Doctoral Dissertation
Doctoral Program in Aerospace Engineering (30th cycle)

# Development of Innovative GNC Algorithms for Aerospace Applications

By

**Matteo Dentis**
******

**Supervisor(s):**
Prof. Giorgio Guglieri
Prof. Fulvia Quagliotti
Ing. Simona Ferraris

**Doctoral Examination Committee:**
Prof. Franco Bernelli Zazzera, Politecnico di Milano
Prof. Salvatore Brischetto, Politecnico di Torino
Prof. Giovanni Mengali, Università di Pisa
Prof. Kimon Valvanis, University of Denver

Politecnico di Torino
2018

# Declaration

I hereby declare that, the contents and organization of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Matteo Dentis
2018

* This dissertation is presented in partial fulfillment of the requirements for **Ph.D. degree** in the Graduate School of Politecnico di Torino (ScuDo).

*A nonno Paolo.*

*To my grandfather Paolo.*

# Ringraziamenti

Questi anni di dottorato sono stati senz'altro impegnativi, ma hanno anche contribuito alla mia crescita personale e professionale, regalandomi molte soddisfazioni. Il risultato di questo lavoro è anche merito di tutti coloro con cui ho avuto il piacere di collaborare e che mi hanno aiutato a superare le numerose sfide che ho dovuto affrontare in questi tre anni, ma è anche merito di chi mi è stato vicino e mi ha supportato, incoraggiandomi sempre e non solo nelle sfide più impegnative.
Vorrei quindi ringraziare i miei tutori, il Prof. Giorgio Guglieri, la Prof.ssa Fulvia Quagliotti e l'Ing. Simona Ferraris, grazie ai quali non sarebbe stato possibile iniziare e concludere questo percorso, Raffaele Aulisio, per tutti i consigli ed i suggerimenti che mi ha dato durante la nostra stretta collaborazione, e tutti i ragazzi del COSE Centre di Thales Alenia Space, che mi hanno nuovamente ospitato in questi tre anni. Assieme a loro, vorrei anche ringraziare tutti coloro con cui ho avuto il piacere di collaborare al Politecnico di Torino, in particolare l'Ing. Elisa Capello, ma anche amici e colleghi dottorandi con cui abbiamo condiviso questo percorso.
Vorrei ringraziare la mia famiglia, che dopo le lauree triennale e magistrale, mi ha nuovamente supportato e sopportato durante il dottorato. Un ringraziamento speciale va alla mia dolce metà, Ilaria, che mi ha accompagnato fin dall'inizio di questo percorso e che più di tutti mi ha sempre aiutato ad affrontare con ottimismo e tenacia tutte le sfide incontrate in questi anni, spronandomi a dare sempre il meglio di me stesso e sopportando tutti i miei sfoghi: sei stata il mio punto di riferimento e lo sarai per sempre. Spero di avervi resi orgogliosi!

*Matteo*

# Acknowledgements

These years of Ph.D. studies have certainly been challenging, but they have also contributed to my personal and professional growth, giving me many satisfactions. The result of this work is also due to all those with whom I had the pleasure of collaborating and who helped me to overcome all the challenges I faced in these three years, but it is also due to those who have been close to me and supported me, always encouraging me and not only in the most demanding challenges.
I would like to thank my tutors, Prof. Giorgio Guglieri, Prof. Fulvia Quagliotti and Eng. Simona Ferraris, thanks to whom it would not have been possible to start and finish this path, Raffaele Aulisio, for all the advice and suggestions he gave me during our close collaboration, and all the guys at COSE Center in Thales Alenia Space, who have hosted me again in these three years. Together with them, I would also like to thank all people with whom I had the pleasure of collaborating at the Politecnico di Torino, in particular Eng. Elisa Capello, but also friends and PhD fellows with whom we shared this path.
I would like to thank my family, who, after bachelor and master degrees, have supported and sustained me during my Ph.D. studies. A special thanks goes to my sweet half, Ilaria, who has accompanied me since the beginning of this journey and that more than anyone has always helped me to face with optimism and tenacity all the challenges encountered in these years, encouraging me to always give the best of my ability and enduring all my outbursts: you have been my point of reference as you will be forever. I hope I have made you proud!

*Matteo*

# Abstract

The main context of the present dissertation is the SAPERE STRONG (Space Advanced Project for Excellence in Research and Enterprise – Sistemi, Tecnologie e Ricerche per l'Operatività Nazionale Globale) project, founded by Italian Ministry of University and Research (MIUR) with the goal to improve Italian access to Space and Space Exploration. For this purpose, extension of the launch capability of the Vega launcher is included in the project, realized with a Space-Tug which is used to deploy in the nominal orbit a payload spacecraft. This thesis has the objective to develop an advanced orbital simulator as a tool which makes the designer able to develop and test the Guidance, Navigation and Control (GNC) software for the Space-Tug spacecraft. The GNC software is developed in collaboration with the leader industrial company of the project, Thales Alenia Space. Thales Alenia Space (TAS) is in charge of developing the Navigation and Control Function and the main structure of flight software, while Politecnico di Torino collaborates with the development of the Guidance function and the orbital simulator. During the whole project has been planned an internship of 1500 hours inside the offices of TAS in Torino. The project includes also a visiting period of international institution. In the specific frame of this Ph. D. thesis, has been spent three months at the University of Sevilla, with the purpose of study and design of a Galileo receiver as an additional input for determination of position in advanced navigation systems, since the Galileo constellation is near to be fully operative in the next future. Details related to all the activities executed during this internship will be presented in Appendix B.

The main objective of this dissertation is the development of innovative GNC algorithms, focusing mainly on the Guidance problem, for aerospace application. An extensive literature review of existing guidance law, control techniques, actuators for attitude and trajectory control, sensors and docking mechanism and techniques has been performed. The Guidance topic has been analyzed focusing on the missile-derived Proportional Navigation Guidance (PNG) algorithm, Zero-Effort-

Miss/Zero-Effort-Velocity (ZEM/ZEV) algorithm and Lambert guidance. Feasibility, performance, pros and cons have been extensively studied in this work, especially in an experimental fashion, and new solutions and implementation strategies have been proposed. The literature review has been completed for Control and Navigation issues, as well. Control strategies, actuation systems and algorithm have been investigated, starting from the classical Proportional/Integrative/Derivative (PID) controllers, to more recent and innovative control law, such as Linear Quadratic Regulator (LQR). As for the Control function, the Navigation topic, intended as navigation filters and algorithms, has been studied in the last period of this work, while the navigation problem form the hardware side (i.e. sensors) has been deeply analyzed in the present work.

In addition to the GNC investigation, the simulation topic has been studied as well, since one of the goals of this dissertation is the realization of an orbital simulator. The orbital simulator is a complete 6 degrees-of-freedom simulator, based on the relative equation of motion (Hill's equations) for the trajectory computation and based on the classical rigid body equation, including the quaternion notation, for the computation of the attitude dynamics. The orbital environment is well defined, including all common disturbances found in Low Earth Orbits (LEO) and affecting the dynamics of an orbiting body. A complete set of sensors is implemented, including an accurate model of common measurement errors affecting the sensors included in the spacecraft configuration (Inertial Measurement Unit, Star Tracker, GPS, Radio Finder, Lidar and Camera). Actuators are carefully modeled, including a reaction wheels system and a reaction control thrusters system. Errors derived for misalignment of the wheels system and non-nominal inertia and shooting and misalignment errors for the thrusters systems are modeled as well.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

In this chapter, the historical background of past space mission will be provided, focusing on rendezvous and docking missions. Modern rendezvous and docking missions will be introduced as well, including a detailed description of rendezvous strategies and docking/berthing mechanisms evolution. Guidance, Navigation and Control systems and techniques are also introduced, mainly focusing on the guidance topic. Eventually, a brief description of the Space-Tug mission scenario investigated within SAPERE STRONG project is provided.

## 1.1 Historical Background

Space exploration found its origin in the Cold War age, when there were an heated rivalry between the United States of America (USA) and the Union of Soviet Socialist Republics (URSS). At that time there were a strong competition between the two factions, not only by the geopolitical point of view, but also in the scientific field, in which is located the *Space Race*. The scientist who can be considered the father of the space exploration is *Wernher von Braun*, a German scientist who designed the weapon V-2 during the World War II, which become the basic design for the future space rockets. Other scientists gave a huge contribution to the rocket science, such as *Robert Goddard*, *Hermann Oberth*, *Sergej Korolëv* and *Konstantin Tsiolkovsky*, who was the first scientist who theorized many aspects of the space flight and rocket propulsion (the famous *Tsiolkovsky's rocket equation*). A non-scientific but romantic contribution to the access to space, before the technology comes, can be found in novels by *Jules Verne* and *Herbert George Wells*, pushing the men's fantasy to realize space vehicles.

There were a lot of tests and failures before the first success occurred with the flight of the Soviet satellite *Sputnik 1* on 4 October 1957, which was a very simple satellite with the goal of transmitting a series of *beep* sounds while operates in orbit. The second success was again by the URSS, with the *Sputnik 2* on 3 November 1957 which carried a little dog named *Laika*, which became the first living being into space. The first man in space was the Soviet cosmonaut *Yuri Gagarin* who flown aboard the *Vostok 1* on 12 April 1961. The first American space flight was made by *Alan Shepard* with the *Mercury-Redstone 3* vector, named also *Freedom 7*, on 5 May 1961. It was a suborbital flight because of performance limitation of the Redstone rocket, which didn't allow the Mercury capsule to reach the Low Earth Orbit, since it was derived by the Redstone ballistic missile which was not designed to reach the Earth orbit. The Shepard's flight was followed by the *John Glenn*'s flight aboard of a *Mercury-Atlas 6* rocket on 20 February 1962. This time the Atlas vector provided enough power to reach the orbit, then Glenn completed three orbits around the Earth before splashing down in the North Atlantic ocean. The first woman flying into space was the cosmonaut *Valentina Tereshkova* on 16 June 1963 aboard the *Vostok 6*, completing 48 orbits.

After the declaration to the Congress of US president *John Fitzgerald Kennedy* on 25 May 1961, the *Apollo Program*, reviewed from the original one conceived during the previous administration of presidents *Dwight Eisenhower*, started with the objective of *"... landing a man on the Moon and returning him safely to the Earth*[1]*"*. The Apollo program introduced such challenging goals, also in terms of time requirements *"before the end of the decade"*, that a second program, the *Gemini Program*, started few years later, with the goal to develop and test all enabling technologies to be implemented in the Apollo program. The problem of reaching the Moon was solved by the *staging* technique, in which the launcher vehicle can be separated in different stages during the ascent to the orbit in order to reduce the mass of each segment once the propellant ran out, making the launch phase more efficient and being able to place into orbit larger spacecraft as final elements to reach the Moon. The output of a large number of studies was that the only affordable way to land on the Moon and return to the Earth was to reach the Moon with two spacecraft,

[1]John F. Kennedy, "We choose to go to the Moon..." - speech of 12 September 1962

land one of them on the Moon - the *Lunar Module (LM)* - and return to the Earth with the second one - the *Command and Service Module (CSM)*. To accomplish this goal, the National Aeronautics and Space Administration (NASA) proposed the solution to assemble the two-module spacecraft, CSM + LM, once in orbit and then reaching the lunar transfer orbit. This methodology of assembling a spacecraft in orbit has been called *orbital docking*. This technology had been tested in the Gemini program.
The docking technology had been tested gradually: indeed, it was tested first the *orbital rendezvous*, a technique which brings two spacecraft, launched at different time, to meet in orbit and getting close enough to have a sort of formation flight until reaching the docking between them. The first attempt of a rendezvous maneuver was executed on June 3, 1965, with a *Gemini 4* spacecraft, piloted by astronaut *James McDivitt*: the goal was to meet the Gemini 4 spacecraft with the spent upper stage of the *Titan II* launch vehicle. The Gemini 4 rendezvous failed because of a number of factors: depth-perception problems of the pilot didn't allow to reach the station-keeping condition, while a stage propellant vent started to kept it moving around. However, the main problem was due to the lack of knowledge of NASA's engineers about relative orbital dynamics, which was not well known at that time. The second rendezvous attempt was a great success: on December 15, 1965, astronauts *Walter Marty Schirra Jr.* and *Thomas Patten Stafford* piloted the *Gemini 6A* spacecraft close to the *Gemini 7*, piloted by *Frank Frederick Borman* and *James Arthur Lovell Jr.*, consolidating the rendezvous strategy. In the same mission, Gemini 7 was establishing a new record of 14 days of a crew in space, the longest period at that time. On March 16, 1966, at 22:14 UTC, the crew of *Gemini 8*, astronauts *Neil Alden Armstrong*, the future first man on the Moon, and *David Randolph Scott*, successfully rendezvous and docked with the autonomous *Agena* space vehicle. This mission was close to be a tragedy in the history of space flight due to problems occurred to the Orbit Attitude and Maneuvering System of the Gemini spacecraft and a mission abort and an emergency landing completed the mission before the scheduled plan, but, despite the problems, the docking execution between two spacecraft was a huge success. The technology progress was finally ready to bring the man to the Moon.

The Gemini program gave an important contribution to the Apollo program, and it was born with this purpose. Starting from *Apollo 9*, all Apollo missions performed orbital rendezvous and docking between CSM and LM to reach the Moon. Apollo 9, with astronauts *James McDivitt*, *David Scott* and *Russell Schweickart* tested the first rendezvous and docking mission in Earth orbit. In the next *Apollo 10* mission, with astronauts *Thomas Stafford*, *John Watts Young* and *Eugene Andrew Cernan*, the previous mission was repeated in Lunar orbit, in addition to other operations such as lunar descent and ascent to get back to the Earth. Form *Apollo 11* to *Apollo 17*, the rendezvous and docking in Earth and Lunar orbit became a common practice of space exploration.

After conclusion of the Apollo program, with Apollo 17 mission, orbital rendezvous and docking experiments continued. From 1973 to 1979, the United States realized the *Skylab*, an orbital laboratory developed by modifying the second stage of a *Saturn IB* rocket. This orbital laboratory had to be supplied with crew and expendables, like food, with additional missions using Apollo CSMs. To re-supply the Skylab, the rendezvous and docking mission was again the leading enabling technology. A collaboration project between USA and URSS, with the *Apollo-Soyuz Test Project (ASTP)*, in July 1975, brought the first international cooperation in space exploration. In this experiment, an Apollo CSM-111 spacecraft had been docked with a Soyuz 7K-TM module, connecting them with a docking module specifically designed for this mission and stored in the upper stage of a Saturn IB rocket, similarly to the Apollo Lunar Modules. The docking module had two different functions: it was an airlock system, because the two spacecraft had very different atmosphere mixtures (340 mbar pure oxygen for the Apollo and 1000 mbar nitrogen/oxygen mixture for the Soyuz), and it was an adapter, since the Apollo was equipped by the probe and drogue docking system while the Soyuz was equipped with the Androgynous Peripheral Attach System (APAS) developed for this mission. This project was a great success for both the scientific and political point of view: this collaboration sentenced the end of the Space Race.

## 1.2 Modern Rendezvous and Docking Missions

Since the beginning of the space exploration, as described in the previous section, the rendezvous and docking technology opened the access to space exploration both in Earth orbit and Lunar missions. The modern era of space exploration, focusing on rendezvous and docking (RVD) missions, started with the Soviet *Mir* orbital station. The Mir station itself is composed as an assembly of different modules and then each assembly operations was a particular docking mission. In addition, since the Mir hosted a stable crew, supply missions became usual, and executed supported by atonomous systems. Indeed, differently from all the previous Gemini and Apollo missions, which was piloted by humans, during the Mir operations there were used the autonomous *Progress* spacecraft, that was, as the current versions, a derivation of the manned Soyuz spacecraft. The Progress spacecraft was one of the first autonomous spacecraft able to execute a rendezvous and docking mission autonomously, even if ground operators and Mir crew supervised all the operations.

After the Apollo program, NASA started developing the next orbital system, the *Space Shuttle*. The Space Shuttle was an innovative space system able to land on ground like a glider and being a reusable system. Its main purpose was to deploy in their operative orbit different space systems and also execute servicing mission to orbiting spacecraft, as occurred several time to the *Hubble Telescope*. The Space Shuttle included new avionic systems and computers, then automatization of a lot of procedures was possible, such as the rendezvous and docking missions. The final approach until the docking was always assisted by human operators. The Space Shuttle was also involved in the *Shuttle-Mir Program*, a second collaboration program between United States an Russia, with the purpose to enhance collaboration between the two national space agencies, NASA (USA) and Roskosmos (Russia), to learn by the Russian experience the long duration space flight experienced during the Mir program, and to prepare the next development of the *International Space Station*.

The International Space Station (ISS) is the result of cooperation among different countries, USA, Russia, Canada , Japan and different European countries members of ESA (European Space Agency). As for the Mir station, the ISS has been assembled module by module since the first mission and it is currently under continuous update. Each module is docked with the other, in which the main of them have been assembled directly with docking missions, while recent modules have been assembled using robotic arms installed on the station, while a rendezvous mission is always required. On board the ISS there is a constant crew composed by six members, rotating in group of three, then the need of supplies, such as food, oxygen and experiments, is constantly required. In the ISS scenario, a step forward in rendezvous and docking operations has been made, introducing more frequent autonomous resupply missions, such as ATV, HTV, Progress, Cygnus, while the crew replacement is currently executed using the well tested Soyuz spacecraft. The docking phases was progressively delegated to the autonomous on-board computer, while cosmonauts and astronauts are asked to supervise the mission and to intervene only in emergency cases. Moreover, in the ISS scenario, a new concept of "docking" has been applied: the *berthing* approach. Differently from the docking mission, in which two vehicles are forced to get in contact with a sort of "controlled collision" and activating the docking mechanism which consolidate the mechanical connection once they are in contact, the berthing operation allows a more smooth connection between two modules: the first step is a rendezvous maneuver which brings the two spacecraft to get close up to few meters, then, a robotic arm operating from the main spacecraft, the *Chaser* vehicle, is moved to grasp the second spacecraft, the *Target* vehicle, and place it close to the docking port, eventually completing the hard docking driving the relative motion between Target and Chaser with the robotic arm. This scenario is followed by the Cygnus rendezvous and docking/berthing resupply missions.

## 1.3 Rendezvous and Docking Strategies

In this section, a brief summary of the common rendezvous and docking/berthing strategies is provided. First, a brief discussion about the local reference frame used for rendezvous maneuver description is provided, even tough a complete discussion about reference frames will be provided in Section 3.3.1. The local frame used as reference for RVD mission is depicted in Fig. 1.1. The origin of the frame is located in the center of mass of the target spacecraft, a Space Station as in the example in Fig. 1.1: the x-axis is aligned along the orbital velocity of the target spacecraft and it is named $V_{BAR}$; the z-axis is pointing the Earth in the radial direction, so it is called $R_{BAR}$; the y-axis is aligned in the out-of-plane direction of the orbital plane, in order to complete the right-handed frame, and it is named $H_{BAR}$.

The goal of a Rendezvous and Docking maneuver is to get closer to the target spacecraft and to dock with the assigned docking port located on the target spacecraft. The docking axis can be aligned with one of the axis of the local reference frame, and consequently the RVD maneuver will follow different approaches, depending by the position of the docking port. Other factor that affect the RVD strategy is the illumination condition: indeed, it is recommended to execute the final approach and docking in sunlight condition. Two examples of RVD maneuver will be briefly described in the following: the Space Shuttle and the Soyuz RVD. Both RVD

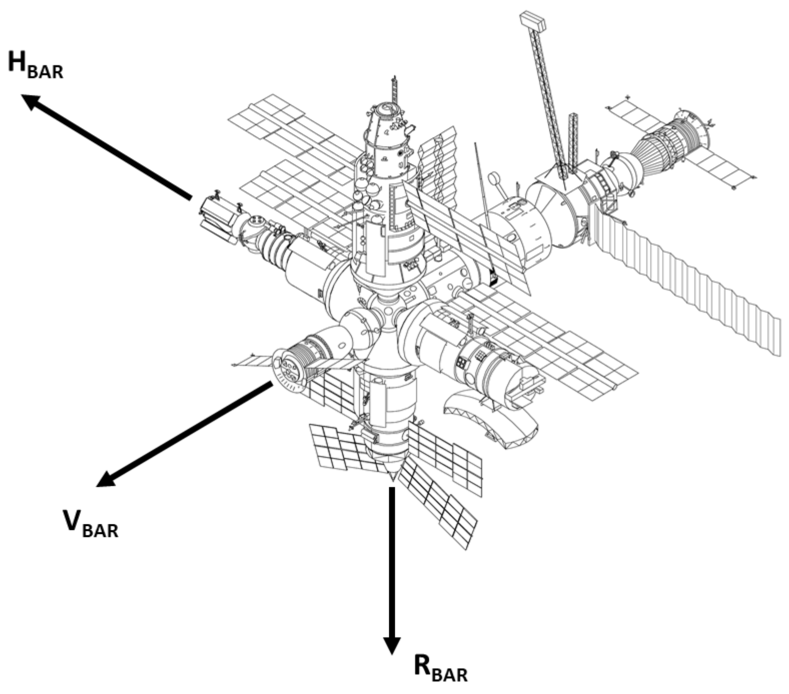


Fig. 1.1 Local frame for rendezvous missions.

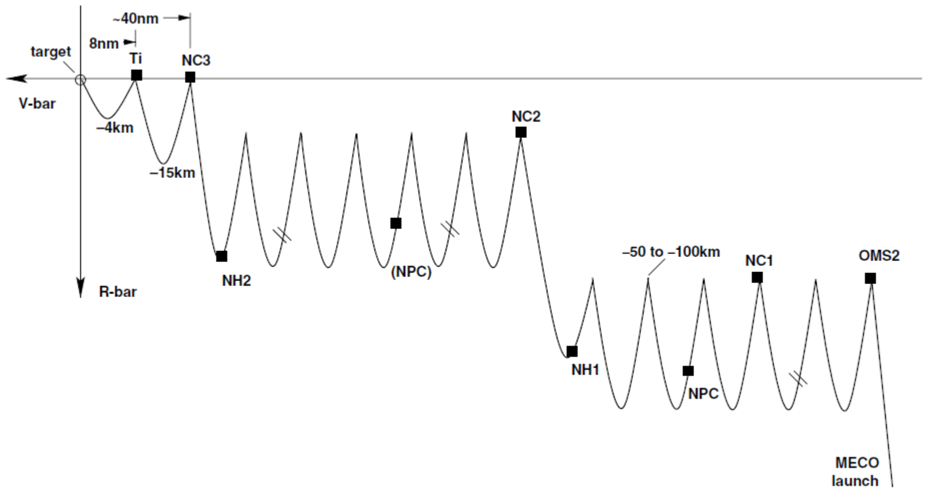
maneuvers have some commonalities which are the basis of the orbital rendezvous maneuver. In general, a rendezvous and docking maneuver can be divided in four main maneuvers: *phasing*, *far range rendezvous*, *close range rendezvous* and *final approach*.

Shortly after the orbital ignition by the launcher system, some corrections are applied in order to correct the actual orbit and execute the *phasing* maneuver. During this phase, the chaser vehicle, the Space Shuttle Orbiter or the Soyuz, is located in a different orbit with respect the target one, usually at a lower altitude and in a different orbital plane, due to launcher limitation and/or orbital ignition errors. Since the chaser in the lower orbit has a shorter orbital period than the target in the higher orbit, the chaser is orbiting faster than the target, and the phase angle between the target and the chaser spacecraft is reduced after a number of orbital revolutions. The Shuttle and the Soyuz follow different strategies to complete this phase. The Space Shuttle executes a series of in-plane maneuvers, *NC* and *NHC*, in order to correct and support phasing, correct thrust errors occurred in previous maneuvers and correct the altitude of the orbit with a lager maneuver, and a series of out-of-plane maneuvers, *NPC*, in order to correct inclination and RAAN during phasing. Differently from the Shuttle strategy, the phasing maneuver of the Soyuz consists to reach a near circular orbit at a lower altitude and to apply a series of corrections to the actual orbit. The phasing maneuver executed with near circular orbits allows an easier computation of maneuver schedule and planning. For both the Space Shuttle and the Soyuz spacecraft, the phasing maneuver is commanded by ground operators. Far range maneuver for both Shuttle and Soyuz spacecraft is depicted in Fig. 1.2.

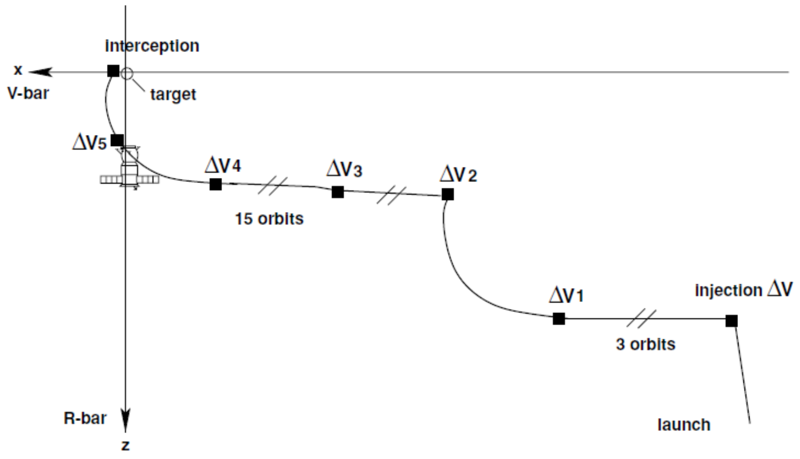
The *far range* maneuver is executed after phasing. Such maneuver usually starts at about 10-15 km far from the target spacecraft. According to the Shuttle strategy, the far range RVD starts at point *NC3* of Fig. 1.2a (or *NC* of Fig. 1.3a). This is the first maneuver controlled autonomously by the on-board control system of the spacecraft. A series of correction are applied in order to reach the final point, $T_i$, of the maneuver with the desired terminal conditions. As for the Space Shuttle, the far range maneuver of the Soyuz spacecraft is controlled by the on-board control system by computing orbital elements uploaded during the previous phase. The maneuver is executed between points *M4* and *M5*, and the goal is to reach the aim point far approximately 1.5 km far from the target.

The successive maneuver, the *close range* RVD, is executed with strongly different strategies between Space Shuttle and Soyuz. The first vehicle executes a maneuver similar to the previous one, starting from the point $T_i$ and ending at a different point depending by the desired final approach: in a $\mathrm{V_{BAR}}$ approach the maneuver ends in a point located about 150 m in front of the target, while in a $\mathrm{R_{BAR}}$ approach the maneuver ends when the Shuttle is crossing the $\mathrm{R_{BAR}}$ axis of the target. During this maneuver, controlled by the on-board control system, a set of mid-course corrections are applied. The close range RVD of the Soyuz spacecraft is executed between points *M6* and *M8*, with intermediate corrections, as in Fig. 1.3b. The goal of this maneuver is to bring the spacecraft into a fly-around maneuver with respect to the target spacecraft with radius of approximately 400 m (point $\mathrm{T_b}$ of Fig. 1.5).

The *final approach* maneuver is the last maneuver prior to the docking/berthing phase. For both Shuttle and Soyuz spacecraft it can be executed following a $\mathrm{V_{BAR}}$ (Fig. 1.4a) or $\mathrm{R_{BAR}}$ (Fig. 1.4b) approach, depending by the available docking port of the target spacecraft. While this maneuver is executed autonomously by the on-board computer of the Soyuz spacecraft, driven by the *Kurs* system, the final approach of the Space Shuttle is manually controlled by the astronauts inside the orbiter, supported by a set of sensors including the Crew Optical Alignment Sight (COAS)

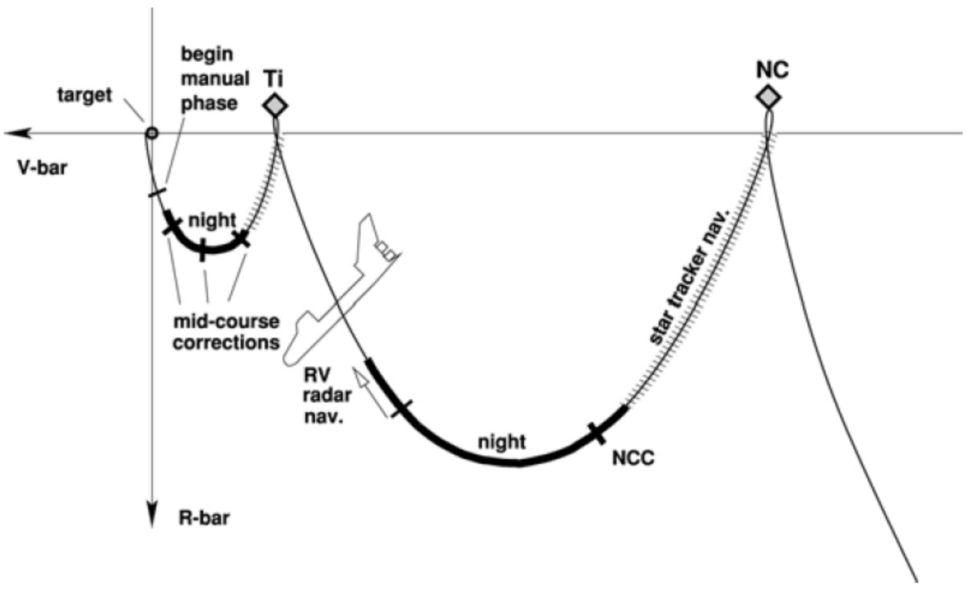


(a) Space Shuttle phasing and far range RVD.

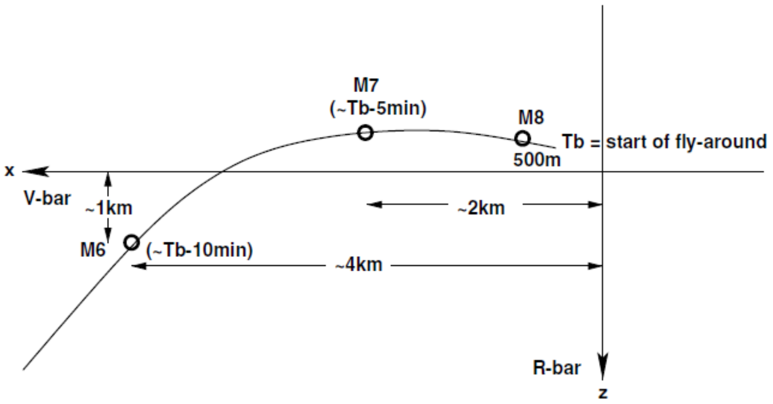


(b) Soyuz phasing and far range RVD.

Fig. 1.2 Space Shuttle and Soyuz early RVD maneuvers.

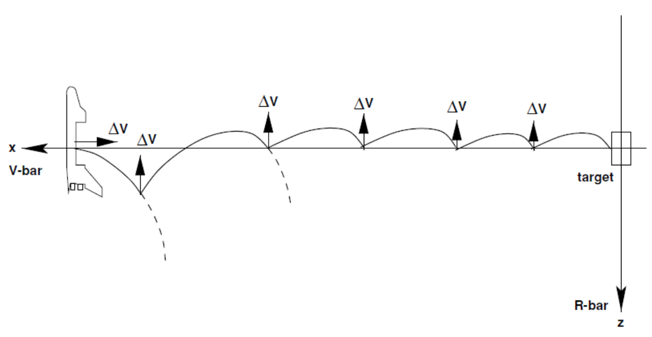


(a) Space Shuttle close range RVD.

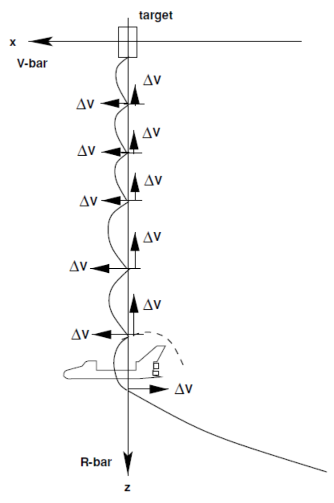


(b) Soyuz close range RVD.

Fig. 1.3 Space Shuttle and Soyuz close range RVD.



(a) Space Shuttle final approach $V_{BAR}$.



(b) Space Shuttle final approach $R_{BAR}$.

Fig. 1.4 Space Shuttle final approach.

(an optical passive alignment system used since Apollo missions), the Closed Circuit Television System (CCTV) (two cameras installed in the cargo bay measuring the alignment angle from the Shuttle and the target) and the Trajectory Control Sensor (TCS) (a laser sensor which track the Line-Of-Sight (LOS) vector). More details about the Space Shuttle reference RVD mission can be found in [1].

This final maneuver is completed by executing the *Docking* or the *Berthing* with the target spacecraft. The *Docking* consists in the chaser getting closer to the docking port of the target until the contact between the two spacecraft: after the first contact the two spacecraft are mechanically connected, but the link still allows a limited relative motion between them, so this first phase is usually named *soft docking*. Once the soft mechanical link is activated, a second mechanism moves closer the two spacecraft and it performs a more robust connection between them: at this step, the two spacecraft are strongly connected and they can be considered as a single vehicle (by the point of view of the dynamical behaviour), and then the *hard docking* is completed.

The *Berthing* maneuver, as for the docking one, consist in getting closer the chaser to the target, but the approaching is stopped at a distance of few meters from the target spacecraft. The chaser is holding such relative position while a robotic arm located on the target spacecraft starts operating in order to catch the chaser with

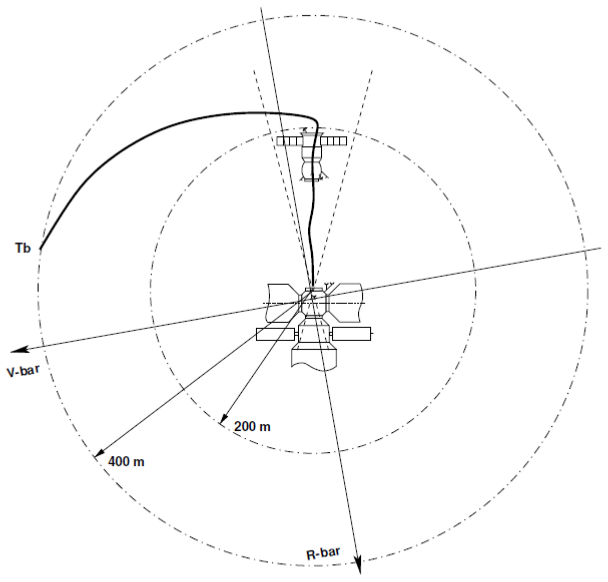


Fig. 1.5 Soyuz $R_{BAR}$ final approach.

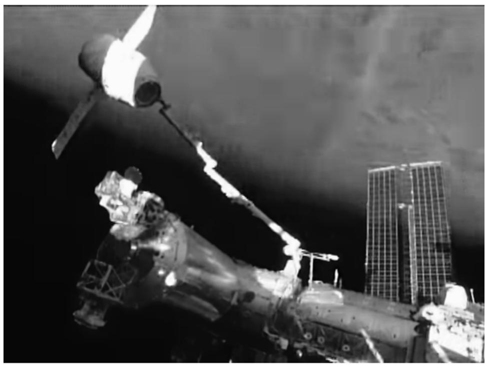
the end effector of the robotic arm and moving it to connect it to the desired docking port. In this case, a "passive" grasping mechanism, or grasping spot, compatible with the "active" grasping mechanism of the robotic arm shall be installed on the chaser. The berthing strategy can be considered safer than the docking one, since the connection by the two spacecraft is executed operating a robotic arm instead of be driven by the on-board computer of the chaser and actuated by the thruster system, and a less number of failures may occurs, in addition to the lower mechanical energy involved. Currently, the berthing maneuver is extensively used in resupply mission to the International Space Station with Dragon or Cygnus spacecraft, and the robotic arm is operated by an astronauts in the ISS. In Fig. 1.6 it is depicted the four main phases of the berthing maneuver of a Dragon spacecraft: the spacecraft is in station keeping few meters from the ISS (Fig. 1.6a); the robotic arm grasp the spacecraft (Fig. 1.6b); the spacecraft is moved towards the docking port (Fig. 1.6c); the spacecraft is connecting to the docking port (Fig. 1.6d).

## 1.4 Docking/Berthing Mechanism

In this section, a brief overview about existing docking and berthing mechanisms will be provided. The first mechanism examined is the *Probe and Drogue* mechanism of

(a) Dragon arrival and station keeping.

(b) Grasping of Dragon.


(c) Moving Dragon to the docking port.

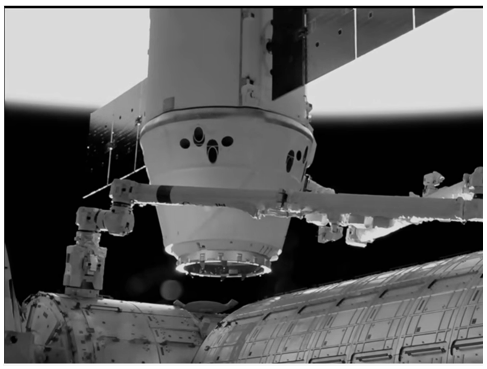

(d) Docking of Dragon.

Fig. 1.6 SpaceX Dragon CRS-13 arriving at the Interational Space Station.

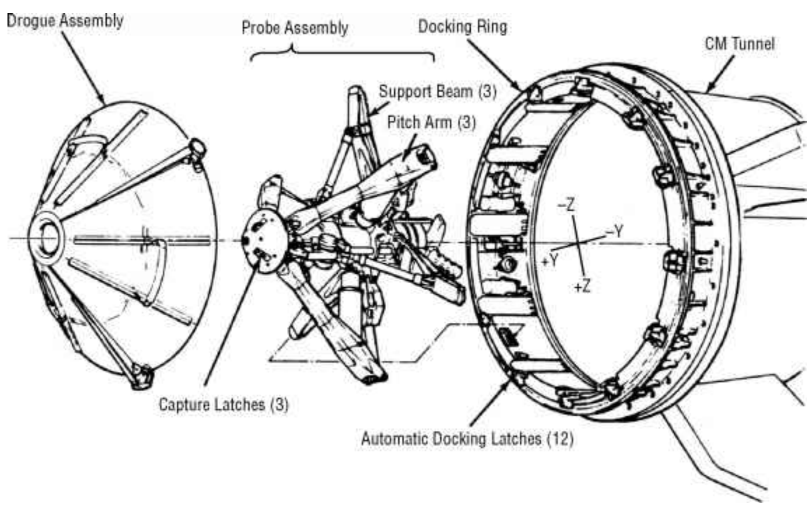
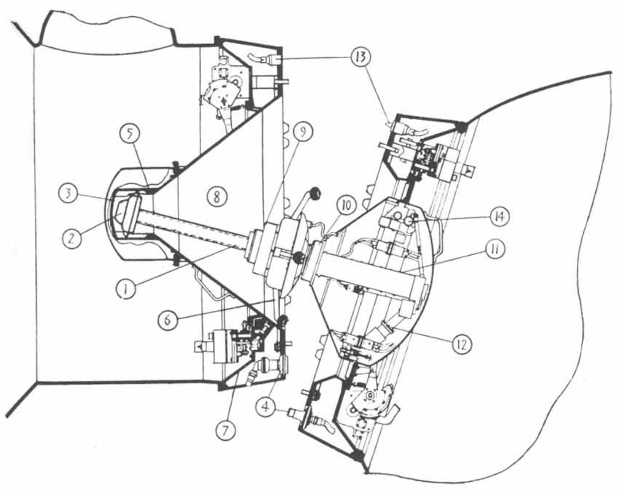


Fig. 1.7 Apollo probe and drogue docking mechanism.

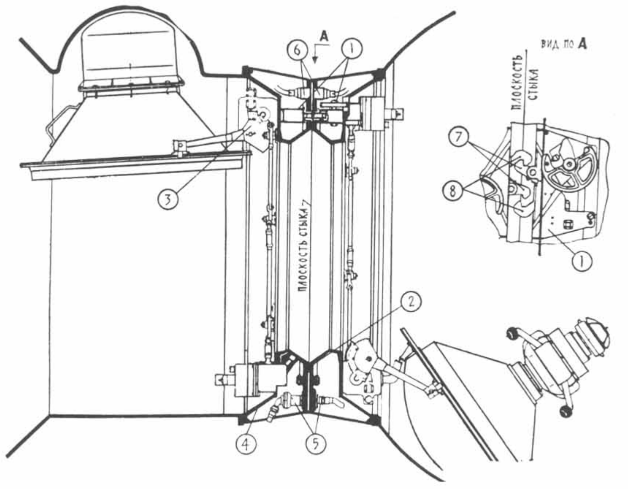
the Apollo missions, depicted in Fig. 1.7. The working principle of the mechanism is the following: during the approaching phase, the probe is extended to the maximum length; the drogue assembly has a conic shape in order to allow a limited alignment error between the two mechanisms and to drive the capture latches of the probe through the hole at the top of the drogue assembly; when the the capture latches of the probe capture the drogue assembly, the soft docking is realized and the probe starts retracting: the petals named *pitch arms* are extending while the probe is retracting, limiting the relative movement during the probe retraction phase; when the probe is completely retracted, the two docking rings located on the chaser and on the target are in contact and latches of the docking rings are activated and connect strongly the two spacecraft: the hard docking is completed. The probe and drogue system, once it is connected, allows the opening of the hatch, which permit the movement of astronaut, supplies and atmosphere between the two environments.

The Soyuz/Progress docking mechanism is similar to the Apollo probe and drogue system. In Fig. 1.8 it is depicted the schematic sketch of the mechanism. As for the Apollo, the probe of the Soyuz is extended and it capture the top of the drogue cone. After the capture (soft docking) the probe is retracted until the hard docking is completed after latching, then the hatch can be opened.

A different type of mechanism is the *Common Berthing Mechanism (CBM)* [2], depicted in Fig. 1.9. This mechanism is used to dock pressurized modules to the International Space Station, such as HTV, Cygnus and Dragon. The mechanism is composed by two parts: the CBM Active Half, installed on the ISS module, and



Soyuz probe and drogue after initial capture: (1) probe; (2) probe head with capture latches; (3) drogue; (4) hydraulic umbilical connector; (5) stop; (6) alignment pins; (7) docking structural ring; (8) drogue cone; (9) electric drive for retraction of probe stem; (10) ball joint; (11) probe guide; (12) lateral shock absober; (13) electrical umbilical; (14) electromechanical damper.



Soyuz docking assembly after latching has been completed and hatches have been swung open to permit transfer of crewmen: (1) peripheral latch, (2) docking interface seal; (3) hatch cover drive; (4) docking (structural) ring; (5) hydraulic umbilical; (6) electrical umbilical; (7) active hooks; (8) passive hooks.

Fig. 1.8 Soyuz/Progress docking mechanism.

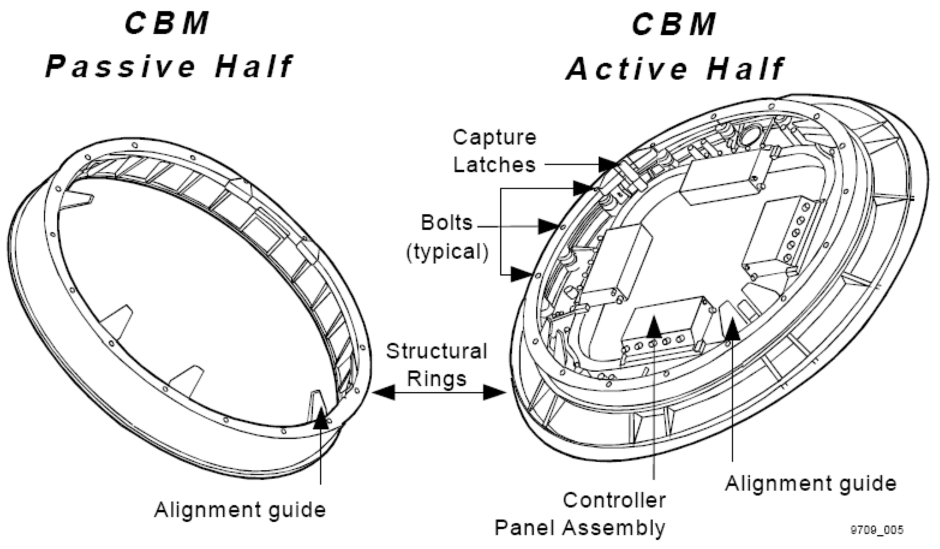


Fig. 1.9 Common Berthing Mechanism.

the CBM Passive Half, installed in the HTV/Cygnus/Dragon module. The passive half is a simple structural ring which allows a complete closure and ensures the pressurizing capability of the modules. The active half, in addition to the structural and pressurization functions, includes also the mechanism (capture latches) which allows the capture, closure, and passive structural connection between the modules. Such docking mechanism is also used to connect modules of the ISS, such as *Unity* which was the first module equipped with this mechanism. The berthing maneuver to the ISS is completed operating the *Space Station Remote Manipulator System (SSRMS)*, also known as *Canadarm 2*[2]. The Canadarm 2 is a robotic arm used for different purposes, such as berthing of arriving resupply modules or installation and/or relocation of modules of the ISS, as well as it can be used in support to EVA (Extravehicular Activity) operations. The Canadarm 2 is equipped with two Latching End Effectors (LEE) at its extremities, both able to grasp different grapple fixtures located on ISS modules or on resupply modules. On-orbit LEE and example of the grapple fixture is depicted in Fig. 1.10. A survey of interfaces used in berthing operations on ISS can be found in [3].

A different category of docking systems can be found in *androgynous* docking systems, such as the Space Shuttle docking mechanism depicted in Fig. 1.11. This

[2]The Shuttle Remote Manipulator System (SRMS), or Canadarm, was the equivalent robotic manipulator used by the Space Shuttle during payload operations. The SSRMS is derived by the SRMS design.

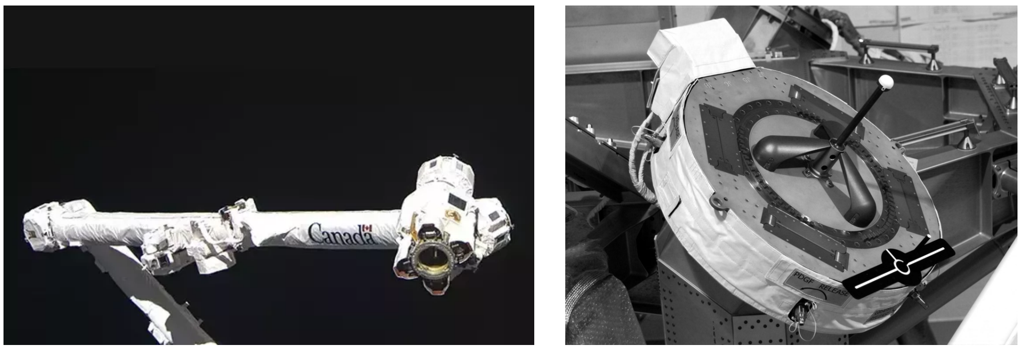

(a) SSRMS and LEE. (b) Grapple fixture.

Fig. 1.10 Canadarm 2 and example of grapple fixture.

mechanism, named APAS-95 (Androgynous Peripheral Assembly System) [4], has been used during the rendezvous and docking missions to the International Space Station, even though it is based on the design of the APAS-89 used for the Space Shuttle/Mir RVD missions. This is an androgynous mechanism, i.e. it is not made up by two different "male" and "female" connectors, as for the Apollo and Soyuz docking mechanisms, but the two connectors are basically designed with the same shape. In order to allow a limited alignment error, the connection ring is mounted on a Gough-Stewart platform. The Gough-Stewart platform is a mechanism which allows a 6 Degrees-of-Freedom (DoF) movement of the active ring (see Fig. 1.11) realized by a set of six actuators moving synergistically. The 6DoF motion of the platform is limited by the maximum and minimum extension of the actuators, but it can realize three displacements and three rotations. Hence, such mechanism of the APAS-95 system is able to correct alignment of the docking mechanism of the Space Shuttle and execute the soft and hard docking with the ISS. In addition, there are a set of dampers in order to dissipate the residual impact energy during the final phase of the docking. The hard docking is completed while the latches on the structural ring are locked with the hooks in the opposite structural ring. The hatch can be opened and the two environments are able to communicate. Other application of the Gough-Stewart platform can be found in the moving mechanism of simulators, mostly in the aerospace branch. A complete survey about docking mechanisms developed by NASA can be found in [5].

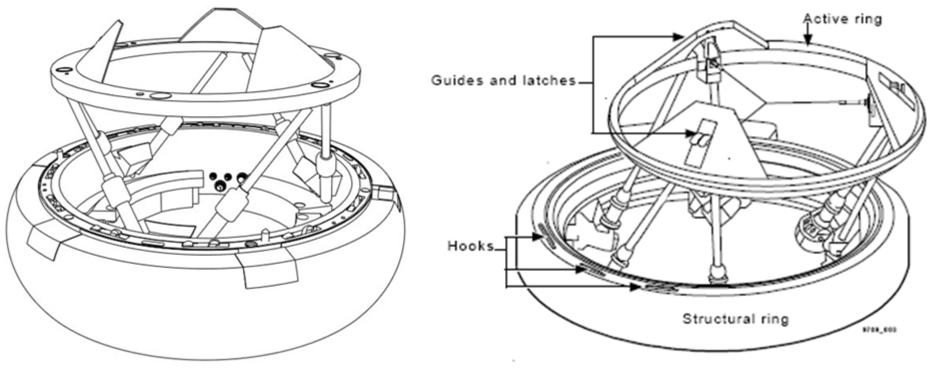


Fig. 1.11 Space Shuttle docking mechanism.

## 1.5 Guidance, Navigation and Control Systems

The design of an efficient Guidance, Navigation and Control (GNC) System is mandatory to execute successfully a space mission. Even if space missions are very different each other, the functional architecture of GNC systems does not differ dramatically from a RVD mission to an Earth observation one. Certainly, differences can be found GNC algorithms driving the system, guidance and control actuators as well as navigation sensors, which differ especially depending by the size of the spacecraft (for example, Cubesats are very limited in terms of hardware that can equip the spacecraft). In the present section existing Guidance, Navigation and Control Systems are extensively investigated, focusing on the state of art in such systems.

### 1.5.1 Guidance

The Guidance function shall, in general, drive the spacecraft to follow a defined path, or trajectory, in order to reach a different orbit or to keep a specific slot in the actual orbit of the spacecraft, counteracting external disturbances. Therefore, an efficient guidance for a rendezvous and docking mission shall compute the optimal trajectory to reach the target spacecraft and shall generate the force or velocity change (i.e. $\Delta V$) command which allows the chaser spacecraft to follow the computed path. Differently, the guidance for an Earth observation mission, a GNSS constellation or a GEO (Geostationary Earth Orbit) application shall maintain the spacecraft within a defined an limited box, or slot, located in the nominal mission orbit. This particular maneuver is called *station keeping*, and it is typically required when orbital elements of the spacecraft must be kept with high accuracy, such as for a GNSS constellation. As it can be noticed by the reader, the two application yet described are very different each other, but in both cases it can be considered a guidance problem too, it doesn't matter if the spacecraft shall reach a different orbit or it shall maintain the nominal one.

The guidance can be realized both autonomously, such as for the present case study of SAPERE STRONG, or manually by ground operators or astronauts (or cosmonauts) commanding the spacecraft. However, even if the guidance is managed by the autonomous on-board computer of the spacecraft, ground operators are continuously supervising the system, as for the ATV rendezvous mission. The ATV RVD operations are managed by the autonomous guidance system which control the spacecraft during all the RVD mission phases but ground operators intervene when the spacecraft reach defined waypoints and they call the "*go*" or "*no-go*" for the subsequent mission phase, depending by if all the parameters are in the correct range or not. Details of the ATV GNC system can be found in [6], [7] and [8]. Differently, the guidance for the Space Shuttle rendezvous maneuver to the ISS is mainly scheduled by ground operators and it depends by the actual orbit of the spacecraft, while the final approach is completely guided by astronauts inside the orbiter, and the autonomous guidance is limited to a single phases or the RVD mission, as discussed in Section 1.3.

Assuming a closed loop guidance for an autonomous GNC system, i.e. the guidance command is computed taking into account the actual state (position and velocity) of the spacecraft fed back by sensors measurements with respect to the desired state, there are two possible strategies in designing the guidance algorithm. Depending by the thrust to mass ratio of the spacecraft, it can be chosen a two-impulse or a continuous thrust maneuver. The *two-impulse maneuver* consists in commanding a first acceleration to reach the commanded $\Delta V$ and a second impulse at the end of the scheduled maneuver. Usually, the second impulse is commanded immediately after half an orbital period is completed, since it is the cruise time typical for an impulsive maneuver. During the maneuver, if a closed loop guidance is implemented, there are a number of correction pulses to keep the spacecraft within the desired trajectory. A real implementation of two-impulse maneuvers can be strongly sub-optimal due to the impossibility to apply an impulsive, i.e. instantaneous, velocity change. Indeed, depending by the thrust to mass ratio and

the magnitude of the desired $\Delta V$, the time required to reach the desired velocity may increase up to a non-negligible time interval compared to the orbital period, hence the $\Delta V$ impulse cannot be considered impulsive anymore, and the maneuver results far from the ideal one. For this reason, choosing this strategy an accurate analysis of the maneuver itself and of the propulsive system is required.
The *continuous thrust maneuver*, instead, consists in providing a continuous thrust along the maneuvering axis (typically $\mathrm{V_{BAR}}$ or $\mathrm{R_{BAR}}$), to reach the desired final position with respect to the target spacecraft. This strategy is suitable for low thrust to mass ratio spacecraft and especially for tunable propulsive systems, such as *electric thrusters*. Since electric thrusters can be found in limited application, mostly for station keeping purposes, a rendezvous maneuver is mainly realized by a set of reaction control thrusters, usually with a on/off working principle, modulating the thrust acting on the opening interval of each thruster, as it will described in the next sections. Hence, this strategy is also limited by the propulsive system of the spacecraft, depending by the maximum and minimum opening time of the flow control valve (FCV) of the thrusters.

Moreover, a good guidance shall be able to manage failures which can occur during the mission, for example a thruster failure, even if a complete failure management is managed at spacecraft system level, mostly including redundancy of systems. Hence, the failure management capability of the guidance system may be limited to the definition of a set of specific safety maneuvers, such as *Collision Avoidance Maneuvers (CAM)*, even though such capability can be realized in conjunction with the Control system.

### 1.5.2 Navigation

The navigation problem dates back to the first European explorers which drove their vessels though the ocean and a good estimation of the actual latitude and longitude measures was fundamental to not get lost in the ocean. Modern navigation techniques are currently based on satellite navigation systems, founding the origin in the TRANSIT navigation system, which was operating from 1964 to the end of 1996, when it was decommissioned and replaced by the *Global Positioning System (GPS)*.
In the framework of autonomous spacecraft, navigation systems can be split in two main layers: the *hardware layer*, which includes all the sensors for position and attitude determination, and the *software layer*, mandatory for all digital systems, which

includes all the software function which performs filtering of sensors measurements and measures data fusion. Indeed, by introducing digital measurement processing techniques it is possible to efficiently mitigate the measurement noise using digital filters and it is also possible to merge multiple sensor measurement to obtain a single estimate of the current state of the spacecraft, for example when more than one sensor is used to detect the same measure due to redundancy requirements. The most used technique to afford the navigation problem is the implementation of a digital *Kalman Filter*, theorized by Rudolf E. Kalman in 1960 and implemented in the Apollo program, and its extended formulation, commonly named *Extended Kalman Filter*, which allows implementation with non-linear systems and data fusion, as discussed below. The extensive study of the Kalman filter is left to the following sections.

The navigation problem analyzed in the present section includes discussions about *position estimation*, both absolute and relative position with respect to the target spacecraft, and the *attitude estimation*, mandatory to satisfy pointing requirements during the whole rendezvous and docking maneuver.

### Position Estimation

The position estimation issue in a RVD mission identify two different problems: (i) to determine the *absolute* position, i.e. the position of the spacecraft with respect to an inertial reference frame, and (ii) to determine the *relative* position with respect to the target spacecraft, i.e. with respect to a relative reference frame. To fulfill the estimation of these two different functions, there are used two different category of sensors, named respectively *absolute* and *relative* sensors. In the category of absolute sensors, the most used one is the GPS sensor, or any equivalent GNSS sensor, such as GLONASS, Galileo or Beidu. A secondary absolute sensor is the *Inertial Measurement Unit (IMU)*, which is a set of accelerometers and gyroscopes used to detect linear acceleration and angular velocity that can be time-integrated to compute linear velocity and position as well as attitude angles. In the past, before the beginning of GNSS, the IMU was the main sensor used to compute velocity and position of vehicles, typically aircraft, moving on Earth. Unfortunately, due to different sources of errors, the IMU cannot be used as a unique sensor to compute the position of a vehicle, even if the ideal model of this measurement technique should provide the exact solution, and the position computed by the IMU had to be frequently corrected. In modern navigation system, the IMU is usually used concurrently with a GNSS system to obtain a better position estimation.

Differently form the IMU sensor, which allow a poor estimation of the position measuring directly the state of motion of the spacecraft, the GNSS navigation consist in determining the position of the spacecraft processing data transmitted by a number of satellites of the GNSS constellation and so estimating the actual position of the spacecraft by a triangulation technique, hence performing an indirect measure. The position measure is obtained by the knowledge of the actual position of, at least, four different GNSS satellites, which transmit their own position with respect to the inertial frame in the navigation message. In principle, it is required the exact knowledge of the position of three satellites to compute the position of the generic GNSS receiver, since the three spheres centered in each satellites are intersecting in only one point, which is the receiver location. Since the signal emitted by the GNSS satellites requires a non-negligible time to reach the receiver, an additional data related to *time* is required to solve the position of the receiver. In addition, due to the high value of the speed of light, which is the vacuum propagation velocity of electromagnet waves (i.e. the navigation message), all the clocks used in the five-element system of receiver and four GNSS satellites must be very accurate an synchronized each other, since an error of few *nanoseconds* could generate a position error up to several kilometers. Hence, since the high-accuracy atomic clocks used in the GNSS constellation are too expensive to be installed also in GNSS receivers, the time error grows in importance and it has to be properly managed. Referring to the scheme depicted in Fig. 1.12, the *pseudorange* measure from each satellites can be computed by

$$\rho^{(i)} = \sqrt{(x^{(i)} - x)^2 + (y^{(i)} - y)^2 + (z^{(i)} - z)^2} - b$$

where the superscript $i$ is referred to the ith satellite, $\rho^{(i)}$ is the pseudorange realted to the satellite $i$, $x$, $y$ and $z$ are the Cartesian unknown coordinate of the receiver, $x^{(i)}$, $y^{(i)}$ and $z^{(i)}$ are the known Cartesian coordinates of the ith satellites and $b$ includes all the clock errors.

As mentioned before, estimation of the chaser/target relative position requires a different typology of sensors. Most of these sensors are optical sensor, such as LIDAR (Laser Imaging Detection and Ranging) or video systems based on cameras. Other system are based on radar system, such as the Kurs system used by Soyuz and

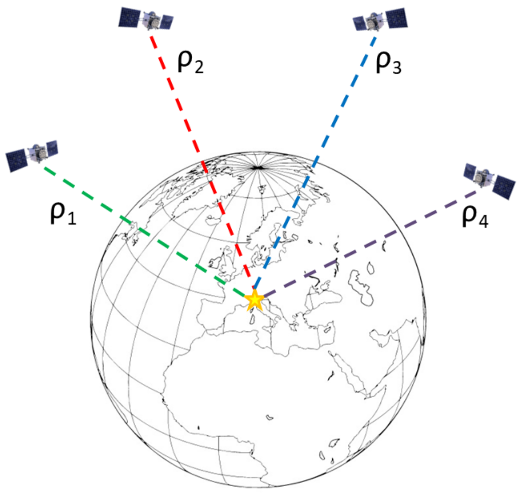


Fig. 1.12 GNSS satellites and receiver minimal configuration.

Progress spacecraft during the RVD maneuvers to the ISS, even if it was extensively used in the past by Russian for docking to the MIR space station, which substituted the previous Igla system. The Kurs system includes four different typology of antennas, installed on both chaser (Soyuz) and target (ISS) vehicles, which are activated differently during the RVD approaching since each antenna is able to measure a specific parameter, such as range and range-rate or attitude angles and rates, with increasing accuracy. Hence, some antennas are used to during the far rendezvous to guide the chaser in the proximity of the target spacecraft, while other antennas are used to guide the chaser to the selected docking port and guide the spacecraft in the final approach to the docking. The navigation executed with the Kurs system allows a complete autonomous rendezvous and docking maneuver since the beginning of using such system in late 60s. More details about the Kurs system and the Soyuz RVD maneuver can be found in [9]. A picture of Kurs antennas installed on both Soyuz spacecraft and International Space Station in shown in Fig. 1.13.

Also the Space Shuttle was including a relative sensor based on a radar system, but such system was able to provide additional information related to the relative attitude only if used in RVD mission to active spacecraft, such as the ISS, while, during servicing mission to different spacecraft, such has the Hubble telescope, the radar of

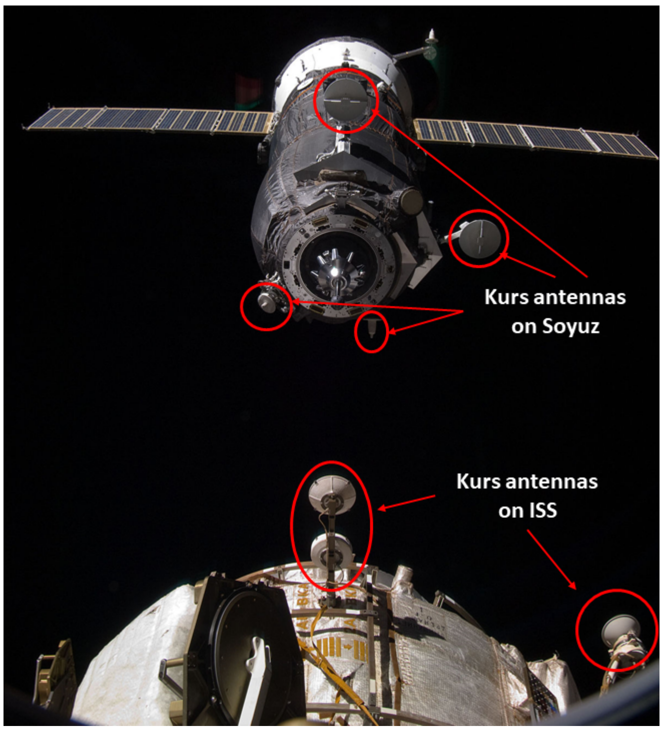


Fig. 1.13 Highlight of Kurs antennas on Soyuz and ISS.

the Space Shuttle was able to provide only range and range-rate measure, since it was measuring only the signal of radio waves reflected by the target.

Advantages of using radio systems for relative navigation are mainly derived by the higher operative range, which in some conditions can result in several hundreds of kilometers [9]. In addition, active radar systems, such as the Kurs system, provide the measure of the relative state of the spacecraft, including both linear and angular parameters, with high accuracy and by using a single sensor, overcoming problems related to performance degradation of radar systems in close distances of few tens of meters. On the contrary, the mass of such systems are quite important (the mass of the Kurs system, including antennas and related hardware, is about 85 kg on the Soyuz spacecraft [9]) and they require to be installed on both chaser and target vehicles, strongly limiting the application in a more general scenario which includes spacecraft not designed with such active systems. Hence, as for the ATV [10], it is common to use different systems for relative navigation, such as *Differential GPS (DGPS)* for far range rendezvous and optical system for closer rendezvous, improving also the reliability of the system.

Optical systems are very accurate compared to passive radar systems, but their operational range is limited to few hundreds of meters. Optical systems, such LIDAR, are based on measuring the phase difference of the signal emitted by the sensor and reflected by markers on the target spacecraft. The laser signal can be of different wavelength, including visible and infra-red light, and accuracy of the system is strongly dependent by the wavelength adopted. Since this system is measuring a specific reflected light, the target spacecraft must be equipped with markers that reflect the laser light, and, again, the application of such system may be limited to target spacecraft designed properly to accept such detection method.
Different optical systems are based on cameras, which detect and process the image of the target spacecraft on the optical sensor (usually a space-qualified CCD) and they are able to compute the relative distance and attitude. The use of advanced image processing techniques may not require installation of dedicated markers on the target spacecraft, even if it is highly recommended the use of markers to reach the prescribed accuracy, especially for attitude estimation. The operative range of camera-based systems is lower than laser-based systems due to the physical principle which governs this measuring techniques: indeed, depending by the quality of the lenses of the optical system, an image of the target detailed enough can be detected only within hundreds of meters of distance, while for higher distances, the target image can be limited to a bright spot in the field of view of the sensor, which can be easily confused with a star.

### Attitude Estimation

Attitude estimation is mandatory non only for rendezvous and docking mission, but also for almost the totality of space mission. The required accuracy in attitude estimation (and control) is different from each specific mission and it derives from specific requirements demanded by payload pointing, thermal or communication issues, and more. For example, the first space missions did not require an accurate attitude estimation, since the most important attitude requirements was dependent from communication and thermal constraints, while a scientific payload, such as an orbital telescope or an Earth observation payload, surely requires a very accurate attitude estimation and fine and stable attitude control. Note that performance of the attitude control system is strictly linked to the accuracy of the attitude estimation.
In the past, a lower number of sensors for attitude determination was available.

As mentioned before, a classical sensor for absolute attitude estimation was the IMU, which requires to be reset on ground before launch and it is sensible to the measure drift after a period of time. The attitude can be estimated by time integration of angular rates measured by IMU gyroscopes, but modern IMU, thanks to huge improvements in digital systems, provide also the attitude estimated by on-sensor computer. Currently, IMU systems are extensively used in modern spacecraft, mainly for measuring of the angular rate of the vehicle, and a great interest is growing in embedding such system in cheap CubeSats [11].

A second attitude estimation sensor is the Sun sensor, which usually consists in photodiodes or photocells used as a voltage sources dependent by the Sun radiation. Simple Sun sensors are producing a Boolean output indicating the presence or not of the Sun in the field of view of the sensor, while more complex configurations are able to provide also the angle of incidence of the vector pointing the Sun, improving the accuracy of the sensor and extending its applicable range. Clearly, due to the working principle of this sensor, it cannot be used as the main attitude determination sensor, since it is able to detect only the Sun and in eclipse period of the orbit of the spacecraft it cannot be used. In addition, the accuracy of this kind of sensor rarely is higher than 1 deg, which is one or two order of magnitude (and more) lower than other attitude sensors, such as star trackers. For those reasons, the application of this sensor is mainly limited to attitude determination in contingency maneuvers, where less restricting pointing requirements are demanded and it is required a simple stabilization of the spacecraft.

The main sensor used currently on spacecraft, including small satellites as CubeSats, is the *star traker* sensor. This sensor consist on a camera which is able to detect and track a large number of fixed stars in the field of view of the sensor. Once the sensor get a "photo" of the sky, the frame is compared with the star catalog stored on the on-sensor computer and so it is possible to evaluate the attitude of the spacecraft with respect such stars, which define an inertial frame. This sensor is able to provide measures with high accuracy, also in cheaper systems. The main limitation of this sensor is that it is very sensible to the direct radiation of the Sun, which can seriously damage the sensor if no protection devices are considered. A further limitation is due to the angular rate of the spacecraft: if the spacecraft is rotating at high angular rate, the accuracy of the sensor is strongly affected, and it cannot be able to detect the attitude due to the refresh frequency of the star detector.

A completely different technique to estimate the attitude of a spacecraft is the Differential GPS. The DGPS technique is mainly used for relative position estimation, as it has been described in the previous section, but it can also be used for attitude estimation. For this particular application, two or more different GPS (or generic GNSS) receivers are installed on the spacecraft in known and fixed positions. Hence, differently from the classical use of DGPS where the computed position of each receiver is known and the problem is to find the relative position between the receivers, the relative position of the two receivers is known and by the knowledge of the absolute position of the receivers is possible to compute the attitude of the spacecraft. The known position of the receivers is commonly named as *baseline*. The computation of the position of the receivers occurs with raw data, which include phase information of each GNSS carries signals. Computing the phase of a GNSS signal is not a problem, while the problem is in computing the integer number of cycles of the sinusoidal wave of the transmitted GNSS signal. This problem is called *integer ambiguity resolution* [12], [13]. By the knowledge of the integer number of cycle of the signal received by the GNSS satellite it is theoretically possible to compute the range vector with accuracy of about 20 cm, which is the wavelength of the L1 band of a GPS signal. This value can be used to correct the computed range vector of each satellites improving the position estimation accuracy, and hence it is possible to get a good estimation of the attitude. However, due to the low accuracy achievable with this technique and other issues related to the use of GPS or GNSS constellation, such as availability of the navigation signal, the use of this technique for attitude determination is limited in space application and current performance allow the application of this technique only for contingency maneuvers. In addition, this method can be used only in Low Earth Orbit applications, since the GNSS signal is not available in higher orbits.

In addition to active radar sensors, such as the Kurs system described previously, a different sensor used is the Earth sensor, or horizon sensor. Again, it is an optical sensor which is able to detect the infra-red emission the Earth, distinguishing the heat emitted by the Earh from the cold space, obtaining an image of the Earth horizon in the field of view of the sensor (so the name *horizon sensor*). Unfortunately, this sensor suffers of low accuracy, mainly due to its working principle, which allows a good detection of roll and pitch angles with respect to the local vertical frame but it has very poor performance, if not completely absent, in detection of the yaw angle. In addition, the maximum altitude of the orbit of the spacecraft using this sensor is strongly limited by the field of view of the sensor itself, because in higher orbit the sensor is not able to detect the Earth horizon if it is forced to maintain the alignment with the local horizontal frame, and detection is possible only pointing the sensor with an angle through the Earth. Due to its poor performance, the use of this sensor is very limited in current space application, and mainly used as contingency sensor.

### 1.5.3 Control

Analysis of control system will focus on actuation systems used for position and attitude control and related control techniques, while investigation of control algorithms will be left to the following sections. Generally, with the term *control* it is intended both *attitude* and *position* control.

#### Attitude Control

Controlling the attitude is mandatory in space application since there are a number of factors which concur in the definition of the attitude pointing requirements, as mentioned in the previous section. Indeed, controlling the attitude of the spacecraft is fundamental to control the temperature, to maintain solar arrays pointing the Sun, to maintain or reorient the payload along with a desired orientation, to align the docking mechanism with the docking port, to align the thrust vector of the apogee engine through a specific direction to execute the desired orbital maneuver, and many other purposes. To fulfill such functions, in the spacecraft configuration there are included a set of actuators which are able to modify and control the attitude of the spacecraft. Actuators for attitude control are usually selected among reaction control thrusters, reaction wheels, gyroscopes, magnetotorquers. Attitude control, or stabilization, can be achieved also with passive tecniques, such as gravity gradient stabilization and spin stabilization.

Attitude control using reaction control thrusters allows a very poor control accuracy, since reaction control thrusters used for attitude control are usually providing low thrust, compared to thruster used for orbital maneuvers, and they have a discrete working principle, i.e. the thruster can be activated or not generating the maximum thrust which is designed for, without the possibility to modulate the magnitude of thrust. The use of modulation techniques, e.g. *Pulse-With Modulation (PWM)*, can partially overcome this problem, but the presence of thresholds and dead-bands due to the minimum impulse that can be provided by thrusters introduce an intrinsic accuracy limitation. For this reason, attitude control using thrusters is executed mainly during the de-saturation of the reaction wheels or other momentum exchange devices, during contingency maneuvers or when the required control torque is too high and it cannot be provided by other actuators, concurrently with them. In order to generate control torque, reaction control thrusters are used coupling activation of two thrusters firing in opposite side and direction in the spacecraft configuration.

Fine attitude control can be achieved using momentum exchange devices, such as *reaction wheels*. Such devices are basically rigid disks which are rotating and controlled in velocity. Since the disk of a reaction wheel has its own inertia, depending by the mass of the disk, it is required a torque to accelerate it to a different speed. The torque is provided by the controlled electric motor and, during the disk acceleration and by the third Newton's law, the reaction torque, with the same magnitude but opposite sign, it is transmitted to the mechanical support of the motor, and eventually to the spacecraft structure, since the motor is mechanically linked with it, changing the attitude of the spacecraft. Such systems are subjected to two main limitations of maximum torque provided by the motor and the maximum angular speed of the wheel.
The first limit, related to the maximum torque, is linked to the maximum torque the electric motor is able to provide, and hence this limit is due to the maximum current which can flow inside the coils of the motor. Increasing the maximum torque will increase the size of the motor, and so the mass of the system, and this issue it should be taken into account in the design of the attitude control system.
The limit related to the maximum speed is instead due to the maximum velocity that can be achieved by bearings which support the disk and the engine. Exceeding the maximum velocity can seriously damage the device, which can results in a mission failure if there are no redundancy due to the impossibility to control the attitude along the axis related to the damaged wheel. When a disk reaches the maximum velocity, the system is not able to provide torque, since it cannot accelerate the disk anymore. In such condition, to make the wheel usable again, it is required to slow down the wheel. Slowing down the wheel means applying a torque in the opposite direction, and hence the attitude of the spacecraft will change if no other tecquniques are used to counteract the decelerating wheel. Commonly, reaction control thrusters and magnetotorquers are used to generate a torque which counteract the deceleration of the wheel, maintaining the attitude of the spacecraft within a limited drift. Wheels

de-saturation supported by magnetotorquers, due to the simpler design and higher operative life than reaction control thrusters, are used in small spacecraft such as CubeSats, but there application in bigger spacecraft as well, such as the Hubble Space Telescope [14].
Reaction wheels systems are usually equipped with self-protection precautions, mainly embedded in the control hardware and software which drives the system, limiting the maximum current supplied to the motor and limiting the velocity of the disk to a safer value lower than the theoretical one.
A further effect which affects reaction wheels is the *friction*, in particular the static friction, since the rolling friction is much lower in magnitude. Static friction is mainly responsible of delay in the response of the wheel system, since it is required a greater torque to start moving an object with respect to change the momentum of a moving one. Static friction can became an important problem if the design of the reaction wheel is not done properly or the thermal control of spacecraft fails. Indeed, mechanical deformation due to temperature can be high enough to cause choking of the bearings, blocking the rotation of the disk. An additional effect of friction is due to the drag acting on the disk rotating at high speed, but in non-airtight system this effect is not present. A common way to minimize the effect of (static) friction is to set a minimum rotation speed as "*rest*" condition: when the wheel is de-saturated is is slowing down to this velocity instead to zero velocity, making the system more reactive and free from related oscillation caused by static starting of the wheel.

Alternative actuators for attitude control are *Control Moment Gyroscopes (CMG)*. Even if they are rotating devices able to store angular momentum, as for reaction wheels, their working principle is completely different. The disk of a CMG is continuously rotating at fixed speed, hence storing angular momentum. To generate torque it is used the gyroscopic effect: a rotating element spinning about an axis, if it starts rotating about an axis perpendicular to the spinning one, generates a torque perpendicular to both the spinning and rotation axis with magnitude proportional to the stored angular momentum and the rotation velocity. Control moment gyros are then constituted of a disk mounted on one or two gimbals which allow the controlled rotation of the spinning disk about one or two axis (or degrees of freedom).
CMGs are subjected to saturation as for reaction wheels, but due to different principles: if a set of two or more CMGs are required to balance the angular momentum of a spacecraft, it can occurs the situation of all the CMGs angular momentum are moving until the are pointing in the same direction, and the system is not able to

provide any additional torque with respect to that axis, and the system is saturated (gimbal lock). To recover the control authority it is required to de-saturate CMGs with same techniques as for reaction wheels, mainly using reaction control thrusters since the torque generated by a control moment gyros is much greater than the torque of a reaction wheels system with similar power.
Application of control moment gyros can be found in past Skylab space station ([15], [16] and [17]) and in the modern International Space Station ([18], [19] and [20]).

*Magnetotorquers* are used for attitude control using electromagnetic principles. Magnetotorquers are basically electric coils in which flows an electric current generating an electromagnetic field. The electromagnetic field generated by the coils interacts with the electromagnetic field of the Earth, generating a reaction torque used to control the attitude of the spacecraft. Since they are not built with moving parts they are more reliable than momentum exchange devices, and they do not suffer of de-saturation problems such as CMGs or reaction wheels. On the other hand, they are able to generate only torque perpendicular to the magnetic field vector, compromising the full axis attitude control. In addition, their effectiveness depend by the intensity of the magnetic field of the Earth, hence they are unusable in non Earth-orbital application, and the control effectiveness is greater in LEO with respect to orbits at higher altitude. Effects related to the dependency by the Earth's magnetic field are reflecting also in the maximum torque that can be generated by this device: due to the weakness of the magnetic field in Earth orbit, high torque can be generated only supplying very high current in the magnetic coils, which can damage the whole power subsystem of the spacecraft. For these reasons, magnetotorquers are mainly used in small spacecraft, even if there are application also in quite large satellites, as for the Hubble telescope [14].

As mentioned at the beginning of this section, attitude control can be achieved also with passive techniques. In this case it is not correct to talk about *attitude control*, but rather it is better to call it *attitude stabilization*. Passive stabilization techniques are based on the dynamic response of a rigid body, in some case rotating, in other case spinning about an axis.
Passive stabilization can be realized by the *gravity gradient* stabilization technique. This technique use inertia properties of the spacecraft in the gravitational field of the Earth. Due to the gravitational law, in an ideal system consisting of a mass-less bar with two equal masses fixed at the extremities of the bar, the mass located closer to the Earth is subjected to an attractive force greater than the attractive force acting on

the mass farther from the Earth. This differential force acting on the system generates a torque which tends to align the system along the local vertical. Extending this principle to objects designed with more complex inertia properties, this results that the object subjected to the gravitational field of Earth will tend to align its axis with lower moment of inertia to the local vertical axis. Some precautions can be adopted to minimize oscillation of spacecraft using this stabilization technique, such as by equipping the vehicle with dampers. This technique is used when less restrictive requirements are adopted, since no control or stabilization can be realized about the local vertical axis using gravity gradient stabilization.

*Spin stabilization* consists in making the spacecraft spinning about an axis and, by conservation of angular momentum, the attitude is stabilized to the initial orientation by the gyroscopic effect. This technique has been used for both upper stages of launchers, such as Delta II, or by interplanetary probes as Pioneer 4. Spin stabilization requires to be started while in orbit, for example using thrusters to starts the spacecraft spinning. Due to imperfection in inertia estimation and thruster errors, the resulting angular momentum is not perfectly related to a pure spin, so there can be present residual nutation movement that has to be damped. In addition, spin stability has to be satisfied: to reach spin stability it is used to spin the spacecraft about the axis with the lower inertia.

Eventually, passive stabilization of a spacecraft can be realized also with magnetic actuators. The working principle is the same as for magnetotorquers, with the advantage that for simple stabilization can be used permanent magnets instead of active magnetic actuators.

### Position Control

Position control is usually required for station keeping or rendezvous and docking purposes, as well as deep space exploration. Limiting the analysis to Earth orbit applications, only the first two applications will be discussed.

The *station keeping* maneuver consists in correcting orbital parameters of the spacecraft, restoring them to the designed values. Main cause of orbital drift in LEO orbits is the effect due to the residual atmospheric drag, while in higher orbits the effects due to the geopotential field of the Earth (the so called *J2 effect*) is the leading one. Other effects, such as third body perturbation and solar radiation are

present, but a non-negligible value can be found only in higher orbits. Depending by a number of design parameters, such as station keeping accuracy, operational life expectation of the spacecraft, spacecraft mass, and more, actuators used to ensure the station keeping affect the design of the propulsion system, since they are *de facto* part of this subsystem.
In many application, such as telecommunication satellites, there can be used *electric thrusters*. This typology of thruster, instead of accelerating a gas flow through a nozzle to generate thrust, they accelerates charged particles of the propellant gas by interaction with an electromagnetic field generated by the thrusters itself. By the reaction principle, even if the mass of the particles is very small, their exhaust velocity is extremely high, in the order of thousand of meters per seconds, which produces a non negligible thrust. However, the magnitude of the thrust depends by the technology of the electric thruster and by the power supplied, and it is usually very limited compared to classical thrusters (few $\mu N$ for ion and plasma thrusters to some $mN$ and rarely up to 0.5-1 N for electrostatic thrusters), but advantages are related to the high specific impulse that can be reached (up to 3000 s for plasma thrusters) which ensure a low propellant consumption, consequently reducing the spacecraft mass and extending the operational life of the spacecraft. Electric thrusters was equipping the GOCE spacecraft for its orbital control [21], [22]. Station keeping of bigger spacecraft can also be realized with classical chemical thrusters, such as for the International Space Station which executes re-boost maneuvers using the propulsion system of visiting spacecraft as Progress or ATV.

*Rendezvous and docking* purposes requires execution of many orbital maneuver in a limited time period. Even if the use of electric thrusters is possible, as for the LEO to GEO orbital maneuver of the Space-Tug studied in the STRONG project [23], the common practice is the use of the reaction control thrusters to execute all the orbital maneuvers. The reaction control system can also be used for attitude control purposes, as discussed in the previous section. The basic principle of a mono-propellant cold gas thruster is very simple: the gas is stored in a pressurize tank and it is connected to the nozzle using a pressure regulation valve and a flow control valve; the pressure regulation valve stabilize the output pressure of the gas to a specific value, and the flow control valve can be opened or closed allowing the gas flowing; the gas flows in the throat of the nozzle, expanding and accelerating, generating thrust. More complex and more effective thrusters include a combustion chamber before the throat of the nozzle, in which two propellants are ignited (fuel and oxidizer) increasing the total pressure inside the combustion chamber and generating higher thrust, increasing the specific impulse as well. Reaction control systems found application in all the spacecraft visiting the International Space Station, as Soyuz/Progress, Space Shuttle, ATV, HTV, Cygnus, Dragon, and the past Apollo space vehicles.

## 1.6 SAPERE STRONG Mission Scenario

The present PhD dissertation is contextualized within the SAPERE STRONG project, founded by Italian Ministry of University and Research (MIUR) with the goal to improve Italian access to Space and Space Exploration. For this purpose, extension of the launch capability of the Vega launcher is included in the project, realized with a *Space-Tug* which is used to deploy in the nominal orbit a payload spacecraft. The Space-Tug is then a spacecraft which is able to execute an automated rendezvous and docking maneuver with a second spacecraft deployed in Low Earth Orbit by Vega, executes an orbital transfer to bring the target spacecraft in its operative orbit, deploys the target spacecraft in the desired orbit and returns to LEO to execute a second mission. According to market analysis studies, identified target orbits are Geostationary Earth Orbits (GEO). Hence, the mission scenario related to a nominal mission profile of the Space-Tug has been defined, as depicted in Fig. 1.14: the Space-Tug shall be able to execute multiple missions consisting in LEO-GEO-LEO transfers, in order to deploy payload spacecraft in their operative orbit and return to LEO to start a new mission. Since a typical LEO-GEO orbital transfer is very energy consuming, classical chemical apogee engines do not allow execution of multiple missions. For this reason, a further enabling technology is the development of electric engines, which allows execution of multiple mission thanks to the greater specific impulse and consequently lower consumption. In addition, it has been scheduled a refueling operation of the Space-Tug using an orbital tank to guarantee service continuity.

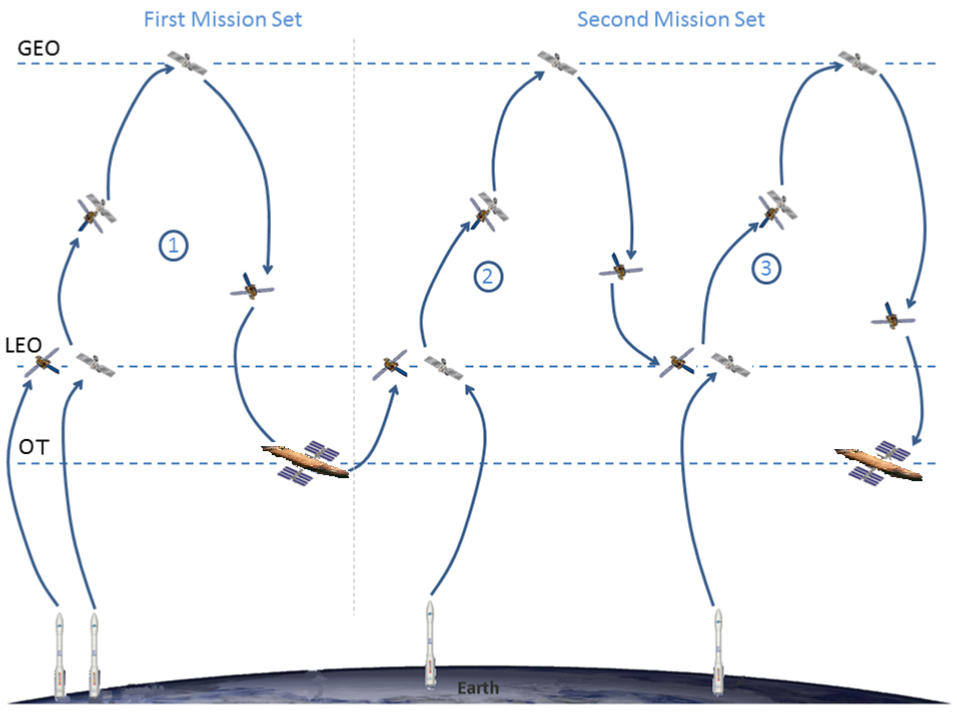


Fig. 1.14 Space-Tug reference mission scenario

# Chapter 2

# GNC Architecture

In this section it will be presented the GNC architecture developed in the context of the project SAPERE STRONG, as well as alternative GNC configuration proposed for academic research studies. Details of Guidance, Navigation and Control algorithms will be provided in related sections, and details about functional architecture and flow charts will be described as well.

## 2.1 Guidance, Navigation and Control Architecture

The main functional architecture of the GNC software is presented. Details of each functional component of the complete GNC will be provided in the following sections, including description of the implemented GNC algorithms. The functional architecture of the GNC software is depicted in Fig. 2.1. The functional architecture of the GNC software includes four main functions: three functions are strictly linked to the algorithms which perform the basic functions of Guidance, Navigation and Control, while a different function, the Flight Manager, is in charge to provide the activation command for internal GNC's sub-functions. Hence, depending by the output of the guidance, navigation and control functions, the flight manager switches the internal mode of each function to the subsequent mode, in order to drive the whole GNC software to execute the complete RVD maneuver.

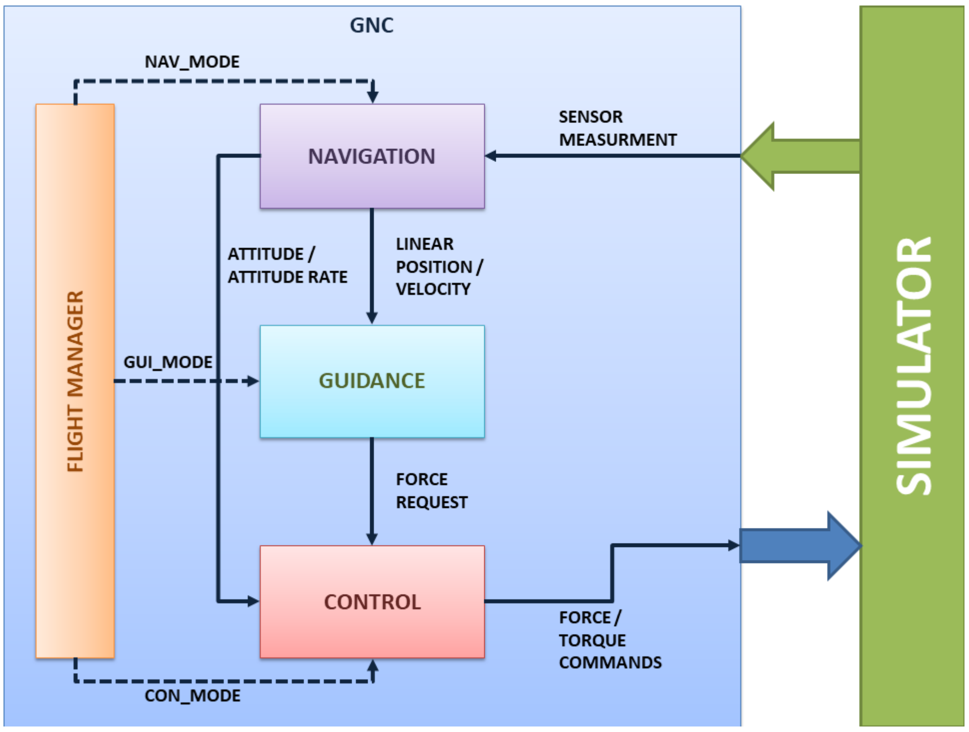


Fig. 2.1 GNC functional architecture.

## 2.2 Guidance

As extensively described in previous sections, the guidance problem is related to the execution of orbital maneuvers of a spacecraft for multiple reasons, such as station keeping purposes, mainly required to a spacecraft belonging to a constellation or GEO satellites, which must be kept inside a specified orbital slot above ground, or to execute rendezvous and docking missions. Guidance algorithms investigated in the present dissertation are focusing about this second purpose, i.e. guidance algorithms suitable for execution of complex orbital maneuvers for a RVD mission.

### 2.2.1 Algorithms Review

For what concern guidance algorithms, in literature can be found a number of applications in rendezvous maneuvers usually limited to ideal cases, hence not considering limitations of the propulsive system of the spacecraft, measurement errors of sensors, external disturbances, and often without considering coupling effects due to the non-ideal attitude control. In addition, most algorithms found in literature are analyzing only the terminal guidance for the final approach, limited to few hundreds of meters from the target. Eventually, the definition of guidance algorithms is often formulated with respect to an inertial frame, and applying such implementation referred to a *relative* frame is not free from errors, as it will be discussed in the following of this dissertation.

**Proportional Navigation**

The first guidance algorithm taking into exam is the *Proportional Navigation (PN) Guidance* algorithm. This algorithm derives from guidance laws for ballistic missile interceptors [24]. Since a missile intercept problem can be related to a rendezvous application, there are example of application of the PN guidance for asteroids intercepts in [25] and [26]. The basic principle of the PN algorithm is simple: the guidance law is commanding a controlled collision with the target keeping a constant angle of the Line-of-Sight (LOS) vector. The formulation is the following

$$a = nV_c\dot{\lambda} \tag{2.1}$$

where $a$ is the acceleration guidance command, perpendicular to the LOS vector from the chaser to the target, $n$ is the PN gain, which is a tunable parameter usually in the range of 3-5, $V_c$ is the closing, or approaching, velocity and $\dot{\lambda}$ is the time-derivative of the LOS angle. A graphical representation of the PN guidance is depicted in Fig. 2.2. In Fig. 2.2 the LOS vector is named $\vec{r}$ and the LOS angle is named $\lambda$. Improvement of the PN guidance law have been introduced by Palumbo in [27] and other similar implementation can be found in [25] and [26] by Hawkins. Both Palumbo and Hawkins introduce the *Augmented Proportional Navigation Guidance (APNG)* in which it has been introduced the time-to-go parameter, $t_{go}$, which is defined by

$$t_{go} = t_{final} - t \tag{2.2}$$

where $t_{final}$ is the desired final time and $t$ is the actual elapsed time. The final formulation of the APNG algorithm is different depending by the assumption considered during the development of the guidance law. Palumbo applied the Linear Quadratic (LQ) optimal control to the PN algorithm of Eq. 2.1 and finding the optimal PN gain $n = 3$. The $t_{go}$ parameter has been used to compute the value of $\dot{\lambda}$. The augmented PN guidance found by Hawkins introduces the *Zero Effort Miss (ZEM)* error perpen-

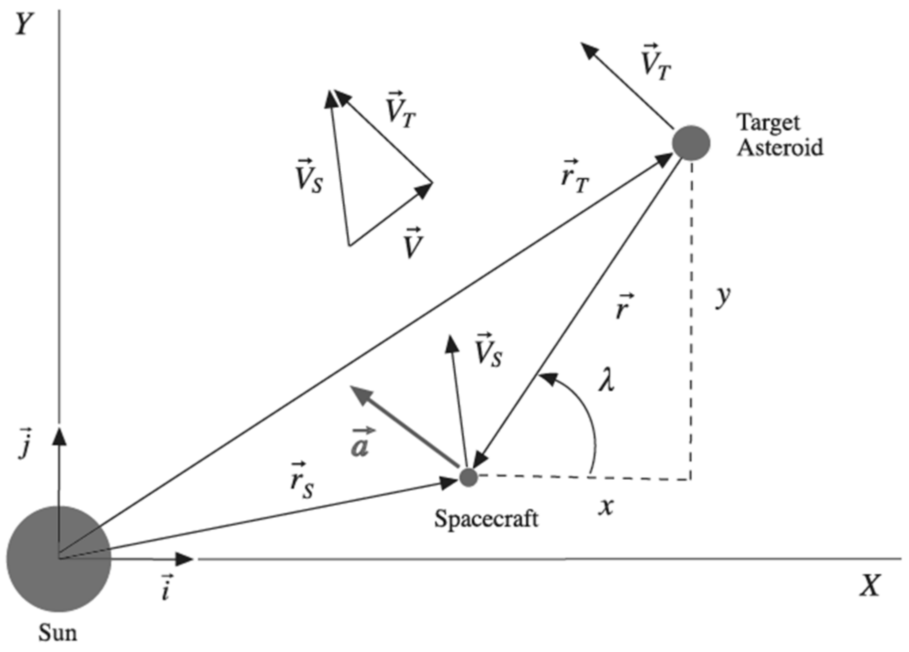


Fig. 2.2 Asteroid intercept geometry (picture by Hawkins, 2012).

dicular to the LOS vector, computed considering the distance of the chaser from the target at the end of the flight if no control action is applied

$$ZEM = r + \dot{r}\, t_{go} \tag{2.3}$$

Introducing Eq. 2.3 in Eq. 2.1 the resulting APNG is then

$$a = n\frac{ZEM}{t_{go}^2} \tag{2.4}$$

It has to be noticed that both the examples presented by Palumbo and Hawkins consider only the planar intercept problem, and the prediction of the position of the chaser vehicle if no control action is applied, as in Eq. 2.3, consider a uniform motion propagation, which results very different from the orbital free dynamics motion.

**Zero-Effort-Miss/Zero-Effort-Velocity**

A further work by Hawkins found in [28] and [29] introduces a new guidance law based on the *Zero Effort Miss/Zero Effort Velocity (ZEM/ZEV)* algorithm. The general formulation of this guidance law is the following

$$a = \frac{6}{t_{go}^2} ZEM - \frac{2}{t_{go}} ZEV \tag{2.5}$$

The time-to-go parameter, $t_{go}$, is the same as defined in Eq. 2.2, $a$ is the command acceleration and $ZEM$ and $ZEV$ errors are defined by

$$ZEM = r_f - \tilde{r}_f \tag{2.6}$$

$$ZEV = v_f - \tilde{v}_f \tag{2.7}$$

where $r_f$ and $v_f$ are respectively the *desired* final position and the *desired* final velocity at $t = t_{final}$, while $\tilde{r}_f$ and $\tilde{v}_f$ are respectively the *predicted* final position and velocity at $t = t_{final}$ if no acceleration are applied to the chaser. According to the Hawkins formulation for the application to an asteroid intercept mission, the predicted final position and velocity can be computed by

$$\tilde{r}_f = r + v t_{go} + \int_t^{t_{final}} (t_{final} - \tau) g(\tau) \mathrm{d}\tau \tag{2.8}$$

$$\tilde{v}_f = v + \int_t^{t_{final}} g(\tau) \mathrm{d}\tau \tag{2.9}$$

where $g(\tau)$ is the contribution of the gravitational acceleration. In some application, such as landing on a small asteroid, due to the weak gravitational field of the target object, the contribution of the gravitational acceleration can be neglected. The computation of $t_{go}$ is a critical issue of the ZEM/ZEV algorithm, as also discussed by [30]. According to the definition of Eq. 2.2, a proper setting of the parameter $t_{final}$ is crucial for the success of the mission. Indeed, as also discussed by [31] and [32], as the elapsed time $t$ is approaching the value of $t_{final}$, the acceleration command may became *infinite* due to divisions by zero. Hence, a particular attention shall be applied in selecting the value of $t_{final}$.

### Lambert Guidance

A different typology of guidance algorithm is found in the Lambert guidance. This algorithm is based on solving the Lambert's problem, i.e. it has to be found the transfer orbit which connect the initial position vector $r_0$ to the desired final position vector $r_f$ within a specified transfer time $T$. A review of solution methods of the Lambert's problem can be found in [33]. In literature can also be found a number of different iterative algorithm for the solution of the Lambert's problem, as in [34], [35] and [36], also including the introduction of genetic optimization algorithms [37], but the main contribution is by Battin [38], [39] and an alternative approach has been proposed by Nelson and Zarchan in [40]. Even though the work of Battin produce a very good solution of the Lambert's problem, Avanzini proposed an alternative algorithm solving the problem with a simple Newton-Raphson iterative method [41], which has comparable performance in convergence velocity and accuracy with respect to the Battin's solution, but it is obtained with a simpler algorithm. The solution of the Lambert's problem is basically a geometrical problem, since the solution consists in finding the elliptical transfer orbit. Criticalities can be found in computing the the transfer time of the maneuver. Due to the complexity of the solution of the Lambert's problem and due to the fact that the formulation is in an inertial frame, no further details will be provided in this section abut such topic, leaving detailed studies to the reader. In the following sections, a different formulation for the Lambert guidance in a relative frame will be discussed, helped by [42].

### Glideslope Guidance

A further guidance algorithm is the so called *glideslope* guidance algorithm. Real implementation of this algorithm can be found in the final approach of the rendezvous and docking maneuvers of the Space Shuttle, as reported by [1], [43] and [44], even if the Shuttle guidance is executed by astronauts who actually commanded the computed guidance commands. The basic principle of the glideslope algorithm is the following: by the knowledge of the initial position $r_0$ and the desired final position $r_T$ of the chaser spacecraft with respect to the local frame centered in the target spacecraft, the guidance command is computed in such a way that while the relative distance is reducing, the relative approaching velocity is reducing too, and

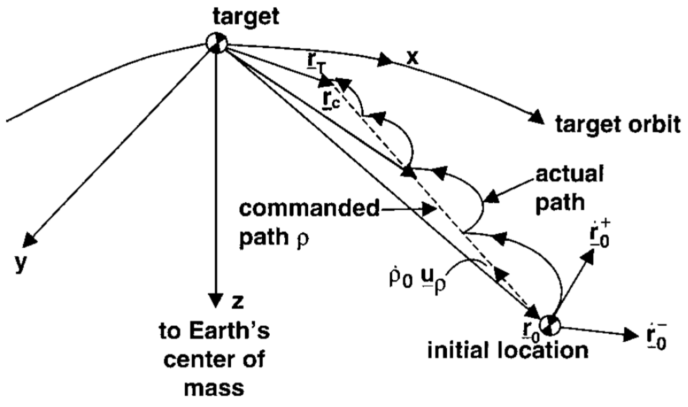


Fig. 2.3 Glideslope path (by Hablani, 2002).

consequently the guidance acceleration command is reducing as well. Assuming $\rho$ as the magnitude of the relative position vector $r$ and $\dot{\rho}$ the magnitude of the relative velocity vector, it is defined the following relationship between $\rho$ and $\dot{\rho}$

$$\dot{\rho} = a\rho + \dot{\rho_T} \tag{2.10}$$

where the subscript "$T$" is referred to the *terminal* condition. Assuming the subscript "0" for the initial condition, the slope $a$ can be evaluated

$$a = \frac{\dot{\rho_0} - \dot{\rho_T}}{\rho_0}$$

and the solution of Eq. 2.10 can be found

$$\rho(t) = \rho_0 e^{at} + \frac{\dot{\rho_T}}{a}(e^{at} - 1) \tag{2.11}$$

The total transfer time is evaluated by

$$T = \frac{1}{a}\ln\left(\frac{\dot{\rho_T}}{\dot{\rho_0}}\right)$$

A graphical representation of the trajectory driven by the glideslope algorithm is depicted in Fig. 2.3. As it can be noticed in the picture, since the algorithm proposed by Hablami in [43] considers a multi-pulse maneuver, the actual path followed by the chaser spacecraft results in different steps related at the instant in which the commanded $\Delta V$ is applied. For a continuous thrust maneuver, the trajectory followed by the chaser spacecraft converges to the dashed commanded path of Fig. 2.3. Even though the work of Hablani is complete and considers different cases, such as $\mathrm{V_{BAR}}$ and $\mathrm{R_{BAR}}$ approach, in-plane and out-of-plane maneuver, fly-around maneuvers, as well as it has been included navigation errors, the algorithm presented in [43] presents a set of limitations, such as (i) the guidance commands are considered applied instantaneously and not affected by errors, (ii) attitude coupling effects are not taken into account and (iii) the proposed guidance algorithm is limited to few hundreds of meters of the final approach and it is not suitable for a complete orbital RVD maneuver due to its not fuel-optimal performance for far range approach. The work of Ariba in [45] applies linear programming techniques to find a fuel-optimal autonomous guidance based on glideslope algorithm, limiting the application to $\mathrm{V_{BAR}}$ and $\mathrm{R_{BAR}}$ terminal approach. Application to a far range RVD can be found in [46], where a Clohessy-Wiltshire targeting technique (which is the basis of the glideslope approach) is implemented.

### LQR Guidance

In the field of optimal control, it has been investigated a different implementation of the *Linear Quadratic Regulator (LQR)* for the guidance algorithm. The general formulation of a LQR controller starts from the state-space system representation

$$\dot{x} = Ax + Bu$$

where $x$ and $u$ are respectively the state and the control action, $A$ is the state matrix and $B$ is the control matrix. The goal is to find a control law in the form

$$u = -Kx \tag{2.12}$$

which minimize the cost function $J$ defined by

$$J = \int_0^\infty (x^T Qx + u^T Ru)\,\mathrm{d}t$$

where $K$ is the LQR gain and $Q$ and $R$ are weighting matrices related respectively to the state and the control action. The optimal solution for $K$ is in the form

$$K = R^{-1}B^T P \tag{2.13}$$

where $P$ is the solution of the Algebraic Riccati Equation (ARE)

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{2.14}$$

Usually, the computation of the matrix $P$ is not trivial and it is computationally expensive. For this reason, the solution of the Riccati equation is computed *off-line*, as in the simple implementation of LQR in [47]. In this work, no specific assumption are provided to support the choice of the matrices $Q$ and $R$. Improvements in LQR guidance control can be found in [48] in which the weighting matrices $Q$ and $R$ are depending by the actual relative position of the chaser with respect to the target, named $r_{goal}$, and hence $Q$ and $R$ matrices are computed *on-line*, since there are time-depending terms in those matrices. In addition to the LQR guidance of [48], in [49] it has been introduced *Artificial Potential Functions (APF)* to manage obstacle avoidance, a very useful tool especially for orbital formation flight. A different approach for the implementation of the LQR guidance will be provided by the author in the following sections.

### Model Predictive Control

Following the optimal control theory, in recent years there was a growing interest in *Model Predictive Control (MCP)* theory. As suggested in [50], MPC is a very useful control techniques since it is based on solving optimal control over a specified time horizon. On the other hand, since it is solving an optimization problem, it is a very computational demanding algorithm. In general, the discrete guidance control law consists in optimizing a cost function $J$, in a similar way of LQR, over a specified time interval to compute the control law $u = K(x)$ at each time step. Hence, the optimization problem has to be solved each time the guidance function is running. Since the guidance law is computed as optimal over the selected time interval, the system response has to be evaluated over all the time interval knowing the control law $u$, and so it is required an accurate prediction model, such as the state-space representation of linear systems. Advantages in using MPC is that constraints about state, control action, system response, and more, can be taken into account in the optimization problem. Other advantage is that MPC can be implemented also in non-linear problems without requiring linearization. In order to solve the high computational demand of MPC, some techniques can be adopted, such as using a linear prediction model and constrains, the problem can be reduced to a linear

programming or quadratic programming if a quadratic cost function is adopted, while in explicit MPC the guidance control law can be computed *off-line* using piecewise-affine functions.
In the context of the present dissertation, MPC is not further analyzed, manly due to the high computational effort required for a real-time implementation, which is limited by GNC requirements listed in the SAPERE STRONG project. In literature can be found a number o application of MPC for rendezvous and proximity operations [51], [52], [53], including case study of rotating target [54] and performing obstacle avoidance [55], and eventually MPC guidance law implemented in RVD maneuver to ENVISAT [56].

### 2.2.2 Development of Guidance Algorithms

Starting from the literature review of actual studies about the guidance problem, it has been identified different potential guidance algorithms as candidate for the implementation in the current Guidance of the Space-Tug spacecraft. Specifically, they have been deeply studied *Modified ZEM/ZEV*, *LQR Guidance*, *Relative Lambert Guidance* and a *Continuous Lambert-type Guidance* algorithms.

#### Modified ZEM/ZEV

The motivation which led the development of an alternative implementation of the ZEM/ZEV algorithm is mainly due to its definition with respect to an inertial reference frame and due to the behaviour of the relative equation of motion. Indeed, differently from the inertial equation of motion, an object subjected to a constant force, for example along $\mathrm{V_{BAR}}$, at the beginning starts moving in a straight line, but not so far it starts to drift the straight motion along $\mathrm{R_{BAR}}$. For example, this happens during the execution of an orbit raising maneuver applying a continuous thrust. This behaviour is strongly in contrast with the correction command provided by the classical ZEM/ZEV, in which an error in position and/or velocity is corrected by a compensation force. In other words, considering the application of ZEM/ZEM to the first maneuver, the desired final position will be the position of waypoint $S2$, while the desired final velocity will be zero. Since the nominal maneuver begins in $S1$ in a free drift condition, the predicted final velocity to insert in Eqs. 2.6-2.7 will be the same as the free drift velocity, while the position will be $\tilde{r}_f = [S1_x + \dot{x}_{fd} t_{go}, 0, S1_z]^T$.

Hence, the computed guidance command is expected to have a component along $V_{BAR}$ which tends to slow down the spacecraft and a component along $R_{BAR}$ which tends to push the spacecraft to an higher orbit. Even if the magnitude of both the $V_{BAR}$ and $R_{BAR}$ components of the guidance command are normalized with respect to $t_{go}$, their effects are non negligible due to the computed high value of ZEM and ZEV errors, hence the resulting commanded trajectory is completely different to the expected one.

Due to this anomalous beheaviour of the classical ZEM/ZEV algorithm, a different formulation of the algorithm is proposed. Instead of computing ZEM and ZEV errors with respect the final desired position and velocity to reach, the prediction horizon, $t_{go}$, has been reduced to few tens of seconds ahead: in this way, the short term behaviour of Hill's equations is more similar to an inertial formulation, since orbital drift effects are strongly reduced in the short time. According to this modification, the desired final position and velocity are not the final position and velocity related to the subsequent waypoint, but they are the desired final position and velocity related of a generic waypoint located $t_{go}$ seconds ahead with respect to the ideal trajectory the spacecraft is expected to follow. Hence, it is required to generate a reference trajectory profile in terms of both position and velocity. For the generation of the reference trajectory profile, it has been used the time-domain formulation of the Hill's equations, as described in [42], i.e.

$$\begin{aligned} x(t) = &\left(\frac{4\dot{x}_0}{\omega_0} - 6z_0\right)\sin(\omega_0 t) - \frac{2\dot{z}_0}{\omega_0}\cos(\omega_0 t) + (6\omega_0 z_0 - 3\dot{x}_0)t + \left(x_0 + \frac{2\dot{z}_0}{\omega_0}\right) + \\ &+ \frac{2}{\omega_0^2}\gamma_z(\omega_0 t - \sin(\omega_0 t)) + \gamma_x\left(\frac{4}{\omega_0^2}(1-\cos(\omega_0 t)) - \frac{3}{2}t^2\right) \end{aligned} \tag{2.15}$$

$$y(t) = y_0\cos(\omega_0 t) + \frac{\dot{y}_0}{\omega_0}\sin(\omega_0 t) + \frac{\gamma_y}{\omega_0^2}(1-\cos(\omega_0 t)) \tag{2.16}$$

$$\begin{aligned} z(t) = &\left(\frac{2\dot{x}_0}{\omega_0} - 3z_0\right)\cos(\omega_0 t) + \frac{\dot{z}_0}{\omega_0}\sin(\omega_0 t) + \left(4z_0 - \frac{2\dot{x}_0}{\omega_0}\right) + \\ &+ \frac{2}{\omega_0^2}\gamma_x(\sin(\omega_0 t) - \omega_0 t) + \frac{\gamma_z}{\omega_0^2}(1-\cos(\omega_0 t)) \end{aligned} \tag{2.17}$$

for the position reference profile and

$$\dot{x}(t) = \left(\frac{4\dot{x}_0}{\omega_0} - 6z_0\right)\omega_0\cos(\omega_0 t) + 2\dot{z}_0\sin(\omega_0 t) + (6\omega_0 z_0 - 3\dot{x}_0) +$$
$$+\frac{2}{\omega_0}\gamma_z(1-\cos(\omega_0 t)) + \gamma_x\left(\frac{4}{\omega_0}\sin(\omega_0 t - 3t)\right) \tag{2.18}$$
$$\dot{y}(t) = -y_0\omega_0\sin(\omega_0 t) + \dot{y}_0\cos(\omega_0 t) + \frac{\gamma_y}{\omega_0}\sin(\omega_0 t) \tag{2.19}$$
$$\dot{z}(t) = -\left(\frac{2\dot{x}_0}{\omega_0} - 3z_0\right)\omega_0\sin(\omega_0 t) + \dot{z}_0\cos(\omega_0 t) + \frac{2}{\omega_0}\gamma_x(\cos(\omega_0 t) - 1) +$$
$$+\frac{\gamma_z}{\omega_0}\sin(\omega_0 t) \tag{2.20}$$

for the velocity reference profile. In the two set of equations (Eq. 2.15 to Eq. 2.20), the terms $x_0$, $y_0$, $z_0$, $\dot{x}_0$, $\dot{y}_0$ and $\dot{z}_0$ are respectively initial position and velocity related to the V$_{\text{BAR}}$, H$_{\text{BAR}}$ and R$_{\text{BAR}}$ axis, while $\gamma_x$, $\gamma_y$ and $\gamma_z$ are the acceleration terms along the same axis sequence. In order to generate the correct reference maneuver profile, initial conditions are exactly the initial conditions of the waypoint in which the maneuver begins, i.e. $S1$ for the maneuver S1-S2, $S2$ for the maneuver S2-S3 and $S3$ for the maneuver S3-S4. Hence, the time $t$ is the time elapsed from the beginning of the maneuver, i.e.

$$t = (t_{elapse} - t_{0_{man}}) + T_{pred} \tag{2.21}$$

where $t_{elapse}$ is the elapsed simulation time, $t_{0_{man}}$ is the initial time of the maneuver, initialized at each waypoint, and $T_{pred}$ is the short term prediction horizon. The time $t$ is the current elapsed time from the beginning of the maneuver plus the prediction horizon for which it has to be computed the reference trajectory and velocity profiles. Then, it is now possible to compute the desired position and velocity to insert in the computation of ZEM and ZEV errors of Eqs. 2.6-2.7

$$r_f = [x(t), y(t), z(t)]^T \tag{2.22}$$
$$v_f = [\dot{x}(t), \dot{y}(t), \dot{y}(t)]^T \tag{2.23}$$

assuming that $t$ is the actual maneuver time computed by Eq. 2.21. Focusing on the acceleration terms, they are mandatory for the computation of the correct maneuver, since they are used to generate the trajectory and velocity profile generated assuming a constant acceleration, which is required to complete the maneuver. Hence, acceleration terms, $\gamma_x$, $\gamma_y$ and $\gamma_z$, are maneuver-dependent and they are computed, for example, according to the altitude of the orbit to reach, as for the maneuver S1-S2 or the $\mathrm{V_{BAR}}$ distance is required to travel, as for the maneuver S2-S3. Hence, acceleration terms are computed by Eq. 3.8 and Eq. 3.14.

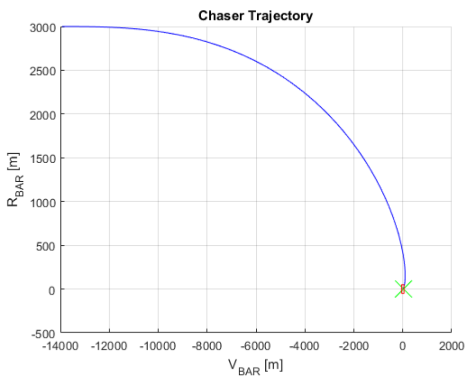
To compute the predicted final position and velocity, $\tilde{r}_f$ and $\tilde{v}_f$, as prescribed by the classical formulation of ZEM/ZEV, it is required the prediction of the *free dynamics* motion, i.e. the position and velocity prediction assuming that no control action is applied to the system. For this reason, the prediction of position and velocity is computed by Eqs. 2.15 to 2.20 neglecting all the acceleration terms $\gamma_x$, $\gamma_y$ and $\gamma_z$. In addition, differently from the computation of $r_f$ and $v_f$, the prediction time-span to insert in the equations is limited to the prediction horizon, hence computation of Eqs. 2.15 to 2.20 requires to set $t = T_{pred}$, and so the resulting predicted postion and velocity are

$$\tilde{r}_f = [x(T_{pred}), y(T_{pred}), z(T_{pred})]^T \tag{2.24}$$

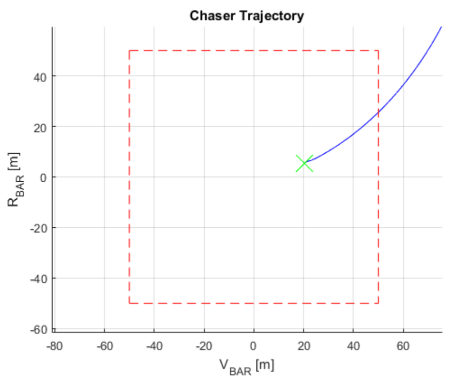
$$\tilde{v}_f = [\dot{x}(T_{pred}), \dot{y}(T_{pred}), \dot{y}(T_{pred})]^T \tag{2.25}$$

While the initial position and velocity used to compute the predicted position and velocity after $T_{pred}$ are the current position and velocity of the spacecraft. It is now possible to compute the ZEM and ZEV errors as in Eqs. 2.6-2.7 and to compute the guidance command as in Eq. 2.5, setting $t_{go} = T_{pred}$.

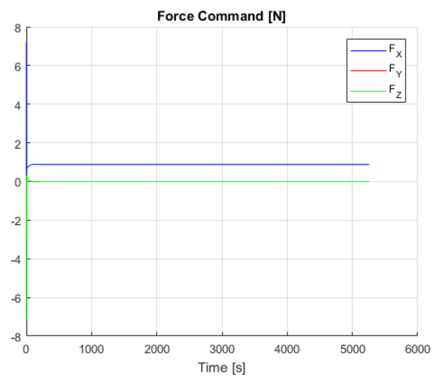
A preliminary test related to the application of the Modified ZEM/ZEV algorithm is depicted in Fig. 2.4. Performance of the Modified ZEM/ZEV has been evaluated by executing a reference maneuver, which is the *orbit raising* maneuver as it will be deeply described in Section 3.1. The goal of this reference maneuver is to reach the position of a waypoint located in the orgin of the local reference frame. The trajectory resulting by the implementation of the Modified ZEM/ZEV algorithm is depicted in Fig. 2.4a and the highlight of the terminal conditions is depicted in Fig. 2.4b. As it can be seen, the final waypoint is reached with high accuracy. The computed guidance command, depicted in Fig. 2.4c, has a quite regular profile and the computed force command is comparable to the ideal one, as it will be computed in Section 3.1. The propellant consumption is comparable with the ideal one as well. This preliminary result allows to candidate the Modified ZEM/ZEV algorithm to be implemented as guidance algorithm in the complete GNC software that will be described and developed in the following of the present dissertation.



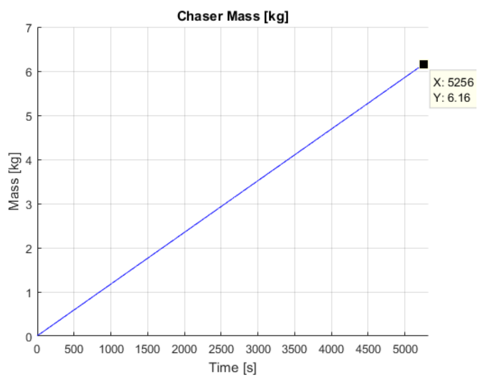
(a) $V_{BAR}$-$R_{BAR}$ trajectory.



(b) $V_{BAR}$-$R_{BAR}$ trajectory: zoom on *S2*.



(c) Guidance force command.



(d) Propellant mass.

Fig. 2.4 Preliminary results of the Modified ZEM/ZEV Guidance algorithm.

### LQR Guidance

The proposed LQR guidance algorithm follows the same principle which bring to the formulation of the Modified ZEM/ZEV algorithm. Differently to what is proposed by Bevilacqua in [49], in which the LQR controller is used to guide the spacecraft to the desired final position, and the gain matrix is computed with time-varying weighting matrices, the proposed approach consider the adoption of fixed weighting matrices, leaving the computation of the gain matrix $K$ off-line and during the initialization of the guidance algorithm. In this way, the proposed LQR guidance is used to drive the spacecraft along a reference trajectory instead of driving the spacecraft directly through the desired final position, i.e. the successive waypoint. The resulting LQR guidance is then controlling to zero the *error* between the actual and the reference trajectory and velocity profiles. Assuming the state-space formulation

$$\dot{\hat{x}} = A\hat{x} + Bu \tag{2.26}$$

where the state $\hat{x}$ is defined by

$$\hat{x} = \begin{Bmatrix} x_{des} \\ y_{des} \\ z_{des} \\ \dot{x}_{des} \\ \dot{y}_{des} \\ \dot{z}_{des} \end{Bmatrix} - \begin{Bmatrix} x_{act} \\ y_{act} \\ z_{act} \\ \dot{x}_{act} \\ \dot{y}_{act} \\ \dot{z}_{act} \end{Bmatrix} \tag{2.27}$$

and the subscript *"des"* is referred to the *desired* state while the subscript *"act"* is referred to the *actual* state. $A$ and $B$ matrix are defined by

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 2\omega_0 \\ 0 & \omega_0^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3\omega_0^2 & -2\omega_0 & 0 & 0 \end{bmatrix} \tag{2.28}$$

$$B = \frac{1}{m_c} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.29}$$

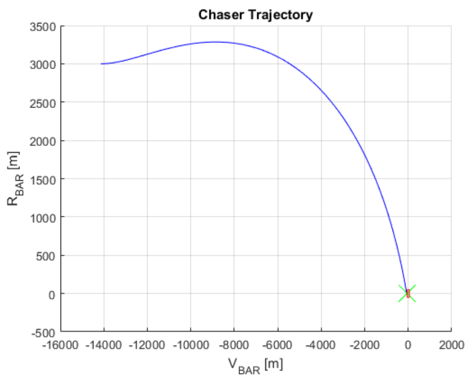
According to the definition of $B$, the control $u$ is defined by

$$u = \begin{Bmatrix} F_x \\ F_y \\ F_z \end{Bmatrix} \tag{2.30}$$

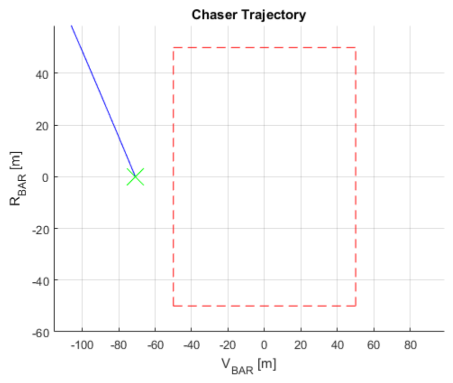
The solution of the Algebraic Riccati Equation is computed by Eq. 2.14, by choosing properly matrices $Q \in \mathbb{R}^{6,6}$ and $R \in \mathbb{R}^{3,3}$. Eventually, the LQR gain is computed by Eq. 2.13, and the control law is eventually obtained by

$$u = -K\hat{x} \tag{2.31}$$

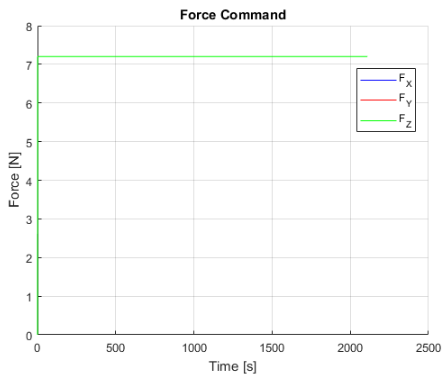
Performance of the LQR Guidance, obtained in preliminary tests are depicted in Fig. 2.5. Such test consists in executing the same maneuver as done for the Modified ZEM/ZEV algorithm. Compared to the Modified ZEM/ZEV, the trajectory obtained by implementing the LQR Guidance is very different and sub-optimal, as depicted in Fig. 2.5a, as well as the final waypoint is not reached with enough accuracy (see Fig. 2.5b). In addition, the LQR Guidance requires a much higher control effort with respect to the Modified ZEM/ ZEV, since the computed guidance command is basically a constant force along $\mathrm{V_{BAR}}$ and $_{\mathrm{BAR}}$ axis, as in Fig. 2.5c. This



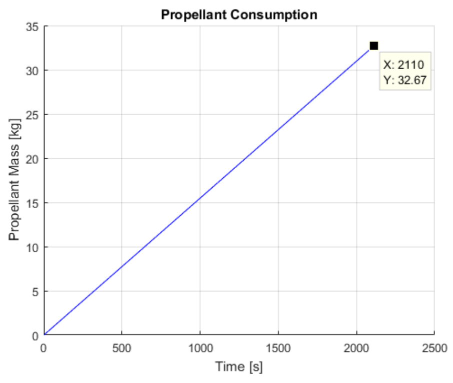
(a) $\mathrm{V_{BAR}}$-$\mathrm{R_{BAR}}$ trajectory.



(b) $\mathrm{V_{BAR}}$-$\mathrm{R_{BAR}}$ trajectory: zoom on *S2*.



(c) Guidance force command.



(d) Propellant mass.

Fig. 2.5 Preliminary results of the LQR Guidance algorithm.

effort can be reduced by modifying properly $Q$ and $R$ and introducing thresholds and dead-bands in the control activation, but this implicates a reduced capability in following the desired state, as well as an higher propellant consumption with respect to the Modified ZEM/ZEM, as in Fig. 2.5d. A possible solution to improve dramatically overall performance of LQR Guidance is to use an *on-line* computation of $Q$ and $R$, but due to limited computational capabilities of the on-board computer of the STRONG Space-Tug, this solution is far to be implemented.

### Relative Lambert Guidance

The Relative Lambert Guidance [42] is a very simple guidance law which allow the execution of a wide set of maneuvers between two arbitrary waypoints and within a prescribed time period $t_{des}$. According to Clohessy-Wiltshire equations of Eqs.

2.15-2.16-2.17 and neglecting acceleration terms $\gamma_x$, $\gamma_y$ and $\gamma_z$, the required initial velocity to reach a point located in $P = [x(t_{des}), y(t_{des}), z(t_{des})]^T$ starting from a point $O = [x_0, y_0, z_0]^T$ can be computed by

$$\dot{x}_0 = \frac{-\omega_0 \sin(\omega_0 t)(x - x_0) + \omega_0[6\omega_0 t \sin(\omega_0 t) - 14(1 - \cos(\omega_0 t))]z_0}{3\omega_0 t \sin(\omega_0 t) - 8(1 - \cos(\omega_0 t))} + \\ + \frac{2\omega_0(1 - \cos(\omega_0 t))z}{3\omega_0 t \sin(\omega_0 t) - 8(1 - \cos(\omega_0 t))} \tag{2.32}$$

$$\dot{y}_0 = \frac{\omega_0[y - y_0 \cos(\omega_0 t)]}{\sin(\omega_0 t)} \tag{2.33}$$

$$\dot{z}_0 = \frac{\omega_0[2(x_0 - x)(1 - \cos \omega_0 t) + (4\sin(\omega_0 t) - 3\omega_0 t \cos(\omega_0 t))z_0]}{3\omega_0 t \sin(\omega_0 t) - 8(1 - \cos(\omega_0 t))} + \\ + \frac{\omega_0[3\omega_0 t - 4\sin(\omega_0 t)z]}{3\omega_0 t \sin(\omega_0 t) - 8(1 - \cos(\omega_0 t))} \tag{2.34}$$

assuming $[x, y, z]^T = [x(t_{des}), y(t_{des}), z(t_{des})]^T$ and setting the desired transfer time $t = t_{des}$. By knowing the desired initial velocity required to complete the maneuver from $O$ to $P$ yet computed and by knowing the *actual* velocity of the spacecraft $v_{0_{act}} = [\dot{x}_{0_{act}}, \dot{y}_{0_{act}}, \dot{z}_{0_{act}}]^T$, it is possible to evaluate the $\Delta V$ required to execute the maneuver

$$\Delta V_{x_0} = \dot{x}_0 - \dot{x}_{0_{act}} \tag{2.35}$$

$$\Delta V_{y_0} = \dot{y}_0 - \dot{x}_{y_{act}} \tag{2.36}$$

$$\Delta V_{z_0} = \dot{z}_0 - \dot{z}_{0_{act}} \tag{2.37}$$

The final velocity after $t_{des}$, $v_{f_{des}} = [\dot{x}(t_{des}), \dot{y}(t_{des}), \dot{z}(t_{des})]^T$, can be evaluated by Eqs. 2.18-2.19-2.20 by setting $t = t_{des}$, $p_0 = [x_0, y_0, z_0]^T$ as initial position of waypoint $O$ and initial velocity $v_0 = [\dot{x}_0, \dot{y}_0, \dot{z}_0]^T$ as the velocity computed by Eqs. 2.32-2.33-2.34. Eventually, the second velocity impulse to apply after $t_{des}$ is computed by

$$\Delta V_{x_{final}} = \frac{3}{2}\omega_0 z(t_{des}) - \dot{x}(t_{des}) \tag{2.38}$$

$$\Delta V_{y_{final}} = -\dot{y}(t_{des}) \tag{2.39}$$

$$\Delta V_{z_{final}} = -\dot{z}(t_{des}) \tag{2.40}$$

which is referred to the free drift velocity which shall be reached at the final waypoint $P$.

Advantages of using Relative Lambert Guidance is that it is possible to execute any orbital transfer between two arbitrary waypoints. Since a free parameter $t_{des}$ is set by the designer, or computed with some technique, the resulting maneuver may be sub-optimal, since the computed $t_{des}$ may be far from the optimal one. For example, executing an orbit raising maneuver, the Relative Lambert Guidance converges to the optimal Hohmann transfer only if it is executed between two waypoints located as prescribed Fig. 3.2b and the maneuver time is set as $t_{des} = T/2$. In addition, Relative Lambert Guidance, if implemented in a closed-loop guidance scheme, is able to provide corrections due to thrust errors induced by the propulsive system and to cope external disturbances.

### Continuous Lambert-type Guidance

The Continuous Lambert-type Guidance is based on the Relative Lambert Guidance algorithm solved for a continuous thrust maneuver. This algorithm is used to find the general solution of the state-space formulation of Eq. 2.26 of Hill's equations Eq. 3.34, assuming the state $\hat{x} = [x, y, z, \dot{x}, \dot{y}, \dot{z}]^T$. As stated in Appendix A of [42], to find the homogeneous solution of Eq. 3.34 it is convenient to solve them by using the *Laplace transformation* method. Hence, the solution in the $s$ domain is easy to found

$$x(s) = \Phi(s)[x(0_+) + Bu(s)] \tag{2.41}$$

where $B$ is the control matrix as in Eq. 2.26, $u(s)$ is the control input in $s$ domain, $x(0_+)$ is the initial state and $\Phi(s)$ is the *state transition matrix* in $s$ domain and defined by

$$\Phi(s) = (sI - A)^{-1} \tag{2.42}$$

where $I$ is the identity matrix and $A$ is the state matrix. Applying the *inverse Laplace transformation*, the solution in the time domain can be found

$$\begin{aligned} x(t) = \mathscr{L}^{-1}[\Phi(s)x(0_+)] & \qquad + \mathscr{L}^{-1}[\Phi(s)Bu(s)] = \\ = \phi(t, t_0)x(t_0) & \qquad + x_p \end{aligned} \tag{2.43}$$

where $\phi(t, t_0)$ is the state transition matrix expressed in time domain and the expression $\phi(t, t_0)x(t_0)$ is the homogeneous solution, and $x_p$ is the particular solution of

Hill's equations, expressed by

$$x_p = \int_{t_0}^{t} \phi(t,t_0) B u(T)\,\mathrm{d}T \tag{2.44}$$

It is important to note that the homogeneous solution corresponds to the Clohessy-Wiltshire equations Eqs. 2.15 to 2.20 neglecting the acceleration terms $\gamma_x$, $\gamma_y$ and $\gamma_z$. For ease of representation, it will be discussed how to obtain the particular solution referred to *in-plane* maneuver, i.e. referring to the $\mathrm{V_{BAR}}$-$\mathrm{R_{BAR}}$ plane, while equations for *out-of-plane* maneuvers ($\mathrm{V_{BAR}}$-$\mathrm{H_{BAR}}$ plane) will be provided at the end of the section. The state transition matrix is the matrix which map the initial state at $t_0$ to the state after a time period $\tau = t - t_0$. A way to compute the state transition matrix is the use of the exponential matrix

$$\phi(t,t_0) = e^{A(t-t_0)} = e^{A(\tau)} = \phi(\tau) \tag{2.45}$$

which in the $\mathrm{V_{BAR}}$-$\mathrm{R_{BAR}}$ plane can be expressed as

$$\phi(\tau) = \begin{bmatrix} 1 & 6(\omega_0\tau - \sin(\omega_0\tau)) & \frac{4}{\omega_0}\sin(\omega_0\tau) - 3\tau & \frac{2}{\omega}(1-\cos(\omega_0\tau)) \\ 0 & 4 - 3\cos(\omega_0\tau) & \frac{2}{\omega_0}(\cos(\omega_0\tau) - 1) & \frac{1}{\omega_0}\sin(\omega_0\tau) \\ 0 & 6\omega_0(1-\cos(\omega_0\tau)) & 4\cos(\omega_0\tau) - 3 & 2\sin(\omega_0\tau) \\ 0 & 3\omega_0\sin(\omega_0\tau) & -2\sin(\omega_0\tau) & \cos(\omega_0\tau) \end{bmatrix} \tag{2.46}$$

assuming the reduced *in-plane* state as $\hat{x}_{red_{in}} = [x, z, \dot{x}, \dot{z}]^T$. To compute the particular solution it is used again the Laplace transformation and it is assumed a command $u$ constant during a period $t_2 - t_1$ and defined as

$$u(t) = k u_{t_1}(t) - k u_{t_2}(t) \tag{2.47}$$

where $k$ is the unknown magnitude of the impulse and $u_{t_1}(t)$ and $u_{t_2}(t)$ are two step functions as

$$u_a(t) = \begin{cases} 0, & \text{if } t < a \\ 1, & \text{if } t \geq a \end{cases}$$

assuming $t_0 \leq t_1 < t_2$. Note that assuming $u(t) = [u_x(t), u_z(t)]^T$, two unknown $k_x$ and $k_z$ have to be computed. Applying the Laplace transformation to the input $u(t)$

defined in Eq. 2.47, the particular solution can be found

$$\mathscr{L}^{-1}[\Phi(s)Bu(s)] = \frac{1}{m_c}\mathscr{L}^{-1}\begin{bmatrix}\Phi_{1,3}(s)F_x(s)+\Phi_{1,4}(s)F_z(s)\\ \Phi_{2,3}(s)F_x(s)+\Phi_{2,4}(s)F_z(s)\\ \Phi_{3,3}(s)F_x(s)+\Phi_{3,4}(s)F_z(s)\\ \Phi_{4,3}(s)F_x(s)+\Phi_{4,4}(s)F_z(s)\end{bmatrix} \tag{2.48}$$

remembering that the reduced control matrix $B$ for the $\mathrm{V_{BAR}}$-$\mathrm{R_{BAR}}$ plane is

$$B_{red_{in}} = \frac{1}{m_c}\begin{bmatrix}0 & 0\\ 0 & 0\\ F_x & 0\\ 0 & F_z\end{bmatrix}$$

Hence, the particular solution of Eq. 2.48 for the case of a maneuver executed with a single constant impulse can be computed

$$x_p = \frac{1}{m_c}Hu = \frac{1}{m_c}\left[h_1, h_3\right]u \tag{2.49}$$

with the two column vector $h_1, h_3 \in \mathbb{R}^{4,1}$ expressed by

$$h_1 = \begin{bmatrix}\frac{4}{\omega_0^2}[\cos(\omega_0(t-t_2))-\cos(\omega_0(t-t_1))]+\frac{3}{2}[(t-t_2)^2-(t-t_1)^2]\\ \frac{2}{\omega_0^2}[\sin(\omega_0(t-t_1))-\sin(\omega_0(t-t_2))+\omega_0(t_1-t_2)]\\ \frac{4}{\omega_0}[\sin(\omega_0(t-t_1))-\sin(\omega_0(t-t_2))]+3(t_1-t_2)\\ \frac{2}{\omega_0}[\cos(\omega_0((t-t_1))-\cos(\omega_0(t-t_2))]\end{bmatrix} \tag{2.50}$$

$$h_3 = \begin{bmatrix}\frac{2}{\omega_0^2}[\sin(\omega_0(t-t_2))-\sin(\omega_0(t-t_1))]+\omega_0(t_2-t_1)]\\ \frac{1}{\omega_0^2}[\cos(\omega_0(t-t_2))-\cos(\omega_0(t-t_1))]\\ \frac{2}{\omega_0}[\cos(\omega_0(t-t_2))-\cos(\omega_0(t-t_1))]\\ \frac{1}{\omega_0}[\sin(\omega_0((t-t_1))-\sin(\omega_0(t-t_2))]\end{bmatrix} \tag{2.51}$$

The general solution of Hill's equation for a single impulse maneuver is eventually found

$$x(t) = \phi(t,t_0)x(t_0) + \frac{1}{m_c}Hu \tag{2.52}$$

and the control command $u$, i.e. the magnitude $k_x$ and $k_z$, may be easily found by

$$u = \left(\frac{1}{m_c}H\right)^{-1} (x(t) - \phi(t,t_0)x(t_0)) \tag{2.53}$$

but it cannot be analytically solved because the matrix $H \in \mathbb{R}^{4,2}$ is not square, and so it does not admit computation of the inverse matrix. This problem can be partially solved by computing the *pseudo-inverse* matrix of $H$

$$H_{pseudo} = (\mathrm{H}^T\,\mathrm{H})^{-1}\mathrm{H}^T \tag{2.54}$$

where $\mathrm{H} = \frac{1}{m_c}H$ and

$$u = H_{pseudo}\,(x(t) - \phi(t,t_0)x(t_0)) \tag{2.55}$$

is the solution obtained by solving Eq. 2.53 using the pseudo-inverse matrix, which means to find a *least square solution*, that is not analytical and it is sub-optimal. On the other hand, this method allows to find a solution for any maneuver to be executed by setting $t_0$, $t_1$ and $t_2$. If the maneuver is, for example, an orbit raising maneuver between proper initial and final waypoints, setting $t_0 = t_1 = 0$ and $t_2 = T$ the solution converges to what prescribed by Eq. 3.8 and depicted in Fig. 3.2c.

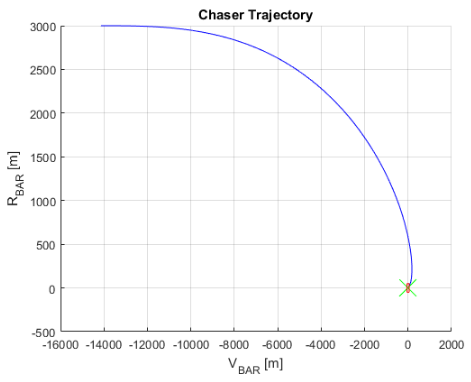
As mentioned before, this method can be easily applied also for out-of-plane maneuvers, assuming the reduced state $\hat{x}_{red_{out}} = [y, \dot{y}]^T$, and by setting the state transition matrix and the $H$ matrix respectively as

$$\phi(\tau) = \begin{bmatrix} \cos(\omega_0\tau) & \frac{1}{\omega_0}\sin(\omega_0\tau) \\ -\omega_0\sin(\omega_0\tau) & \cos(\omega_0\tau) \end{bmatrix} \tag{2.56}$$

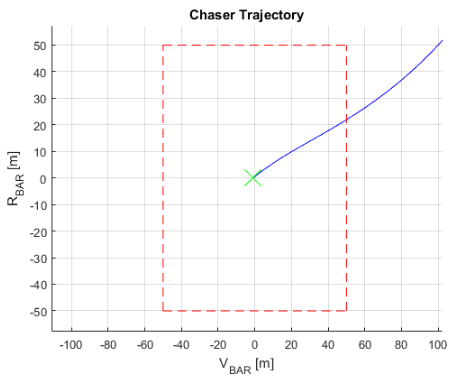
$$H = [h_2] = \begin{bmatrix} \frac{1}{\omega_0^2}[\cos(\omega_0(t-t_2)) - \cos(\omega_0(t-t_1))] \\ \frac{1}{\omega_0}[\sin(\omega_0(t-t_1)) - \sin(\omega_0(t-t_2))] \end{bmatrix} \tag{2.57}$$

and considering $u(t) = k_y$. Certainly, re-arranging the state transition and $H$ matrices for both in-plane and out-of-plane dynamics it is possible to find a global state transition matrix and an $H$ matrix for the complete solution, finding the unknown $u(t) = [k_x, k_y, k_z]^T$.

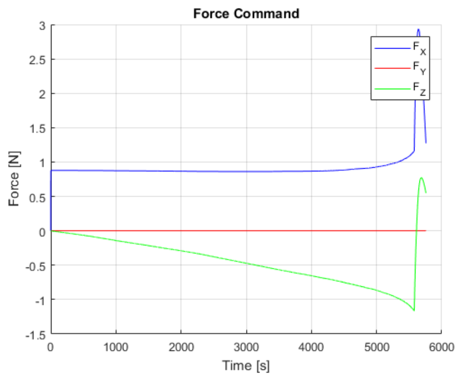
The Continuous Lambert-type algorithm has been tested by executing the same maneuver of the previous cases and performance of the algorithm have been prelimi-



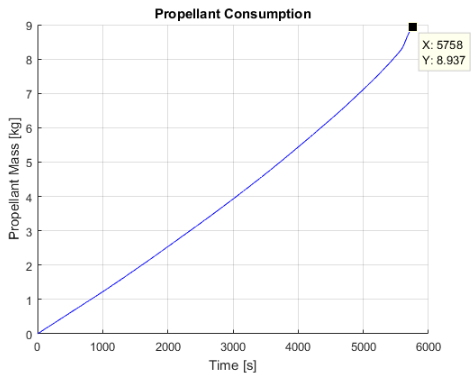
(a) $V_{BAR}$-$R_{BAR}$ trajectory.



(b) $V_{BAR}$-$R_{BAR}$ trajectory: zoom on *S2*.



(c) Guidance force command.



(d) Propellant mass.

Fig. 2.6 Preliminary results of the Continuous Lambert-type Guidance algorithm.

nary evaluated, as summarized in Fig. 2.6. The resulting trajectory is depicted in Fig. 2.6a and it is very similar to the trajectory followed by the Modified ZEM/ZEV algorithm. The final waypoint is reached with higher accuracy with respect to the Modified ZEM/ZEV (see Fig. 2.6b). The guidance command is quite regular, but there is a residual component along $R_{BAR}$ which is not expected by theoretical computations, as described in Section 3.1. The computed propellant consumption, as in Fig. 2.6d is comparable with the ideal one. These preliminary results allow the Continuous Lambert-type Guidance algorithm to be further tested and implemented in the complete GNC software.

### 2.2.3 Guidance Implementation

In this section it will be discussed the implementation of the guidance function, focusing on the guidance flowchart which describes the operations executed by the guidance function. The flowchart related to the guidance function is depicted in Fig. 2.7.

The guidance function is initialized, including initialization of waypoints $S2$, $S3$ and $S4$ since they are defined with a fixed position with respect to the target spacecraft, as it will described in Section 3.1.2. Computation of waypoint $S1$ is running continuously: indeed, if the initial altitude of the chaser spacecraft is different from the nominal one, the fixed location of $S1$ may became not optimal, since it has been computed in order to be compliant with a continuous thrust orbit raising maneuver. For this reason, the location of $S1$ is computed according to $\text{V}_{\text{BAR}}$-$\text{R}_{\text{BAR}}$ actual position of the spacecraft as estimated by the navigation function. In this way, the location of $S1$, which is mainly dependent by the orbital altitude of the Chaser with respect to the Target orbit, is continuously varying during each cycle of the guidance function, until it is not reached during the phasing maneuver. If the actual location of the Chaser is coincident with the computed location of waypoint $S1$, the first maneuver is initialized and then executed, until the spacecraft reaches waypoint $S2$ and executes station keeping. When the station keeping in $S2$ has been reached, the Flight Manger function commands the Guidance to execute the next maneuver. Hence, the guidance function initialize and execute the second maneuver through the next waypoint. In $S3$, after reaching the station keeping, the final maneuver is initialized and executed according to the command of the Flight Manager function.

## 2.3 Navigation

The navigation function is in charge to collect noisy data provided by sensors and to compute the estimated state of the spacecraft, including both position and attitude estimation. This function is realized by filtering the measurement noise introduced by sensors and to merge data from multiple sensors in order to obtain a better state estimation, thanks to the adoption of multiple sensors, through data fusion techniques. The purpose of this section is to discuss the implementation of the digital navigation filter of the navigation function.

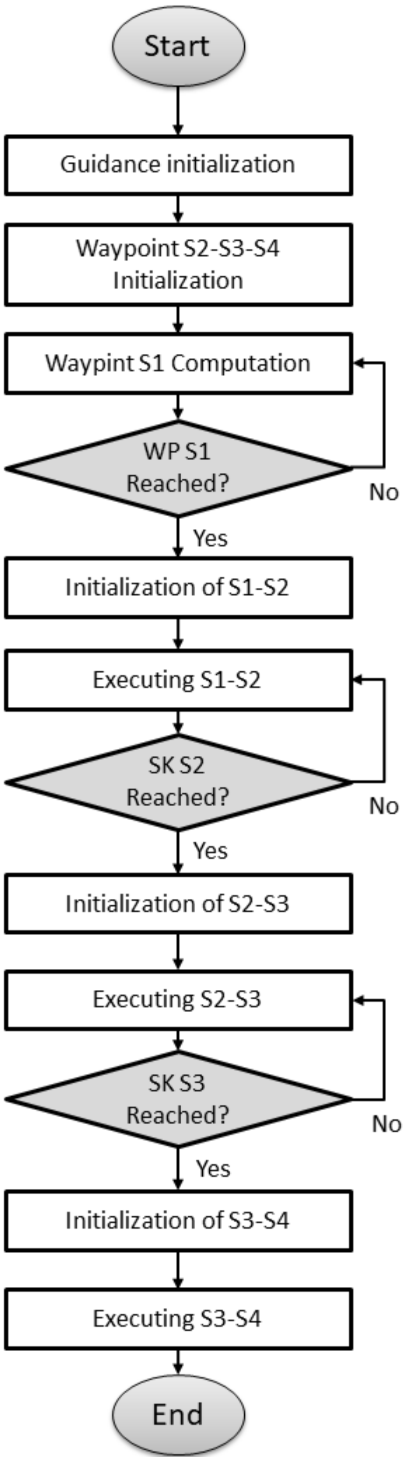


Fig. 2.7 Guidance function flowchart.

### 2.3.1 Navigation Filter

The navigation filter implemented in the actual GNC architecture is the *Kalman filter*. The Kalman filter is designed as an optimal state estimator for linear systems, and it is very effective also in linearized systems and in non-linear systems prior local linearization. The Kalman filter is a recursive filter, which means that the optimal estimation is achieved after a number of iterations. In addition, it allows to merge output of different sensors to obtain the state estimation, hence it can be easily used for data fusion purposes. The Kalman filter implemented in the navigation function is a discrete filter, which means that is based on the discretization of the state space system

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) + w(t) \\ y(t) = Cx(t) + v(t) \end{cases}$$

where $x(t)$ is the state, $u(t)$ the control action, $w(t)$ and $v(t)$ are respectively the process and measurement uncertainties, which are uncorrelated Gaussian processes. The discrete state space formulation is obtained by

$$\begin{cases} x(k+1) = Fx(k) + Gu(k) + w(k) \\ y(k) = Hx(k) + v(k) \end{cases} \tag{2.58}$$

where the term $k+1$ is referred to the successive time step and matrices $F$, $G$ and $H$ are the discrete formulation of matrices $A$, $B$ and $C$ obtained by

$$\begin{cases} F = e^{AT} \\ G = \int_0^T e^{A\sigma} B \,\mathrm{d}\sigma \\ H = C \end{cases} \tag{2.59}$$

where $T$ is the discretization time step. The recursive process of the Kalman filter is based on a prediction-correction approach: a first state estimation is computed by propagating the state with actual measurments, as in Eq. 2.58, and the covariance matrix is projected ahead, then the Kalman gain is computed and it is used to update the state estimation by the knowledge of the computed covariance matrix, which is updated in the last step of the recursive scheme. The Kalman filtering scheme can be resumed in seven steps:

1. *State prediction*. The state at the next step $k+1$, $\hat{x}(k+1|k)$, is predicted with information about the actual state at the present step $k$, $\hat{x}(k|k)$:

$$\hat{x}(k+1|k) = F\hat{x}(k|k) + Gu(k)$$

2. *Covariance prediction*. The covariance matrix is propagated forward using information related to the actual time step and by the knowledge of the covariance matrix $W$ related to the process noise $w(k)$:

$$\Sigma(k+1|k) = F\Sigma(k|k)F^T + W(k)$$

3. *Innovation*. It is computed the error between the actual state measurement and the estimated one:

$$e(k) = y(k) - H\hat{x}(k+1|k)$$

4. *Innovation of the covariance matrix*. The covariance matrix is computed taking into account covariace matrix $V$ related to measurement errors $v(k)$:

$$S(k) = H\Sigma(k+1|k)H^T + V(k)$$

5. *Kalman gain*. The Kalman gain is then computed:

$$K(k) = \Sigma(k+1|k)H^T S^{-1}(k)$$

6. *State estimation update*. The estimated state is updated by the knowledge of the Kalman gain. This is the output of the Kalman filter:

$$\hat{x}(k+1|k+1) = \hat{x}(k+1|k) + K(k)e(k)$$

7. *Covariance update*. The covariance matrix $\Sigma$ is updated according to the new information about covariance and Kalman gain:

$$\Sigma(k+1|k+1) = (I - K(k)H)\Sigma(k+1|k)$$

Output of step 6 and step 7 will be the input respectively of step 1 and step 2 at the successive iteration cycle. This iterative cycle can be used also for the implementation of the *Extended Kalman Filter (EKF)*. The EKF is basically the extension of the Kalman filter to non-linear systems in the form

$$\begin{cases} x(k+1) = f(x(k), u(k)) + w(k) \\ y(k) = h(x(k)) + v(k) \end{cases} \tag{2.60}$$

where $f$ and $h$ are non-linear functions. In this case, the state propagation of step 1 is obtained by

$$\hat{x}(k+1|k) = f(\hat{x}(k|k), u(k)) \tag{2.61}$$

while $F$ and $H$ matrices are substituted by the Jacobian matrices of $f$ and $g$

$$F = \left(\frac{\partial f}{\partial x}\right)_{x=\hat{x}} \tag{2.62}$$

$$H = \left(\frac{\partial h}{\partial x}\right)_{x=\hat{x}} \tag{2.63}$$

which means that it is applied a local linearization for the computation of the Kalman gain and the the state estimation.

## 2.4 Control

The Control function computes the output to be sent to the actuators. In other word, the control function is in charge to drive the spacecraft trough a specific trajectory according to the guidance commands and to orient the and stabilize the attitude of the spacecraft with respect to a defined reference frame. The control function developed in the present dissertation has been implemented in order to support the design of the guidance function and to preliminary test the implemented guidance algorithm introducing misalignment, delays and errors that can be induced by a real attitude control. In addition, the control function is continuously monitoring the telemetry of the reaction wheels in order to execute wheels de-saturation commands and restore the availability of the wheels. To control thrust and torque provided by the reaction control system, the control function implements a thrusters modulation function, based on the Pulse-Width Modulation (PWM) technique.

### 2.4.1 Basics of Attitude Control

As extensively mentioned in the previous section, attitude control is a fundamental function to be implemented in the GNC software for almost all space applications. Passive stabilization techniques will not be further discussed, since the present dissertation is based on active control systems.

#### Proportional Integral Derivative Controller

The Proportional Integral Derivative (PID) Controller is commonly used in space application due to its simple design and its inherently robustness, especially if it is designed with proper techniques, such as Linear Matrix Inequalities (LMI) [57]. The general PID control law is expressed by

$$u(t) = K_p e(t) + K_i \int e(t)\,\mathrm{d}t + K_d \frac{\mathrm{d}e(t)}{\mathrm{d}t} \tag{2.64}$$

where $e(t)$ is the error between the reference and actual state, $K_p$ is the proportional gain, $K_i$ is the integral gain and $K_d$ is the derivative gain. In order to ensure stability and robustness, such gains have to be tuned properly. The *proportional action*, $K_p e(t)$, is the basic control action: the control is commanded proportionally to the error. In many application, the proportional control action alone is enough to design a satisfactory controller. The *integral action*, $K_i \int e(t)\,\mathrm{d}t$ integrates the error and compute the control action in the long term. Applying only the proportional action, it can be present a residual error which cannot be reduced only with the proportional control, and the integral action tends to reduce such error and allows to reach exactly the desired reference. Eventually, the *derivative action*, $K_d\,\mathrm{d}e(t)/\mathrm{d}t$, acts on the error derivative $\dot{e}(t)$, hence it is used to increase dynamic performance of the controller and reducing residual oscillations derived by the proportional action alone. In many applications, including space applications, not all the control action are included in the design of the PID controller, which may result in a simpler PD or PI controller, depending by control requirements specified for each application.
Using the quaternion notation, the error can be expressed as

$$e(t) = q_{err} = q_{BE}^{-1} * q_{des} \tag{2.65}$$

where $q_{err}$ is the quaternion error, $q_{BE}$ is the Body quaternion with respect to the ECI frame, $q_{des}$ is the desired reference quaternion and the symbol $*$ express the quaternion product.

### Linear Quadratic Regulator Controller

The *Linear Quadratic Regulator (LQR)* is based on the optimal control theory. A brief introduction to LQR has been provided in section 2.2.1. Differently from PID controllers, LQR controllers are not inherently robust [58], but a proper design of the controller ensures stability and good performance of the plant. Application of LQR to control the attitude dynamics of a spacecraft requires a linearization of the system dynamics, since LQR can be applied only to linear systems. The classical formulation of an LQR controller, the control law of Eq. 2.31 drives the state $\hat{x}$ to zero. According to this definition, defining a control law related to the local vertical frame means to regulate the attitude of the spacecraft aligned with local vertical frame. For this reason, it has been computed the linear relative attitude dynamics. First, is necessary to define the relative state in quaternion form

$$\hat{x} = [q_1, q_2, q_3, \omega_1, \omega_2, \omega_3]^T \tag{2.66}$$

where the scalar component of the quaternion, $q_0$, is neglected, since it is dependent by the vectorial components according to $\mathtt{qnorm}(\mathbf{q}) = 1$ (the quaternion norm is equal to 1). The angular velocity of the spacecraft expressed in body axis with respect the local vertical frame is

$$\omega^b_{lob} = \omega^b_{ECIb} - \omega^b_{ECIlo} = \omega^b_{ECIb} - DCM_{blo}\omega^{lo}_{ECIlo} \tag{2.67}$$

where $\omega^b_{ECIb}$ is the inertial angular velocity, as computed by Eq. 3.40, $\omega^b_{ECIlo}$ is the angular velocity of the local vertical frame expressed in body frame, $DCM_{blo}$ is the rotation matrix from local vertical to to body frame and $\omega^{lo}_{ECIlo}$ is the angular velocity of the local vertical frame expressed in the local vertical frame, defined by

$$\omega^{lo}_{ECIlo} = [0, -\omega_0, 0]^T \tag{2.68}$$

The rotation matrix $DCM_{blo}$ can be computed as function of the related quaternion as

$$DCM_{blo} = \begin{bmatrix} q_0^2+q_1^2-q_2^2-q_3^3 & 2(q_1q_2+q_0q_3) & 2(q_1q_3-q_0q_2) \\ 2(q_1q_2-q_0q_3) & q_0^2-q_1^2+q_2^2-q_3^3 & 2(q_2q_3+q_0q_1) \\ 2(q_1q_3+q_0q_2) & 2(q_2q_3-q_0q_1) & q_0^2-q_1^2-q_2^2+q_3^3 \end{bmatrix} \tag{2.69}$$

Assuming the reference quaternion as

$$q_{ref} = [1, 0, 0, 0]^T$$

it is possible linearize the rotation matrix 2.69 as

$$DCM_{blo} = \begin{bmatrix} 1 & 2q_3 & -2q_2 \\ -2q_3 & 1 & 2q_1 \\ 2q_2 & -2q_1 & 1 \end{bmatrix} \tag{2.70}$$

and consequently the linearized relative angular velocity is

$$\omega_{lob}^b = \omega_{ECIb}^b - \begin{bmatrix} -2q_3\omega_0 \\ -\omega_0 \\ 2q_1\omega_0 \end{bmatrix} \tag{2.71}$$

and the time derivative

$$\dot{\omega}_{lob}^b = \dot{\omega}_{ECIb}^b - \begin{bmatrix} -2\dot{q}_3\omega_0 \\ 0 \\ 2\dot{q}_1\omega_0 \end{bmatrix} \tag{2.72}$$

Eventually, the time derivative of the relative quaternion can be computed by Eq. 3.41, which results

$$\dot{q}_{rel} = \frac{1}{2}[0, \omega_1, \omega_2, \omega_3]^T \tag{2.73}$$

Assuming the inertia matrix of the spacecraft related to the principal axis of inertia, the inertia matrix $J$ of Eq. 3.40 is diagonal, and neglecting the contribution of reaction wheels, the matrix $A$ of the state space system will result as

$$A = \begin{bmatrix} 0 & 0 & 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2} \\ -2\omega_0^2 k_x & 0 & 0 & 0 & 0 & -(k_x+1)\omega_0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -2\omega_0^2 k_z & -(1+k_z)\omega_0 & 0 & 0 \end{bmatrix} \tag{2.74}$$

where $k_x = (I_{yy} - I_{zz})/I_{xx}$ and $k_z = (I_{xx} - I_{yy})/I_{zz}$, and $I_{xx}$, $I_{yy}$ and $I_{zz}$ are the diagonal terms of the principal inertia matrix $I$. The control matrix will become

$$B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{1}{I_{xx}} & 0 & 0 \\ 0 & \frac{1}{I_{yy}} & 0 \\ 0 & 0 & \frac{1}{I_{zz}} \end{bmatrix} \tag{2.75}$$

According to the definition of the state $\hat{x}$, the LQR controller results to be the equivalent of a PD controller.

### 2.4.2 Implementation of the Control Function

In order to develop a control function which is able to support the design and the implementation of the guidance algorithm, it has been chosen to implement an LQR regulator, performing a tuning of $Q$ and $R$ matrices in order to fit limitations of the reaction wheels system which equips the reference spacecraft. The computation of the gain $K_{LQR}$ related to the attitude control law has been executed in MATLAB®environment, using the `lqr` function. The tuning of the weighting matrices resulted in

$$Q = 0.045 \cdot \mathrm{I}$$
$$R = \mathrm{I}$$

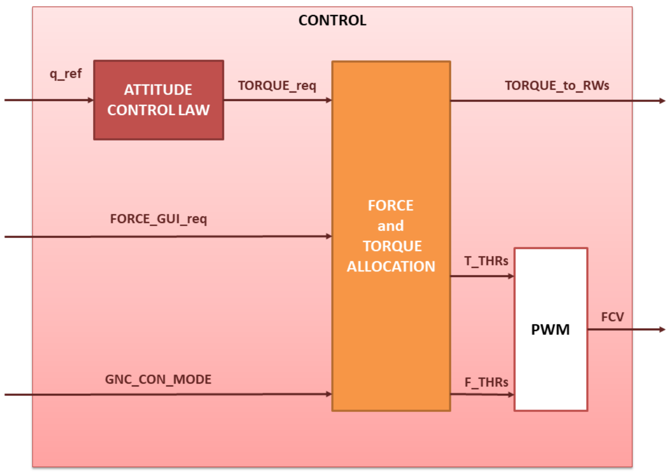


Fig. 2.8 Control functional architecture.

where I is the identity matrix. The design of the control function requires the implementation of the thrust modulating function and, in addition, it is required a torque allocation function, since the attitude control can be executed using both thrusters or reaction wheels. In addition, the torque allocation function shall manage the de-saturation of reaction wheels. The functional architecture of the control function is depicted in Fig. 2.8. The navigation function provides the reference quaternion related to the local vertical frame, $q_{ref}$, which is computed by the LQR controller, producing the required control torque. The *Force and Torque Allocation* function allows the torque input of the control law, the force input from the guidance function and the GNC mode, which drives the wheels de-saturation command. The function manage the output torque to the reaction wheels system and the torque and force command to be modulated for the thrusters activation by the *PWM* function. Finally, the PWM function compute the opening command of the flow control valves (FCV) of each thruster.

The Force and Torque Allocation function is needed to allocate the amount of torque to be provided by reaction wheels and thrusters. During nominal operations, the torque is allocated only to the reaction wheels, except for higher torque commands required for large attitude maneuvers. During wheels de-saturation, the torque is allocated to both reaction wheels and thrusters, but with opposite sign. During torque generation by the thrusters system, the force modulation is inhibited, since the attitude control has the priority with respect to the position control.

The PWM function modulates the opening and the closing of the flows control valves of the thrusters system. According to the force or torque request, the PWM computes the opening time of each thrusters and it allocates the opening time to the related couple of thrusters, according to the sign and to the force axis. Depending by the force or torque request, the opening time is computed as

$$t_{on_i} = \frac{|F_{req_i}|}{F_{max}} \qquad\qquad t_{on_i} = \frac{|M_{req_i}|}{M_{max}} \tag{2.76}$$

where the index $i$ is related to the i$^{\text{th}}$-thruster. Depending by the sign and by the axis of the force, $t_{on_i}$ is allocated to a specific couple of thrusters: indeed, if, for example, the force modulation is requested along the +x-axis, only the couple of thruster which act along the +x-axis will be opened for a time equal to the computed $t_{on_i}$. The same for the torque request. Once the value of $t_{on_i}$ is computed, such value has to be compared to the minimum opening time, $t_{on_{min}}$ and the maximum opening time, which corresponds to the modulation period of the PWM function, $\Delta t_{mod}$. Hence, the value of $t_{on_i}$ is adjusted such as

$$\begin{cases} t_{on_i} = \Delta t_{mod} & \text{if } t_{on_i} \geq \Delta t_{mod} \\ t_{on_i} = t_{on_i} & \text{if } t_{on_{min}} \leq t_{on_i} < \Delta t_{mod} \\ t_{on_i} = 0 & \text{if } t_{on_i} < t_{on_{min}} \end{cases} \tag{2.77}$$

Eventually, a timer implemented in the PWM function is reset after each modulation cycle

$$t - t_{0_{PWM}} = \Delta t_{mod}$$

where $t$ is the on-board clock time and $t_0$ is reset to the value $t$ after each PWM cycle. Hence, once $t_{on_i}$ is adjusted, the FCV output is computed as

$$FCV_{cmd_i} = \begin{cases} 1 & \text{if } t - t_{0_{PWM}} \leq t_{on_i} \\ 0 & \text{if } t - t_{0_{PWM}} > t_{on_i} \end{cases} \tag{2.78}$$

The final vector $FCV_{cmd} = [FCV_{cmd_1}, \dots, FCV_{cmd_{12}}]^T$ is containing the opening command two the 12 FCVs of each thruster.

# Chapter 3

# Orbital RVD Modelling and Simulator

In this section, all the mathematical models, including reference frames, orbital dynamics, environmental disturbances, sensors and actuators, developed during the design of the functional orbital simulator will be described. First of all, the reference rendezvous and docking maneuver is described, in addition to the detailed description of the functional architecture of the orbital simulator.

## 3.1 RVD Maneuver

The nominal rendezvous and docking maneuver developed in the framework of the SAPERE STRONG mission scenario will be described in the following. In addition, a brief review on basic orbital maneuvers will be provided, in order to better understand the complete RVD maneuver and each mission phase.

### 3.1.1 Basic Orbital Maneuvers

This section will focus on description of basic orbital maneuvers defined in a relative frame. Understating such basic trajectories related to relative orbital dynamics computation will help to understand the relative orbital motion of an object with respect to a relative reference frame, as depicted in Fig. 1.1 and deeply discussed in Section

3.3.1. Tangential, radial and horizontal maneuvers, as well as the special cases of straight line maneuvers will be analyzed in the following of the present section.. Each kind of maneuver will be discussed about both impulsive and continuous thrust execution. All pictures included in the present section have been taken from the textbook of W. Fehse [42].

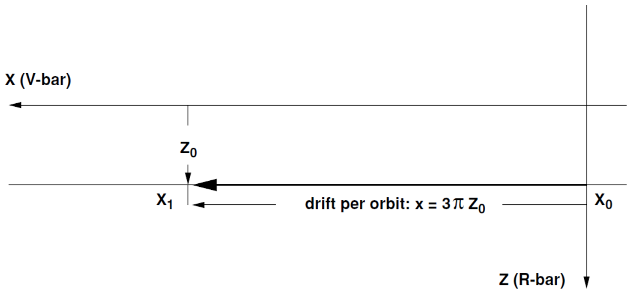
### Straight Line Maneuvers

In the class of straight line maneuvers it has been studied both free and forced motion straight line maneuvers. The *free drift* maneuver is depicted in Fig. 3.1a. This maneuver is basically a straight line maneuver executed at constant velocity and at constant altitude without acting any commands. This maneuver is usually executed to reduce the phase angle between the target and the chaser spacecraft: the chaser orbiting at a lower altitude will move faster and it will get closer (along $\mathrm{V_{BAR}}$) to the target, while a chaser orbiting at an higher altitude will move slower and it will get farther from the target. Assuming the orbital angular velocity $\omega_0$ of the chaser as defined in Eq. 3.33 and the orbit altitude of the chaser as $Z_0$, the free drift velocity is computed by

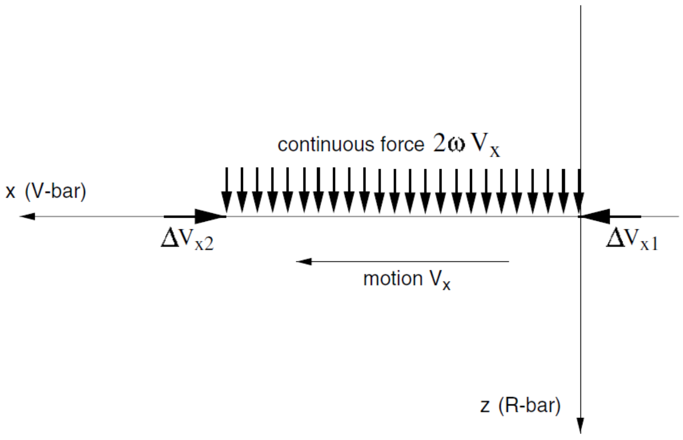
$$\dot{x}_{fd} = \frac{3}{2}\omega_0 Z_0 \tag{3.1}$$

Similar to the free drift maneuver, a second motion at constant velocity along $\mathrm{V_{BAR}}$ it obtained by a *forced $V_{BAR}$ straight line maneuver*, depicted in Fig. 3.1b. This maneuver is usually executed during the last meters of a RVD maneuver, in which it is mandatory to maintain a constant velocity profile for safety and scheduling reasons, and hence it is executed at the same altitude of the target spacecraft (zero altitude of the relative reference frame), but the theory can be extended to a forced motion at different altitude. To execute the maneuver, a first impulse $\Delta V_{x1}$ is provided to accelerate the spacecraft to the desired velocity $V_x$, while a second impulse $\Delta V_{x2}$, with same magnitude and opposite sign of $\Delta V_{x1}$ is provided at the end of the maneuver to stop the motion of the spacecraft. During the cruise phase, a constant force along the $\mathrm{R_{BAR}}$ axis ($F_z = m \cdot \gamma_z$, where $m$ is the mass of the spacecraft and $\gamma_z$ is the force per unit mass) it has to be applied to counteract drift effects and it is computed by

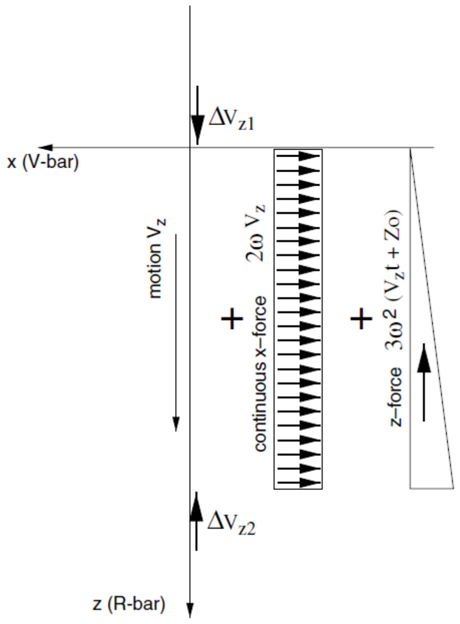
$$\gamma_z = 2\omega_0 V_x \tag{3.2}$$



(a) Free drift maneuver.



(b) Forced $V_{BAR}$ straight line maneuver.


$\Delta V_{z1}$
x (V-bar)
motion $V_z$
+ continuous x–force $2\omega V_z$
+ z–force $3\omega^2 (V_z t + Zo)$
$\Delta V_{z2}$
z (R-bar)

(c) Forced $R_{BAR}$ straight line maneuver.

Fig. 3.1 Basics straight line maneuvers.

The third straight line maneuver is executed along the R$_{\text{BAR}}$ axis, i.e. a *forced $R_{BAR}$ straight line maneuver*, depicted in Fig. 3.1c. As for the forced V$_{\text{BAR}}$ straight line maneuver, this maneuver is the equivalent maneuver to be execute when the docking port is located along the R$_{\text{BAR}}$ axis. Two impulses have to be applied to accelerate the spacecraft to the desired velocity $V_z$ and stop the motion, respectively $\Delta V_{z1}$ and $\Delta V_{z2}$. At the same time, a constant force along V$_{\text{BAR}}$ and a variable force along R$_{\text{BAR}}$ have to be applied to counteract orbital drift effects, i.e.

$$\gamma_x = -2\omega_0 V_z \tag{3.3}$$

$$\gamma_z = -3\omega_0^2 (V_z t + z_0) \tag{3.4}$$

where $\gamma_x$ and $\gamma_z$ are forces per unit mass related respectively to the V$_{\text{BAR}}$ and R$_{\text{BAR}}$ forces, $t$ is the maneuver time and $z_0$ is the initial altitude of the maneuvering spacecraft.

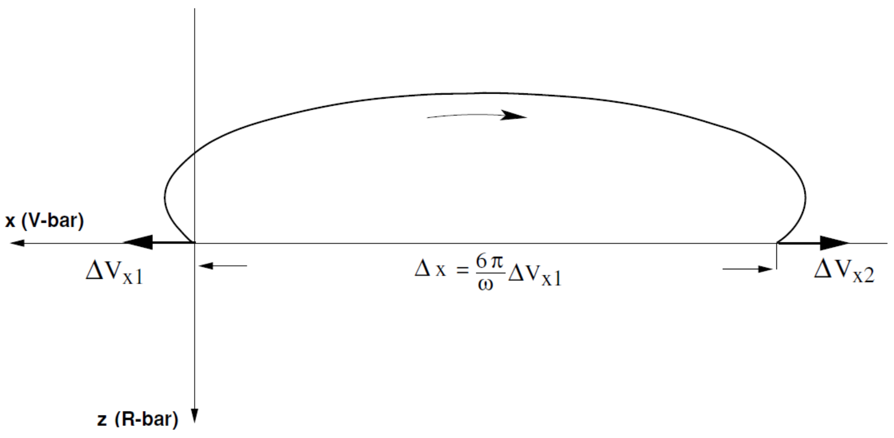
## Tangential Maneuvers

Tangential maneuvers are intended such maneuvers in which the control command is actuated along the V$_{\text{BAR}}$ axis. Applying a $\Delta V$ along the tangential direction allows the execution of two possible maneuvers: a tangential transfer along V$_{\text{BAR}}$ (Fig. 3.2a) and an orbit raising maneuver, called *Hohmann transfer* (Fig. 3.2b). In the first case, it is possible to move the spacecraft along the V$_{\text{BAR}}$ axis of the desired amount by applying a velocity impulse. The distance traveled along V$_{\text{BAR}}$ depends by the magnitude and the sign of the first velocity impulse

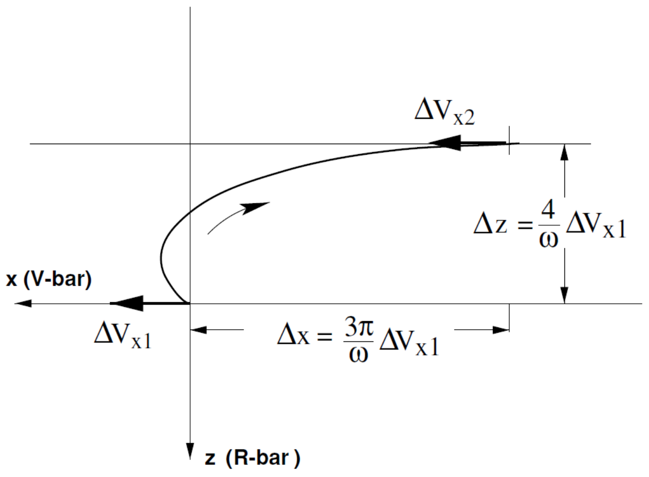
$$\Delta x = \frac{6\pi}{\omega_0} \Delta V_{x1} \tag{3.5}$$

and to stop the maneuver it is required to apply a second impulse, $\Delta V_{x2}$ of the same magnitude but opposite sign, after a time equal to an orbital period $T = 2\pi/\omega_0$.

Since after half an orbital period the tangential maneuver reaches the maximum (or minimum) altitude along R$_{\text{BAR}}$, applying the second impulse after half an orbital period can be used to execute a different maneuver. An *orbit raising* maneuver can be realized in this way: depending by the total amount of altitude it is required to

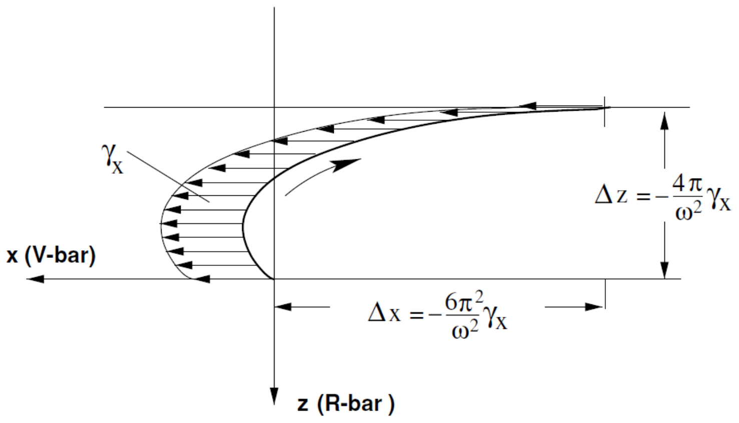


(a) Tangential $V_{BAR}$ transfer maneuver.



$\Delta V_{x2}$
$\Delta z = \frac{4}{\omega} \Delta V_{x1}$
x (V-bar)
$\Delta V_{x1}$
$\Delta x = \frac{3\pi}{\omega} \Delta V_{x1}$
z (R-bar )

(b) Tangential orbit raising maneuver (Hohmann).



$\gamma_x$
$\Delta z = -\frac{4\pi}{\omega^2} \gamma_x$
x (V-bar)
$\Delta x = -\frac{6\pi^2}{\omega^2} \gamma_x$
z (R-bar )

(c) Tangential orbit raising continuous thrust maneuver.

Fig. 3.2 Tangential maneuvers.

change, $\Delta z$, the first impulse can be evaluated

$$\Delta V_{x1} = \frac{\omega_0}{4}\Delta z \tag{3.6}$$

and applied at $t = 0$. Then, after half an orbital period, $t = T/2$, it is applied a second impulse $\Delta V_{x2}$ with same magnitude and direction of the first one. After application of the second impulse, the chaser go on moving in a straight line with free drift velocity magnitude computed at the final altitude. The distance traveled between the two pulses can be evaluated by

$$\Delta x = \frac{3\pi}{4}\Delta z \tag{3.7}$$

This specific maneuver is also called *Hohmann transfer* and it is the optimal maneuver to execute an in-plane orbital change.

An orbit raising maneuver can also be realized with a *continuous thrust* maneuver (Fig. 3.2c). Differently from the Hohmann transfer, the propulsion system should provide a continuous force along $\mathrm{V_{BAR}}$, $F_x$, and the maneuver is stopped after *one* orbital period by ending to provide thrust. To change the orbit by the desired amount $\Delta z$, the acceleration $\gamma_x = F_x/m_c$ that should be provided during the whole maneuver is

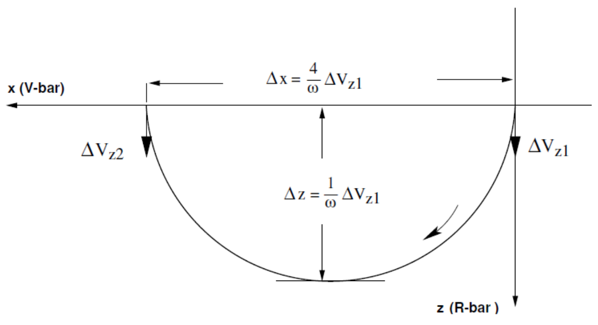
$$\gamma_x = -\frac{\omega_0^2}{4\pi}\Delta z \tag{3.8}$$

considering that $\Delta z = z_{final} - z_0$, increasing the orbit results in a negative $\Delta z$ and to execute the maneuver is required a *positive* acceleration. After one orbital period the maneuver is stopped by setting $\gamma_x = 0$ and the distance traveled along $\mathrm{V_{BAR}}$ is

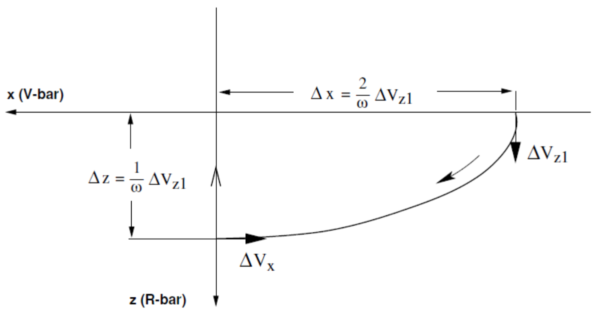
$$\Delta x = -\frac{6\pi^2}{\omega_0^2}\gamma_x \tag{3.9}$$

**Radial Maneuvers**

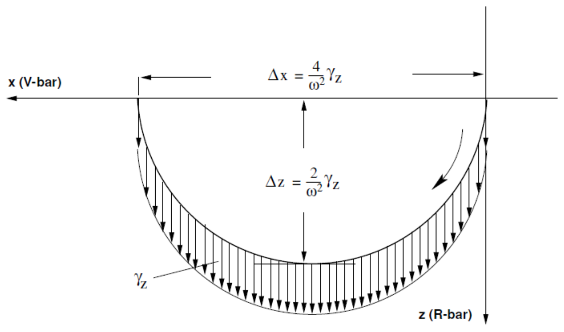
Radial maneuvers are executed applying the control command along the $\mathrm{R_{BAR}}$ axis. The first maneuver, also called *radial boost maneuver*, is used to modify the relative distance along $\mathrm{V_{BAR}}$, as for the example of the tangential $\mathrm{V_{BAR}}$ transfer maneuver. A first impulse, $\Delta V_{z1}$, is commanded at the beginning of the maneuver and a second impulse, $\Delta V_{z2}$, is commanded after half orbital period to stop the maneuver.


x (V-bar)
$\Delta x = \frac{4}{\omega}\Delta V_{z1}$
$\Delta V_{z2}$
$\Delta V_{z1}$
$\Delta z = \frac{1}{\omega}\Delta V_{z1}$
z (R-bar)

(a) Radial $V_{BAR}$ transfer maneuver.


x (V-bar)
$\Delta x = \frac{2}{\omega}\Delta V_{z1}$
$\Delta V_{z1}$
$\Delta z = \frac{1}{\omega}\Delta V_{z1}$
$\Delta V_x$
z (R-bar)

(b) Radial fly around maneuver.


x (V-bar)
$\Delta x = \frac{4}{\omega^2}\gamma_z$
$\Delta z = \frac{2}{\omega^2}\gamma_z$
$\gamma_z$
z (R-bar)

(c) Radial $V_{BAR}$ transfer continuous thrust maneuver.

Fig. 3.3 Radial maneuvers.

Differently for the equivalent tangential maneuver, the second impulse have the same magnitude and direction of the first one. The distance traveled is

$$\Delta x = \frac{4}{\omega_0}\Delta V_{z1} \tag{3.10}$$

This maneuver can be stopped after a quarter of orbital period to execute, for example, a *fly around maneuver* which allows the spacecraft to reach a docking port located along the $\mathrm{R_{BAR}}$ axis, as in Fig. 3.3b. In this case, the maneuver is not a *pure* radial maneuver, since after the first impulse $\Delta V_{z1}$ it is required to apply a impulse along the $\mathrm{V_{BAR}}$ axis, $\Delta V_x$, to stop the maneuver. This impulse is applied after a quarter of orbital period and the magnitude is

$$|\Delta V_x| = 2 \cdot |\Delta V_{z1}| \tag{3.11}$$

After a quarter of orbital period, the distance traveled along $\mathrm{V_{BAR}}$, $\Delta x$, and the relative $\mathrm{R_{BAR}}$ distance, $\Delta z$, can be evaluated by

$$\Delta x = \frac{2}{\omega_0}\Delta V_{z1} \tag{3.12}$$

$$\Delta z = \frac{1}{\omega_0}\Delta V_{z1} \tag{3.13}$$

A $\mathrm{V_{BAR}}$ transfer may also be executed with a *radial continuous thrust maneuver*, as in Fig. 3.3c. In this case, a force along $\mathrm{R_{BAR}}$, $F_z = m \cdot \gamma_z$, is provided continuously during the maneuver. The force per unit mass, $\gamma_z$, depends b the relative distance it is required to travel, $\Delta x$, and it is computed by

$$\gamma_z = \frac{\omega_0^2}{4\pi}\Delta x \tag{3.14}$$

The total time required to complete the maneuver is one orbital period $T$.

### 3.1.2 Reference Rendezvous and Docking Maneuver

The first step for the definition of the nominal rendezvous and docking maneuver is to set a number of waypoints the chaser spacecraft shall follow to reach the target spacecraft until docking. In the definition of all the waypoints, safety issues shall

| Waypoint ID | Location [m] $[V_{BAR}, H_{BAR}, R_{BAR}]$ | Desciption |
|---|---|---|
| S0 | [-30000, 0, 3000] | Initial Chaser position |
| S1 | [-17137, 0, 3000] *Nominal condition | Beginning of the I maneuver |
| S2 | [-3000, 0, 0] | End of the I maneuver / Beginning of the II maneuver |
| S3 | [-500, 0, 0] | End of the II maneuver / Beginning of the III maneuver |
| S4 | [0, 0, 0] | Target position / Docking |

Table 3.1 Waypoint definition.

be taken into account, ensuring a failure tolerant mission scenario. Hence, five waypoints have been defined, and four maneuvers, or mission phases, have to be completed sequentially. Waypoints definition and location are summarized in Table 3.1. According to the definition of waypoints of Table 3.1, four mission phases have been defined:

1. *S0-S1: Free drift*. This maneuver is used to allow convergence of digital filters and orient the attitude of the chaser along the desired direction. During this maneuver the phase between target and chaser spacecraft is reducing and sensors start tracking the position of the target spacecraft.
2. *S1-S2: Orbit raising*. After reaching waypoint S1, the Chaser starts the maneuver to reach the same orbit of the Target. The maneuver ends reaching station keeping conditions in S2.
3. *S2-S3: Radial $V_{BAR}$ transfer*. Executing this maneuver the Chaser is getting closer to the Target with a radial boost-like maneuver. The maneuver ends with station keeping condition when wapoint S3 has been reached.
4. *S3-S4: Final approach*. This is the last maneuver to dock the Chaser with the Target. It has been selected as a straight line maneuver, in which a conic envelope has been included to support the straight line approach. The maneuver ends with docking terminal conditions. The velocity profile of the maneuver is controlled in order to be constant during the first 450 m, then the spacecraft is slowed down to a velocity compliant to docking requirements.

The complete rendezvous and docking maneuver is depicted in Fig. 3.4. All the

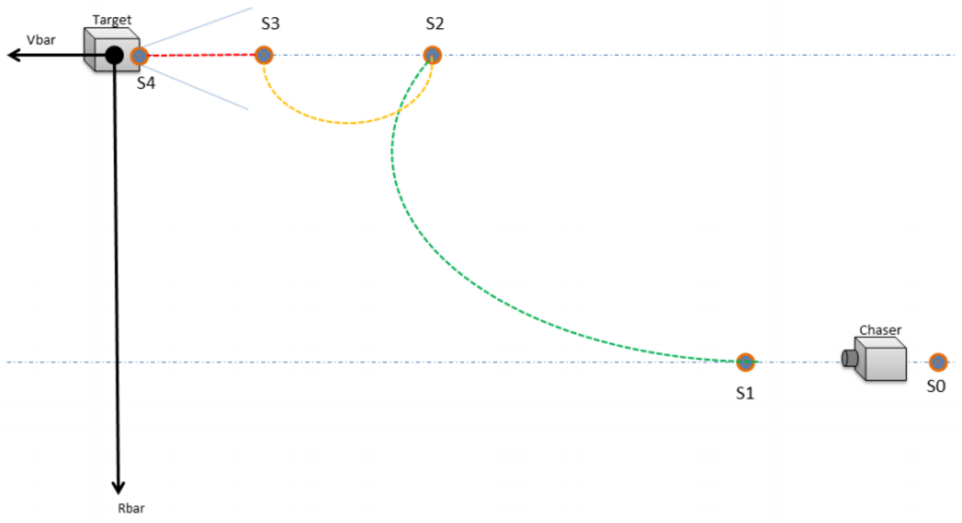


Fig. 3.4 Rendezvous and docking reference maneuver.

maneuvers are intended to be executed as continuous thrust maneuvers, according to the thrust to mass ration of the current configuration of the chaser spacecraft. For this reason, the $V_{BAR}$ position of waypoint $S1$ has been chosen in order to be compliant with the distance traveled along $V_{BAR}$ for a continuous thrust maneuver, as computed in Eq. 3.9. This assumption is valid only for nominal initial condition: indeed, in order to ensure robustness of the guidance law and so to take into account non-nominal initial orbit, the position of waypoint $S1$ is computed continuously during the free drift maneuver, in order to set correctly the position of $S1$ according to the real orbit altitude of the Chaser. Details of such computation have been described in Section 2.2.3
The position of waypoint $S3$ with respect to $S2$ has been selected in such a way that the minimum altitude reached during the maneuver S2-S3 ensure that the target is within the field of view of all active relative sensors during the entire maneuver, assuming the attitude of the Chaser aligned with the local vertical frame.
Differently, the position of the initial waypoint, $S0$, has been selected in order to study a *worst case* scenario in which the on-board computer is switched on the first time at $S0$, and so the GNC sowftware has to be initialized and the digital filters implemented in the navigation function require several time to reach convergence, since the target spacecraft have to be detected at least by the far range relative sensor.

As mentioned before, safety issues has been taken into account during the definition of the reference rendezvous and docking maneuver. The first maneuver,

S1-S2, since it is executed as a continuous thrust, it is inherently failure tolerant: if a double failure of active thrusters occurs, the maneuver cannot be completed and the chaser start drifting in an elliptic orbit that never reach the altitude the target spacecraft. In addition, if the thrusters opposite to the maneuvering couple have not been failed, a *Collision Avoidance Maneuver (CAM)* can be executed and they can be used to stop the motion of the Chaser and place it on a free drift orbit or to an orbit falling through the Earth, avoiding any collision with the Target spacecraft. If single thruster failure occurs, the active thruster must be switched off in order to avoid generation of torque which may start the spacecraft spinning without control, and this condition must be manged as the double thrusters failure.
Similar consideration can be assumed for the maneuver S2-S3: again, if a failure occurs, the spacecraft starts orbiting at a safe distance from the Target, and CAMs can be safely executed using the remaining active thrusters, in a similar way of maneuver S1-S2.
Differently, the last maneuver, S3-S4, does not ensure the passive safety of the previous maneuvers. A failure of thrusters along $R_{BAR}$ direction causes a drift of the spacecraft to higher orbits which ensure passive safety, but safety strictly depends by how much the two spacecraft are close each other: if the failure occurs in the last meters, the Chaser may don't have enough margin to overcome the Target without colliding with it, hence CAMs must be executed, possibly along $H_{BAR}$. An $H_{BAR}$ CAM ensure the Chaser passing over the Target with an orbit with different inclination. A CAM maneuver is mandatory for a failure of thrusters which should slow down the spacecraft, since the *controlled collision* of the docking phases will occurs with an higher speed which cannot be sustained by the docking mechanism, resulting in a catastrophic conclusion.

## 3.2 Simulator Architecture

The orbital simulator has been designed in order to include all mathematical models required to perform an accurate simulation of a spacecraft orbiting the Earth, comprehensive of a complete set of sensors and actuators. Since the design of an orbital simulator is not a trivial task, a system engineering approach has been followed during the design and development of the tool. The first step has been to define a list of requirements the simulator shall satisfy, then it has been defined the functions required to accomplish the specification, the functions have been allocated to functional blocks which define the simulator architecture, and eventually the source code has been developed and all the functions have been tested and validated. More details about the test and validation process will be provided in Appendix A. The high level of definition of the simulator architecture is depicted in Fig. 3.5. The detailed level of decomposition of the functional architecture is defined as follows:

- *Level 0 Functions:* it is the highest level of detail of the system. For this purpose, the highes level of detail is the Simulator itself.
- *Level 1 Functions:* it is the first level of decomposition of the system. For the simulator case study, at this level have been defined all function in charge to simulate specific tasks.
- *Level 2 Functions:* it is the second level of decomposition. Level 1 functions may be detailed in more Level 2 functions. This further partition is useful to define simpler functions that can be tested more easily and accelerate the debug process.
- *Level 3 Functions:* a further partition of Level 2 functions ...

In the present work, the level of detail has been limited to Level 2 function, since a further detail results useless and entails in a more complicated debug and testing process. In the category of Level 3 functions may be included all mathematical operation functions, such as matrices and vector operations, quaternions operations, and more: these functions has been implemented and extensively tested, but they are not included in the definition of the simulator architecture due to the lower level of detail and because they are more related to the implementation issue, rather than the architecture definition.

The Functional Architecture of the simulator is depicted in Fig. 3.6. It has been highlighted the system level of detail. As in Fig. 3.5, the Level 0 function, the Simulator, has been detailed in five main Level 1 functions and each function has been detailed further in Level 2 functions. As already mentioned, the test and validation process starts from this level of detail and moves up until validation of the Level 0 function, the Orbital Simulator. In the following, each function will be extensively described and description of input and output will be provided.

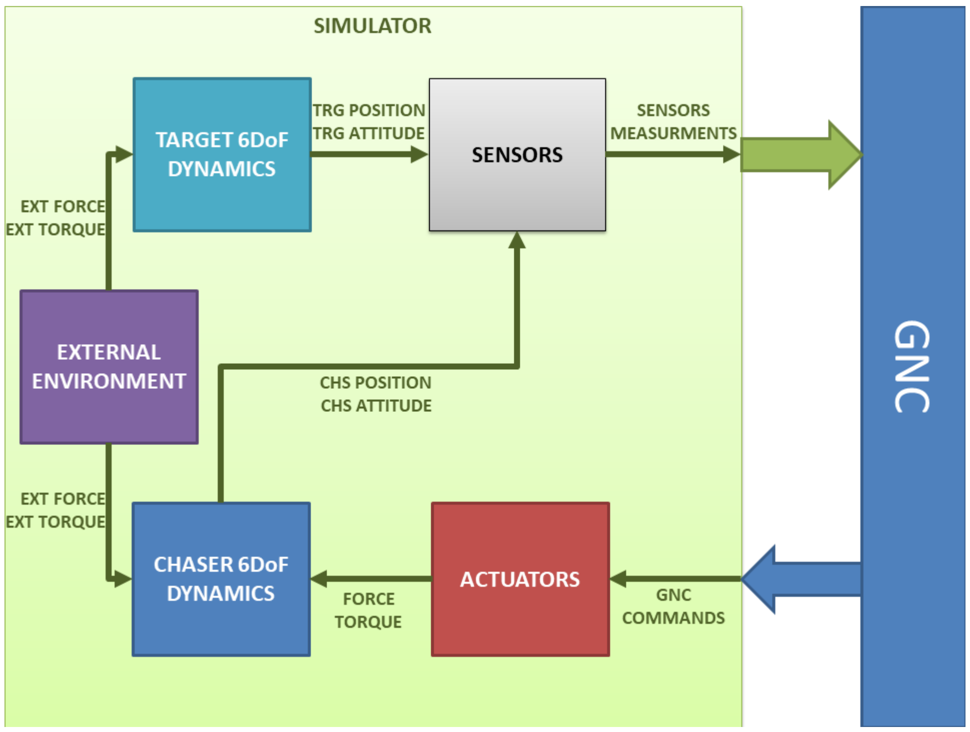


Fig. 3.5 Simulator Architecture.

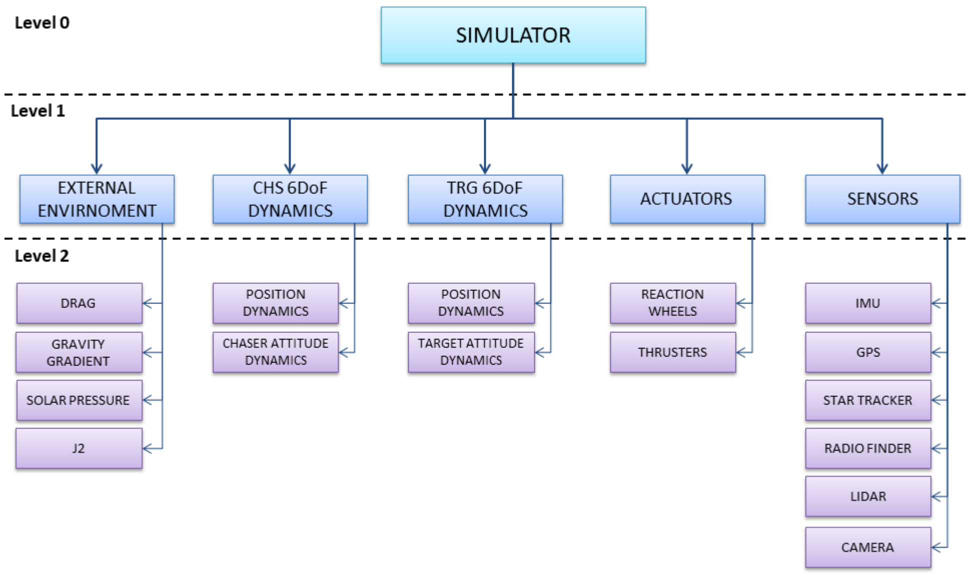


Fig. 3.6 Simulator Functional Architecture.

### 3.2.1 External Environment

The *External Environment* function includes all functions related to disturbances which affect the orbital motion of spacecraft in LEO. Analyzed disturbances are:

- Aerodynamic Drag;
- Gravity Gradient;
- Solar Pressure;
- J2 Effect.

#### Aerodynamic Drag

This disturbance is due to the presence of atmospheric gases at very high altitude which combined with the high orbital velocity of spacecraft produce a non-negligible effect, responsible of orbital decay of orbiting objects. Input and output of the function are summarized in Table 3.2.

| Input | | | | | |
|---|---|---|---|---|---|
| Name | Unit | Size | Type | Frame | Origin |
| Velocity | m/s | 3 | Double | ECI | Postion Dynamics |
| Position | m | 3 | Double | ECI | Position Dynamics |
| Area | $m^2$ | 1 | Double | - | Chaser parameters |
| Drag Coefficient | - | 1 | Double | - | Chaser parameters |
| **Output** | | | | | |
| Name | Unit | Size | Type | Frame | Destination |
| Force | N | 3 | Double | LVLH | Position Dynamics |

Table 3.2 Input and output definition of function *Drag*.

#### Gravity Gradient

The Gravity Gradient disturbance is a torque disturbance which force the spacecraft to align its minimum moment of inertia axis with the local vertical axis. This disturbance is also used as attitude stabilization technique. Input and output are summarized in Table 3.3.

| Input | | | | | |
|---|---|---|---|---|---|
| Name | Unit | Size | Type | Frame | Origin |
| Position | m | 3 | Double | ECI | Position Dynamics |
| DCM$_{BLVLH}$ | - | 3x3 | Double | LVLH to ECI | Attitude Dynamics |
| Inertia Matrix | kgm$^2$ | 3x3 | Double | Body | Chaser parameters |
| **Output** | | | | | |
| Name | Unit | Size | Type | Frame | Destination |
| Torque | Nm | 3 | Double | Body | Attitude Dynamics |

Table 3.3 Input and output definition of function *Gravity Gradient*.

#### Solar Pressure

The Solar Pressure torque is a disturbance acting on the spacecraft due to the solar radiation pressure. Since the origin of this disturbance is the Sun, this disturbance is not affecting the spacecraft attitude while it is in the eclipse of the Earth. Input and output are summarized in Table 3.4.

| Input | | | | | |
|---|---|---|---|---|---|
| Name | Unit | Size | Type | Frame | Origin |
| Position | m | 3 | Double | ECI | Position Dynamics |
| ECI to Body DCM | - | 3x3 | Double | ECI to Body | Main |
| True Anomaly | rad | 1 | Double | OP | Main |
| **Output** | | | | | |
| Name | Unit | Size | Type | Frame | Destination |
| Torque | Nm | 3 | Double | Body | Attitude Dynamics |

Table 3.4 Input and output definition of function *Solar Wind*.

#### J2 Effect

The J2 Effect is a disturbance force caused by the non-spherical shape of the Earth, which generates perturbation in the gravitational potential of the planet, which is not perfectly spherical. Input and output are summarized in Table 3.5.

| Input | | | | | |
|---|---|---|---|---|---|
| Name | Unit | Size | Type | Frame | Origin |
| Position | m | 3 | Double | ECI | Position Dynamics |
| Inclination | rad | 1 | Double | ECI | Initialization |
| RAAN | rad | 1 | Double | ECI | Initialization |
| Output | | | | | |
| Name | Unit | Size | Type | Frame | Destination |
| Force | N | 3 | Double | Body | Position Dynamics |

Table 3.5 Input and output definition of function *J2*.

### 3.2.2 Chaser 6 DoF Dynamics

The Chaser 6 degrees-of-freedom dynamics is the function which propagates the rigid body orbital dynamics of the Chaser spacecraft. It is mainly composed by two level 2 functions: *Position Dynamics* and *Chaser Attitude Dyamics*.

#### Position Dynamics

The *Position Dynamics* function propagates the position dynamics of the spacecraft. It is based on the *Hill's equations* of relative dynamics. Input and output of the function are summarized in Table 3.6.

| Input | | | | | |
|---|---|---|---|---|---|
| Name | Unit | Size | Type | Frame | Origin |
| Velocity | m/s | 3 | Double | Hill | Position Dynamics |
| Position | m | 3 | Double | Hill | Position Dynamics |
| Force | N | 3 | Double | Hill | Spacecraft Actuators<br>External Disturbances |
| Mass | kg | 1 | Double | - | Spacecraft Configuration |
| Angular rate | rad/s | 1 | Double | - | Orbit Parameters |
| Output | | | | | |
| Name | Unit | Size | Type | Frame | Destination |
| Acceleration | $m/s^2$ | 3 | Double | Hill | IMU |
| Velocity | m/s | 3 | Double | Hill | Position Dynamics |
| Position | m | 3 | Double | Hill | Position Dynamics<br>Sensors |

Table 3.6 Input and output definition of function *Position Dynamics*.

### Chaser Attitude Dynamics

The Chaser attitude dynamics is based on the classical rigid body equation formulation. The attitude is computed following the quaternion notation. Input and output of the function are summarized in Table 3.7.

| Input | | | | | |
|---|---|---|---|---|---|
| Name | Unit | Size | Type | Frame | Origin |
| Angular rate | rad/s | 3 | Double | Body | CHS Attitude Dyn. |
| Quaternion | - | 4 | Double | Body | CHS Attitude Dyn. |
| External torque | Nm | 3 | Double | Body | External Disturbances |
| Thrust torque | Nm | 3 | Double | Body | Actuators |
| RWs torque | Nm | 3 | Double | Body | Actuators |
| Angular momentum | Nms | 3 | Double | Body | Actuators |
| Inertia | kgm$^4$ | 3 | Double | Body | Spacecraft Config. |

| Output | | | | | |
|---|---|---|---|---|---|
| Name | Unit | Size | Type | Frame | Destination |
| Angular acceleration | rad/s$^2$ | 3 | Double | Body | Sensors |
| Angular rate | rad/s | 3 | Double | Body | CHS Attitude Dyn.<br>Sensors |
| Quaternion | - | 4 | Double | Body | CHS Attitude Dyn.<br>Sensors |

Table 3.7 Input and output definition of function *Chaser Attitude Dynamics*.

## 3.2.3 Target 6 DoF Dynamics

As for the Chaserr 6 DoF dynamics, the Target complete dynamics is composed by both position and attitude dynamics.

### Target Position Dynamics

The Target position dynamics, as for the Chaser position dynamics is based on propagation of relative dynamics equations. Due to this high similarity, it has been used the same function described in Section 3.2.2. Table of input and output of the Target Position Dynamics is the same as Table 3.6.

### Target Attitude Dynamics

Also for the Target spacecraft it has been required to compute its attitude dynamics. Since the Target spacecraft has been supposed to be an active spacecraft able to control its attitude, the attitude dynamics function differs form the Chaser attitude dynamics: instead of propagating rigid body equations, the Target attitude dynamics is computed by imposing a three-axial oscillatory motion. Input and output of the function are summarized in Table 3.8.

| Input | | | | | |
|---|---|---|---|---|---|
| Name | Unit | Size | Type | Frame | Origin |
| Quaternion | - | 4 | Double | Body | TRG Attitude Dynamics |
| Amplitude | rad | 3 | Double | - | Spacecraft Configuration |
| Frequency | rad/s | 3 | Double | - | Spacecraft Configuration |
| Output | | | | | |
| Name | Unit | Size | Type | Frame | Destination |
| Quaternion | - | 4 | Double | Body | TRG Attitude Dynamics<br>Sensors |

Table 3.8 Input and output definition of function *Target Attitude Dynamics*.

## 3.2.4 Actuators

The function Actuators includes functions related to actuators installed on the Chaser spacecraft. In the current Chaser configuration, it has been included two different type of actuators: *Reaction Wheels* and *Thrusters*.

### Reaction Wheels

The function Reaction Wheels is in charge to produce the output in Body frame of the reaction wheels system, which can be composed by many reaction wheels, each of them producing a mono-axial torque. It is the main actuation system able to control the attitude of the Chaser spacecraft. Input and output are summarized in Table 3.9.

| Input | | | | | |
|---|---|---|---|---|---|
| Name | Unit | Size | Type | Frame | Origin |
| RWs Command | Nm | 3 | Double | Body | GNC.CON |
| RWs Rates | rad/s | 3 | Double | Body | Reaction Wheels |
| Output | | | | | |
| Name | Unit | Size | Type | Frame | Destination |
| RWs Torque | Nm | 3 | Double | Body | CHS Attitude Dynamics |

Table 3.9 Input and output definition of function *Reaction Wheels*.

**Thrusters**

The function Thruster describes the output of the thrusters system (RCS) of the spacecraft. This actuation system is the main actuation system able to perform orbital RVD maneuvers and fine position and velocity control during the docking phase, and it is composed by a set of many thrusters, each of them producing a mono-axial thrust force. Input and output are summarized in Table 3.10.

| Input | | | | | |
|---|---|---|---|---|---|
| Name | Unit | Size | Type | Frame | Origin |
| FCV Command | - | 12 | Boolean | - | GNC.CON |
| CoG Location | ms | 3 | Double | Body | Spacecraft Configuration |
| Output | | | | | |
| Name | Unit | Size | Type | Frame | Destination |
| Thrust Force | N | 3 | Double | Body | CHS Position Dynamics |
| Thrust Torque | Nm | 3 | Double | Body | CHS Attitude Dynamics |

Table 3.10 Input and output definition of function *Thrusters*.

### 3.2.5 Sensors

A complete set of sensor is described by the Sensors function. In the current Chaser configuration, a set of three absolute sensors and three relative sensors has been considered.

### IMU

The IMU (Inertial Measurement Unit) function is describing the absolute sensor able to measure acceleration and angular rates of the spacecraft. Input and output are summarized in Table 3.11.

| Input | | | | | |
|---|---|---|---|---|---|
| Name | Unit | Size | Type | Frame | Origin |
| Acceleration | m/s$^2$ | 3 | Double | Hill | Position Dynamics |
| Angular rate | rad/s | 3 | Double | Body | CHS Attitude Dyn. |
| Angular acceleration | rad/s$^2$ | 3 | Double | Body | CHS Attitude Dyn. |
| CoG Location | m | 3 | Double | Body | Spacecraft Config. |
| **Output** | | | | | |
| Name | Unit | Size | Type | Frame | Destination |
| Acceleration | m/s$^2$ | 3 | Float | Body | GNC.NAV |
| Angular rate | rad/s | 3 | Float | Body | GNC.NAV |
| Status | - | 1 | Integer | - | GNC.NAV |
| Counter | - | 1 | Interger | - | GNC.NAV |

Table 3.11 Input and output definition of function *IMU*.

### GPS

The GPS (Global Positioning System) function describes the GNSS (Global Navigation Satellite System) sensor which measures absolute position and velocity of the spacecraft with respect to the ECEF frame. This sensor is also used to compute the local vertical vector by processing position and velocity data. This is also a precise time sensor, thanks to the atomic clocks present in the GPS constellation. Input and output are summarized in Table 3.12.

### Star Tracker

The Star Tracker function describes the Star Tracker sensor, which measure absolute attitude of the spacecraft with respect to an ECI frame. Input and output are summarized in Table 3.13.

| Input | | | | | |
|---|---|---|---|---|---|
| Name | Unit | Size | Type | Frame | Origin |
| Position | m | 3 | Double | ECI | Position Dynamics |
| Velocity | m/s | 3 | Double | ECI | Position Dynamics |
| Rotation Matrix | - | 3x3 | Double | ECI to ECEF | Main |
| Time | s | 1 | Double | - | Main |
| **Output** | | | | | |
| Name | Unit | Size | Type | Frame | Destination |
| Position | m | 3 | Double | ECEF | GNC.NAV |
| Velocity | m/s | 3 | Double | ECEF | GNC.NAV |
| Time of Week | - | 1 | Double | - | GNC.NAV |
| Week | - | 1 | Integer | - | GNC.NAV |
| Leap Seconds | - | 1 | Integer | - | GNC.NAV |

Table 3.12 Input and output definition of function *GPS*.

| Input | | | | | |
|---|---|---|---|---|---|
| Name | Unit | Size | Type | Frame | Origin |
| Quaternion | - | 4 | Double | Body | CHS Attitude Dyn. |
| Angular rate | rad/s | 4 | Double | Body | CHS Attitude Dyn. |
| **Output** | | | | | |
| Name | Unit | Size | Type | Frame | Destination |
| Quaternion | - | 4 | Double | Body | GNC.NAV |
| Status | - | 1 | Integer | - | GNC.NAV |
| Counter | - | 1 | Integer | - | GNC.NAV |

Table 3.13 Input and output definition of function *Star Tracker*.

**Radio Finder**

The function Radio Finder describes the relative sensor used for long range target detection. It is basically a radar with good performance in range measurement but with low precision for both azimuth and elevation detection. Input and output are summarized in Table 3.14.

| Input | | | | | |
|---|---|---|---|---|---|
| Name | Unit | Size | Type | Frame | Origin |
| Position CHS | m | 3 | Double | Hill | Position Dynamics |
| Position TRG | m | 3 | Double | Hill | Position Dynamics |
| Rotation Matrix | - | 3x3 | Double | LVHL to Body | Main |
| CoG Location | m | 3 | Double | Body | Spacecraft Config. |
| **Output** | | | | | |
| Name | Unit | Size | Type | Frame | Destination |
| Range | m | 1 | Double | Body | GNC.NAV |
| Elevation | rad | 1 | Double | Body | GNC.NAV |
| Azimuth | rad | 1 | Double | Body | GNC.NAV |
| Status | - | 1 | Integer | - | GNC.NAV |
| Counter | - | 1 | Integer | - | GNC.NAV |

Table 3.14 Input and output definition of function *Radio Finder*.

**Lidar**

The function Lidar describes the relative laser sensor used for medium/low range target detection. Unilke the Radio Finder, the Lidar has better performance in azimuth and elevation tracking than range estimation. Table of input and output of the Lidar function is the same as Table 3.14.

**Camera**

The Camera function describes the vision system sensor based on a Camera. This sensor is used for low target tracking and it is able to detect both relative position and relative attitude. Input and output are summarized in Table 3.15.

| Input | | | | | |
|---|---|---|---|---|---|
| Name | Unit | Size | Type | Frame | Origin |
| Position CHS | m | 3 | Double | Hill | Position Dynamics |
| Position TRG | m | 3 | Double | Hill | Position Dynamics |
| Quaternion CHS | - | 4 | Double | LVLH | CHS Attitude Dynamics |
| Quaternion TRG | - | 4 | Double | LVLH | TRG Attitude Dynamics |
| CoG Location | m | 3 | Double | Body | Spacecraft Configuration |
| **Output** | | | | | |
| Name | Unit | Size | Type | Frame | Destination |
| Range | m | 1 | Double | Body | GNC.NAV |
| Elevation | rad | 1 | Double | Body | GNC.NAV |
| Azimuth | rad | 1 | Double | Body | GNC.NAV |
| Roll | m | 1 | Double | Body | GNC.NAV |
| Pitch | rad | 1 | Double | Body | GNC.NAV |
| Yaw | rad | 1 | Double | Body | GNC.NAV |
| Status | - | 1 | Integer | - | GNC.NAV |
| Counter | - | 1 | Interger | - | GNC.NAV |

Table 3.15 Input and output definition of function *Camera*.

## 3.3 Mathematical Models

In this section, all the mathematical models implemented in the development of the orbital simulator will be extensively described. In addition to models of orbital dynamics, sensors, actuators and external disturbances, it will be briefly introduced all the reference frames used in the orbital simulator.

### 3.3.1 Reference Frames

In this section, a brief description about coordinate frames used in the present dissertation to describe the orbital motion will be provided. Reference frames used to design the orbital simulator are:

- Earth-Centered Inertial frame - ECI;
- Earth-Centered Earth-Fixed frame - ECEF;
- Orbital Plane frame - OP;
- Local Orbital Local Horizontal frame - LVLH;

- Spececraft Body Geometrical frame - BG;
- Spacecraft Body frame - B;

### Earth-Centered Inertial Frame

The Earth-Centered Inertial frame, $F_{ECI}$, is used to describe the orbital motion of an object with respect to the center of the Earth. In this frame, the Earth is considered perfectly spherical, so the origin of the frame, $O_{ECI}$, is located in the center of the Earth. The $x$ axis of the ECI frame is located in the equatorial plane and pointing toward the vernal equinox ($\gamma$ in Fig. 3.7). The $z$ axis is pointing the geographic North pole and the $y$ axis lies in the equatorial plane in order to complete the right-handed frame, such that $z = x \times y$. Note that this frame is not truly inertial: the Earth is orbiting the Sun and it is subjected to nutation and precession motions, so the reference frame is not inertial at all, but for application such as small object orbiting the Earth the assumption of inertial frame can be assumed.

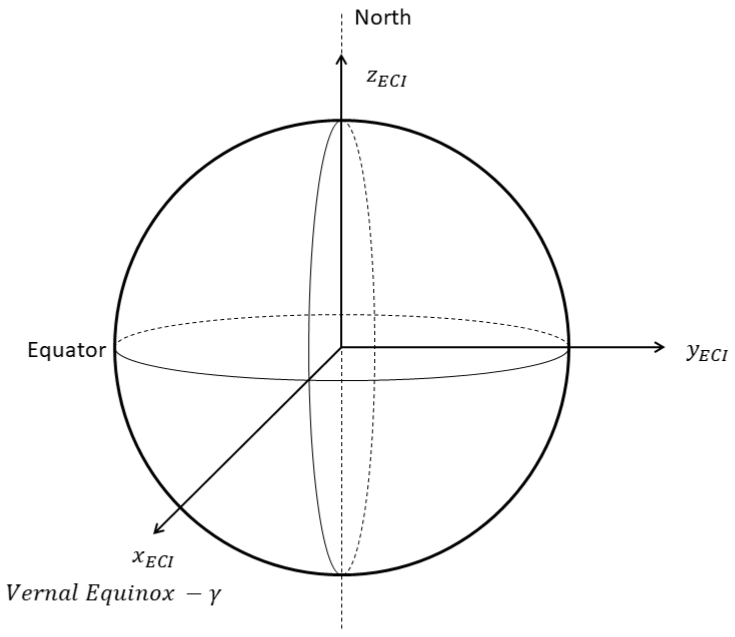


Fig. 3.7 ECI frame.

### Earth-Centered Earth-Fixed Frame

The Earth-Centered Earth-Fixed frame, $F_{ECEF}$, is used to describe the orbital motion of an object with respect to the center of the Earth, such as in the ECI frame, but with axes fixed to the surface of the Earth. In other words, the ECEF frame corresponds to an ECI frame rotating about its $z$ axis with angular velocity equal to the duration of the solar day, then the orgin $O_{ECEF}$ of $F_{ECEF}$ is located in the center of the Earth too. This frame is used to depict the ground track of satellites and to compute the line-of-sight vector from a ground station to an orbiting object. Then, the $z$ axis of $F_{ECEF}$ is coincident with the $z$ axis of $F_{ECI}$, while $x$ and $y$ axes are still in the equatorial plane, whith a phase of 90 deg between them, but there is an angle $\Theta$ between $x_{ECEF}$ and $x_{ECI}$. The angle $\Theta$ is computed as:

$$\Theta = \Theta_0 + \Omega_{EARTH}(t - t_{0\,J200}) \tag{3.15}$$

where $\Theta_0$ is the initial phase angle computed with reference to the epoch *J2000*, $\Omega_{EARTH}$ is the angular velocity of the Earth about its rotation axis, $t$ is the actual elapsed time, $t_{0\,J200}$ is the inital time for which $\Theta_0$ has been computed. The rotation matrix from $F_{ECI}$ to $F_{ECEF}$ is:

$$DCM_{ECEF\;ECI} = \begin{bmatrix} \cos\Theta & \sin\Theta & 0 \\ -\sin\Theta & \cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.16}$$

The transformation of a coordinate system from ECI to ECEF is computed as:

$$\begin{Bmatrix} x_{ECEF} \\ y_{ECEF} \\ z_{ECEF} \end{Bmatrix} = DCM_{ECEF\;ECI} \cdot \begin{Bmatrix} x_{ECI} \\ y_{ECI} \\ z_{ECI} \end{Bmatrix} = \begin{bmatrix} \cos\Theta & \sin\Theta & 0 \\ -\sin\Theta & \cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{Bmatrix} x_{ECI} \\ y_{ECI} \\ z_{ECI} \end{Bmatrix} \tag{3.17}$$

The ECEF frame have been used to compute position and velocity data provided by the GPS sensor.

### Orbital Plane Frame

The Orbital Plane frame, $F_{OP}$, is used to describe the motion of an object within the orbital plane. Due to this assumption, a characteristic of $F_{OP}$ is that its $z_{OP}$

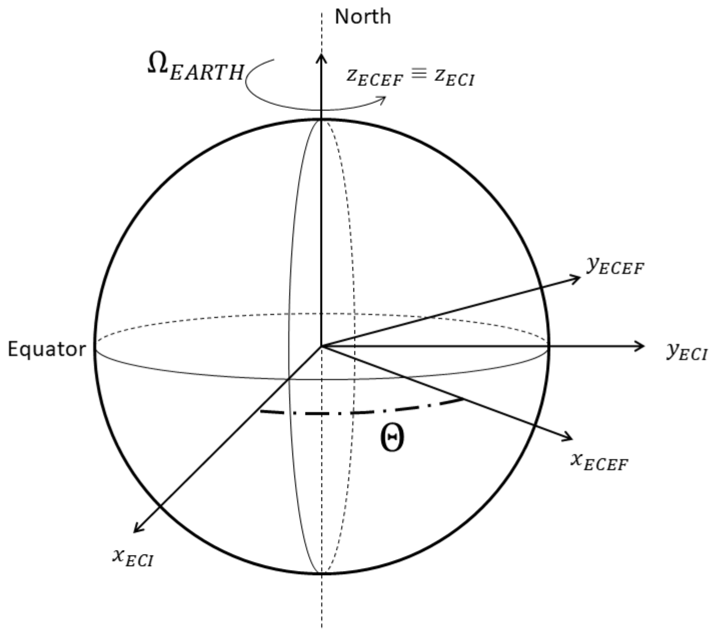


Fig. 3.8 ECEF frame.

coordinates is always zero, since the orbiting object, by its own motion, is defining the frame. The orgin $O_{OP}$ of $F_{OP}$ is located in the center of the Earth. The $x_{OP}$ axis is located in the orbital plane and points the ascending node, $z_{OP}$ is normal to the orbital plane and inclined with respect to the North direction ($z_{ECI}$) of an angle $i$, and $y_{OP}$ lies in the orbital plane such that $z_{OP} = x_{OP} \times y_{OP}$. The transformation between $F_{ECI}$ and $F_{OP}$ is obtained by a rotation about $z_{ECI}$ of an angle $\Omega$ and a rotation of an angle $i$ about $x_{OP}$ yet computed:

$$\begin{Bmatrix} x_{OP} \\ y_{OP} \\ z_{OP} \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos i & \sin i \\ 0 & -\sin i & \cos i \end{bmatrix} \cdot \begin{bmatrix} \cos\Omega & \sin\Omega & 0 \\ -\sin\Omega & \cos\Omega & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{Bmatrix} x_{ECI} \\ y_{ECI} \\ z_{ECI} \end{Bmatrix} \tag{3.18}$$

The two angles in Eq. 3.18 are two orbital parameters, respectively $i$ is the inclination angle and $\Omega$ is the Right Ascesion of the Ascending Node (RAAN), the angle between

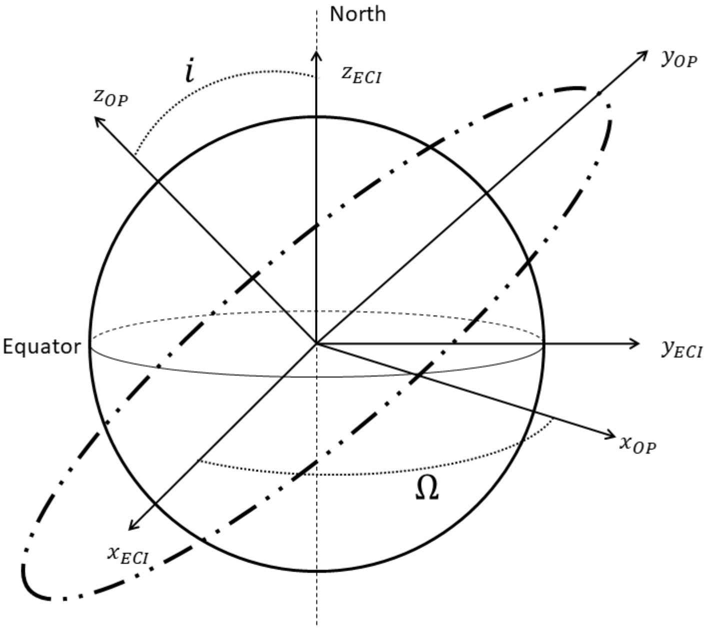


Fig. 3.9 Orbital Plane frame.

$x_{ECI}$ and $x_{OP}$. The rotation matrix $DCM_{OP\,ECI}$ is then the product of two matrices:

$$DCM_{OP\,ECI} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos i & \sin i \\ 0 & -\sin i & \cos i \end{bmatrix} \cdot \begin{bmatrix} \cos\Omega & \sin\Omega & 0 \\ -\sin\Omega & \cos\Omega & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.19}$$

### Local Orbital Local Horizontal Frame

The Local Orbital Local Horizontal frame, $F_{LVLH}$, is used to describe the motion of the spacecraft with respect to the orbital velocity and the direction towards the center of the Earth. This frame is mainly used to compute the local vertical of the spacecraft and so to compute its attitude in order to re-orient or maintain the required pointing of the spacecraft. The origin $O_{LVLH}$ of $F_{LVLH}$ is located in the center of mass of the spacecraft. The $x_{LVLH}$ axis is aligned such that $x_{LVLH} = y_{LVLH} \times z_{LVLH}$, approximately pointing the direction of the orbital velocity but not necessarily aligned with it (this is true only for circular orbits). The $y_{LVLH}$ axis is pointing the opposite direction of the angular momentum vector (then opposite to $z_{OP}$). The $z_{LVLH}$ axis is

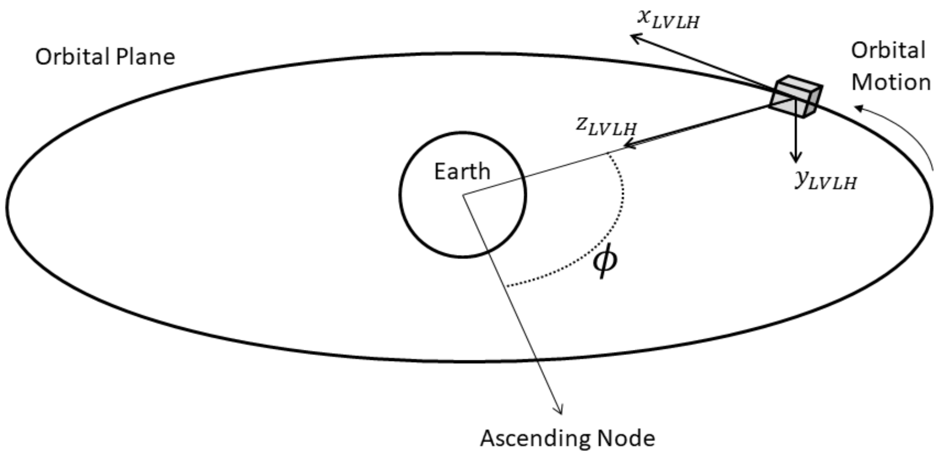


Fig. 3.10 LVLH frame.

pointing towards the radial direction, aligned with the vector connecting the center of mass of the spaceraft to the center of the Earth. The trasformation from $F_{OP}$ to $F_{LVLH}$ can be obtained with three subsequent rotations: the first about $z_{OP}$ by the orbital phase angle $\phi$, also named true anomaly; the second rotation is of 90 deg and it is needed to put $x_{LVLH}$ in the orbital velocity directoin; the third rotation is of $-90$ deg and it is needed to put $z_{LVLH}$ towards the center of the Earth. The resulting transformation is:

$$\begin{Bmatrix} x_{LVLH} \\ y_{LVLH} \\ z_{LVLH} \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{Bmatrix} x_{OP} \\ y_{OP} \\ z_{OP} \end{Bmatrix} \tag{3.20}$$

Considering the case of circular orbits, the true anomaly is computed as:

$$\phi = \omega(t - t_0) \tag{3.21}$$

where $\omega$ is the angular orbital velocity, $t$ is the current elapsed time and $t_0$ is the initial time for which the orbit begins to be computed. Eventually, the rotation matrix between $F_{OP}$ and $F_{LVLF}$ is:

$$DCM_{LVLH\,OP} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.22}$$

### Spacecraft Body Geometric Frame

The Spacecraft Body Geometric frame, $F_{BG}$, is mainly used to define location of center of mass, sensors, actuators, docking mechanism, and all the systems included in the spacecraft configuration with respect to its physical shape. Since the Body Geometric frame usually depends strongly by the definition of the reference frame during the design of a spacecraft, its definition is not univocally assumed. In the present work, as it is common for rendezvous and docking applications, the origin $O_{BG}$ of $F_{BG}$ is located in the center of the docking mechanism. The $x_{BG}$ is pointing the docking axis, while $y_{BG}$ and $z_{BG}$ are located in the plane normal to $x_{BG}$ and they must satisfy $x_{LVLH} = y_{LVLH} \times z_{LVLH}$. Assuming the spacecraft orbiting with the docking axis aligned with the velocity vector, if and astronaut is sitting on it with the legs towards the Earth, $y_{BG}$ results pointing the right-hand side of the astronaut and consequently $z_{BG}$ is pointing the center of the Earth. The spacecraft Body Geometric frame is depicted in Fig. 3.11.

### Spacecraft Body Frame

The Spacecraft Body frame, $F_B$ is used to describe rotations of the spacecraft about its center of mass. The origin $O_B$ of $F_B$ is located in the center of mass of the spacecraft. Since the mass of the spacecraft is usually varying during the mission, mainly due to propellant consumption, also the center of mass is moving within the spacecraft geometry, making $F_B$ not fixed. In addition, the center of mass of a spacecraft is moving also due to propellant sloshing, solar arrays deployment, probe release, movement of appendices such as robotic arms, and more. The definition of the orientation of the axis of $F_B$ is not unique. In the present work, $F_B$ is assumed to be aligned with $F_{BG}$, while it results shifted with respect to $F_{BG}$ by the position of the center of mass, computed in $F_{BG}$ frame as well. The attitude of the spacecraft is evaluated with respect to a second reference frame, usually the Local Horizontal Local Vertical frame. Even though the orientation of $F_B$ with respect to $F_{LVLH}$ is univocally defined, the sequence of rotations required to define the relative attitude is not unique, and the rotation angles are named differently depending by the sequence applied. In the present work, it has been used the sequence 3-2-1, or z-y-x, and angles $\alpha_z$, $\alpha_y$ and $\alpha_x$ are named respectively *Yaw*, *Pitch* and *Roll* angles, following the aeronautical nomenclature. The transformation between $F_B$ and $F_{LVLH}$ is then:

$$\begin{Bmatrix} x_B \\ y_B \\ z_B \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha_x & \sin\alpha_x \\ 0 & -\sin\alpha_x & \cos\alpha_x \end{bmatrix} \cdot \begin{bmatrix} -\sin\alpha_y & 0 & \cos\alpha_y \\ 0 & 1 & 0 \\ \cos\alpha_y & 0 & \sin\alpha_y + \end{bmatrix} \cdot \dots \cdot \begin{bmatrix} \cos\alpha_z & \sin\alpha_z & 0 \\ -\sin\alpha_z & \cos\alpha_z & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{Bmatrix} x_{LVLH} \\ y_{LVLH} \\ z_{LVLH} \end{Bmatrix} \tag{3.23}$$

The resulting rotation matrix is the product of three rotation, such as:

$$DCM_{B\,LVLH} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha_x & \sin\alpha_x \\ 0 & -\sin\alpha_x & \cos\alpha_x \end{bmatrix} \cdot \begin{bmatrix} -\sin\alpha_y & 0 & \cos\alpha_y \\ 0 & 1 & 0 \\ \cos\alpha_y & 0 & \sin\alpha_y + \end{bmatrix} \cdot \dots \cdot \begin{bmatrix} \cos\alpha_z & \sin\alpha_z & 0 \\ -\sin\alpha_z & \cos\alpha_z & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.24}$$

In Fig. 3.11 it has been depicted both $F_{BG}$ and $F_B$ frames. The vector $r_{CoG}$ is the position vector of the center of mass of the spacecraft with respect to $F_{BG}$.

### 3.3.2 Orbital 6 DoF Dynamics

The orbital dynamics developed in the present work includes both position dynamics and attitude dynamics. The position dynamics is based on the relative equations of motion, which are named *Hill's equations* or *Clohessy-Wiltshire equations*, and it is computed in a similar way for both the Chaser and the Target vehicles. The attitude dynamics, instead, it is computed differently for the two vehicles: the Chaser attitude dynamics is computed propagating the classical rigid body equations, while the Target attitude dynamics is computed by imposing a three-axial oscillatory motion with limited amplitude and frequency. This assumption derives by the fact that the orbital rendezvous and docking maneuver taken into account in this work considers a docking maneuver with a Target vehicle which is supposed to be able to control its attitude. Indeed, an example of Target spacecraft suitable for the STRONG scenario could be a telecommunication satellite, which, once operative, it will be able to maintain pointing requirements driven by its own attitude control system.

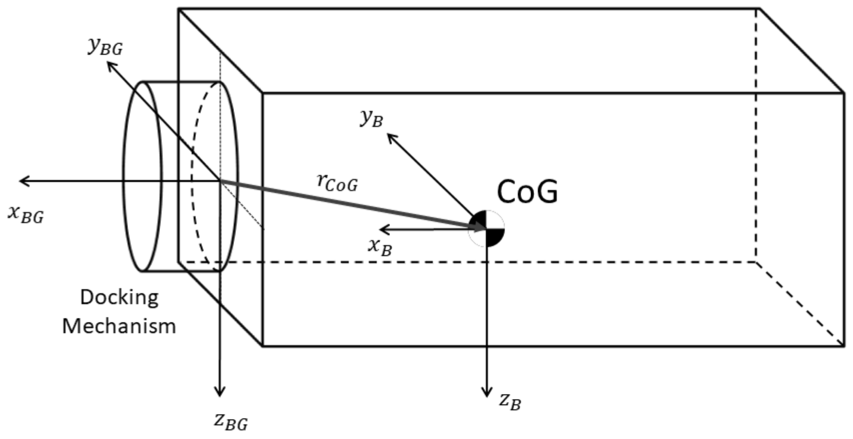


Fig. 3.11 Body Geometric and Body frame.

**Position Dynamics**

The general equation of motion for orbiting objects is based on the Newton's law of gravitation [42]

$$\mathbf{F}_g(\mathbf{r}) = -GM\frac{m}{r^2}\cdot\frac{\mathbf{r}}{r} = -\mu\frac{m}{r^3}\cdot\mathbf{r} \tag{3.25}$$

where $\mathbf{F}_g \in \mathbb{R}^3$ is the gravitational force, $\mathbf{r} \in \mathbb{R}^3$ is the position vector with respect to an inertial frame, $r$ is the magnitude of the vector $\mathbf{r}$, $G$ is the gravitational universal constant, $M$ is the mass of the central body, i.e. the Earth, $m$ is the mass of the orbiting object and $\mu = GM$ is the gravitational parameter of the Earth. To characterize the forces per unit mass acting on both chaser and target spacecraft can be used

$$\begin{cases} \ddot{\mathbf{r}}_c = \frac{\mathbf{F}_g}{m_c} + \frac{\mathbf{F}}{m_c} = \mathbf{f}_g(\mathbf{r}_c) + \frac{\mathbf{F}}{m_c} = -\frac{\mu}{r_c^3}\mathbf{r}_c + \frac{\mathbf{F}}{m_c} \\ \ddot{\mathbf{r}}_t = \frac{\mathbf{F}_g}{m_t} = \mathbf{f}_g(\mathbf{r}_t) = -\frac{\mu}{r_t^3}\mathbf{r}_t \end{cases} \tag{3.26}$$

where $\ddot{\mathbf{r}}_c$ and $\ddot{\mathbf{r}}_t$ are respectively the chaser and target acceleration and $\mathbf{F}$ is the vector of non-gravitational forces acting on the chaser spacecraft, since is it assumed that non-gravitational forces does not affect the target dynamics. Assuming the relative position between the chaser and target spacecraft as

$$\mathbf{s} = \mathbf{r}_c - \mathbf{r}_t$$

and differentiating twice with respect to time it is obtained

$$\ddot{\mathbf{s}} = \ddot{\mathbf{r}}_c - \ddot{\mathbf{r}}_t = \mathbf{f}_g(\mathbf{r}_c) - \mathbf{f}_g(\mathbf{r}_t) + \frac{\mathbf{F}}{m_c} \tag{3.27}$$

Since the goal is to find equations for relative dynamics, it is possible to linearize the vector $\mathbf{f}_g(\mathbf{r}_c)$ in the neighborhood of $\mathbf{r}_t$ using a first order Taylor expansion

$$\mathbf{f}_g(\mathbf{r}_c) = \mathbf{f}_g(\mathbf{r}_t) + \left(\frac{\mathrm{d}\mathbf{f}_g(\mathbf{r})}{\mathrm{d}\mathbf{r}}\right)_{\mathbf{r}=\mathbf{r}_t} (\mathbf{r}_c - \mathbf{r}_t) \tag{3.28}$$

where $\mathrm{d}\mathbf{f}_g(\mathbf{r})/\mathrm{d}\mathbf{r}$ is the *Jacobian* matrix. Neglecting intermediate steps (the reader can be found the complete theory in Appendix A of [42]), it is found

$$\frac{\partial f_g(r_i)}{\partial r_i} = -\frac{\mu}{r^3}\left(1 - \frac{r_i^2}{r^2}\right) \text{ if } i = j \tag{3.29}$$

$$\frac{\partial f_g(r_i)}{\partial r_i} = -\frac{\mu}{r^3}\left(1 - \frac{r_i r_j}{r^2}\right) \text{ if } i \neq j \tag{3.30}$$

where $i$, $j$ assume the value $x$, $y$ and $z$, which are the Cartesian coordinates. Hence, Eq. 3.27 can be rewritten as

$$\ddot{\mathbf{s}} = -\frac{\mu}{r_t^3}\mathbf{M}\mathbf{s} + \frac{\mathbf{F}}{m_c} = -\frac{\mu}{r_t^3}\begin{bmatrix} 1 - 3\frac{r_x^2}{r_t^2} & 3\frac{r_x r_y}{r_t^2} & 3\frac{r_x r_z}{r_t^2} \\ 3\frac{r_y r_x}{r_t^2} & 1 - 3\frac{r_y^2}{r_t^2} & 3\frac{r_y r_z}{r_t^2} \\ 3\frac{r_z r_x}{r_t^2} & 3\frac{r_z r_y}{r_t^2} & 1 - 3\frac{r_z^2}{r_t^2} \end{bmatrix}\mathbf{s} + \frac{\mathbf{F}}{m_c} \tag{3.31}$$

Since $\mathbf{s}$, and consequently $\ddot{\mathbf{s}}$, is defined in the inertial frame, it is possible to compute $\mathbf{s}^*$ in the rotating frame centered in the target spacecraft as

$$\ddot{\mathbf{s}}^* + \omega \times (\omega \times \mathbf{s}^*) + 2\omega \times \dot{\mathbf{s}}^* + \dot{\omega} \times \mathbf{s}^* + \frac{\mu}{r_t^3}\mathbf{M}\mathbf{s} = \frac{\mathbf{F}}{m_c} \tag{3.32}$$

assuming $\mathbf{s}^* = [x, y, z]^T$, $\mathbf{r}_t = [0, 0, -r]^T$, $\omega = [0, -\omega_0, 0]^T$ the angular velocity of the target expressed in the local rotating frame and $\dot{\omega} = 0$ since the the target orbit is assumed to be circular and hence the angular velocity of the target spacecraft orbiting the Earth can be written as

$$\omega_0 = \sqrt{\frac{\mu}{r^3}} \tag{3.33}$$

By completing all the calculations it is found the relative dynamics of $\ddot{\mathbf{s}}^*$, of which the resulting equations are

$$\begin{aligned} \ddot{x} &= 2\omega_0 \dot{z} + \frac{F_x}{m_c} \\ \ddot{y} &= \omega_0^2 y + \frac{F_y}{m_c} \\ \ddot{z} &= -2\omega_0 \dot{x} + 3\omega_0^2 z + \frac{F_z}{m_c} \end{aligned} \tag{3.34}$$

which are also called *Hill's equations*. In Eq. 3.34 the terms $\ddot{x}$, $\ddot{y}$ and $\ddot{z}$ are the relative accelerations, $\omega_0$ is the orbital angular velocity computed as in Eq. 3.33, $F_x$, $F_y$, and $F_z$ are the three components of the total force acting on the spacecraft expressed in Hill frame, and $m_c$ is the mass of the spacecraft. Time integration of $\ddot{x}$, $\ddot{y}$ and $\ddot{z}$ allows computation of relative velocities $\dot{x}$, $\dot{y}$ and $\dot{z}$ and relative positions $x$, $y$ and $z$. The total force acting on the spacecraft and affecting its dynamics includes forces due to external disturbances and control forces provided by the RCS. These last forces allows orbital maneuvering of the Chaser vehicle.
By time-integrating Eq. 3.34 it is computed the 6 degrees-of-freedom dynamics of an object with respect to a local reference frame orbiting the Earth defined in a similar way of the LVLH frame, named *Hill's frame*.

Since the computed relative dynamics is an approximated solution of the orbital motion, its accuracy is reducing if the relative position of the spacecraft with respect to the reference Hill's frame is increasing. A more accurate solution of the relative orbital motion of the chaser spacecraft with respect to the reference Hill's frame can be derived by propagating the inertial equation of motion Eq. 3.25 and compute the relative distance as

$$r_{rel_{ECI}} = r_{CHS_{ECI}} - r_{HF_{ECI}} \tag{3.35}$$

where both $r_{CHS_{ECI}}$ is the chaser position in ECI frame and $r_{HF_{ECI}}$ is the position of the reference Hill's frame in ECI coordinates. The relative postion in Hill's coordinates can be easily computed as

$$r_{rel_{hill}} = DCM_{HILL,ECI} \cdot r_{rel_{ECI}} \tag{3.36}$$

where $DCM_{HILL,ECI}$ is the rotation matrix from ECI frame to the local vertical frame centered in the Hill's frame. This solution is the same as the solution computed by propagating Hill's equations 3.34 except for the error due to not using curvilinear coordinates. The relative velocity in ECI frame is easily computed by

$$v_{rel_{ECI}} = v_{CHS_{ECI}} - v_{HF_{ECI}} \tag{3.37}$$

and it can be rewritten as

$$v_{rel_{ECI}} = v_{rel_{HILL}} + \omega \times r_{rel} \tag{3.38}$$

and then

$$v_{rel_{HILL}} = v_{rel_{ECI}} - \omega \times r_{rel} \tag{3.39}$$

where term $\omega \times r_{rel}$ is the transport velocity of the rotating Hill's frame with respect to the inertial frame, and $v_{rel_{HILL}}$ is the velocity of the chaser spacecraft with respect to the Hill's frame. The same procedure may be applied to the target spacecraft to compute relative postion and velocity with respect to the reference Hill's frame by propagating the inertial position dynamics.

### Attitude Dynamics

The attitude dynamics is defined with respect to the inertial ECI frame. Basic equation for attitude dynamics is

$$\dot{\omega} = \mathbf{J}^{-1}\left(\mathbf{M}_{ext} + \mathbf{M}_{thr} - \dot{\mathbf{H}}_{rws} - \omega \times (\mathbf{J}\omega + \mathbf{H}_{rws})\right) \tag{3.40}$$

where $\mathbf{J} \in \mathbb{R}^{3,3}$ is the inertia matrix of the spacecraft, $\mathbf{M}_{ext} \in \mathbb{R}^3$ is the external torque due to orbital disturbances, $\mathbf{M}_{thr} \in \mathbb{R}^3$ is torque due to the thrusters system, $\dot{\mathbf{H}}_{rws} \in \mathbb{R}^3$ is the torque provided by the reaction wheels assembly, $\mathbf{H}_{rws} \in \mathbb{R}^3$ is the angular momentum of the reaction wheels assembly and $\omega = [\omega_x, \omega_y, \omega_z]^T$ is the angular velocity of the spacecraft expressed in body frame. By time integration of Eq. 3.40 it is obtained the angular velocity of the spacecraft $\omega$.

The attitude is expressed in *quaternion* form, in order to avoid singularities typical of computing kinematic equation in terms of *Euler's angles*. The quaternion notation considered in the present dissertation considers the scalar component of the quaternion as the first element $q_0$ such as

$$\mathbf{q} = [q_0, q_1, q_2, q_3]^T$$

where $q_1$, $q_2$ and $q_3$ are the vectorial components of the quaternion. Detailed discussion about quaternion mathematics is left to the reader, for example studying [59]. Hence, kinematic equations in quaternion form are expressed by

$$\dot{\mathbf{q}} = \frac{1}{2}\tilde{\Omega} \cdot \mathbf{q} = \frac{1}{2}\begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{bmatrix} \cdot \mathbf{q} \tag{3.41}$$

### 3.3.3 External Disturbances

As described in Section 3.2.1, four different external disturbances affecting the LEO environment have been analyzed and modeled.

#### Atmospheric Drag

The atmospheric drag disturbance found in Low Earth Orbit environment is due to residual traces of atmospheric gases at high altitude. Above about 100 km from the Earth surfaces, the atmospheric density decrease dramatically and the atmosphere cannot be considered as a continuous medium anymore. Indeed, the drag force acting on an orbiting object is due to gas particles impacting the object, exchanging momentum and then reducing the speed of the object, in a similar way compared to the aeronautical drag force but with a different principle. Since collisions with gas particles progressively reduce the orbital velocity, the orbital equilibrium is then affected and the slowing spacecraft starts decreasing is altitude, literally falling into the Earth. The drag force is the main cause of orbital decay of spacecraft orbiting lower orbits, which often require corrections in order to stabilize the orbit, such as the International Space Station (ISS) which requires regular boost maneuvers to rise its orbit, realized using servicing modules docked with it (ATV, Soyuz, Progress,

...). The formulation of the atmospheric drag is similar to the classical aeronautical formulation:

$$F_{DRAG} = -\frac{1}{2}\rho V^2 S C_D \tag{3.42}$$

where $\rho$ is the residual atmospheric density, $V$ is the linear orbital velocity, $S$ is the frontal area of the spacecraft and $C_D$ is the drag coefficient of the spacecraft, commonly set in 2.2. The density $\rho$ is computed as

$$\rho = \exp(f_x) \tag{3.43}$$

where $f_x$ is a polynomial formulation used to model the atmospheric density within an altitude interval of 100 to 1000 km above the Earth surface, taking into account the altitude from the surface of the Earth $h$, expressed in [m]:

$$f_x = p_1 h^5 + p_2 h^4 + p_3 h^3 + p_4 h^2 + p_5 h + p_6 \tag{3.44}$$

and the polynomial coefficients $p_i$ are:

$$\begin{array}{lll} p_1 = -3.817 \cdot 10^{-28} & p_2 = 1.187 \cdot 10^{-21} & p_3 = -1.403 \cdot 10^{-15} \\ p_4 = 7.929 \cdot 10^{-10} & p_5 = -2.301 \cdot 10^{-4} & p_6 = 1.844 \end{array}$$

The *minus* sign in Eq. 3.42 is due to the fact that the drag force is opposite to the orbital motion. The drag force is expressed in LVLH frame. Only the $x$ component of the force is non-null. The atmospheric density modeled by Eqs. 3.43 and 3.44 is related to the polynomial approximation of the *US76* atmospheric model. The polynomial formulation of Eq. 3.44 is a very good approximation of the US76 atmospheric model [60].

### J2 Effect

The J2 effect is the result of orbital perturbations due to the non-uniform geopotential of the Earth. This is a perturbation of the Keplerian orbit defined by Eq. 3.25 and it is an additive terms that affects orbital parameters, causing their changing over time. According to [61], equation of position dynamics Eq. 3.31 should include the term $\mathbf{J}_2(\mathbf{r}) + \nabla\mathbf{J}_2(\mathbf{r}) \cdot \mathbf{r}$. The relative distance during the whole RVD maneuver analyzed in the present work is always smaller than the orbit radius, hence it can be assumed that both the spacecraft are subjected to the same perturbative force, since the $J_2$

effect is function of inclination and RAAN angles, and hence difference in the $J_2$ acting on the two spacecraft is function of the relative phase angle. For this reason, the term $\nabla \mathbf{J}_2(\mathbf{r}) \cdot \mathbf{r}$ can be neglected, and the resulting $J_2$ perturbative force can be computed as

$$F_{J2} = -\frac{3}{2}\frac{J_2 \mu R_E}{r^4} \begin{Bmatrix} 2\sin(i^2)\sin\Omega\cos\Omega \\ 2\sin i \cos i \sin\Omega \\ 1 - 3\sin(i^2)\sin(\Omega^2) \end{Bmatrix} \tag{3.45}$$

where $R_E$ is the mean radius of the Earth and

$$J_2 = 0.0010827$$

### Gravity Gradient

The gravity gradient disturbance is a torque disturbance acting on the spacecraft due to non-uniform mass distribution. The gravity gradient torque is computed by

$$M_{GG} = 3\frac{\mu}{R^3}\hat{o}_3 \times (I\hat{o}_3) \tag{3.46}$$

where $\hat{o}_3 = -r/\|r\|$, which is also the third column of the $DCM_{BLVLH}$ matrix, expressing the misaligment of the attitude of the spacecraft with respect to the local gravity vector. $I$ is the inertia matrix of the spacecraft, $R$ is the orbit radius from the center of the Earth, and $r$ is the ECI position vector, such that $R = \|r\|$. An important result derived from Eq. 3.46 is that the gravity gradient torque is null when $\hat{o}_3 \times (I\hat{o}_3) = 0$, which means that one principal axis of inertia of the spacecraft is aligned with the local gravity vector $\hat{o}_3$. Due to this important result, the gravity gradient torque is also a passive attitude stabilization technique used when relaxed pointing requirements are required, or it can be used combined with other active attitude control techniques in order to reduce the control effort.

### Solar Pressure

The solar radiation pressure produce an effect analogous to the residual atmospheric drag along the Sun-spacecraft vector, but such effect is two to three order of magnitude lower than the drag effect, while the torque perturbation due to the Sun radiation has the same order of magnitude of the gravity gradient [42]. The solar wind is assumed to be a force aligned with X$_{\text{ECI}}$, hence producing a torque along Y$_{\text{ECI}}$ and Z$_{\text{ECI}}$. The disturbance torque is assumed to be computed as a random value as

$$M_{sun_{ECI}} = \texttt{randu} \cdot [0, pAl, pAl]^T \tag{3.47}$$

where $p$ is the solar radiation pressure ($p \in [4.38, 4.68] \cdot 10^{-6}$ N/m$^2$), $A$ is the reference area of the spacecraft, $l$ is the reference leght set as $l = 0.5 \cdot \sqrt{A}$ and `randu` is a random number in the range $[-1, 1]$ with uniform distribution. The solar wind torque expressed in body frame can be easily found by

$$M_{sun_{Body}} = DCM_{BE} M_{sun_{ECI}} \tag{3.48}$$

where $DCM_{BE}$ is the ECI to Body rotation matrix. For obvious reasons, this disturbance is null when the spacecraft is in the eclipse period, since the Sun is shaded from the Earth shadow. According to Fig. 3.12, the half eclipse angle can be computed as

$$\beta = \frac{\pi}{2} - \arccos \frac{R_E}{r} \tag{3.49}$$

where $R_E$ and $r$ are respectively the Earth radius and the radius of the spacecraft orbit. The total eclipse angle is then $2\beta$ and hence the solar perturbation will be active for $\Theta_{Sun} = 2\pi - 2\beta$. Assuming the initial position of the spacecraft $\phi_0$ at the end of the eclipse period, the solar torque can be computed as

$$\begin{cases} M_{sun_{Body}} = DCM_{BE} M_{sun_{ECI}} & \text{if } 0 \le \phi \le \Theta_{Sun} \\ M_{sun_{Body}} = 0 & \text{if } \Theta_{Sun} < \phi < 2\pi \end{cases} \tag{3.50}$$

## 3.3.4 Spacecraft Actuators

The Chaser spacecraft is equipped with a set of actuators used to orient and maintain the spacecraft attitude within the desired orientation and to execute orbital rendezvous maneuvers. The main actuation system used to execute orbital maneuvers is the *Thrusters System*, while the main actuation system used to control the attitude of the spacecraft is the *Reaction Wheels System*. This system is assisted by the Thrusters System when the torque required to orient the spacecraft to a desired attitude is higher than the maximum torque that can be provided by the wheels assembly along the

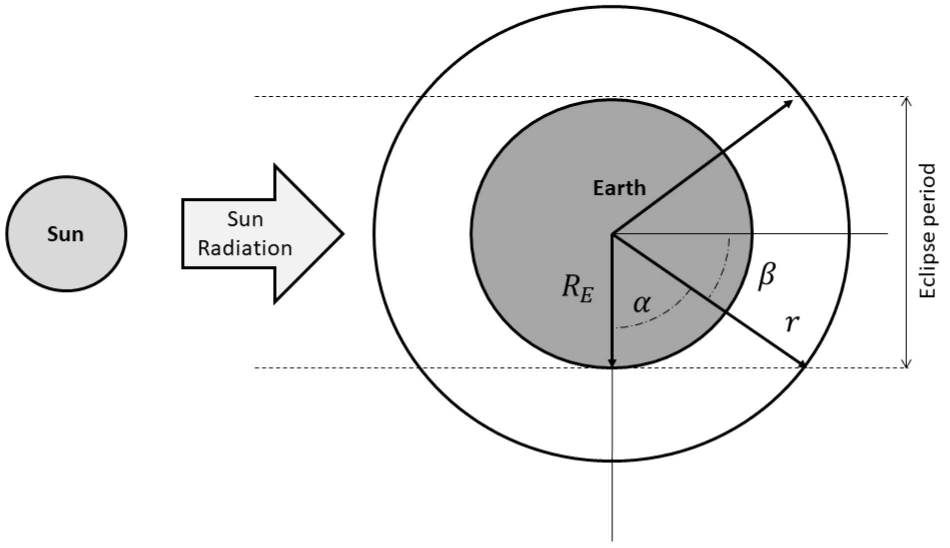


Fig. 3.12 Solar radiation and Earth eclipse.

desired axis. In addition, the Thrusters System is also used to de-saturate Reaction Wheels.

### Reaction Control Thrusters System

The Reaction Control Thrusters System is composed by 12 cold gas thrusters arranged in the configuration depicted in Fig. 3.13. The adopted configuration allows the use of the thrusters system to generate both pure force and pure torque along a desired axis. To generate pure force, it is required to activate the couple of thrusters firing in the same direction; instead, to generate pure torque, it is required to activate the couple of thrusters firing in opposite direction. For example, to generate pure force along the +X$_{BODY}$ axis it is necessary to activate thrusters 1x and 3x, while to generate pure torque along +X$_{BODY}$ it is necessary to activate thrusters 1y and 3y. In Table 3.16 are summarized the nominal firing direction and the nominal position of each thruster with respect to the Body Geometric frame. The vector $r_{4_i}$ of Fig. 3.13 is the position vector with respect to BG frame of the 4th thruster firings in the direction *i* (which can be x, y or z).

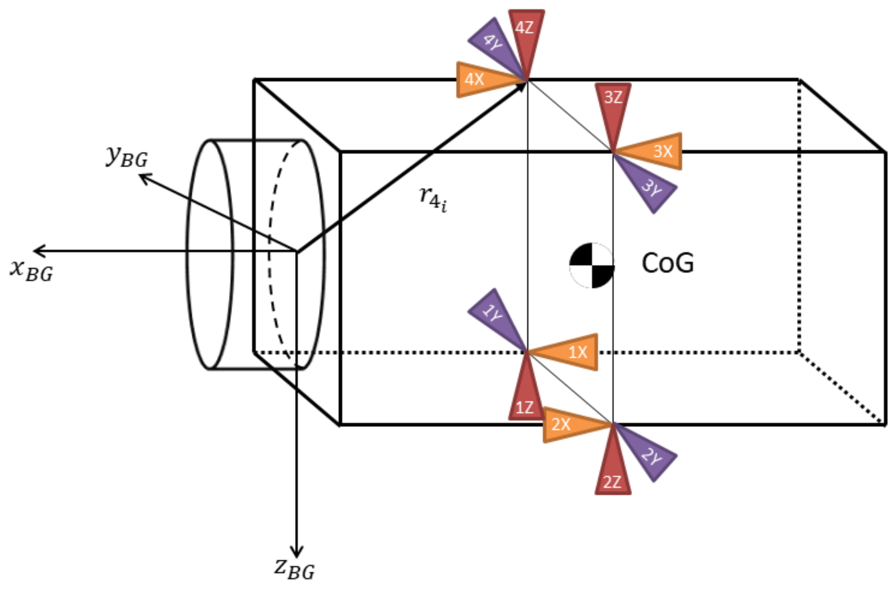


Fig. 3.13 Reaction Control Thrusters System configuration.

| Thruster ID | Nominal firing direction [x, y, z]$_{BODY}$ | Nominal position in BG [x, y, z]$_{BODYGEOM}$ [m] |
|---|---|---|
| 1X | [1, 0, 0] | [-2.0, 0.8, 0.8] |
| 2X | [-1, 0, 0] | [-2.0, -0.8, 0.8] |
| 3X | [1, 0, 0] | [-2.0, -0.8, -0.8] |
| 4X | [-1, 0, 0] | [-2.0, 0.8, -0.8] |
| 1Y | [0, -1, 0] | [-2.0, 0.8, 0.8] |
| 2Y | [0, 1, 0] | [-2.0, -0.8, 0.8] |
| 3Y | [0, 1, 0] | [-2.0, -0.8, -0.8] |
| 4Y | [0, -1, 0] | [-2.0, 0.8, -0.8] |
| 1Z | [0, 0, -1] | [-2.0, 0.8, 0.8] |
| 2Z | [0, 0, -1] | [-2.0, -0.8, 0.8] |
| 3Z | [0, 0, 1] | [-2.0, -0.8, -0.8] |
| 4Z | [0, 0, 1] | [-2.0, 0.8, -0.8] |

Table 3.16 Direction and position of the thrusters system.

The modeling process for the reaction control thrusters system can be divided in three main layers: the mono-dimensional (1D) model of the single thruster, the three-dimensional (3D) model of the single thruster and the three-dimensional model of the whole system of 12 thrusters. The 1D model of the single thruster consists in modeling the magnitude of the thrust provided by the nozzle including non-nominal effect, such as thrust errors and delays in reaching the maximum thrust. The magnitude of the thrust is modeled as follows

$$F_{mag_i} = F_{nom} + \Delta F_{bias_i} + \Delta F_{rand_i} \tag{3.51}$$

where $F_{mag_i}$ is the 1D magnitude of the thrust provided by the thruster $i$, $F_{nom}$ is the nomial *as-designed* thrust, $\Delta F_{bias_i}$ is the bias thrust error affecting the thruster $i$, $\Delta F_{rand_i}$ is the random thrust error affecting the thruster $i$. The index $i$ is referred to the thruster ID as summarized in Table 3.16. The value of $F_{nom}$ is equal to each thrusters, since it is the nominal value of thrust for which the thruster has been designed. The bias error $\Delta F_{bias_i}$ is the thrust error occurred during the manifacturing of the thruster: since the nozzle of each thruster may present very small flaws with respect to the original design, also the maximum thrust provided by each nozzle may be different from the nominal one. This bias error has been introduced in order to model this effect. Typical values of this error are in the range of 3-5 % of the nominal thrust. Since each thruster is tested before be sold, the value of this error is usually known and reported in the datasheet of the thruster. Finally, the random error $\Delta F_{rand_i}$ is introduced to modeled the effect that the maximum thrust provided by the thruster may vary at each activation. In this error are also included thrust errors due to delays in reaching the maximum thrust: this effect affects the thrust magnitude particularly for short opening intervals of the Flow Control Valvle (FCV) of the nozzle. Typical values of this error are about 5 % of the nominal thrust. Contrary to the $\Delta F_{bias_i}$, which is a constant error, the $\Delta F_{rand_i}$ error is a time-varying error computed during each activation of the thruster: the error is different by each activation but it is constant during the whole activation interval and does not depend by the length of the interval.

The 3D model of each thruster consists in projecting the magnitude of the thrust $F_{mag_i}$, as computed in Eq. 3.51, along the direction versor expressed in Body Geometric frame

$$F_{thr_i} = DCM_{rand_i} \cdot DCM_{bias_i} \cdot \hat{e}_i \cdot F_{mag_i} \tag{3.52}$$

As for the model of the mono-directional thrust, also the thrust direction is affected by two errors, respectively bias and random errors. In Eq. 3.52, $F_{thr_i}$ is the computed 3D thrust of the thruster $i$, $F_{mag_i}$ is the 1D magnitude of the thrust, $\hat{e}_i$ is the nominal direction of the thruster $i$ as summarized in Table 3.16, $DCM_{bias_i}$ is the rotation matrix which models the misalignment with respect to the nominal direction, mainly due to mounting errors, and $DCM_{rand_i}$ is the rotation matrix which models random errors due to the misalignment induced by vibrations caused by the activation of the thruster. Again, the $DCM_{bias_i}$ is a constant matrix, while $DCM_{rand_i}$ is different by each activation of the thruster but it is constant whithin the activation interval.

The last step is to model the whole thrusters system. This model consists in evaluating the total force and torque provided by the thrusters system. Evaluating the total force is a trivial task, since it is the sum of the 3D force generated by each thrhuster:

$$F_{RCS} = \sum F_{thr_i} \tag{3.53}$$

To evaluate the total torque provided by the RCS, it is first evaluated the position of each thruster with respect to the center of mass of the spacecraft

$$r_{thr_i} = r_{i_{BG}} - r_{cog_{BG}} \tag{3.54}$$

and the total torque is computed by

$$M_{RCS} = \sum r_{thr_i} \times F_{thr_i} \tag{3.55}$$

Eventually, the propellant mass flow rate of the thrusters system is evaluated. The mass flow rate of the single thruster is computed by

$$\dot{m}_{thr_i} = \frac{F_{mag_i}}{I_{sp} \cdot g_0} \tag{3.56}$$

where $I_{sp}$ is the specific impulse of the thruster and $g_0$ is the gravitational acceleration of Earth at sea level. The total propellant mass flow rate is evaluated by

$$\dot{m}_{RCS} = \sum \dot{m}_{thr_i} \tag{3.57}$$

and the mass variation of the chaser spacecraft during the mission is computed by time integration of the total mass flow rate

$$m_{chs} = m_{chs_0} - \int \dot{m}_{RCS}\, dt \tag{3.58}$$

### Reaction Wheels System

The Reaction Wheels System is the main actuation system used to control the attitude of the chaser spacecraft. The system is composed by a number $N_{wheels}$ of reaction wheels arranged with different types of configuration, depending by the specific application, control strategy, redundancy, and more. The single reaction wheel it is a massive disk connected to an electric motor controlled in both torque and velocity: to accelerate the wheel, the motor apply a torque to the disk and, since the wheel assembly is mechanically connected to the entire spacecraft structure, the reaction torque of the accelerating disk (equal in magnitude but with opposite direction) is transmitted to the motor and hence it is transmitted to the spacecraft, which reacts modifying its attitude. The reaction wheel is subjected to two main saturation conditions: torque saturation and velocity saturation. The torque saturation is the maximum torque the wheel can provide: it is a physical limitation of the maximum current that flows into the electric motor before damaging it. Usually, the torque saturation is implemented in the control law, in such a way that the torque required by the controller should never exceed the maximum torque provided by the wheel, preventing damaging of the electric motor. The velocity saturation is the maximum angular velocity that can be sustained by bearings of the electric motor. If the velocity is greater than the maximum allowable, the bearings may be damaged and makes the wheel unusable. Since the wheel has reached the maximum velocity it cannot accelerate anymore, then the torque provided in this condition is null. In order to make the wheel usable again, it is necessary to slow down the disk, reducing the velocity. Decelerating the disk means applying a torque opposite to the rotation, but this will cause a modification of the attitude of the spacecraft due to the reaction torque, hence it is necessary to apply an external torque to counteract the the torque required to decelerate the wheel. This external torque is provided by the thrusters system for almost all the typologies of spacecraft, except for the recent *CubeSat* typology which use magnetotorquers to de-saturate reaction wheels.

The reaction wheels system model is divided in two sub-models: the 1D model of the single wheel and the 3D model the the whole reaction wheels system. The 1D model is a simple model of the accelerating wheel. In order to take into account the effect that the moment if inertia of the wheel may be a little different form the nominal one due to production defects, it is introduced a scale factor, $K_{wheel_i}$, to model the non nominal moment of inertia, and the true moment of inertia of the $i^th$ wheel is modeled by

$$I_{wheel_{true_i}} = \frac{I_{wheel_{nom}}}{K_{wheel_i}} \tag{3.59}$$

The input torque accelerating the wheel is

$$\dot{h}_{in_i} = K_{wheel_i} \cdot M_{cmd_i} \tag{3.60}$$

where $M_{cmd_i}$ is the command torque required by the controller for the wheel $i$. The angular acceleration of the wheel is computed as

$$\dot{\omega}_{wheel_i} = \frac{\dot{h}_{in_i}}{I_{wheel_{true_i}}} \tag{3.61}$$

and the output torque of the wheel is then

$$\dot{h}_{out_i} = -\dot{\omega}_{wheel_i} \cdot I_{wheel_{true_i}} \tag{3.62}$$

The angular velocity of the wheel $i$ is computed by

$$\omega_{wheel_i} = \omega_{wheel\,0_i} + \int \dot{\omega}_{wheel_i}\, dt \tag{3.63}$$

and the angular momentum of the wheel is

$$h_{wheel_i} = \omega_{wheel_i} \cdot I_{wheel_{true_i}} \tag{3.64}$$

Saturation in torque and velocity are implemented as well.

The 3D model of the reaction wheels system is used to model the total torque and the total angular momentum of the reaction wheels system in Body Frame, which means to mix all the $N_{wheels}$ models of the single wheel into a three-dimensional one.

Three vectors of size $N_{wheels}$ have to be defined:

$$\dot{h}_{RWs} = [\dot{h}_{out_{wheel_1}} \dots \dot{h}_{out_{wheel_{N_{wheels}}}}]^T$$
$$h_{RWs} = [h_{wheel_1} \dots h_{wheel_{N_{wheels}}}]^T$$
$$\omega_{RWs} = [\omega_{wheel_1} \dots \omega_{wheel_{N_{wheels}}}]^T$$

The vector $\dot{h}_{RWs} \in \mathbb{R}^{N_{wheels}x1}$ is the vector collecting all the 1D output torque of the wheels, the vector $h_{RWs} \in \mathbb{R}^{N_{wheels}x1}$ is the vector collecting all the 1D angular momentum of the wheels and $\omega_{RWs} \in \mathbb{R}^{N_{wheels}x1}$ is the vector collecting all the angular velocities of each wheel. This last vector is vector used to transmit the telemetry of the wheels to the GNC software in order to prevent and fix saturation of the wheels. To link the 1D model to the 3D model of the reaction wheels system it is introduced the matrix $Z \in \mathbb{R}^{3xN_{wheels}}$, which include the orientation of each wheel with respect to the spacecraft Body frame, and project the torque generated by each wheel in the nominal direction of the wheel in Body frame:

$$\dot{H}_{RWs} = Z \cdot \dot{h}_{RWs} \tag{3.65}$$

where $\dot{H}_{RWs}$ is the total torque of the reaction wheels system expressed in Body frame, and the total angular momentum of the reaction wheels system is

$$H_{RWs} = Z \cdot h_{RWs} \tag{3.66}$$

For a simple system of three reaction wheels aligned along three axis of the Body frame, the matrix $Z$ is a 3-by-3 identity matrix and $\dot{H}_{RWs} = \dot{h}_{RWs}$ and $H_{RWs} = h_{RWs}$.

### 3.3.5 Spacecraft Sensors

The orbital simulator includes a complete set of sensors used for absolute and relative state estimation. The absolute state estimation is referred to estimation of position, velocity, attitude and angular velocity of the spacecraft with respect to the inertial frame, while the relative state estimation is referred to estimation of position and velocity of the Chaser vehicle with respect to the Target one.

The three relative sensors, Radio Finder, LIDAR and Camera, have been modeled in a simular way for what concern the relative position measurment, as it will be described in the following of the section. In addition, all sensors, have a common output modeled in order to include some features related to the GNC software implementation. The common output includes a variable named *counter* and a variable named *status*. The status variable is the first step of validation of the sensor measurement. For all sensors, the status variable assumes the following values:

- STATUS = 0 (NOP): the sensor is not operative.
- STATUS = 1 (NACQ): the sensor is operative but the measure is not valid.
- STATUS = 2 (ACQ): the sensor is operative and the measure is valid.

For the Star Tracker sensor, the status NACQ occurs while the spacecraft is rotating with an excessive angular velocity and the stars in the field of view of the sensors cannot be processed and the attitude estimated. For IMU and GPS, failure of the sensor (status NOP and NACQ) is not modeled. For relative sensors, the status NACQ means that the Target is out of range of the sensor, while ACQ meas a complete operativity of the sensor. For the Camera sensor, an additional status, *STATUS = 3 (CAMACQ)*, is used when the sensor is able to measure the relative attitude of the Target spacecraft with respect to the Camera.

The counter is the second step of validation of the sensor measurment, and it is used by the GNC software to determine if the output of the sensor can be considered as valid: indeed, in the normal operation of the sensor, the counter is continuously increasing of one unit at the same frequency of the sensors output update. If two measured output have an identical value of the counter it means that something wrong have been occurred in the sensor and the related measure cannot be considered reliable, and the Navigation function must estimates the navigation solution in a different way. In the GPS sensor model, the counter variable is not implemented, but the same validation procedure executed by the Navigation function uses the time measure instead.

### Global Positioning System - GPS

The GPS sensor is used to measure position and velocity of the spacecraft with respect to the ECEF frame. In addition, since each satellite of the GPS constellation is equipped with very precise atomic clocks, the GPS sensor is also used as a timing sensor. The physical architecture of a complete GPS receiver consists in an antenna installed in the external side of the spacecraft, possibly pointing through the GPS constellation, and a receiver chip inside the spacecraft, connected to both the antenna and the on-board computer of the spacecraft. Installing more antennas makes possible to estimate also the attitude of the spacecraft using differential GPS techniques, even if the precision of the attitude estimation is lower than estimation provided by more precise sensors (for example star trackers or horizon sensors).

The ECEF position measured by the GPS is computed by

$$r_{ECEF} = DCM_{ECEF\,ECI} \cdot r_{ECI} \tag{3.67}$$

and the measurment error is added

$$r_{GPS} = r_{GPS_{err}} + r_{ECEF} \tag{3.68}$$

where $r_{GPS_{err}} \in \mathbb{R}^3$ is the position measurment error of the GPS modeled with a Gaussian distribution. Then, it is computed the velocity in ECEF frame

$$\dot{r}_{ECEF} = DCM_{ECEF\,ECI} \cdot (\dot{r}_{ECI} - V_{Earth_{perif}}) \tag{3.69}$$

where $V_{Earth_{perif}} = \Omega_{EARTH} \times r_{ECI}$ is the peripheral velocity of the spacecraft. Then, the Gaussian measument error is added

$$\dot{r}_{GPS} = \dot{r}_{GPS_{err}} + \dot{r}_{ECEF} \tag{3.70}$$

The time signal of the GPS is defined in the format Leap Seconds, GPS Week and Time of Week. All the time data are referred to the GPS epoch of January 6, 1980. The leap seconds are seconds added to the current time in order to align the Coordinated Universal Time (UTC) to the Solar Day. In 2017, the total leap seconds from the initial GPS time are 17 s. The GPS Week data is the number of week from January 6, 1980. The GPS Time of Week data is the number of seconds from the beginning of the current week. The output of the GPS time signal is then

$$GPS_{TIME} = \begin{cases} LS \\ GPS_{Week} \\ GPS_{ToW} + t_{err} \end{cases} \tag{3.71}$$

where $LS$ is the leap seconds data, $GPS_{Week}$ is the GPS Week data, $GPS_{ToW}$ is the GPS Time of Week data and $t_{err}$ is the time error with Gaussian distribution.

### Inertial Measurment Unit - IMU

The IMU sensor is used to measure both linear acceleration and angular rate. To measure linear acceleration, a set of accelerometers is aligned along the three main axis of the IMU box. The basic design of an accelerometer consist in a mass connected to a spring and able to move only along one axis. If it is applied an acceleration to the sensor, the mass push the spring in the opposite direction. Measuring the compression or extension of the spring with respect to the rest position and knowing precisely the value of the mass it is possible to compute the value of the acceleration. Many other and more precise configuration of accelerometers can be found nowadays in the market, depending by the desired accuracy and the miniaturization level required. To measure the angular rate is used a set of gyroscopes. The gyroscope is simply a rotating mass, as for the reaction wheels described in the previous section. For the gyroscopic effect, if it is applied a torque perpendicular to the rotation axis, the mass starts moving with a precession motion perpendicular to the other two axis. According to this effect, it is possible to measure the torque derived by the motion of the gyroscope along a third axis and then estimate such angular velocity. Modern gyroscopes are based on laser or optical devices with a complete different working principles, which allows much better performances without using moving parts.

The IMU model includes the output of linear acceleration and angular velocity measured by the accelerometers and the gyroscopes. The accelerometer sensor measures linear inertial acceleration, i.e. accelerations mainly due to the thrusters system, since environmental disturbances causes very low values of acceleration difficult to be detected by the sensor. The acceleration in body frame have to be referred to the IMU location, so the true lever arm correction must be applied

$$a_{IMU_B} = a_B + \omega_B \times (\omega_B \times r_{IMU}) + \dot{\omega}_B \times r_{IMU} \tag{3.72}$$

where $a_{IMU_B}$ is the linear acceleration referred to the IMU location, $a_B$ is the linear acceleration in Body frame, $\omega_B$ is the angular velocity of the spacecraft, $\dot{\omega}_B$ is the angular acceleration of the spacecraft and $r_{IMU}$ is the IMU location with respect to the center of mass computed by

$$r_{IMU} = r_{IMU_{BG}} - r_{cog} \tag{3.73}$$

where $r_{IMU_{BG}}$ is the IMU location in Body Geometric frame and $r_{cog}$ is the location of the center of mass in Body Geometric frame. To evaluate the linear acceleration referred to the IMU axis, it has to be taken into account the misalignment of the IMU box with respect to the Body frame, then

$$a_{@IMU} = DCM_{IMU\,B} \cdot a_{IMU_B} \tag{3.74}$$

where $a_{@IMU}$ is the linear acceleration referred to the IMU axis and $DCM_{IMU\,B}$ is the rotation matrix which considers the misalignment of the IMU frame with respect to the Body frame.
The acceleration referred to the accelerometers sensors is including a small misalignment errors, then

$$a_{@ACCEL} = DCM_{ACCEL\,IMU} \cdot a_{@IMU} \tag{3.75}$$

where $a_{@ACCEL}$ is the acceleration in the accelerometer sensor frame and the misalignment between the accelerometer and the IMU is represented by the rotation matrix $DCM_{ACCEL\,IMU}$.

Similarly, also the angular velocity has to be referred to the gyroscopes axis. First, it is computed the angular velocity referred to the IMU axis

$$\omega_{@IMU} = DCM_{IMU\,B} \cdot \omega_B \tag{3.76}$$

and then it is computed the angular velocity referred to the gyroscope sensor

$$\omega_{@GYROS} = DCM_{GYROS\,IMU} \cdot \omega_{@IMU} \tag{3.77}$$

Both accelerometers and gyroscopes are subjected to different errors, depending by the specific sensor, such as calibration errors or scale factors, and by external factors, such as temperature and vibrations. Some of these errors included in the IMU model presented here are:

- *Bias error*: it is constant error and it can be tested averaging the output of the sensor over a long period. This error usually have a Gaussian distribution.
- *Scale Factor*: the scale factor is a static error due to the limited attenuation or amplification of the output with respect to the input, for which the input/output scheme is not mapped 1-to-1. This error is usually measured in part-per-million (ppm).
- *Angle Random Walk (ARW)*: this error is due to thermomechanical processes affecting both accelerometers and gyros sensors, modeled as a white noise. This error is caused by integration of random numbers which produce the so called *random walk* of the integrated value.
- *Bias Stability error*: this errors is similar to the ARW but referred to a limited period of time. This drift error does not increase indefinitely.

All these errors are included in the IMU model. For the accelerometer sensor, the measured acceleration is

$$a_{meas} = a_{bias} + a_{SF} + a_{bias\,stab} + a_{noise} \tag{3.78}$$

where $a_{bias}$ is the acceleration due to the bias error, $a_{SF}$ is the acceleration due to the scale factor error, $a_{bias\,stab}$ is the acceleration due to the bias stability error and $a_{noise}$ is the acceleration to to the ARW error. The bias error is a constant error computed in the initialization process and it has different value for each of the three accelerometers. The scale factor error is a constant error and the acceleration due to this error is

$$a_{SF} = (1 + SF) \cdot a_{@ACCEL} \tag{3.79}$$

where $SF = [SF_x, SF_y, SF_z]^T$ is the vector including three different values of scale factor affected the three accelerometers aligned in the $x_{IMU}$, $y_{IMU}$ and $z_{IMU}$ direction, including the misalignment error. The bias stability error is computed by

$$\dot{b}_{ACCEL} = -\frac{1}{t_{corr}} \cdot b_{ACCEL} + w_{bias_{ACCEL}} \tag{3.80}$$

where $t_{corr}$ is the correlation time of the sensor, $b_{ACCEL}$ is the drift of the bias state computed by

$$b_{ACCEL} = \int \dot{b}_{ACCEL} dt \tag{3.81}$$

and $w_{bias_{ACCEL}}$ is the white noise process with 1-$\sigma$ value

$$\sigma_{bias\,stab} = \sqrt{2}\frac{B_{ACCEL}}{\sqrt{t_{corr}}\sqrt{\Delta t}}$$

where the parameter $B_{ACCEL}$ is used to model the magnitude of the bias stability drift. The last error, $a_{noise}$ is the accelerometer white noise equivalent to the ARW error.

In a similar way such as for the accelerometer measurement, the gyroscopes measure is modeled as

$$\omega_{meas} = \omega_{bias} + \omega_{SF} + \omega_{bias\,stab} + \omega_{noise} \tag{3.82}$$

in which all the terms $\omega_{SF}$, $\omega_{bias\,stab}$ and $\omega_{noise}$ are computed in the same way as respectively Equations 3.79, 3.80 and 3.81, considering $\omega_{bias}$ computed in the initialization process.

The last step to complete the model of the IMU is to consider that the IMU produce a digital output, so the measured output is subjected to the quantization error of the discretization of the measure. Then, considering a 8-bit output, the Less Significant Bit (LSB) of accelerometers and gyroscopes measurements can be computed

$$LSB_{ACCEL} = \frac{a_{max}}{\mathsf{INT}_{\mathsf{MAX}}}$$
$$LSB_{GYROS} = \frac{\omega_{max}}{\mathsf{INT}_{\mathsf{MAX}}}$$

where $a_{max}$ and $\omega_{max}$ are the maximum acceleration and angular velocity that can be measured and $\mathsf{INT}_{\mathsf{MAX}}$ is the maximum limit of the *integer* definition of a variable in C language. To apply the truncation error due to the digitalization of the output it is required the conversion of the measurement from the *double* definition into a *integer* one, then

$$a_{meas}^{INT} = a_{meas} \cdot \frac{\mathsf{INT}_{\mathsf{MAX}}}{a_{max}} \tag{3.83}$$

$$\omega_{meas}^{INT} = \omega_{meas} \cdot \frac{\mathsf{INT}_{\mathsf{MAX}}}{\omega_{max}} \tag{3.84}$$

Then, the *integer* measure is converted again into a *double* measure with the truncation error:

$$a_{meas}^{DOUB} = a_{meas}^{INT} \cdot LSB_{ACCEL} \tag{3.85}$$

$$\omega_{meas}^{DOUB} = \omega_{meas}^{INT} \cdot LSB_{GYROS} \tag{3.86}$$

The IMU model includes also saturation: according to the value of $a_{max}$ and $\omega_{max}$, if the measured acceleration or angular velocity, including errors, is greater of the maximum, the IMU output is saturated to the maximum value.

### Star Tracker

The star tracker is basically an optical device which recognize a set of stars in the field of view of the sensor, compare them with an accurate database and estimate the attitude of the sensor with respect to the stars. The position of the stars is accurately known with respect to the inertial frame ECI, hence the attitude of the spacecraft is measured in the same frame. Depending by the quality of the lenses of the sensor, by the optical sensor for stars acquisition, by the accuracy of the stars catalog, performance of the star tracker are very different from different devices.

Star trackers are affected by a number of errors, such as spherical and chromatic aberration, the detection algorithm may be be not be able to distinguish sunlight reflection by other spacecraft or may be detect wrongly plumes of the spacecraft, and more. Due to these sources of errors, the device shall be placed correctly in the spacecraft configuration, and possibly may be useful to consider a number of redundancy of the sensor. For example, if it is present only one sensor in the spacecraft configuration, a particular attitude may point the sensor towards the Earth, making it unable to track the attitude since there are no stars in the field of view but the surface of the Earth.

In the model of star tracker presented in the present work, there are considered two main errors affecting the attitude measure: a bias and a random errors. The measured output of the sensor is then

$$q_{meas} = q_{rand} * q_{bias} * q_{STR,B} * q_{B,ECI} \tag{3.87}$$

where $q_{meas}$ is the measured attitude of the spacecraft in ECI frame, $q_{rand}$ and $q_{bias}$ are respectively the quaternions of the random and bias errors, $q_{STR,B}$ is the quaternion expressing the attitude of the sensor with respect to the Body frame and $q_{B,ECI}$ the attitude of the spacecraft with respect to the ECI frame. The bias error is included in order to take into account misalignment of the sensor due to mounting errors, while the random error is basically the noise affecting the measure of the sensor, depending by all the sources of errors listed before, including illumination conditions, and depending by the angular velocity of the spacecraft. Indeed, the measurement error depends strongly by the angular velocity of the spacecraft: the faster it is spinning, the greater is the error. For very high angular rates, typically less than 10 deg/s, the star tracker may not be able to measure the attitude at all, providing a not valid attitude measurement.

### Radio Finder

The Radio Finder is a long-range sensor for relative position tracking. It is used to detect the position of the Target spacecraft with respect to the Chaser one. It is a radar sensor, which uses radio frequency to detect position and velocity of an object. The sensor emits radio waves with a specific frequency, usually micro-waves, the waves are reflected by the object and detected by the sensor, which measure the received power and differences in the phase between the transmitted and received signals and elaborates the position and the velocity of the object. According to its working principle, this sensor has a very good estimate of the distance (or range) of the object with respect to the sensor, but low performance in terms of azimuth and elevation tracking. The use of radio waves extend the operative range of this sensor up to tens of kilometers far from the object to track. For the purpose of this work, the Radio Finder sensor is modeled including only the position tracking and not the velocity tracking.

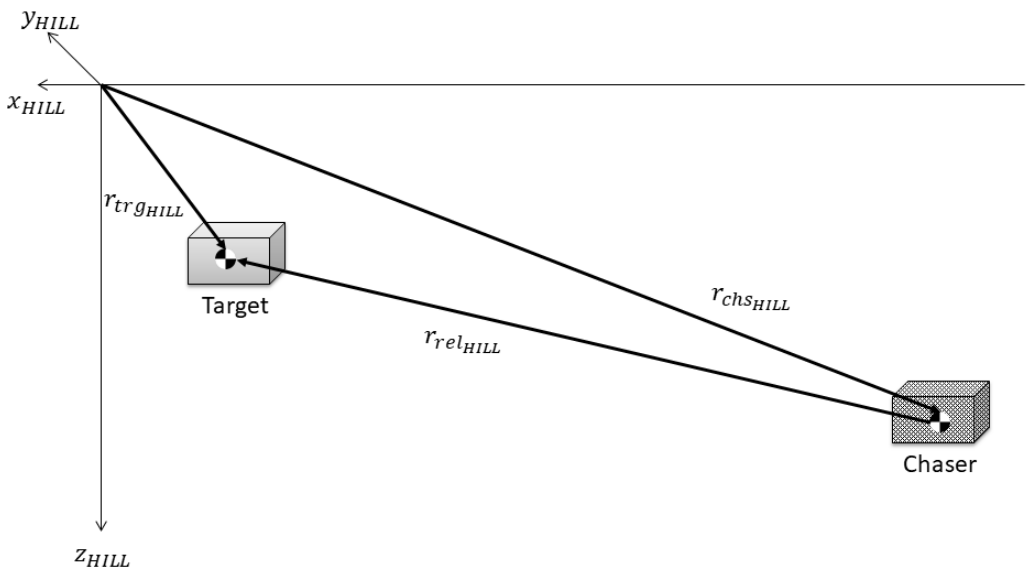


Fig. 3.14 Relative position of Target and Chaser spacecraft.

Since the Radio Finder is the long-range relative sensor, it is used to track the position of the center of mass of the target spacecraft. The relative position between the Chaser and the Target spacecraft with respect to the Hill frame is

$$r_{rel_{HILL}} = r_{trg_{HILL}} - r_{chs_{HILL}} \tag{3.88}$$

where $r_{trg_{HILL}}$ and $r_{chs_{HILL}}$ are respectively the position of the center of mass of the Target and Chaser spacecraft, and so $r_{rel_{HILL}}$ is the relative position of the centers of mass of the two vehicles, as in Fig. 3.14. The relative position of the two spacecraft has to be expressed in the Radio Finder sensor frame. Since the rotation matrix from Hill to Body frame, $DCM_{B,H}$, is known, the rotation between Body and Body Geometric frames is null, then $DCM_{BG,B} = I$, the relative position vector with respect to the BG frame is then

$$r_{rel_{BG}} = DCM_{BG,B} \cdot DCM_{B,H} \cdot r_{rel_{HILL}} \tag{3.89}$$

Once the position of the sensor with respect to the Body Geometric frame, $r_{RF_{BG}}$, and the position of the CoG of the Chaser, $r_{CoG_{BG}}$, are both known, as well as it is know the misalignment of the Radio Finder frame with respect to the Body Geometric frame, $DCM_{RF,BG}$, the relative position vector expressed in Radio Finder sensor

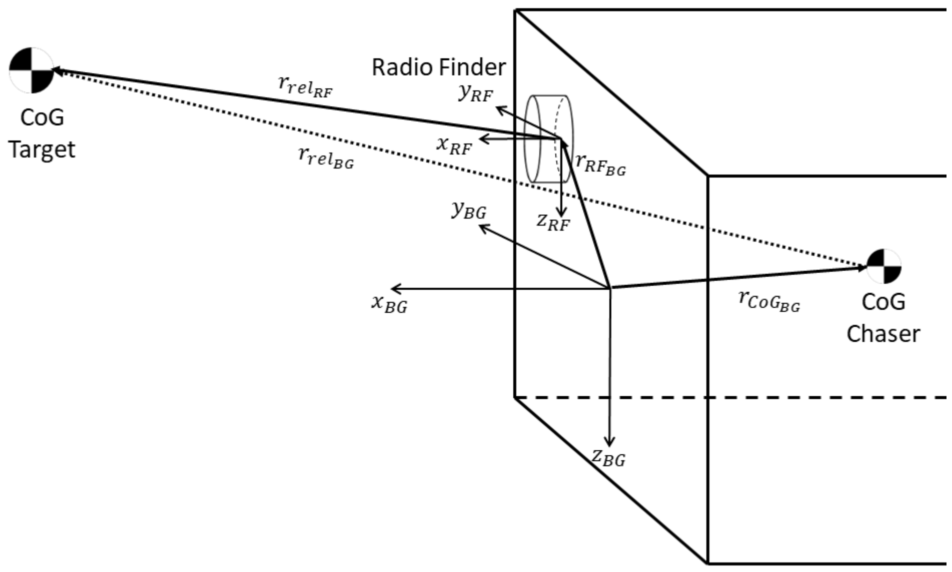


Fig. 3.15 Relative postion in Radio Finder sensor frame.

frame is then

$$r_{rel_{RF}} = DCM_{RF,BG} \cdot (r_{rel_{BG}} - r_{CoG_{BG}} + r_{RF_{BG}}) \tag{3.90}$$

as depicted in Fig. 3.15. The relative position $r_{rel_{RF}}$ has to be converted in the output format typical of this kind of relative sensor, i.e. it has to be produced the measure of *range*, the norm of the relative position vector, *azimuth*, the lateral angle, and *elevation*, the angle with respect to the "horizon" of the sensor. Assuming the vector $r_{rel_{RF}} = [x_{rel_{RF}}\, y_{rel_{RF}}\, z_{rel_{RF}}]^T$ the conversion is

$$R_{RF} = \sqrt{x^2_{rel_{RF}} + y^2_{rel_{RF}} + z^2_{rel_{RF}}} \tag{3.91}$$

$$\Psi_{RF} = \text{atan2}\,\frac{y_{rel_{RF}}}{x_{rel_{RF}}} \tag{3.92}$$

$$\Theta_{RF} = \text{atan2}\,\frac{z_{rel_{RF}}}{\sqrt{x^2_{rel_{RF}} + y^2_{rel_{RF}}}} \tag{3.93}$$

In Fig. 3.16 it has been depicted the definition of azimuth and elevation angles. The Radio Finder is subjected to measurement errors, depending by different factors, both environmental, such as multipath and propagation characteristics of radio waves,

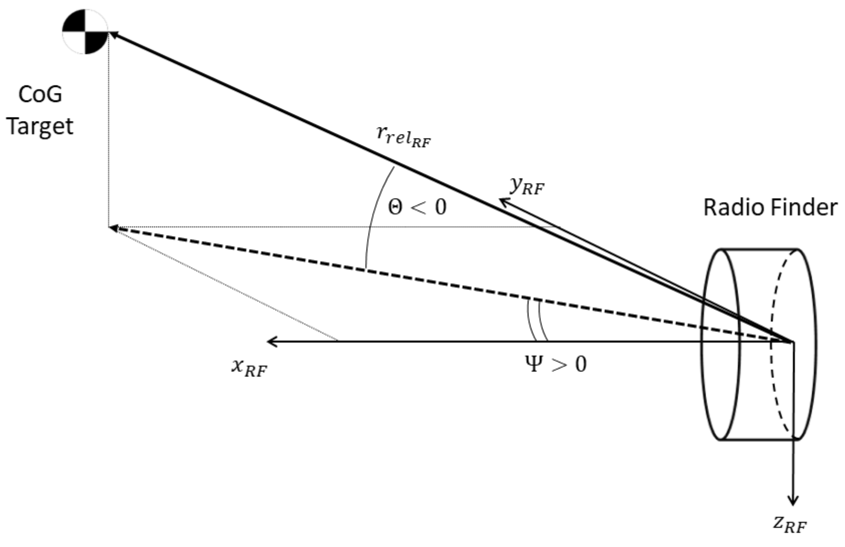


Fig. 3.16 Azimuth and elevation definition.

and target configuration, such as scintillation and glint of the target. In order to take into account such errors, the Radio Finder measure is including both bias and random errors, such as

$$\begin{cases} R_{meas_{RF}} = R_{RF} + R_{bias_{RF}} + R_{rand_{RF}} \\ \Psi_{meas_{RF}} = \Psi_{RF} + \Psi_{bias_{RF}} + \Psi_{rand_{RF}} \\ \Theta_{meas_{RF}} = \Theta_{RF} + \Theta_{bias_{RF}} + \Theta_{rand_{RF}} \end{cases} \tag{3.94}$$

where $R_{bias_{RF}}$, $\Psi_{bias_{RF}}$ and $\Theta_{bias_{RF}}$ are bias errors, depending by a constant source of errors, and $R_{rand_{RF}}$, $\Psi_{rand_{RF}}$ and $\Theta_{rand_{RF}}$ are random errors depending mainly by noise. The two types of error, bias and random, depend by the distance form the target object: the closer is the target, the lower are the errors, decreasing linearly. Limitations of the sensor have been modeled as well, including maximum and minimum operative range and the field of view of the sensor, in terms of maximum elevation and azimuth.

**LIDAR**

The LIDAR (Laser Imaging Detection and Ranging) sensor is a relative sensor able to acquire a target object in the medium/low range, in the order of kilometers. The working principle of the LIDAR is similar to the radar, but it uses different frequencies of the electromagnetic waves, which are in the range of ultraviolet, visible and infrared light. Thanks to the shorter wavelength than the radar, it has a better accuracy in azimuth and elevation measurement. In addition, the shorter wavelength make the sensor able to detect small chemical species in the atmosphere, extending the application field of the LIDAR. In the present work, the LIDAR sensor is used as a range tracker with better performance than the Radio Finder sensor.

The model of the LIDAR is very similar to the model of the Radio Finder. The relative position of Chaser and Target is computed as Eq. 3.88 and it is computed the relative position in BG frame as in Eq. 3.89. According to Eq. 3.90, the relative position in the LIDAR frame is computed as

$$r_{rel_{LID}} = DCM_{LID,BG} \cdot (r_{rel_{BG}} - r_{CoG_{BG}} + r_{LID_{BG}}) \tag{3.95}$$

where $DCM_{LID,BG}$ is the rotation matrix between the Body Geometric and the LIDAR frame. The output format of the LIADR measure is the same as for the Radio Finder, then the $r_{rel_{LID}}$ has to be converted in range, azimuth and elevation applying the conversion in Eqs. 3.91, 3.92 and 3.93, by substituting the subscript *RF* with the subscript *LID*. Eventually, bias and random errors are added, then the LIDAR measurment is

$$\begin{cases} R_{meas_{LID}} = R_{LID} + R_{bias_{LID}} + R_{rand_{LID}} \\ \Psi_{meas_{LID}} = \Psi_{LID} + \Psi_{bias_{LID}} + \Psi_{rand_{LID}} \\ \Theta_{meas_{LID}} = \Theta_{LID} + \Theta_{bias_{LID}} + \Theta_{rand_{LID}} \end{cases} \tag{3.96}$$

Bias and random errors are added in order to model all factors that affect the measurements, intrinsic of the sensor or depending by the environment in which it operates. As for the Radio Finder, limitation of range, azimuth and elevation have been modeled as well.

### Camera Sensor

The Camera sensor is an optical device that process video images and it is able to detect the position of an object and the relative attitude between the object and the sensor. Since it operates detecting an image of the target object, it can be used at a maximum distance of hundreds of meters to detect the position and tens of meters to detect the attitude. This is due to the fact that for higher distances, the perceived size of the markers placed on the target object is very small and comparable to the pixel size of the optical sensor, and it may not be possible to detect satisfactory the position or the relative attitude between the two spacecraft. Indeed, measurement errors depends strongly by the distance of the target object from the sensor due to the pixel resolution issue, and they are amplified by aberration of the lenses and illumination conditions, including flashing due to reflection of the sun light on the target object or other objects in the proximity of the sensor.

As for the Radio Finder and LIADR sensors, the position of the target object with respect to the Camera is evaluated by

$$r_{rel_{CAM}} = DCM_{CAM,BG} \cdot \left(r_{rel_{BG}} - r_{CoG_{BG}} + r_{CAM_{BG}}\right) \tag{3.97}$$

where $r_{rel_{BG}}$ is computed by Eq. 3.89, $r_{CAM_{BG}}$ is the position of the Camera with respect to the Bopdy Geometric frame and $DCM_{CAM,BG}$ is the rotation matrix between BG and Camera sensor frame. Applying conversions of Eqs. 3.91, 3.92 and 3.93 and including bias and random errors, the position measurement provided by the Camera is

$$\begin{cases} R_{meas_{CAM}} = R_{CAM} + R_{bias_{CAM}} + R_{rand_{CAM}} \\ \Psi_{meas_{CAM}} = \Psi_{CAM} + \Psi_{bias_{CAM}} + \Psi_{rand_{CAM}} \\ \Theta_{meas_{CAM}} = \Theta_{CAM} + \Theta_{bias_{CAM}} + \Theta_{rand_{LID}} \end{cases} \tag{3.98}$$

As mentioned before, the Camera sensor is also able to detect the relative attitude between the sensor and the target object. This is done by processing the image of a set of markers, placed in a specific configuration, on the target object, and evaluating the perspective of the markers, the relative attitude is then evaluated. An example of visual markers used for the rendezvous and docking maneuver of the Apollo missions is depicted in Fig. 3.17[1]. For autonomous rendezvous and docking maneuver, markers used by the camera to track the relative attitude may be a simple

[1]Picture from www.hq.nasa.gov

set of bright devices (three or more) in visible or infrared light, depending by the optical sensor of the Camera, placed with a specific configuration. The attitude of the Target spacecraft, $q_{B,LVLH_{trg}}$, and the attitude of the Chaser spacecraft, $q_{B,LVLH_{chs}}$ with respect to their local vertical frame is known, and the rotation matrices $DCM_{B,LVLH_{trg}}$ and $DCM_{B,LVLH_{chs}}$ can be easily derived. The rotation matrix between the two local vertical frame of Target and Chaser can be computed by knowing the relative distance along $V_{BAR}$, since it is a simple rotation along the Y axis

$$DCM_{LVLH_{chs},LVLH_{trg}} = \begin{bmatrix} -\sin\phi_{rel} & 0 & \cos\phi_{rel} \\ 0 & 1 & 0 \\ \cos\phi_{rel} & 0 & \sin\phi_{rel} \end{bmatrix} \tag{3.99}$$

where $\phi_{rel}$ is the relative phase angle between the Target local vertical frame and the Chaser local vertical frame, depicted in Fig. 3.18 and computed by

$$\phi_{rel} = \phi_{trg} - \phi_{chs} \tag{3.100}$$

where $\phi_{trg}$ and $\phi_{chs}$ are the phase angles with respect to the reference Hill frame between respectively $F_{LVLH_{trg}}$ and $F_{LVLH_{chs}}$, computed by considering the formula of the arc of circumference, $l = \theta r$, where $l$ is the length of the arc, $\theta$ is the angular aperture of $l$ and $r$ is the radius of the circumference. Applying this formula to compute the phase angles, considering the $V_{BAR}$ distance as $l$ and the radius $r = r_{EARTH} + h_{HILL} - R_{BAR}$, assuming $r_{hill\,frame} = r_{EARTH} + h_{HILL}$ the phase angles can be evaluated as

$$\phi_{trg} = \frac{x_{HILL_{trg}}}{r_{hill\,frame} - z_{HILL_{trg}}} \tag{3.101}$$

$$\phi_{chs} = \frac{x_{HILL_{chs}}}{r_{hill\,frame} - z_{HILL_{chs}}} \tag{3.102}$$

Since the angles $\phi_{trg}$ and $\phi_{chs}$ are computed using the non-curvilinear $V_{BAR}$ coordinate, values computed in Eqs. 3.101 and 3.102 are affected by an error deriving by the use of a Cartesian coordinate. However, since the operative range of Camera, and consequently the relative position between Chaser and Target, is few hundreds of meters and the Target spacecraft is within a neighborhood of hundreds of meters with respect to the origin of the Hill frame during the simulation of an RVD maneuver, the phase angle $\phi_{rel}$ is usually very small, and hence the matrix $DCM_{LVLH_{chs},LVLH_{trg}}$ tends to the identity matrix, and then the computational error of Eqs. 3.101 and

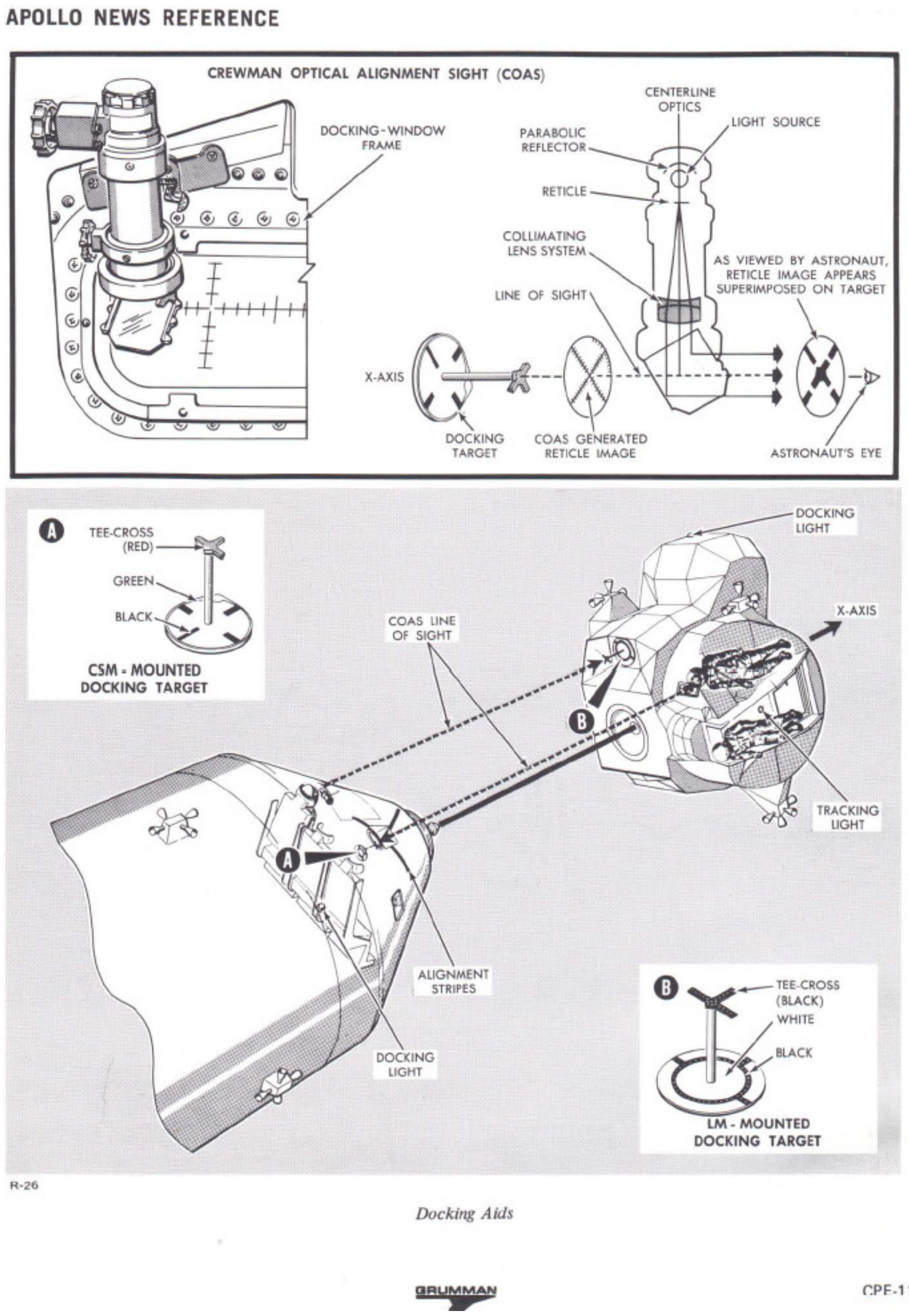


Fig. 3.17 Crewman Optical Alignment System (COAS) and Mounted Docking Target for CMS and LM.

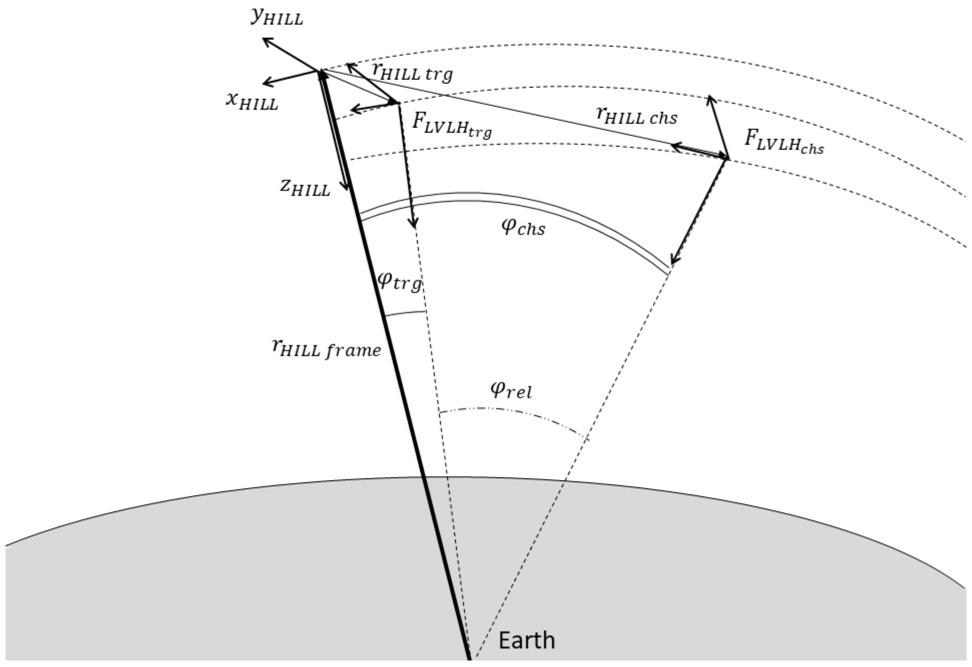


Fig. 3.18 Phase angle between $F_{LVLH_{trg}}$ and $F_{LVLH_{chs}}$.

3.102 is negligible. The relative attitude of the Target with respect to the Body frame of the Chaser is computed by

$$DCM_{B_{chs},B_{trg}} = DCM_{B,LVLH_{chs}} \cdot DCM_{LVLH_{chs},LVLH_{trg}} \cdot DCM_{LVLH_{trg},B} \tag{3.103}$$

The relative attitude of the Target expressed in body frame has to be further rotated in the Camera sensor frame, then

$$DCM_{trg_{CAM}} = DCM_{CAM,B} \cdot DCM_{B_{chs},B_{trg}} \tag{3.104}$$

where $DCM_{CAM,B}$ is the rotation matrix from Body Geometric frame and Camera sensor frame and considering a null rotation between Body and Body Geometric frame. From the $DCM_{CAM,B}$ can be derived the relative attitude angles in terms of roll, $\Psi_{trg}$, pitch, $\Theta_{trg}$, and roll, $\Phi_{trg}$, assuming a rotation sequence 3-2-1 (or Z-Y-X)

$$\begin{cases} \Psi_{trg} & = \text{atan2} \frac{DCM_{CAM,B}(1,2)}{DCM_{CAM,B}(1,1)} \\ \Theta_{trg} & = \arcsin\left(-DCM_{CAM,B}(1,3)\right) \\ \Phi_{trg} & = \text{atan2} \frac{DCM_{CAM,B}(2,3)}{DCM_{CAM,B}(3,3)} \end{cases} \tag{3.105}$$

The three angles of Eq. 3.105 have to be minimized as possible: a null value for all the angles means the ideal condition for docking. Eventually, measurement errors are added to the attitude computation, then

$$\Psi_{meas_{CAM}} = \Psi_{trg} + \Psi_{bias_{CAM}} + \Psi_{rand_{CAM}} \tag{3.106}$$

$$\Theta_{meas_{CAM}} = \Theta trg + \Theta bias_{CAM} + \Theta rand_{CAM} \tag{3.107}$$

$$\Phi_{meas_{CAM}} = \Phi_{trg} + \Phi_{bias_{CAM}} + \Phi_{rand_{CAM}} \tag{3.108}$$

# Chapter 4

# Results

In this section numerical results produced during the progress of the PhD work will be presented and discussed. The section will focus on GNC software integration activities executed in collaboration with Thales Alenia Space. A Monte Carlo analysis has been executed in order to check the robustness of the GNC software subjected to a number of different initial conditions and spacecraft configuration initialized in the orbital simulator. Extensive discussions about results of simulations will be presented. Discussion about the test and validation campaign of the simulator software tool is reported in Appendix A.

## 4.1 Simulator and GNC Integration

The integration between the orbital simulator and the GNC software (GNC SW) has been conducted in strong collaboration with the GNC team of Thales Alenia Space. The integration process consists in the definition of a communication link between the two software, implementation of a communication library to be included in both simulator and GNC software, to check the communication link and to execute preliminary tests to check all the GNC function in a closed-loop system. Indeed, as required by ESA standards [62], to test and validate the GNC software it is required to keep separated the software which performs physical simulation, the orbital simulator, and the software which emulates the on-board computer that implements GNC algorithms. For this purpose, it has been implemented a *Transmission Control Protocol (TCP)*, a communication protocol similar to the one used by internet connections, in which it has been defined a set of messages to be exchanged during simulations by simulator and GNC software using a local connection. Implementation of such protocol has been supported by the existing `winsock` library included in the Windows-based operative system. The role assigned to the orbital simulator is the *server* role, while the GNC software has been assigned as *client*. Differently from the classical information flow used by internet connections, the server is the actor which is asking for data while the client (GNC software) receives the data request and input and process the output. The data flow is depicted in Fig. 4.1. The two executable application are launched, the simulator application first. The simulator application opens the local port and starts waiting the client connection. The GNC application is launched and it is connecting to the simulation-server local port, generating the *handshake signal*. Once the TCP connection is locked, the GNC SW is set in *listen*, the simulator executes internal initialization and starts the first simulation step. The sensors measure the state of the chaser spacecraft and the position of the target spacecraft and sensors data are sent to the GNC SW. Once data are sent, the simulator sends the command request message and then it is set in *listen*. The GNC SW detects the command request and acquires the sensors data, then it processes the input data and computes the output commands. After commands are sent, the GNC SW is set back to *listen*. The simulator detects the commands sent by the GNC SW and propagates chaser and target dynamics actuating the received GNC commands. The simulation step is completed and it starts the successive simulation step, starting from sending sensor measurement to the GNC SW with measurements of the propagated dynamics and the *send/listen* process of simulation and GNC continues until the maximum simulation time expires. When the simulation ends, a *bye bye* message is sent by the simulator to the GNC SW and both applications are closed. Data storage is executed continuously during the simulation in both simulator and GNC applications. File pointers are closed only at the end of the simulation process and results files can be read by post-processing applications.

Messages exchanged between simulator and GNC software are listed below:

- Simulator to GNC messages:
  - *GPS Data*: this message includes the output of the GPS model, including position, velocity and GPS time data.

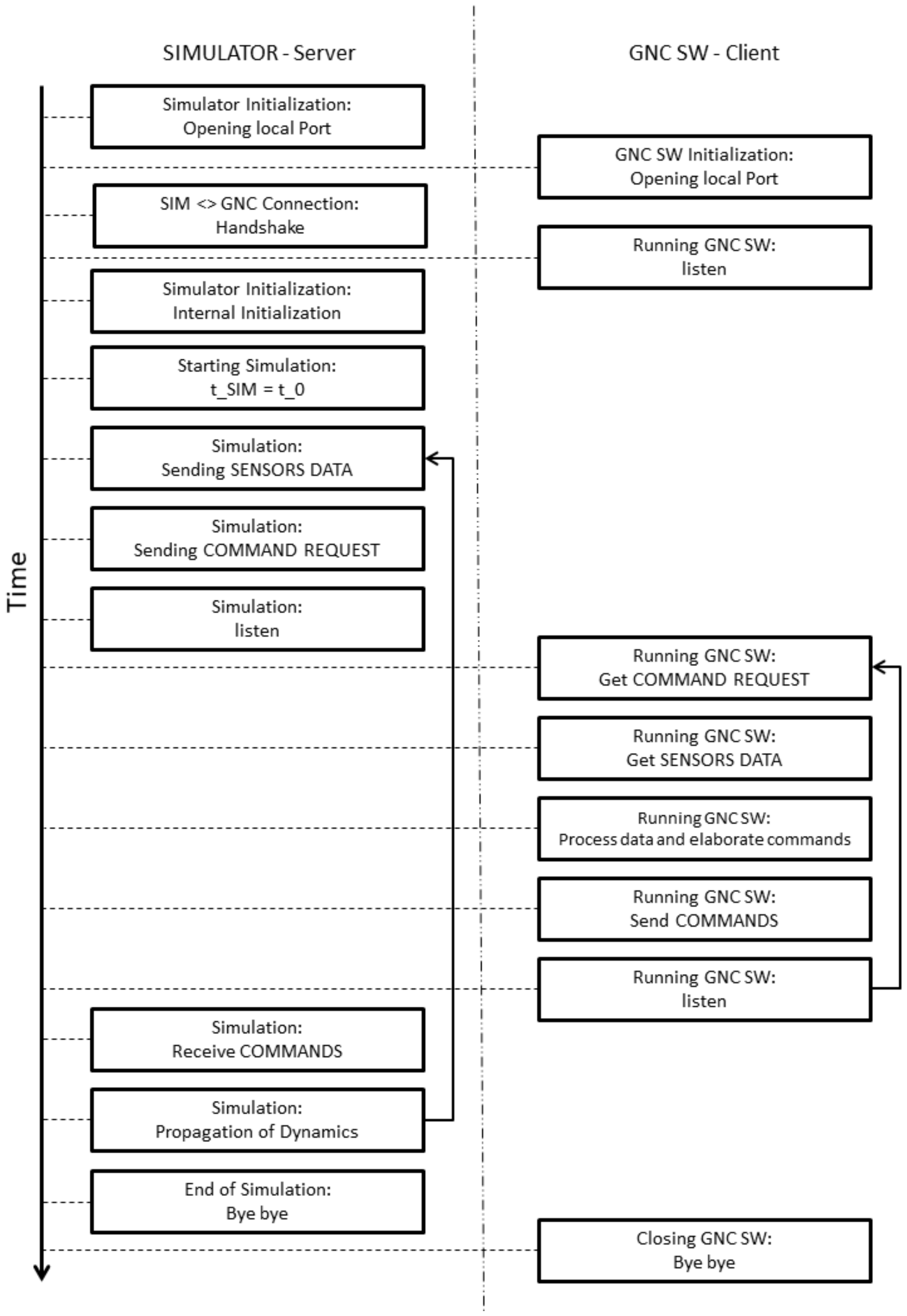

SIMULATOR - Server
GNC SW - Client
Time
Simulator Initialization: Opening local Port
GNC SW Initialization: Opening local Port
SIM <> GNC Connection: Handshake
Running GNC SW: listen
Simulator Initialization: Internal Initialization
Starting Simulation: t_SIM = t_0
Simulation: Sending SENSORS DATA
Simulation: Sending COMMAND REQUEST
Simulation: listen
Running GNC SW: Get COMMAND REQUEST
Running GNC SW: Get SENSORS DATA
Running GNC SW: Process data and elaborate commands
Running GNC SW: Send COMMANDS
Running GNC SW: listen
Simulation: Receive COMMANDS
Simulation: Propagation of Dynamics
End of Simulation: Bye bye
Closing GNC SW: Bye bye

Fig. 4.1 TCP data flow.

  - *IMU Data*: this message includes the output of the IMU sensor, comprehensive of measures of linear acceleration and angular velocity, acquisition status and sensor counter.
  - *Star Tracker Data*: this message includes the measured quaternion by the star tracker, acquisition status and sensor counter.
  - *Radio Finder Data*: this message includes range, azimuth and elevation of the detected target, acquisition status and sensor counter.
  - *LIDAR Data*: as for the Radio Finder, this message includes range, azimuth and elevation data, acquisition status and sensor counter.
  - *Camera Data*: this message includes range, azimuth and elevation of the detected target and relative attitude in terms of yaw, pitch and roll angles, as well as acquisition status and sensor counter.
  - *Reaction Wheels Telemetry Data*: this message includes data related to the actual angular velocity of the reaction wheels.

- GNC to Simulator messages:
  - *Reaction Wheel Torque*: this message includes the requested torque commands to control the reaction wheels.
  - *FCV Activation*: this message contains the Boolean vector to control the flow control valves of the thrusters system.

All these messages are exchanged during the simulation between the two executable applications of simulator and GNC software. A further aspects of Simulator/GNC integration has been the introduction of delays in order to simulate real delays in data transmission of sensors and actuators dynamical response. An input and output buffers has been implemented in the simulator application. The input buffer is related to delays induced by actuators. Since the simulator frequency, set to 100 Hz, is not high enough to model accurately the thrust rise time and inertial response of reaction wheel, hence the input buffer has been introduced to model such effects, even if it has a very limited effect on the simulation behaviour. Indeed, the maximum delays of the input buffer is set to one simulation step (0.01 s).
Differently from the input buffer, the output buffer has been introduced to model delays in processing measurements by sensors. For this reason, according to the update frequency of each sensor, the maximum size of the output buffer is set with

different value for each sensor, and specifically 10 ms for the IMU, 100 ms for LIDAR, Star Tracker and Radio Finder, 500 ms for the Camera sensor and 1 s for the GPS.

In addition, the Simulator/GNC integration process has been used to check that all the GNC functions, in particular the Control function, which provides the guidance and control commands to the spacecraft, and the sensor output runs at the proper frequency. The Control function runs at 100 Hz, the same as the main function of the simulator, and so FCV and reaction wheels commands are varying every 10 ms. The sensor update frequency has been modeled with different values according to the datasheet of the specific sensor: 1 Hz for GPS, Camera, LIDAR and Radio Finder sensors; 10 Hz for Star Tracker sensors; 100 Hz for the IMU sensor.

## 4.2 Simulation Results

Simulation results produced during the integration and validation process of the simulator and GNC application will be presented. First, of all related to the nominal maneuver performance of Modified ZEM/ZEV and Continuous Lambert-type algorithms have been evaluated and compared. The LQR guidance proposed in Section 2.2.2 has been not further investigated due to poor performance highlighted in terms of propellant consumption in preliminary tests, as deeply discussed in Section 2.2.2. The Relative Lambert Guidance has been implemented in $H_{BAR}$ corrections during the S1-S2 maneuver.

### 4.2.1 Nominal Maneuver

Results related to the nominal maneuver as described in Section 3.1.2 will be presented for both the candidate algorithms *Modified ZEM/ZEM Guidance* and *Continuous Lambert-type Guidance*. First of all, it has been tested both the algorithms without introducing measurement errors, while PWM thrust modulation and attitude control are affecting the spacecraft dynamics. Then it has been tested the algorithms including sensor noises and biases, hence activating the navigation filter. Results obtained with both the algorithms are compared with ideal results listed in Table 4.1. The ideal $\Delta V$ has been computed according to the maneuvers defined in Section 3.1.1, by summing the $\Delta V$ related to each maneuver. The $\Delta V$ for the final maneuver has been computed assuming two different terminal velocity profiles: (i) a constant velocity profile of 0.1 m/s during the whole maneuver S3-S4 and (ii) an initial velocity of 0.1 m/s for the first 450 m and then a velocity profile of 0.03 m/s for the last 50 m of the maneuver S3-S4. The adoption of two differet terminal strategies results in different $\Delta V_{tot}$ and related propellant consumption. The consumed propellant mass has been computed by using the Tsiolkovsky rocket equation

$$\beta = \frac{m_{init}}{m_{final}} = e^{\left(\frac{\Delta V_{tot}}{I_{sp}\, g_0}\right)}$$

and the burned mass has been computed by

$$m_{burn} = m_{init} - m_{final} = m_{init}\left(1 - \frac{1}{\beta}\right) \tag{4.1}$$

where $m_{init}$ is the initial mass, $m_{final}$ is the final mass, $I_{sp}$ is the specific impulse of the thruster system and $g_0$ is the gravitational acceleration at sea level.

The nominal maneuver has been tested by implementing the two guidance algorithms in different configuration of the GNC software: the Modified ZEM/ZEV algorithm has been tested using a development version of the GNC software, which includes attitude control based on LQR and a Kalman filter for navigation, while the Continuous Lambert-type Guidace has been tested usign the GNC software developed in collaboration with Thales Alenia Space, which includes attitude control based on PID controller and a complete absolute and relative Extended Kalman Filters for the navigation function.

Indeed, the definition and implementation of the Continuous Lambert-type Guidance has been performed after identification of the high sensitivity to measurement noise of the Modified ZEM/ZEV algorithm during the Simulator/GNC software integration. Preliminary integration tests have been executed by introducing in the development version of the GNC an active attitude control based on LQR and simula-

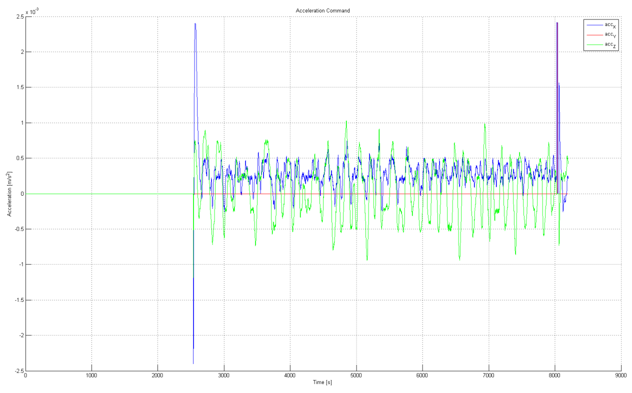
| Terminal Velocity Profile | $\Delta V_{tot}$ [m/s] | Burned Mass [kg] |
|---|---|---|
| Constant Velocity | 3.36 | 15.28 |
| Variable Velocity | 4.96 | 22.54 |

Table 4.1 Ideal $\Delta V$ and propellant consumption.

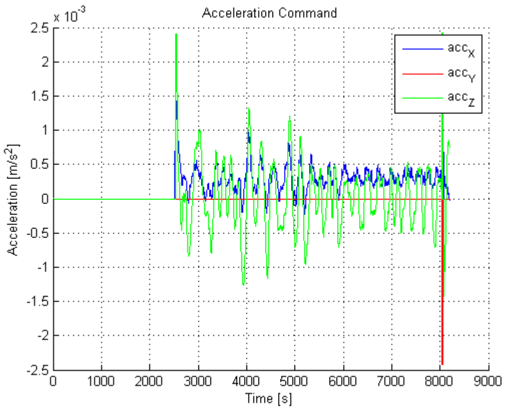
tion of measurement noise as additive quantities to the real position and velocity with decreasing magnitude according to the Chaser-Target relative distance. Introduction of such errors has been evaluated in cooperation with the team of Thales Alenia Space. Such low time varying errors did not affect performance of the Modified ZEM/ZEV Guidance, which drove the Chaser trough the complete RVD maneuver using limited propellant comparable with the ideal value. The sensitivity to measurement noise has been highlighted when complete integration tests have been executed including the navigation filter. Even though the navigation filter produced a very accurate state estimation, the output of the filter resulted still noisy with respect to simulated navigation errors considered in previous tests, and the Modified ZEM/ZEV algorithm didn't work properly. A sensitivity analysis of the Modified ZEM/ZEV algorithm has been executed in order to evaluate a proper tuning of the prediction horizon parameter to improve performance of the algorithm. Results of this analysis are briefly summarized in Fig. 4.2, where it is depicted the acceleration commands generated by the guidance function. The sensitivity analysis has been executed focusing on the first maneuver. In addition, the navigation filter is active but bias and random errors of sensors has been set to zero, hence the analysis has been executed by a noisy navigation output related to null measurement noises. As it can be seen in Fig. 4.2, the guidance command results quite noisy and with a non negligible effect on the $R_{BAR}$ axis, which is ideally not commanded for execution of the S1-S2 maneuver. As it will deeply discussed in the following sections, such sensitivity of the Modified ZEM/ZEV Guidance brought to development and implementation of the alternative Continuous Lambert-type Guidance algorithm.

### Modified ZEM/ZEV: Ideal Case

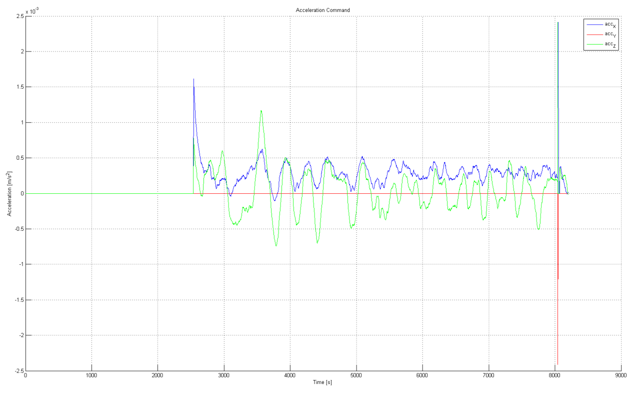
The ZEM/ZEV algorithm has been tested executing the ideal RVD maneuver. Complete results are depicted in Fig. 4.3 and Fig. 4.4. The Modified ZEM/ZEV algorithm allows to complete the rendezvous and docking mission with high accuracy, as it can be noticed in the complete maneuver in Fig. 4.3a and in the final approach in Fig. 4.3d. In addition, all the intermediate waypoints have been reached with high accuracy as in Fig. 4.3b and Fig. 4.3c. The attitude control, based on LQR, has been tuned properly, as it can be seen in Fig. 4.4a and Fig. 4.4b which depicts respectively the relative quaternion and the relative angular velocity, as well as the control torque requested by reaction wheels is below the maximum torque during all the



(a) Prediction horizon of 50 s.



(b) Prediction horizon of 100 s.



(c) Prediction horizon of 200 s.

Fig. 4.2 Modified ZEM/ZEV sensitivity analysis.

maneuver, as in Fig. 4.4e. The guidance force command produced by the Modified ZEM/ZEV algorithm is depicted in Fig. 4.4c: as it can be seen, the commanded force is very close to the ideal value which can be computed by Eq. 3.8 and Eq. 3.14, which are $F_x = 0.8801$ N for the orbit raising maneuver and $F_z = 0.7334$ N for the second maneuver. Values computed by the Modified ZEM/ZEV algorithm are about $F_x = 0.9709$ N and $F_z = 0.7920$ N, which are comparable with the ideal results. Similar considerations can be assumed for the final approach maneuver. Finally, the propellant consumption of about 19 kg is comparable with the ideal prescribed one. The higher propellant consumption is due to two main factors:

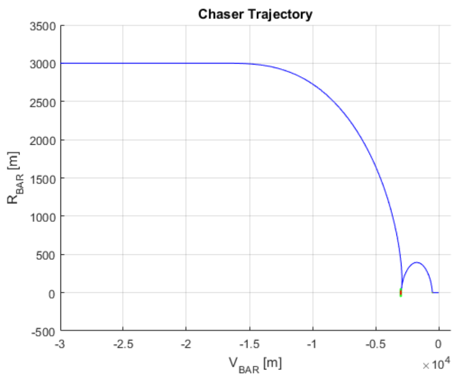
1. in the ideal computation of the burned mass in Table 4.1 it has not been included the propellant consumption due to station keeping purposes, since the ideal maneuvers S1-S2 and S2-S3 ends both with zero velocity exactly in the prescribed waypoint;

2. the thrust is modulated with a PWM technique, hence induced thrust errors causes an higher propellant consumption to follow the prescribed trajectory.

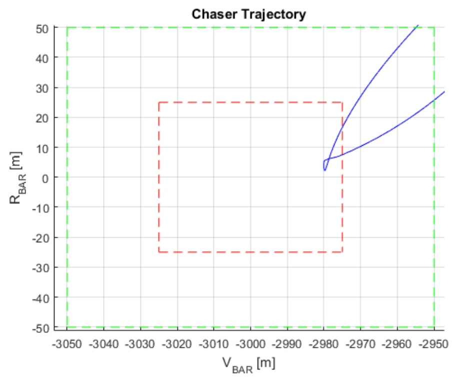
As it can be seen in Fig. 4.4c, peaks in the commanded force are related to station keeping commands, since the spacecraft does not reach exactly the location of the waypoint, then requiring additional effort to reach station keeping conditions. In the ideal case, thrust errors due to the induced misalignment of the non-ideal attitude control has a negligible effect.

### Modified ZEM/ZEV: Measurement Errors

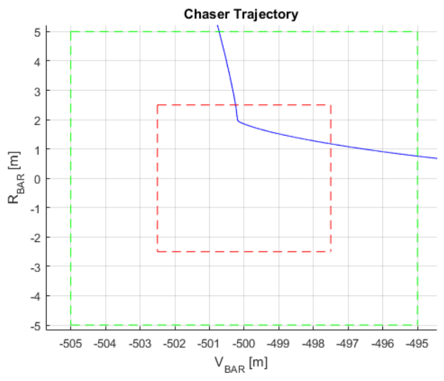
The good performance of the Modified ZEM/ZEV algorithm have been tested introducing measurement errors. In Fig. 4.5, the estimated state computed by the navigation filter is depicted in *magenta* lines. Even if the complete maneuver and the final approach has been completed successfully, as in Fig. 4.5a and Fig. 4.5d, as well as the attitude control is working properly, the guidance command produced by the Modified ZEM/ZEM algorithm is computing the high variable command of Fig. 4.6c, which causes a very high propellant consumption of about 46 kg. These results highlights the high sensitivity of the Modified ZEM/ZEV algorithm to measurement noise. This sensitivity is to be attributed to the short term prediction horizon.



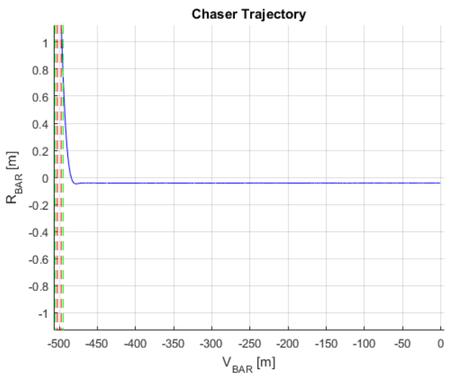
(a) $V_{BAR}$-$R_{BAR}$ maneuver.



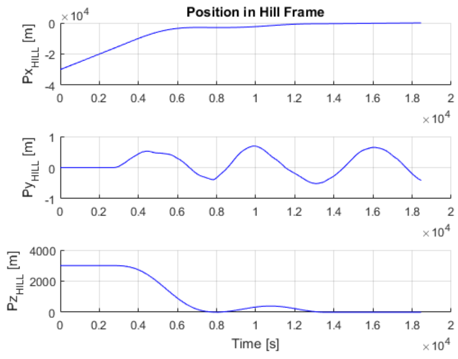
(b) Zoom on waypoint S2.



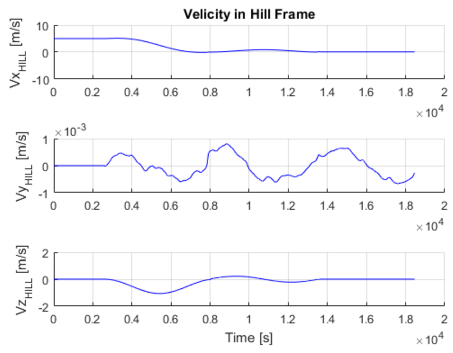
(c) Zoom on waypoint S3.



(d) Zoom on final approach S3-S4.

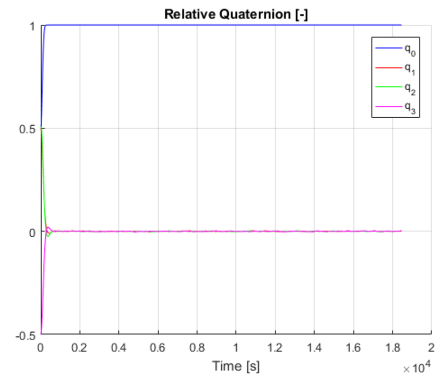


(e) Position in Hill's frame.

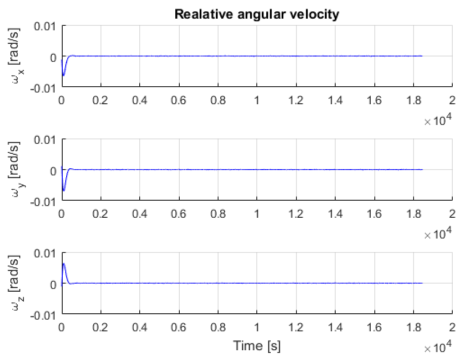


(f) Velocity in Hill's frame.

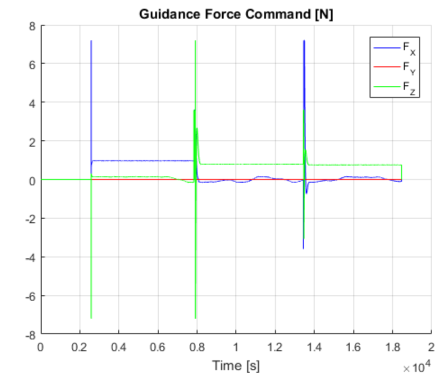
Fig. 4.3 Nominal maneuver with Modified ZEM/ZEV (1).



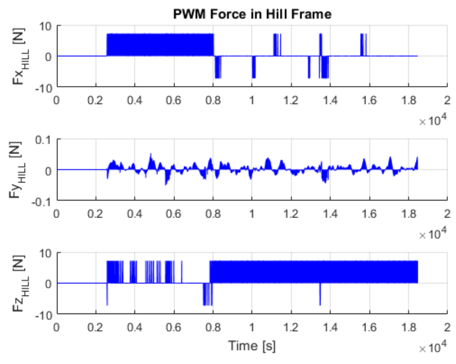
(a) Relative quaternion.



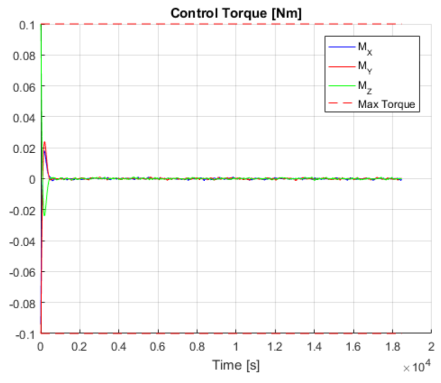
(b) Relative angular velocity.



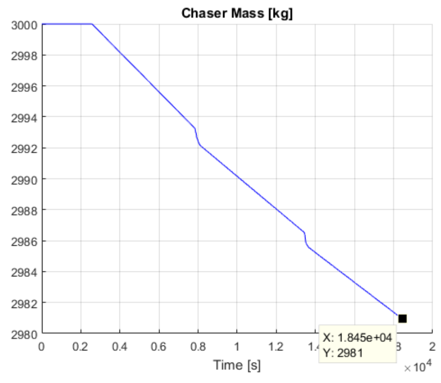
(c) Guidance force command.



(d) Thrust force - PWM.

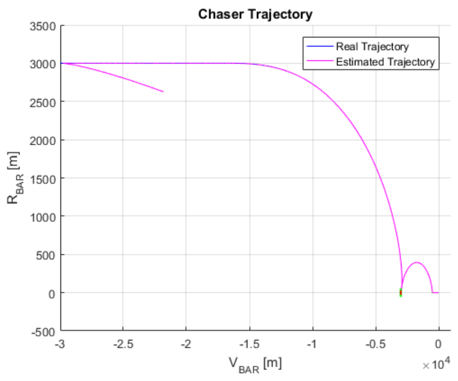


(e) Control torque.

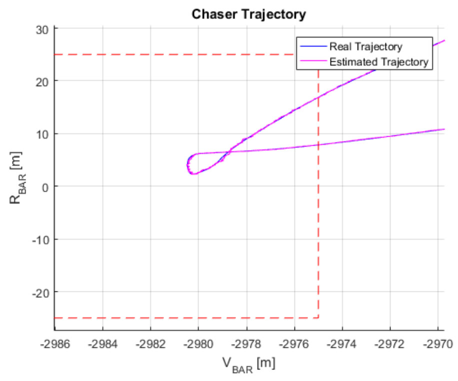


(f) Chaser mass

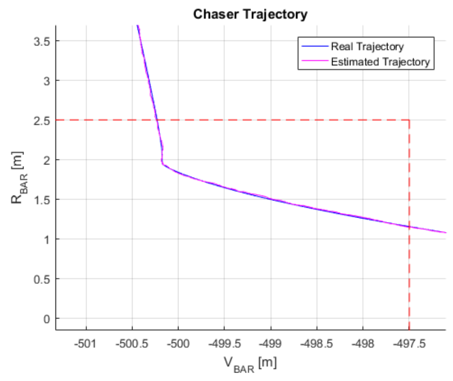
Fig. 4.4 Nominal maneuver with Modified ZEM/ZEV (2).



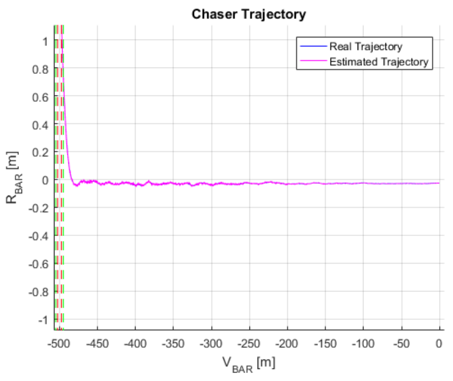
(a) $V_{BAR}$-$R_{BAR}$ maneuver.



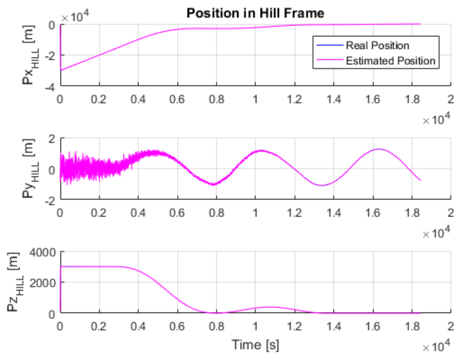
(b) Zoom on waypoint S2.



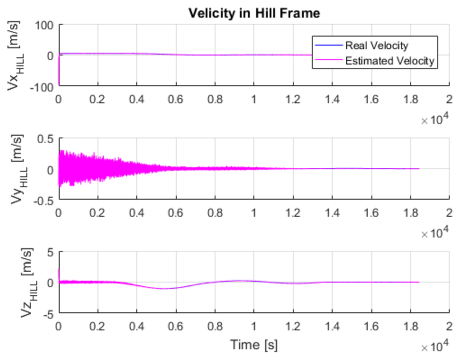
(c) Zoom on waypoint S3.



(d) Zoom on final approach S3-S4.

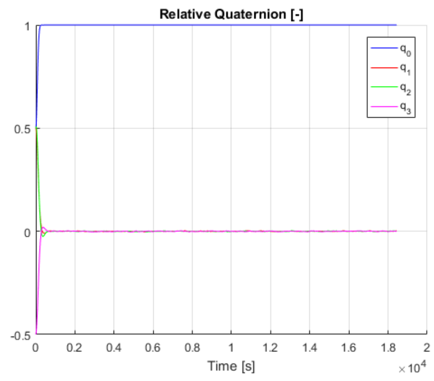


(e) Position in Hill's frame.

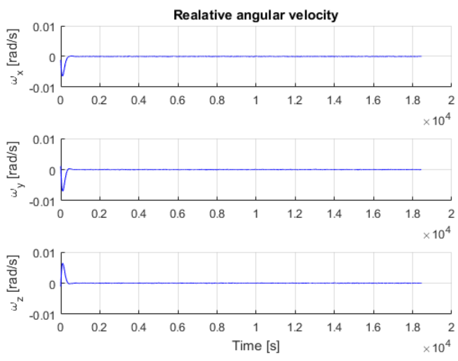


(f) Velocity in Hill's frame.

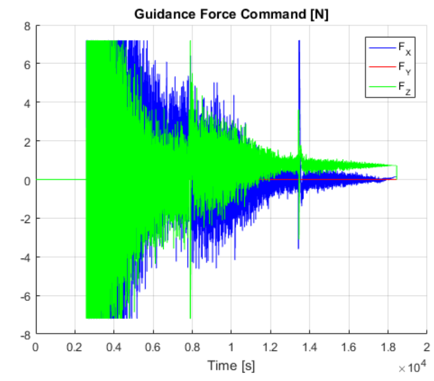
Fig. 4.5 Modified ZEM/ZEV with measurement errors (1).



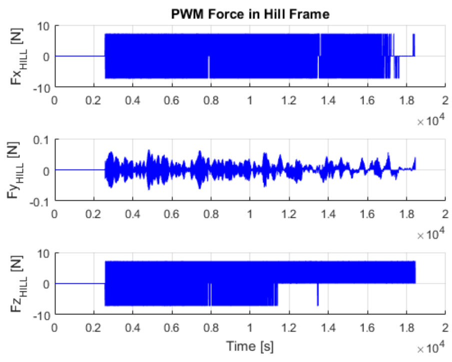
(a) Relative quaternion.



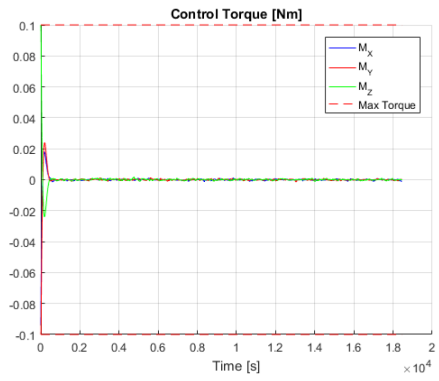
(b) Relative angular velocity.



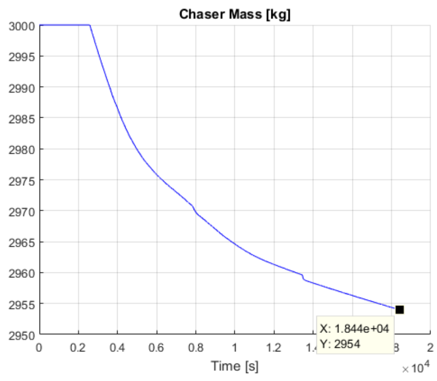
(c) Guidance force command.



(d) Thrust force - PWM.



(e) Control torque.



(f) Chaser mass

Fig. 4.6 Modified ZEM/ZEV with measurement errors (2).

### Modified ZEM/ZEV: Additional Filter

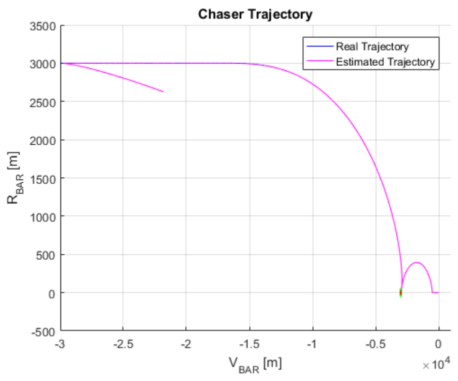
An attempt to improve performance of the Modified ZEM/ZEV algorithm has been tested implementing an additional filter in the guidance function. Such additional filter has been introduced with the purpose to further smooth the measurement noise which is the main responsible of the high propellant consumption. This goal has been partially achieved, as highlighted by the resulting propellant consumption of about 30 kg (see Fig. 4.8f) which is still the double of the ideal one. Indeed, even if the resulting guidance command computed with the additional filter is smoother than the command with a single filter implementation, but it is still too noisy, as shown in Fig. 4.8c.

### Continuous Lambert-type: Ideal Case

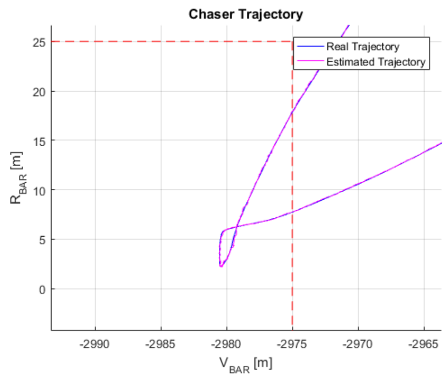
As mentioned before, poor performances of the Modified ZEM/ZEV algorithm in terms of propellant consumption brought to the development of the Continuous Lambert-type Guidance. This algorithm has been tested within the GNC software developed in collaboration with Thales Alenia Space for the Space-Tug GNC subsystem. The configuration of such GNC is more detailed with respect to the development version of the GNC SW used to test the Modified ZEM/ZEV Guidance, which implements simpler functions. The complete GNC software includes:

- a *Flight Manager* function which switches GNC modes according by the mission phases to execute;
- a *Navigation* function which implements two complete Extended Kalman filters for relative and absolute navigation and for attitude and position estimation;
- a *Control* function based on PID controller, implementing thrust modulation and reaction wheels de-saturation;
- a *Guidance* function implementing the Continuous Lambert-type Guidance.

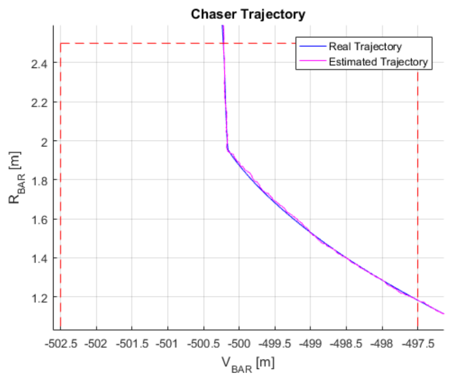
Results of nominal maneuver without measurement errors are depicted in Fig. 4.9 and Fig. 4.10. Overall performance are good an comparable with the Modified ZEM/ZEV in ideal condition, even if the final approach produces different trajectory



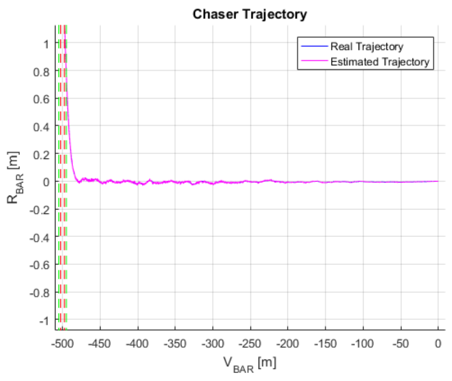
(a) $V_{BAR}$-$R_{BAR}$ maneuver.



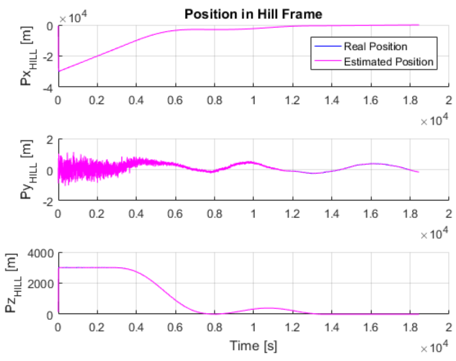
(b) Zoom on waypoint S2.



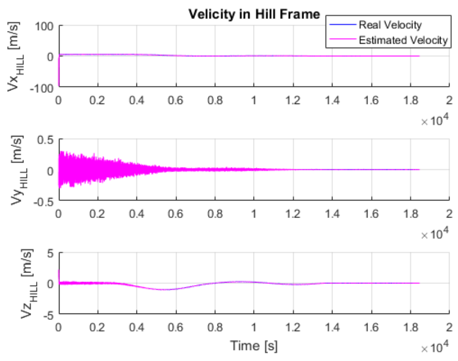
(c) Zoom on waypoint S3.



(d) Zoom on final approach S3-S4.

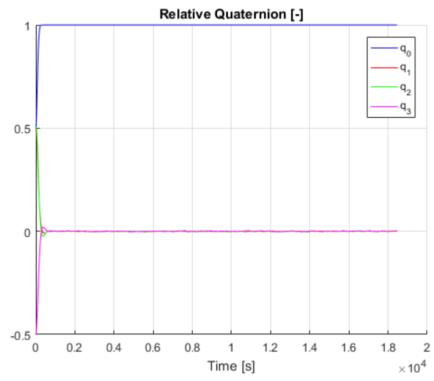


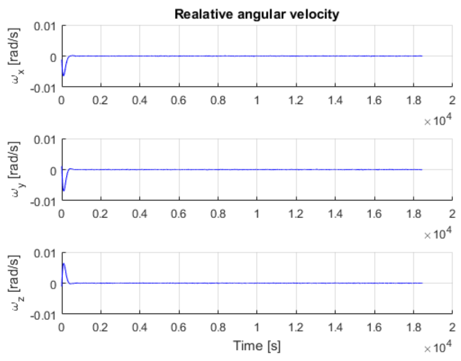
(e) Position in Hill's frame.

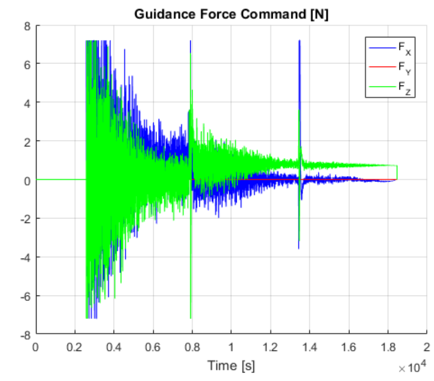


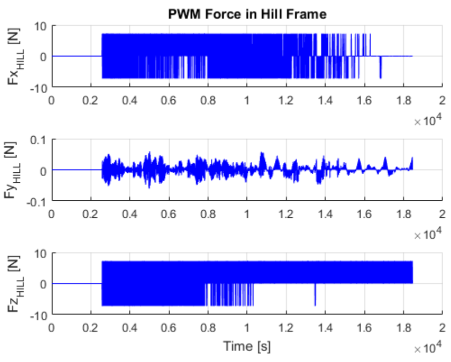
(f) Velocity in Hill's frame.

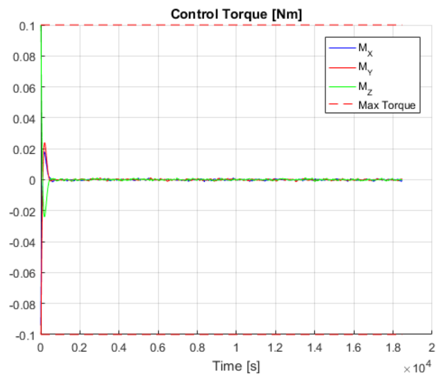
Fig. 4.7 Modified ZEM/ZEV with additional filter (1).


(a) Relative quaternion.


(b) Relative angular velocity.


(c) Guidance force command.


(d) Thrust force - PWM.


(e) Control torque.

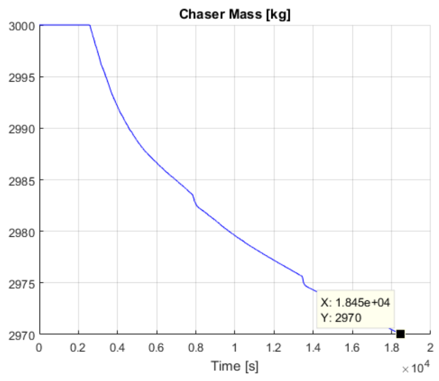
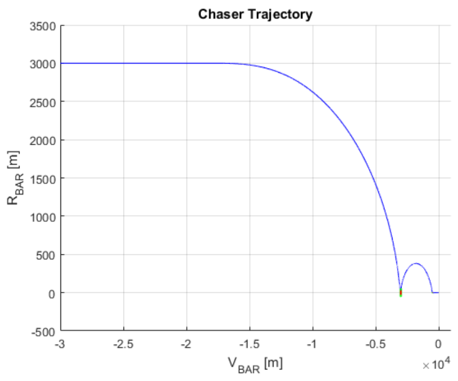

(f) Chaser mass

Fig. 4.8 Modified ZEM/ZEV with additional filter (2).

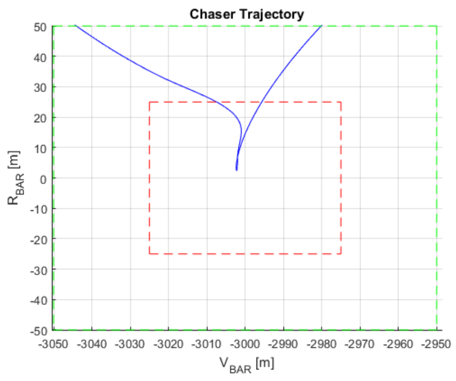
due to the fact that they are different algorithms, as it can be seen in Fig. 4.9d. The guidance command depicted in Fig. 4.10c is more noisy than the Modified ZEM/ZEV command generated in similar conditions but it is comparable with it. Moreover, the attitude control action requires more effort (see Fig. 4.10e) since the control algorithm is based on different control laws, even if attitude control performance are still good (Fig. 4.10a and Fig. 4.10b). The resulting propellant consumption is about 25 kg, which is comparable with ideal consumption computed in Table 4.1 for the variable velocity profile implemented in the final maneuver by the Continuous Lambert-type algorithm.

### Continuous Lambert-type: Measurement Errors

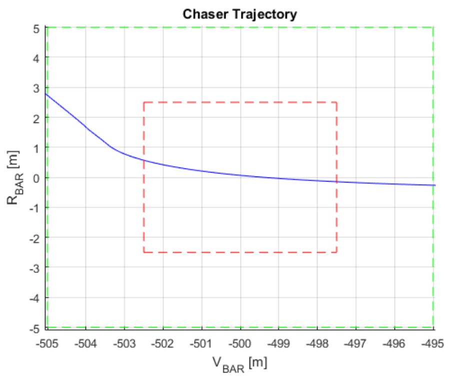
As previously done for the Modified ZEM/ZEV Guidance, the Continuous Lambert-type Guidance has been tested by introducing measurement errors. Complete results are depicted in Fig. 4.11 and Fig. 4.12. The trajectory and attitude control maintains good performances as in the ideal case. The guidance command results to be noisy as well, see Fig. 4.12c, but with a very smoother behaviour with respect to the Modified ZEM/ZEV subjected to measurement errors. Since the implemented Continuous Lambert-type algorithm is different from the Modified ZEM/ZEV, in the terminal phase there is a huge effort in controlling the $H_{BAR}$ axis as well. Although it seems to have performance comparable with the Modified ZEM/ZEV affected by measurement errors, the propellant consumption is of about 29 kg, which is less than the consumption of the Modified ZEM/ZEV with additional filter which drives the same maneuver. This improvement is related to the fact that the computed guidance command is generated in order to correct the position and velocity error over the whole remaining maneuver, instead of correcting the trajectory over a reduced time span of few tens of seconds as the Modified ZEM/ZEV. Due to its better performance, the Continuous Lambert-type Guidance algorithm has been selected to be implemented in the GNC of the Space-Tug developed in SAPERE STRONG.



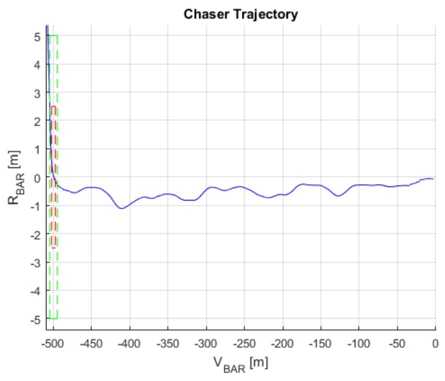
(a) $V_{BAR}$-$R_{BAR}$ maneuver.



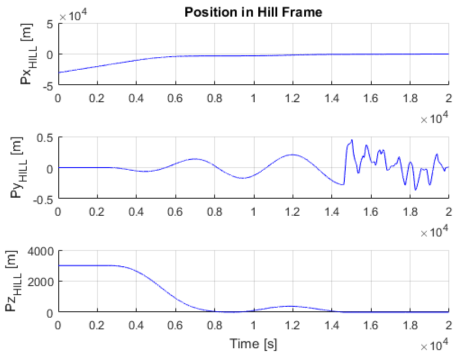
(b) Zoom on waypoint S2.



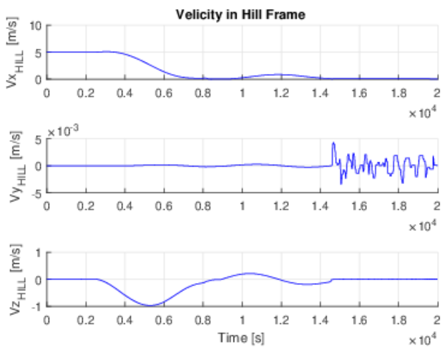
(c) Zoom on waypoint S3.



(d) Zoom on final approach S3-S4.

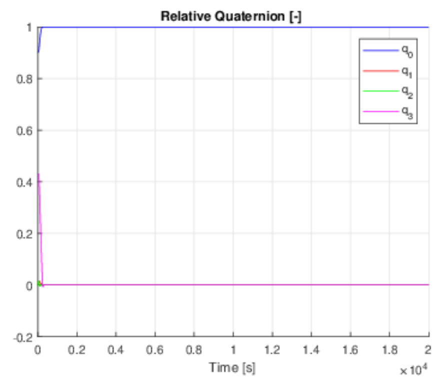


(e) Position in Hill's frame.

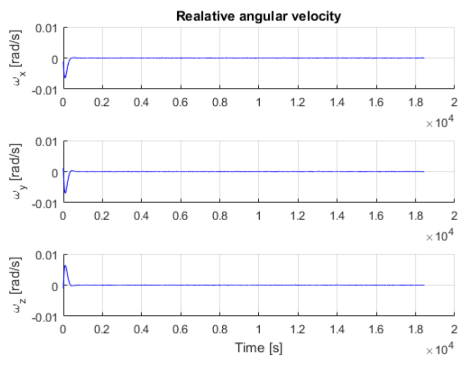


(f) Velocity in Hill's frame.

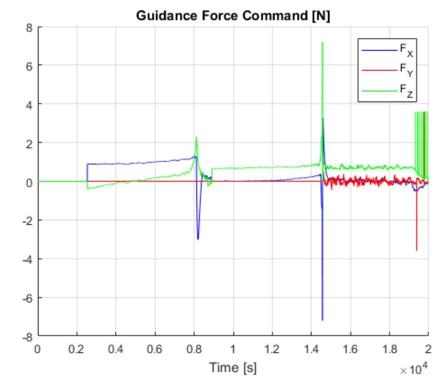
Fig. 4.9 Nominal maneuver with Continuous Lambert-type Guidance (1).



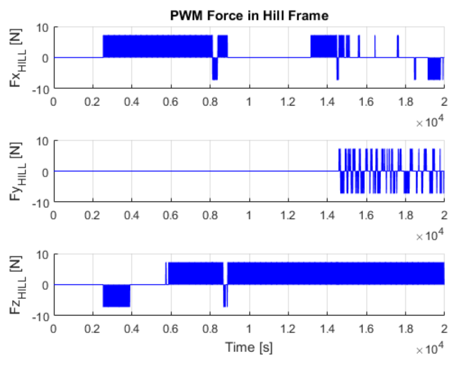
(a) Relative quaternion.



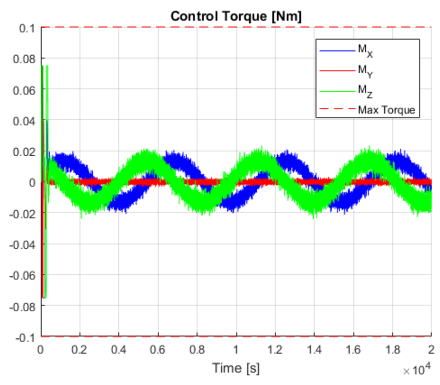
(b) Relative angular velocity.



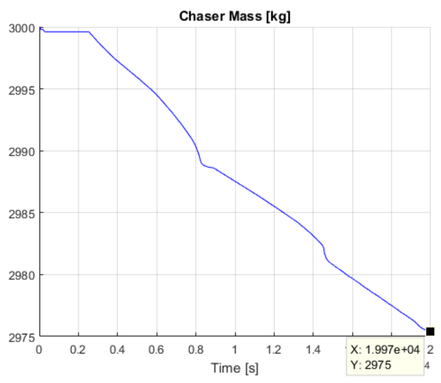
(c) Guidance force command.



(d) Thrust force - PWM.

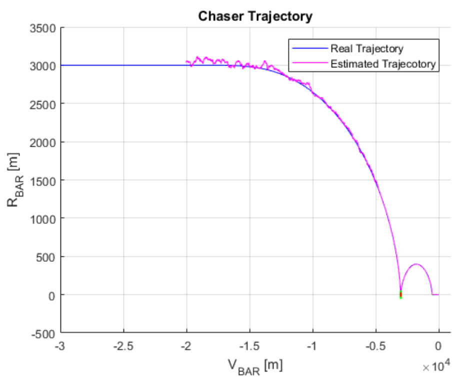


(e) Control torque.

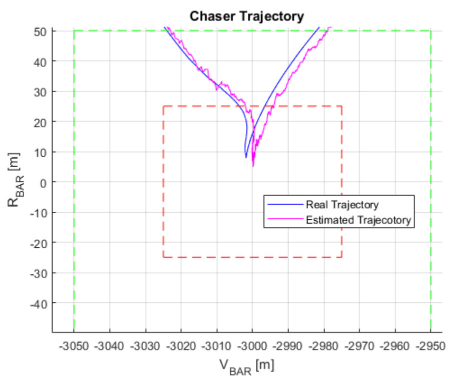


(f) Chaser mass

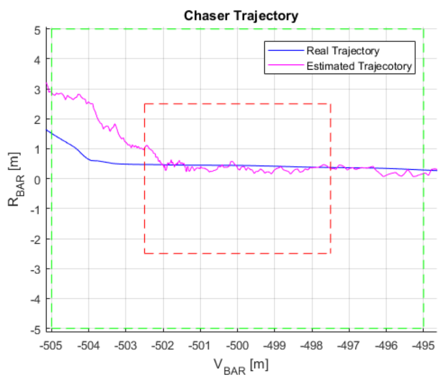
Fig. 4.10 Nominal maneuver with Continuous Lambert-type Guidance (2).



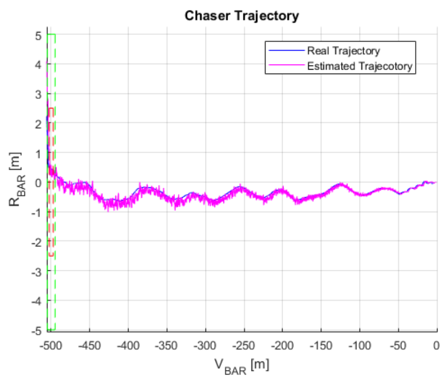
(a) $V_{BAR}$-$R_{BAR}$ maneuver.



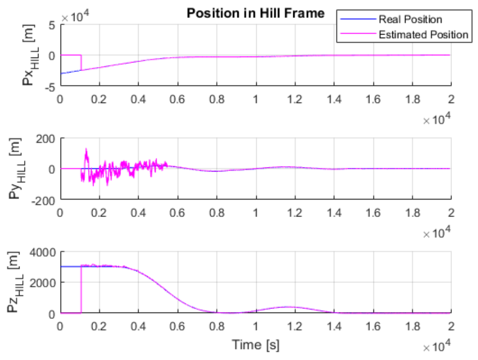
(b) Zoom on waypoint S2.



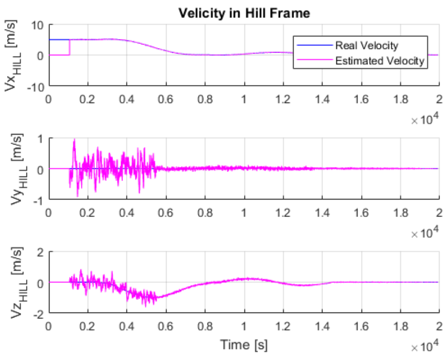
(c) Zoom on waypoint S3.



(d) Zoom on final approach S3-S4.

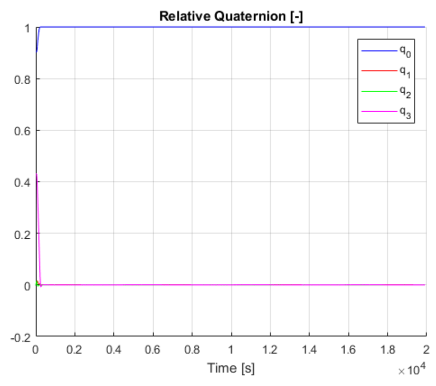


(e) Position in Hill's frame.

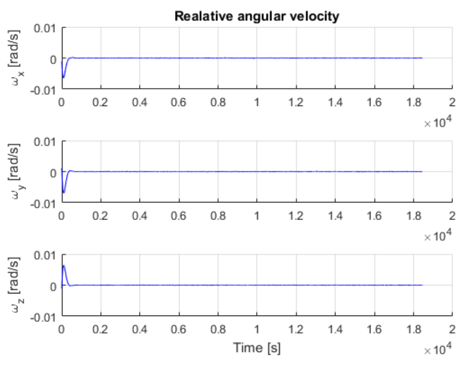


(f) Velocity in Hill's frame.

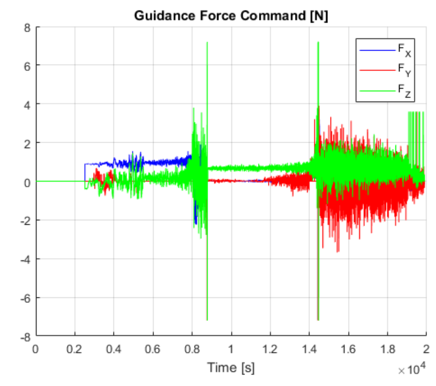
Fig. 4.11 Continuous Lambert-type Guidance with measurement errors (1).



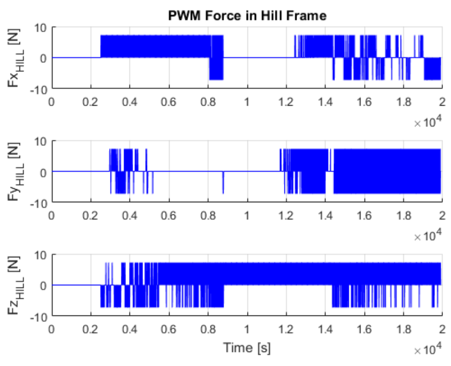
(a) Relative quaternion.



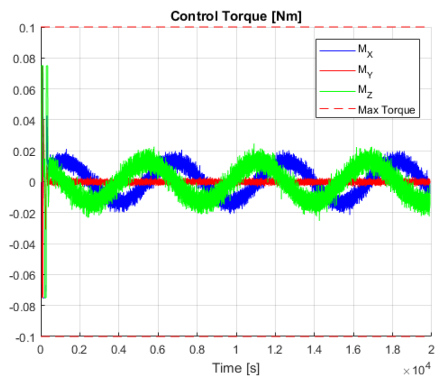
(b) Relative angular velocity.



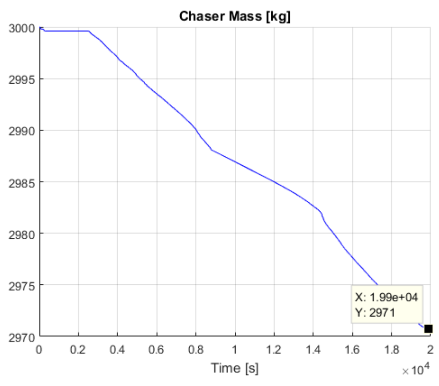
(c) Guidance force command.



(d) Thrust force - PWM.



(e) Control torque.



(f) Chaser mass

Fig. 4.12 Continuous Lambert-type Guidance with measurement errors (2).

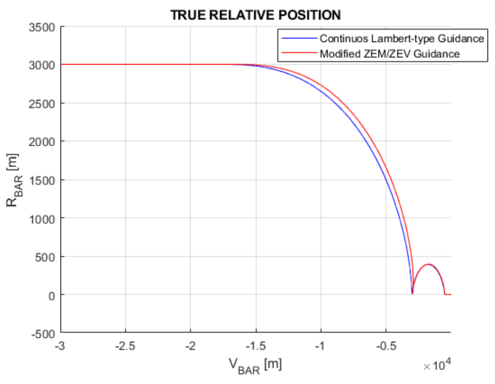
### 4.2.2 Algorithms Comparison

A comparative analysis of the two algorithms presented in previous section is provided, in order to better understand and compare performance of the Modified ZEM/ZEV and the Continuous Lambert-type guidance algorithm tested in similar initial conditions and to investigate their limitations, being able to evaluate the applicability range of each algorithm. The algorithms are compared in both the ideal case and including measurement errors, i.e. activating the navigation filter. The comparison of both the algorithm is limited to study the $V_{BAR}$-$R_{BAR}$ plane, which is the most sensitive to the differences in the guidance algorithm implemented.

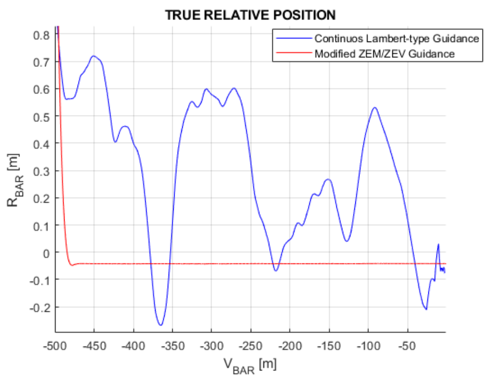
#### Ideal Case

Comparison of the two algorithms in the ideal case, i.e. without introducing measurement errors, is summarized in Fig. 4.13 to Fig. 4.15. In the ideal case, the complete RVD maneuver is executed completely and successfully driven by both the algorithms, even if there are small differences, especially in execution of the first maneuver. According to Fig. 4.13a, the Modified ZEM/ZEV algorithm seems to reach all the waypoints with higher accuracy than the Contiuous Lambert-type algorithm, and the final approach driven by the Modified ZEM/ZEM Guidance follows a very good straight line path, as depicted in Fig. 4.13b.

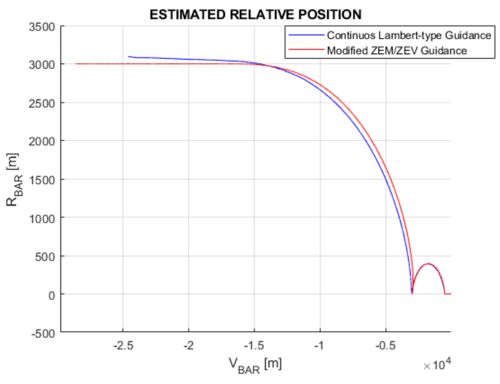
The guidance force command is depicted in Fig. 4.14, and it is plotted with respect to the $V_{BAR}$ distance, instead of with respect to time: using this plotting strategy it is possible to synchronize the guidance command computed by the two algorithms, since the maneuver is completed in a different time span depending by which algorithm is implemented, as well as the waypoints are reached in different time, which make not possible a satisfactory comparison of the two guidance commands. Both the guidance commands presents a quite regular and constant shape, which well fit the ideal force command which should be applied to execute each phase of the complete RVD maneuver (only a constant force along $V_{BAR}$ in the first maneuver and a constant force along $R_{BAR}$ in the second maneuver and in the final approach). As mentioned before, peaks in the guidance commands are related to station keeping maneuvers. In Fig. 4.14b it is highlighted the guidance command computed in the final approach: the command computed by the Modified ZEM/ZEV is smoother than the command computed by the Continuous Lambert-type algo-



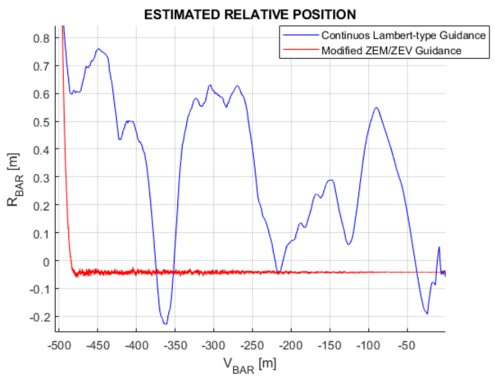
(a) $V_{BAR}$-$R_{BAR}$ true trajectory.



(b) $V_{BAR}$-$R_{BAR}$ true trajectory of final approach.

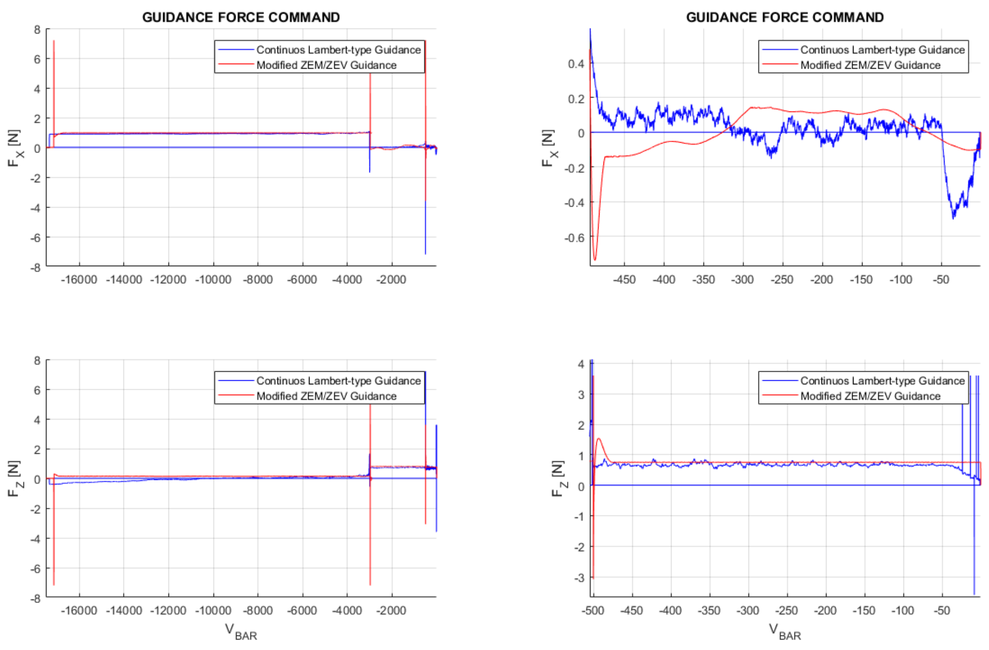


(c) $V_{BAR}$-$R_{BAR}$ estimated trajectory.



(d) $V_{BAR}$-$R_{BAR}$ estimated trajectory of final approach.

Fig. 4.13 True and estimated trajectories in the $V_{BAR}$-$R_{BAR}$ plane - Ideal Case.



(a) Force command. (b) Force command during the final approach.

Fig. 4.14 Guidance Force command - Ideal Case.

rithm. The propellant mass consumption is depicted in Fig. 4.15. The propellant consumption is an additional performance index to evaluate overall performance of the guidance algorithm. In the ideal case, the Modified ZEM/ZEV algorithm causes a lower propellant consumption than the Continuous Lambert-type algorithm. According to results yet presented, the Modified ZEM/ZEV algorithm has better overall performance in executing the complete nominal RVD maneuver.

### Introducing Measurement Errors

Comparison of the two algorithms introducing measurement errors is summarized in Fig. 4.16 to Fig. 4.18. As it is depicted in Fig. 4.16c the estimated trajectory is very different between the two guidance algorithm: this is due to the adoption of different navigation filters in the GNC software, which produces an apparently better position estimation while using the Modified ZEM/ZEV algorithm. Indeed, even if measurement errors are introduced, the Modified ZEM/ZEV algorithm ensures a smoother trajectory profile and a better tracking of the ideal trajectory. A completely

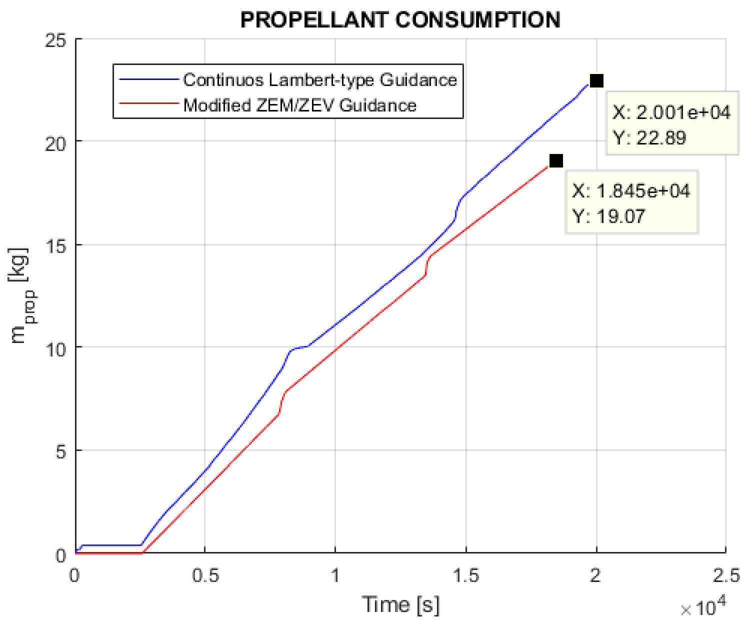
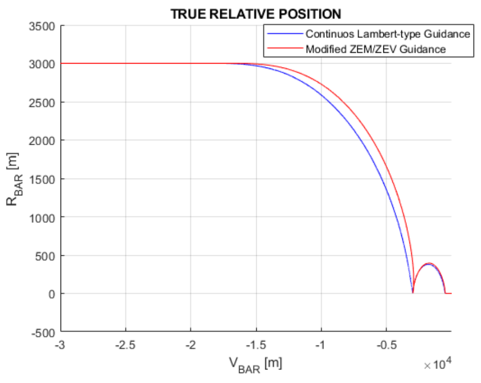
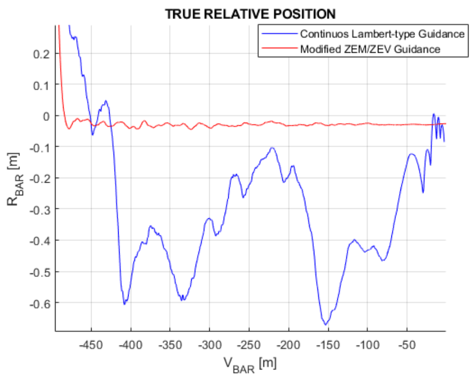


Fig. 4.15 Propellant consumption - Ideal Case.

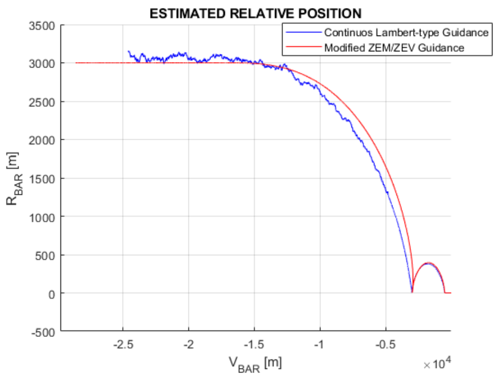
different result is found by evaluating the guidance commands computed by the two algorithms. As depicted in Fig. 4.17, the guidance command computed by the Modified ZEM/ZEV algorithm is continuously switching between the positive and negative saturation limit for almost all the maneuver, while the Continuous Lambert-type algorithm computes a more regular command along both $V_{BAR}$ and $R_{BAR}$ axis, while oscillations and a noise behaviour is present anyway. Such oscillatory command computed by the Modified ZEM/ZEV is expected to cause an higher propellant consumption than the Continuous Lambert-type algorithm, as it is depicted in Fig. 4.18 indeed. Even if the Modified ZEM/ZEV presents better performance in terms of maneuver execution, the Continuous Lambert-type Guidance allows a propellant saving of more than 15 kg, however the *relaxed* maneuver execution, with poorer overall performance than the Modified ZEM/ZEV, ensure to reach terminal conditions within the prescribed accuracy required by the docking mechanism. Since the objective of the present dissertation is the design of a GNC software which ensure to execute a complete rendezvous and docking maneuver satisfying the required terminal conditions with the less propellant consumption, the Continuous Lambert-type algorithm has been selected to be implemented in the GNC software of the Space-Tug, and extensive analysis to evaluate performance of the algorithm



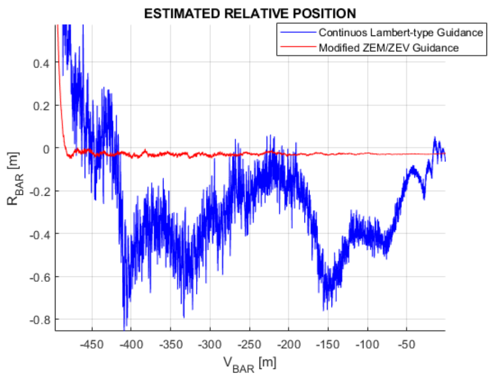
(a) $V_{BAR}$-$R_{BAR}$ true trajectory.



(b) $V_{BAR}$-$R_{BAR}$ true trajectory of final approach.

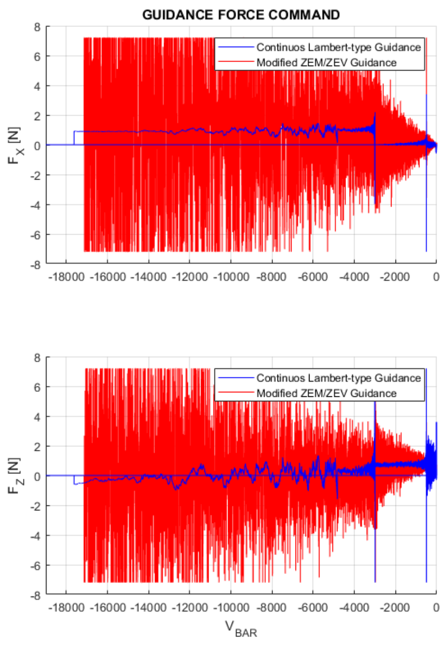


(c) $V_{BAR}$-$R_{BAR}$ estimated trajectory.

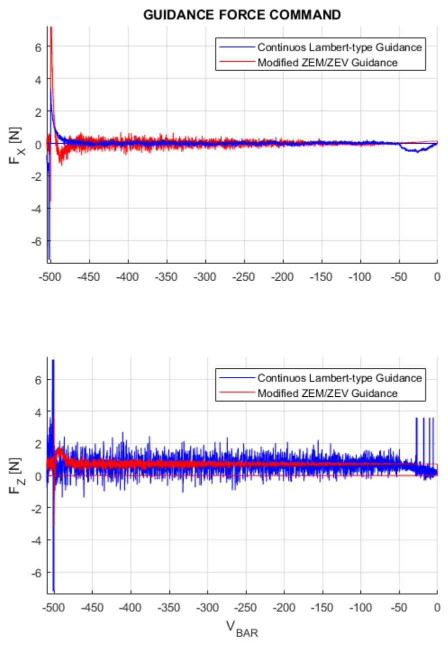


(d) $V_{BAR}$-$R_{BAR}$ estimated trajectory of final approach.

Fig. 4.16 True and estimated trajectories in the $V_{BAR}$-$R_{BAR}$ plane - Measurement Errors.



(a) Force command.



(b) Force command during the final approach.

Fig. 4.17 Guidance Force command - Measurement Errors.

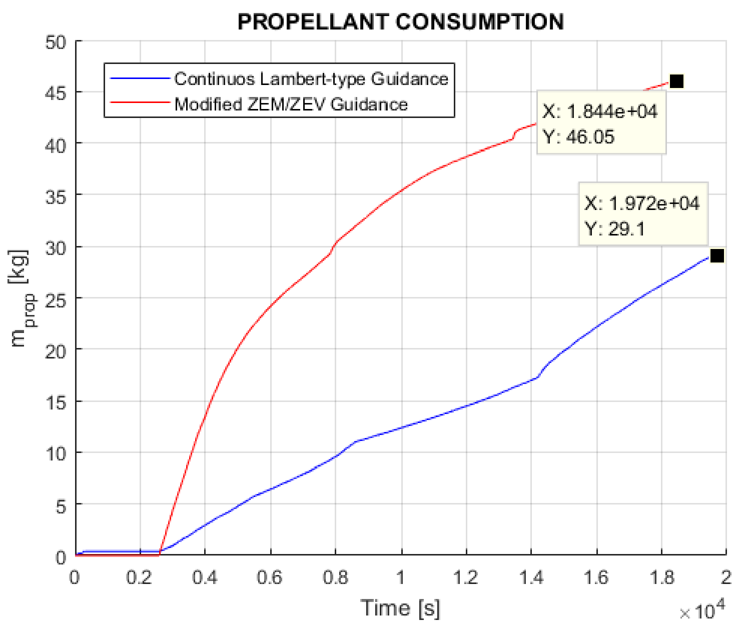


Fig. 4.18 Propellant consumption - Measurement Errors.

in executing non-nominal maneuver, as well as stress tests as Monte Carlo analysis have been completed and summarized in the following of the dissertation.

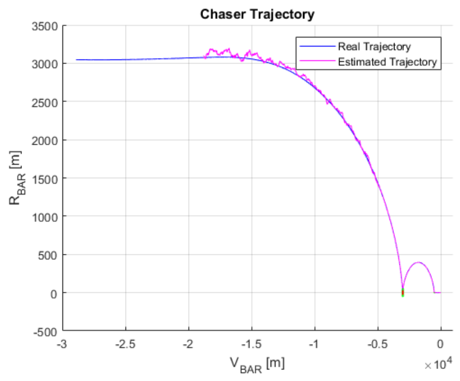
### 4.2.3 Non-nominal Maneuver

After having deeply investigated performance of the Continuous Lambert-type Guidance in executing the nominal RVD maneuver as described in Section 3.1.2, the proposed algorithm has been tested introducing a number of parameter dispersion of initial conditions. Indeed, it is not possible to model exactly anything existing in the real word, since each model developed by humans is subjected to limitation in the knowledge of the specific phenomenon. In addition, some configuration parameters, such as inertia matrix of a spacecraft or measurement biases of an IMU sensor are not known *a priori* and they are subjected to be different from nominal values. Initial orbit of chaser and target spacecraft is affected to uncertainties as well, for example, the orbit deployment of the Chaser spacecraft is well known but residual errors due to orbit injection or errors in determining its position may affects initial conditions. For this purpose, a dispersion with respect to nominal conditions has been implemented. Dispersion implemented are related to:

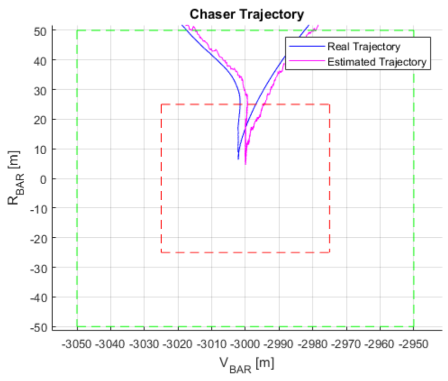
- *Orbital dispersion*: the initial orbit of Chaser and Target is subjected to initial position and velocity errors computed randomly within defined limits. Initial attitude of the chaser spacecraft is subjected to dispersion as well.
- *Configuration dispersion*: mass and inertia of the Chaser are computed introducing errors, including wet and dry properties as well as initial mass configuration and drag coefficient for computation of aerodynamic drag disturbance.
- *Actuators dispersion*: reaction wheels and thrusters are subjected to dispersion too. Mounting errors in terms of misalignment and positioning errors are initialized for both the actuators. Manufacturing errors such as non-nominal inertia of reaction wheels and nozzle defects are introduced too.
- *Sensors dispersion*: dispersion in modeling sensors are related to mounting errors (misalignment and positioning errors) and initialization of biases for all sensors.
- *Buffer dispersion*: as mentioned in Section 4.1, the input and output buffer has been introduced to model actuation and measurements delays. For this purpose, dispersion in the input and output buffer size has been introduced in order to test the GNC SW under different conditions.

In order to take into account all possible configuration of parameter dispersion, a Monte Carlo analysis has been executed, initializing the orbital simulator each time with different initial condition computed randomly within dispersion limits. This analysis will be deeply discussed in Section 4.3.

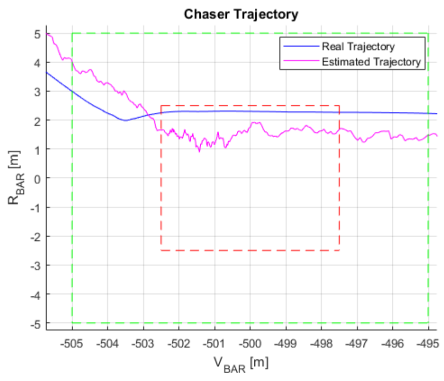
An example of RVD maneuver subjected to all the described dispersion is depicted in Fig. 4.19 and Fig. 4.20. Orbital dispersion can be seen in Fig. 4.19a, in which the free drift maneuver is not a perfect straight line as in previous cases. In addition, the greater command along $\mathrm{H_{BAR}}$, especially in the first maneuver, is suggesting that the Chaser is not in the same orbital plane of the Target ($\mathrm{H_{BAR}}$ relative error) and hence such error is compensated during the maneuver S1-S2. In addition, the initial mass of the Chaser is grater (about 3064 kg) than the nominal initial mass (3000 kg), and the propellant consumption of this maneuver is about 33 kg, mainly due to much orbital corrections to execute. Even if the actual maneuver has different



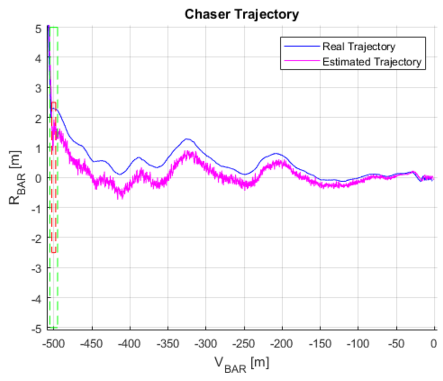
(a) $V_{BAR}$-$R_{BAR}$ maneuver.



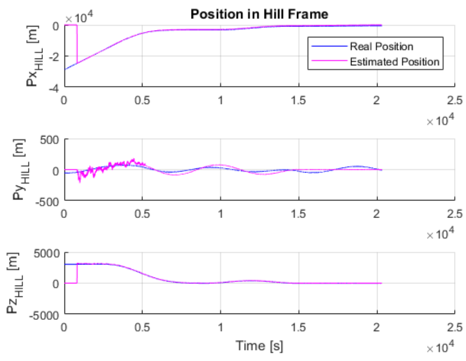
(b) Zoom on waypoint S2.



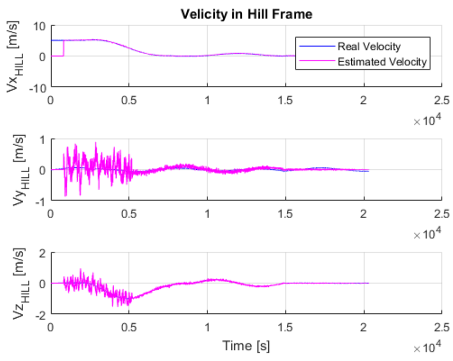
(c) Zoom on waypoint S3.



(d) Zoom on final approach S3-S4.

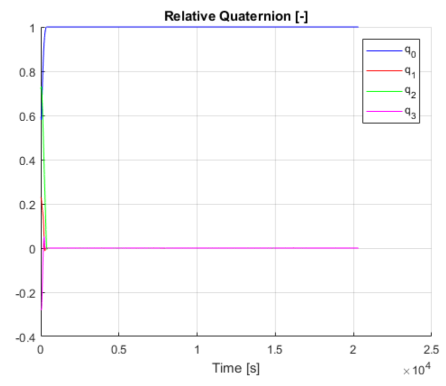


(e) Position in Hill's frame.

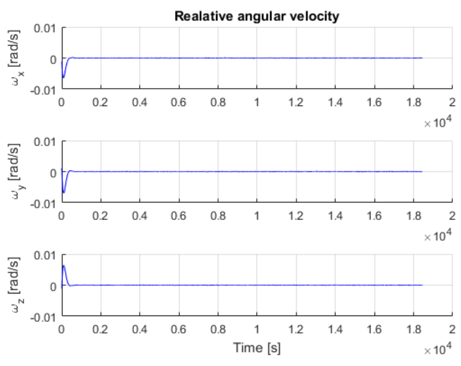


(f) Velocity in Hill's frame.

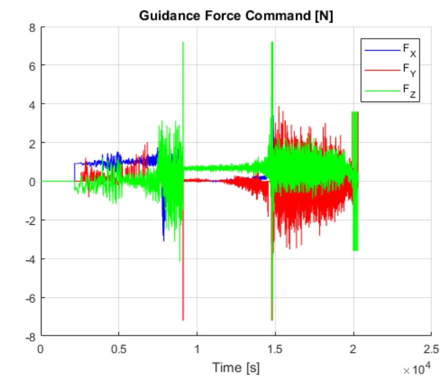
Fig. 4.19 Continuous Lambert-type Guidance with measurement errors (1).



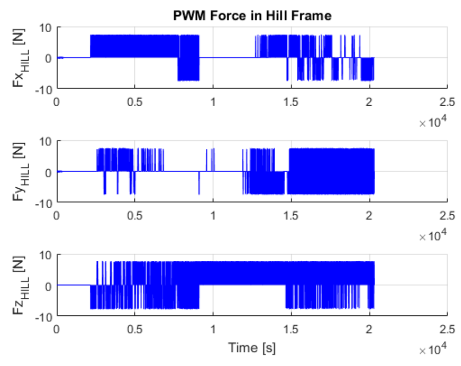
(a) Relative quaternion.



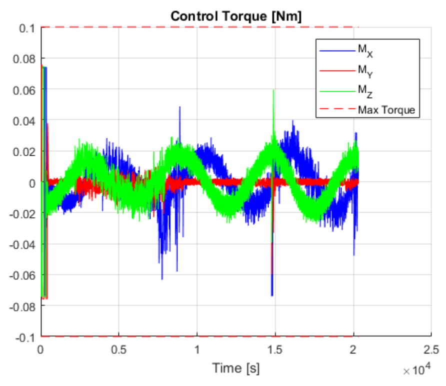
(b) Relative angular velocity.



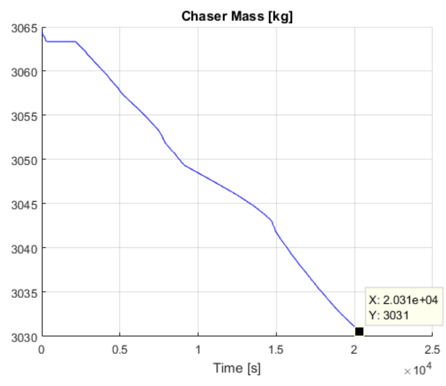
(c) Guidance force command.



(d) Thrust force - PWM.



(e) Control torque.



(f) Chaser mass

Fig. 4.20 Continuous Lambert-type Guidance with measurement errors (2).

initial conditions and spacecraft configuration with respect to the nominal one, the Continuous Lambert-type Guidance algorithm is able to cope such dispersion.

To complete the analysis of the non-nominal maneuver, in the following pictures, from Fig. 4.21 to Fig. 4.26, are reported results related to sensor measurements.

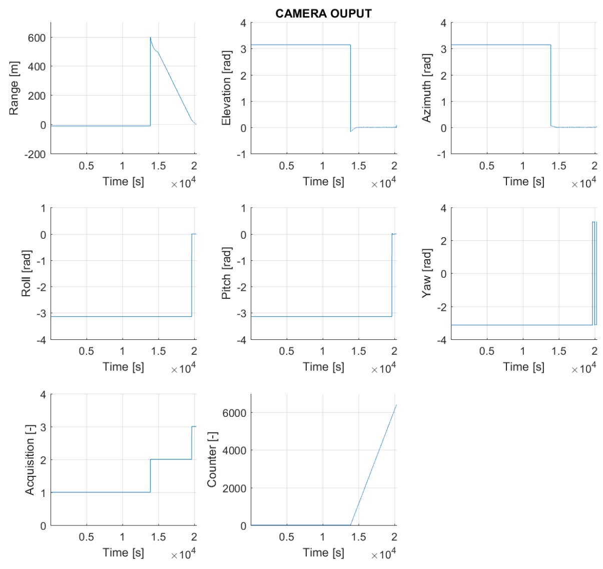


Fig. 4.21 Camera measurement.

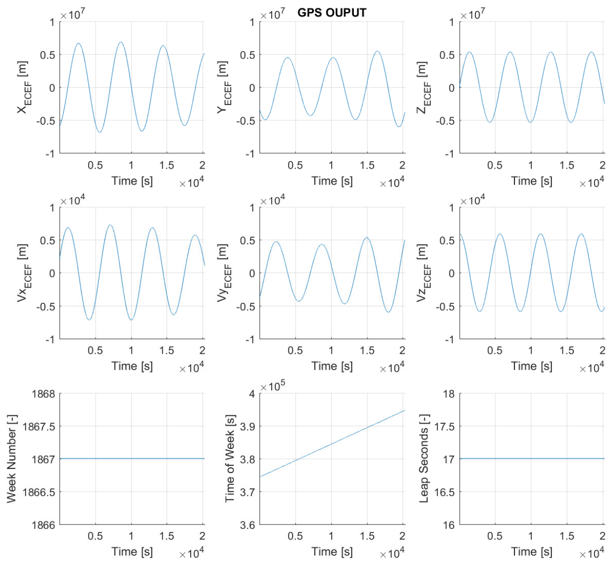


Fig. 4.22 GPS measurement.

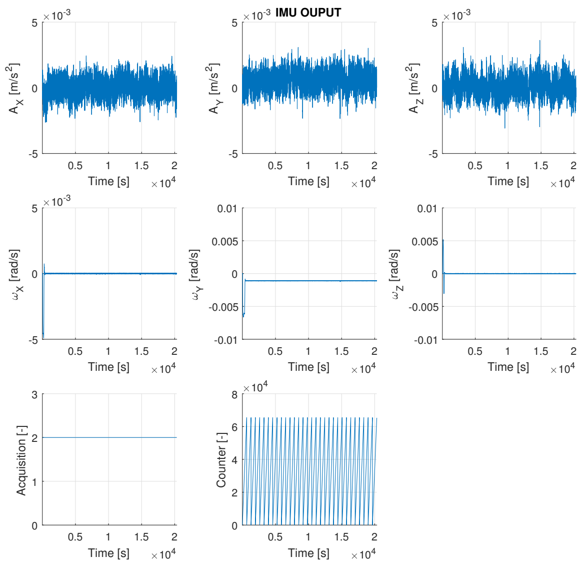


Fig. 4.23 IMU measurement.

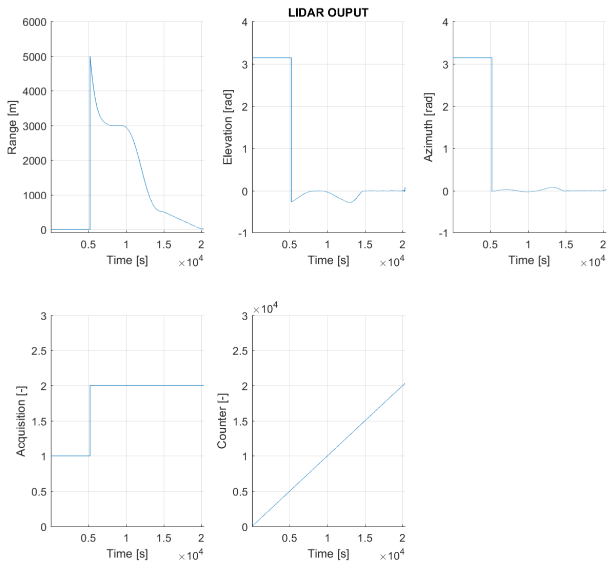


Fig. 4.24 Lidar measurement.

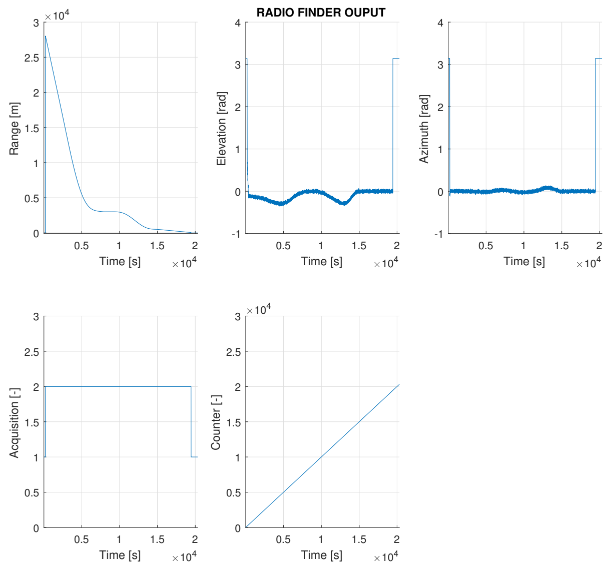


Fig. 4.25 Radio Finder measurement.

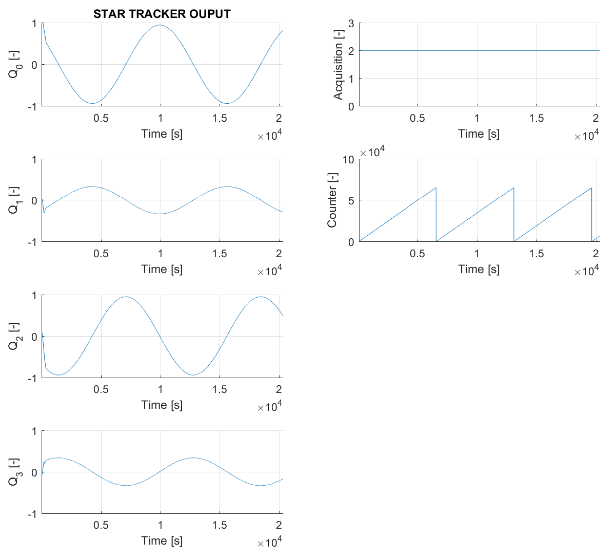


Fig. 4.26 Star Tracker measurement.

## 4.3 Monte Carlo Analysis

In order to verify the feasibility of the RVD mission it is required to extensively test the GNC software considering, as possible, all the potential combination of all initialization parameters defined in the simulator and GNC software. Examined parameters are sensors and actuators biases and misalignment, mounting errors, orbit dispersion, mass properties uncertainties, and more, as discussed in the previous section. All these uncertainties can potentially affect the success of the mission: if the GNC is badly designed, a parameter with a different value form the nominal one can disturb calculation of the flight software, producing wrong results which can cause the failure of the mission. For example, if the initial orbit is significantly different from the nominal one, sensors cannot be able to track the target spacecraft, or the navigation filter cannot find a good solution of the estimated state, causing a wrong initialization of waypoints by the Guidance function and then cause the failure of the whole mission.

In order to take into account the variation of parameters, it can be used the *Monte Carlo Method*. This method has been conceived to solve problems with probabilistic interpretation, hence numerically solving complex mathematical problems described by a large numbers of variables exploring the variability set of all parameters. Exploration of all possible combination of parameters is completed by generating a set of random numbers, uncorrelated each others, which is know the probability distribution of the parameters representing the analyzed phenomenon. An example of application of the Monte Carlo method is the computation of the value of $\pi$. The implementation of the Monte Carlo method is not unique, but all methods tends to converge to the following scheme:

1. Definition of the domain of possible input data.
2. Generation of random numbers included in the defined domain whit specific probability distribution.
3. Deterministic computation of the output with the random input.
4. Aggregation of results into the final one.

As mentioned before, the Monte Carlo analysis preformed in the present work is focused to explore all the possible combinations of uncertainties of parameters derived from the initialization procedure and verify that the GNC software can afford all the tested parameters combination and complete successfully the RVD mission in all conditions. The proposed approach is very simple: it has been defined a number of simulation runs to be executed; the initial configuration of the simulator (sensors, actuators, mass properties, orbit, ecc.) is stored in a specific configuration file containing all initialization parameters; results of each simulation is stored; finally, results are collected, post-processed and analyzed to verify the mission success. This process is set up by development of a MATLAB® script which automates the simulation setup and runs, data storage and post-process of data in a very effective way. A flow chart of the process is depicted in Fig. 4.27. The flowchart has been implemented in a MATLAB® script which is used to store simulation data (configuration, simulation and GNC output) and to launch the two executable files of the simulator and GNC applications, as well as to plot simulation data. In the simulator initialization, all parameters related to orbit, spacecraft configuration, actuators and sensors are subjected to a parameter dispersion as discussed in the previous section.

Monte Carlo results of a 100 simulation runs campaign are summarized below. According to guidelines suggested by Thales Alenia Space, data collected in Monte Carlo simulations are: Chaser and Target Hill's position, Chaser and Target Hill's velocity, Chaser local quaternion, torque provided by the reaction wheels system, consumed propellant and guidance force command. Such results are logged at different frequency, from 100 to 0.1 Hz, depending by the variation rate of each data.

The position of Chaser and Target with respect to the Hill frame is depicted in Fig. 4.28. The effect of orbital dispersion is well highlighted by the amout of 100 different trajectories followed by Chaser and Target in the $V_{BAR}$-$R_{BAR}$ and $V_{BAR}$-$H_{BAR}$ planes. The goal of the RVD maneuver is to get the final position of the target spacecraft depicted in Fig. 4.28c and Fig. 4.28d. In order to improve the readability of the complete maneuver, it has been depicted the relative trajectory of the chaser spacecraft with respect to the Hill trajectory of the target spacecraft, in order to obtain the relative trajectory as depicted in Fig. 3.4. The result of this conversion is depicted in Fig. 4.29. As it can be seen, the RVD maneuver has been completed in all the 100 simulation runs, demonstrating the robustness of the developed GNC software. The trajectory in the $V_{BAR}$-$R_{BAR}$ plane, Fig. 4.29a, shows that the RVD maneuver is completed in all the simulation runs. Effects of corrections along $H_{BAR}$ are depicted in Fig. 4.29b: the trajectory in the $V_{BAR}$-$H_{BAR}$ plane is within a neighborhood of

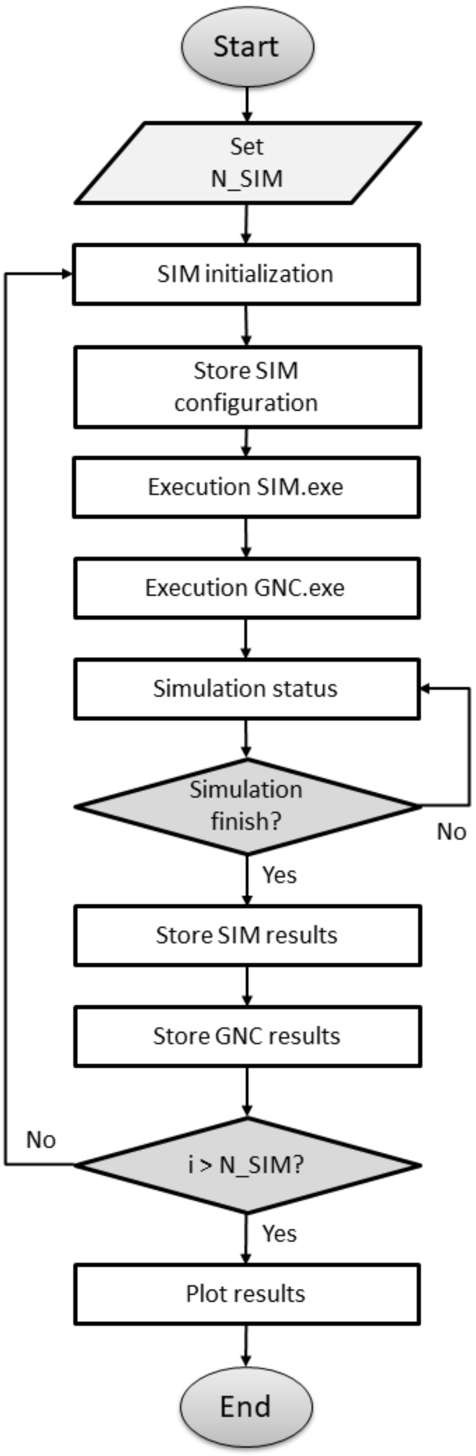
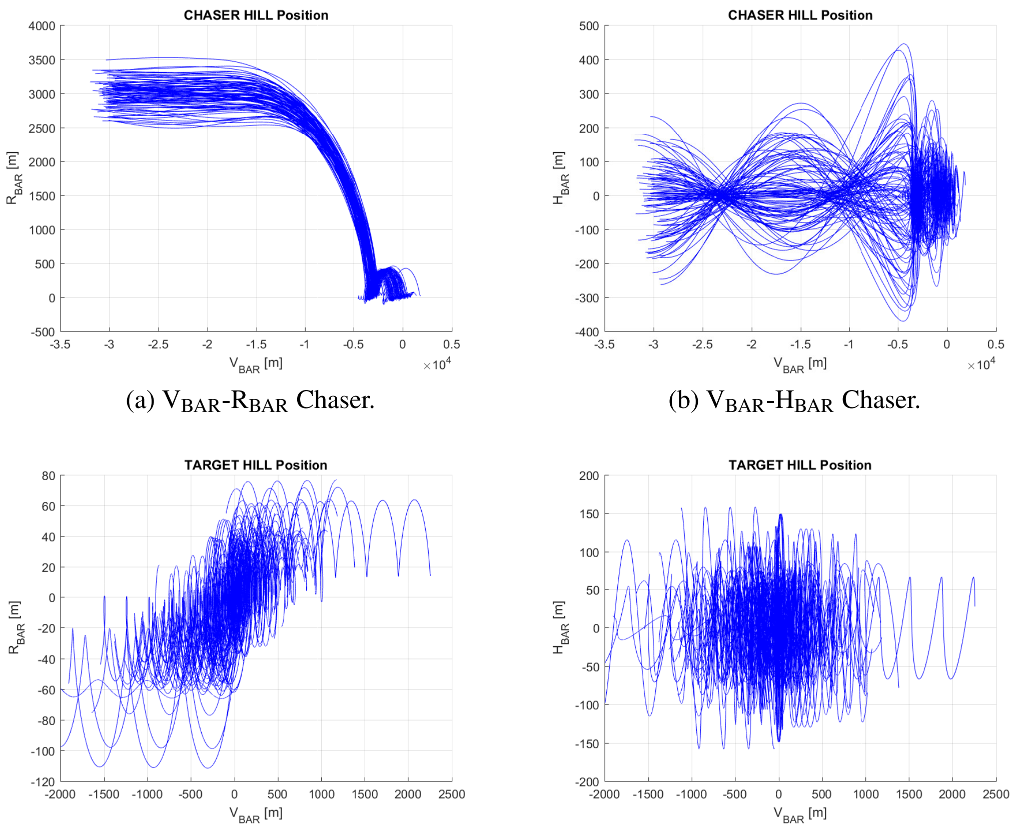


Fig. 4.27 Monte Carlo analysis flow chart.



(a) $V_{BAR}$-$R_{BAR}$ Chaser.

(b) $V_{BAR}$-$H_{BAR}$ Chaser.

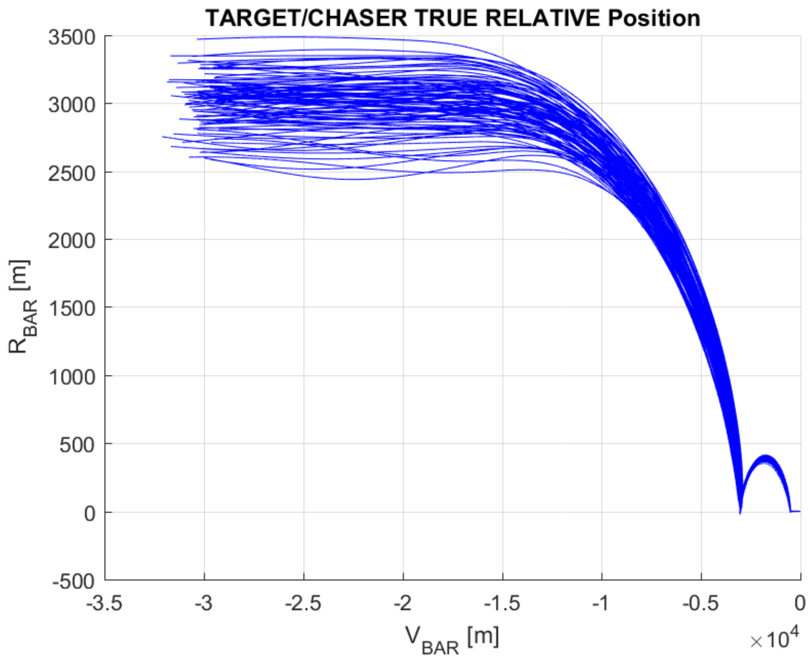
(c) $V_{BAR}$-$R_{BAR}$ Target.

(d) $V_{BAR}$-$H_{BAR}$ Target.

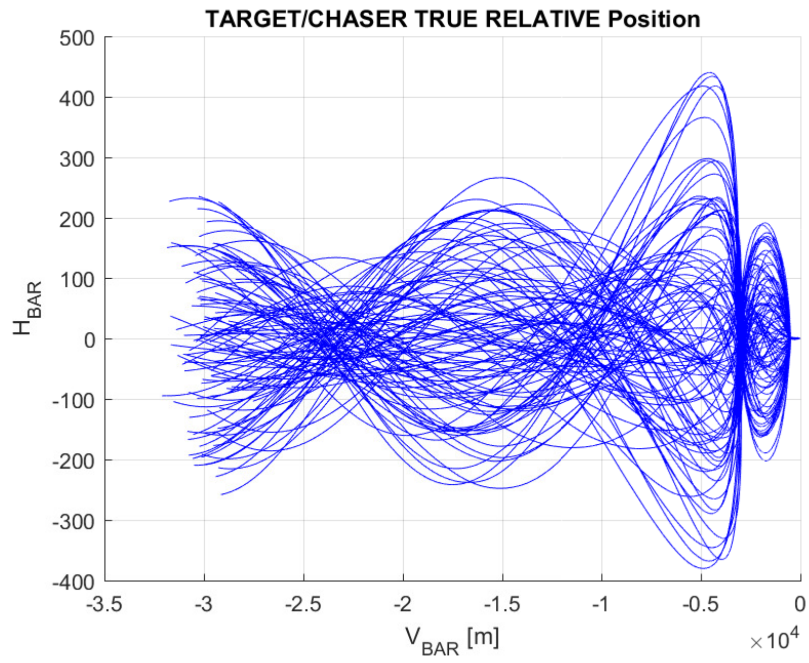
Fig. 4.28 Chaser and Target trajectory in Hill's frame.

about 300 m with respect to the $V_{BAR}$ axis and it progressively converges to such axis during the final approach. Highlight of the final approach are depicted in Fig. 4.30, while Hill velocity of the chaser spacecraft is depicted in Fig. 4.31. As it can be seen in Fig. 4.30, terminal conditions are reached with very high accuracy.

The attitude control has been resulted very effective since the local quaternion of the chaser spacecraft is controlled effectively over all the simulated missions (see Fig. 4.32a). The control action of reaction wheels is depicted in Fig. 4.32b and it can bee seen that some torque saturation occurs while the spacecraft is executing station keeping maneuvers. In Fig. 4.33 it has been depicted the output of the commanded force of the guidance function over all the 100 simulations. Command peaks in the neighborhood of 8000 and 15000 s are related to station keeping maneuvers in waypoints $S2$ and $S3$. Further peaks are related to the guidance algorithm itself. Eventually, the propellant consumption for each simulation run is summarized in Fig. 4.34. In almost all the simulated maneuvers the propellant consumption is within the range of 30-40 kg, which is acceptable and compliant with ideal consumption, considering that dispersion and uncertainties, including thrust magnitude errors, have been included is such analysis. Three peaks in the range of 60-65 kg of propellant consumption can be related to particular *unlucky* missions, where the initial position of the Chaser has been the worst case as well as the Target trajectory, an so it has been required more effort to correct the Chaser trajectory to complete the rendezvous and docking mission. Despite this anomalous propellant consumption in a limited number of cases, the overall performance of the developed GNC software are satisfactory.



(a) $V_{BAR}$-$R_{BAR}$ plane.



(b) $V_{BAR}$-$H_{BAR}$ plane.

Fig. 4.29 Chaser/Target relative trajectory.

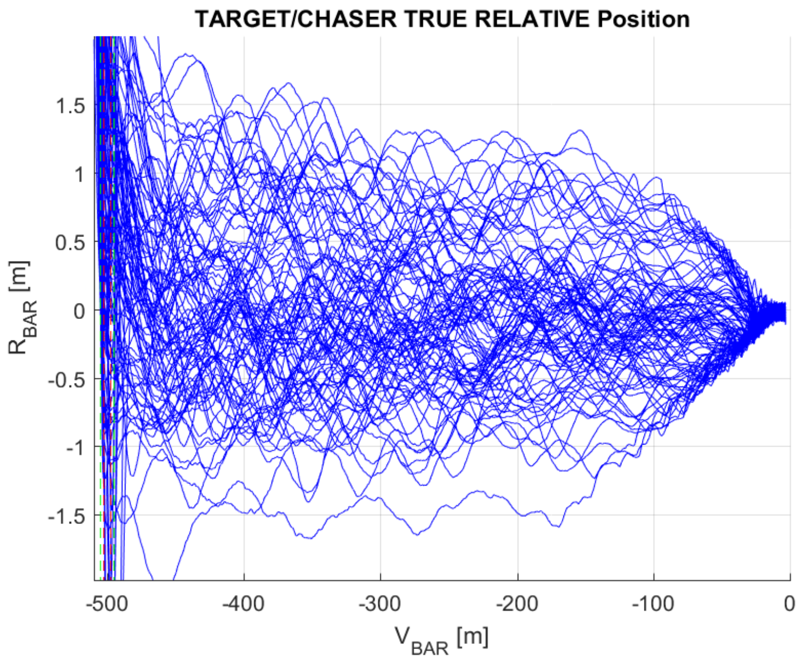


Fig. 4.30 Highlight of the final approach.

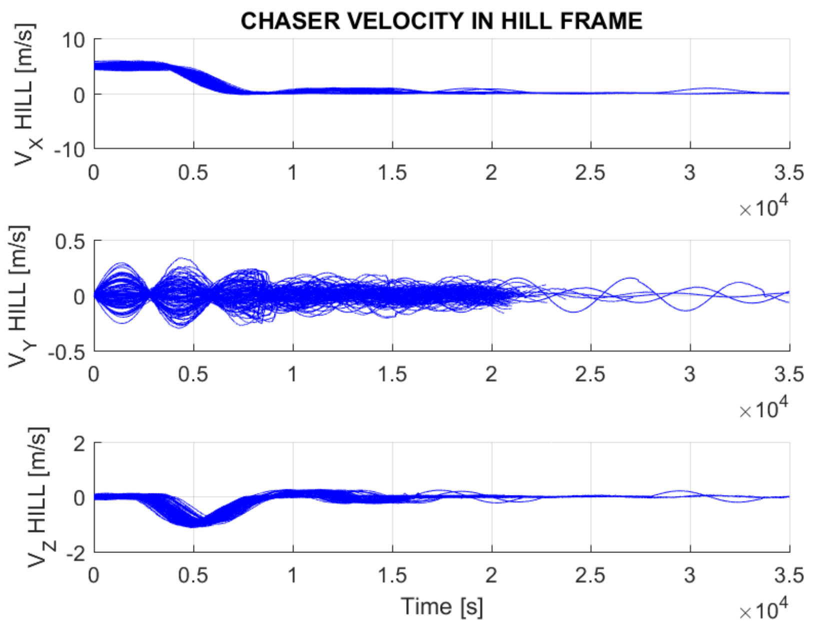
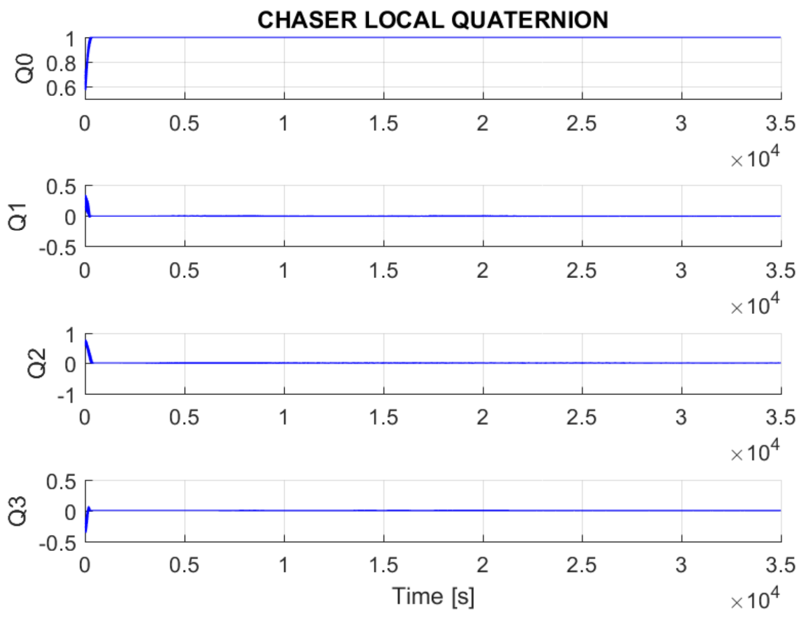
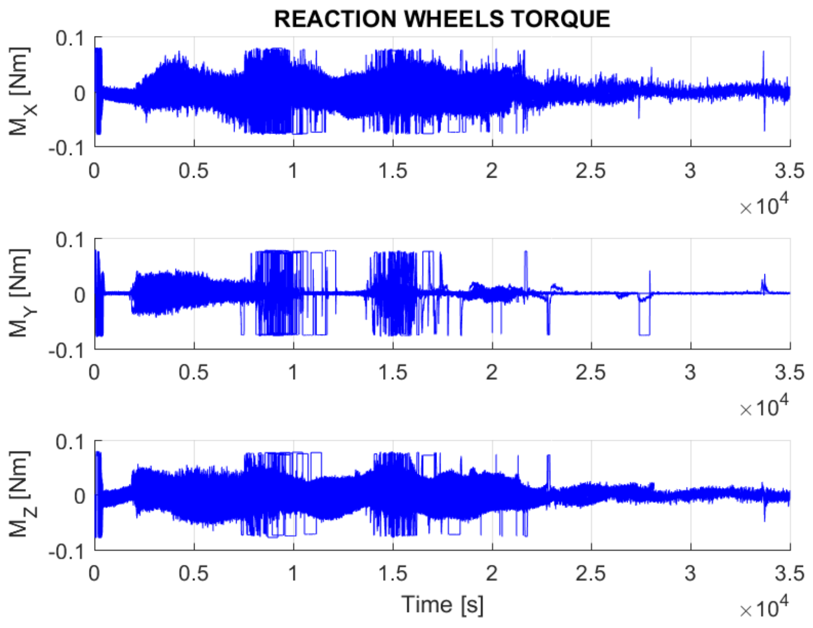


Fig. 4.31 Chaser velocity in Hill's frame.



(a) Chaser local vertical quaternion.



(b) Control torque of reaction wheels.

Fig. 4.32 Chaser/Target relative trajectory.

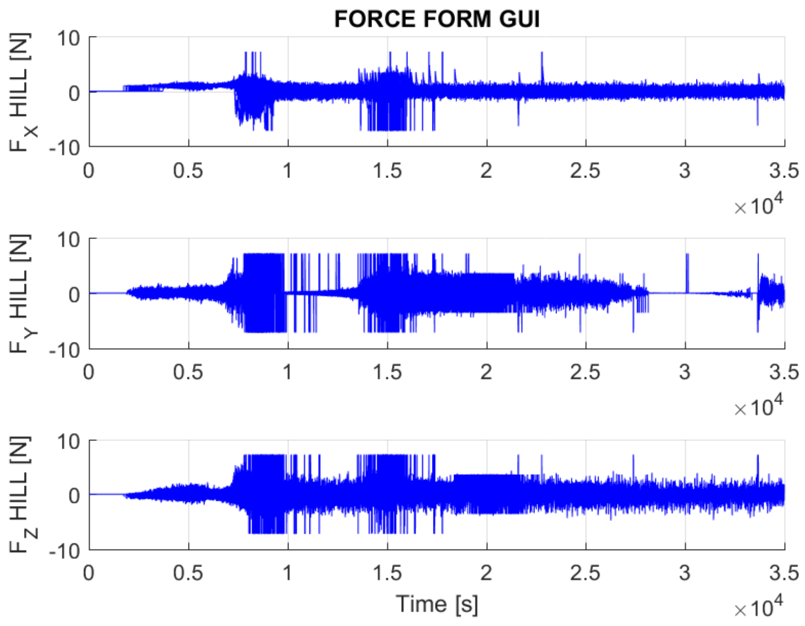


Fig. 4.33 Computed Guidance force command.

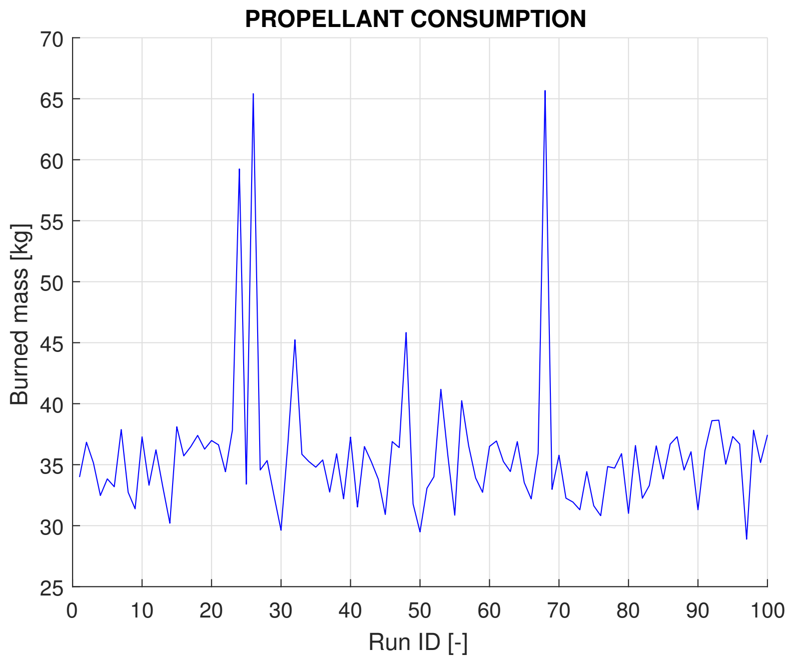


Fig. 4.34 Propellant consumption over 100 simulation.

## 4.4 Real-time Testing

At the beginning of the SAPERE STRONG project, it has been scheduled a set of activities with the goal to test the developed GNC SW in a real-time test bench, executing hardware in the loop simulations and Monte Carlo analysis as well. The purpose of real-time testing was to verify the computational time of each function of the GNC SW. Indeed, as prescribed by ECSS standards[1], the software running on a real machine shall not use all the computational resources in the worst case, and so the complete GNC SW shall run using a limited percentage of the maximum computational capability of the hardware machine, even in the worst case.

As suggested by ESA, microprocessors for space application are the space-qualified LEON platforms, based on the SPARC V8 architecture[2]. The original purpose was to run the GNC SW on the selected LEON hardware platform, while the orbital dynamics, sensors and actuators had been simulated by the developed orbital simulator running on a different powerful machine, as a common workstation with processor based on the Intel i7 family. For scheduling and program re-modulation occurred over the three-year time span of the SAPERE STRONG project such activities has been postponed to future works.

[1]ECSS-E-ST-60-30C and ECSS-E-TM-10-21A
[2]ESA website

# Chapter 5

# Conclusions

Rendezvous and docking missions are an enabling technology not only for space exploration, as it has been proven in the past lunar missions completed during the Apollo program, but also for providing innovative commercial services, such as resupply missions to the International Space Station or more complex services as defined in the Space-Tug mission scenario, part of the SAPERE STRONG project. In particular, providing a commercial payload relocation service executing many LEO-GEO-LEO orbital transfers, requires the implementation of a fleet of Space-Tugs, in order to provide a continuous service. For this purpose, each spacecraft of the fleet shall be equipped with a robust and reliable GNC software able to drive orbital transfers and rendezvous and docking maneuvers, which are the most critical, with an high degree of autonomy, up to complete autonomous operations.

For this purpose, an extensive literature review of existing guidance algorithms and navigation and control techniques has been executed and discussed in Chapter 1 and Chapter 2, focusing on automated GNC architectures. As prescribed by the SAPERE STRONG project, the output of this activity has been the development of advanced guidance algorithm, which has been resulted in the developments of the Modified ZEM/ZEV Guidance algorithm, which resulted having very good performance in ideal cases and in presence of slow-varying measurement errors. Concurrently, as described in the SAPERE STRONG project, it has been developed a functional orbital simulator comprehensive of a complete 6 degrees-of-freedom spacecraft dynamics of both Chaser and Target spacecraft, models of orbital disturbances and detailed models of sensors and actuators. The motivation to design a functional orbital simulator has been to provide an effective simulation tool to support the development of Guidance, Navigation and Control algorithms and to execute stress test, as Monte Carlo analysis, to verify the robustness of the software subjected to a number of parameter dispersion. The use of *C* language for the design and implementation of the functional orbital simulator has been demonstrated to be an optimal choice, especially during the simulator and GNC software integration successfully completed with low effort.

In the simulator and GNC software integration phase, a complete debugging of both simulator and GNC software developed by the team of Thales Alenia Space, complete with Flight Manager, Navigation and Control functions, has been executed, fixing residual code bugs related to the implementation of TCP communication libraries. After debugging, it has been implemented the developed Modified ZEM/ZEV algorithm within the GNC software, in order to start testing the complete autonomous GNC. During this phase it has been highlighted the high sensitivity of the Modified ZEM/ZEV algorithm to measurement noise, making it inapplicable to a real GNC software due to the high propellant consumption related to the excessive required control effort, even if its performance in terms of maneuver accuracy are satisfactory. This results brought to the development of an alternative guidance algorithm. In order to support the development of a second algorithm, its has been designed a lighter development version of the GNC software, including a control function based on LQR and a navigation function implementing a discrete Kalman filter. This has been realized in order to take into account effects of an active attitude control, which potentially induce fatal thrust misalignment, and state estimation errors deriving by the navigation filter. Supported by this light version of the GNC software, the development of the Continuous Lambert-type Guidance algorithm has been completed, and successfully tested in the complete version of the GNC software developed in collaboration with Thales Alenia Space. Results have been extensively discussed in Chapter 4.

Stress tests have been successfully completed, executing Monte Carlo analysis, to test the robustness of the complete GNC software, implementing the Continuous Lambert-type Guidance algorithm, subjected to different initial conditions and spacecraft configuration. Dispersion of the initial orbit and attitude of both Chaser and Target spacecraft has been implemented in the functional orbital simulator, as well as initialization of different spacecraft configurations such ad errors in the actuation system (reaction wheel and reaction control thrusters) and initialization of sensors biases. A test script to run Monte Carlo analysis, store simulation data and post-process simulation results has been implemented in MATLAB®. Differently from what prescribed in the initial description of work of the SAPERE STRONG project, real-time tests have not been executed, as described in Chapter 4, due to several re-modulation and re-planning of the scheduled activities occurred during the three-year project plan, which eventually deleted such activities and leaving real-time testing to future developments.

The goal to take a step forward in design of a complete autonomous GNC for rendezvous missions has been reached. Even if many years are still required to develop commercial services based on a fleet of Space-Tugs, the present work is providing reliable tools and methodologies to support the design of more complex GNS software able to manage autonomous operations, in particular rendezvous maneuvers. Even though the proposed Continuous Lambert-type Guidance algorithm has been tested in execution of only the reference RVD maneuver, it can be potentially execute different orbital maneuvers, as partially demonstrated by Monte Carlo analysis. Indeed, future works may include testing the GNC in a payload release scenario, as well as implementation of collision avoidance maneuvers is envisaged. In addition, future improvements of the current GNC software may implement relative GNSS navigation techniques based on Galileo constellation, in a similar way as done in the rendezvous and docking mission scenario of the ATV spacecraft, which was using the GPS constellation.

The internship at Turin offices of Thales Alenia Space scheduled by the SAPERE STRONG project has been completed. The scheduled visiting period at foreign institution has been conducted at the University of Sevilla for a three month period, from September to December 2016. During this visiting period, advanced navigation techniques based on relative satellite navigation have been studied, focusing on the implementation of relative navigation algorithm using the Galileo constellation. Two electronic boards equipped with a Galileo-compatible GNSS receivers have been realized and preliminary measurement of Galileo signals have been carried out. A complete implementation of investigated algorithms was not possible to complete, mainly due to the difficulty to track an enough number of satellites required to get reliable measures, since the Galileo constellation on December 2016 was counting only eight satellites. Summary of the activities done during the visiting period is reported in Appendix B.

# References

[1] John L Goodman. History of Space Shuttle rendezvous. *NASA Johnson Space Center*, 2011.

[2] Richard J McLaughlin and William H Warr. The Common Berthing Mechanism (CBM) for International Space Station. Technical report, SAE Technical Paper, 2001.

[3] Phillip Callen. Robotic Transfer and Interfaces for External ISS Payloads. *$3^{rd}$ Annual ISS Research and Development Conference*, 2014.

[4] Sean M. Kelly and Scott P. Cryan. International Docking Standard (IDSS) Interface Definition Document (IDD) Revision D. Technical report, ISS Multilateral Control Board, 2016.

[5] John Cook, Valery Aksamentov, Thomas Hoffman, and Wes Bruner. ISS Interface Mechanisms and their Heritage. In *Proceedings, AIAA SPACE 2011 Conference & Exposition*, pages 27–29, 2011.

[6] Martine Ganet, Isabelle Quinquis, Jerome Bourdon, and Patrick Delpy. ATV GNC during rendezvous with ISS. In *DCSSS Conference*, 2002.

[7] Josian Fabrega, Michel Frezet, and J.L. Gonnaud. ATV GNC during rendezvous. In *Spacecraft Guidance, Navigation and Control Systems*, volume 381, page 85, 1997.

[8] J.L. Gonnaud and V. Pascal. ATV guidance, navigation and control for rendezvous with ISS. In *Spacecraft Guidance, Navigation and Control Systems*, volume 425, page 501, 2000.

[9] David C Woffinden and David K Geller. Navigating the road to autonomous orbital rendezvous. *Journal of Spacecraft and Rockets*, 44(4):898–909, 2007.

[10] Didier Pinard, Stéphane Reynaud, Patrick Delpy, and Stein E Strandmoe. Accurate and autonomous navigation for the ATV. *Aerospace Science and Technology*, 11(6):490–498, 2007.

[11] Mayara Mesiano Carrasco and André Luís da Silva. Attitude determination for low cost imu and processor board using the methods of triad, Kalman filter and Allan variance. *REVISTA BRASILEIRA DE INICIAÇÃO CIENTÍFICA*, 3(2), 2016.

[12] JG García, PI Mercader, and Carlos H Muravchik. Use of GPS carrier phase double differences. *Latin American applied research*, 35(2):115–120, 2005.

[13] Nandakumaran Nadarajah and P.J.G. Teunissen. Instantaneous GPS/Galileo/QZSS/SBAS Attitude Determination: A Single-Frequency (L1/E1) Robustness Analysis under Constrained Environments. *Navigation*, 61(1):65–75, 2014.

[14] G.A. Beals, R.C. Crum, H.J. Dougherty, D.K. Hegel, J.L. Kelley, and J.J. Rodden. Hubble Space Telescope precision pointing control system. *Journal of Guidance, Control, and Dynamics*, 11(2):119–123, 1988.

[15] W.B. Chubb and S.M. Seltzer. Skylab attitude and pointing control system. *National Aeronautics and Space Administration*, 1971.

[16] W.B. Chubb, H.F. Kennel, C.C. Rupp, and S.M. Seltzer. *Flight performance of Skylab attitude and pointing control system*, volume 8003. National Aeronautics and Space Administration, 1975.

[17] Thomas R. Coon and John E. Irby. Skylab attitude control system. *IBM Journal of Research and Development*, 20(1):58–66, 1976.

[18] S.R. Vadali and H.S. Oh. Space Station attitude control and momentum management-A nonlinear look. *Journal of Guidance, Control, and Dynamics*, 15(3):577–586, 1992.

[19] Carlos M. Roithmayr. International Space Station attitude control and energy storage experiment: effects of flywheel torque. *National Aeronautics and Space Administration*, 1999.

[20] Nazareth Bedrossian, Sagar Bhatt, Mike Lammers, and Louis Nguyen. Zero-Propellant Maneuver [TM] Flight Results for 180 deg ISS Rotation. *National Aeronautics and Space Administration*, 2007.

[21] D Muzi and A Allasio. GOCE: the first core Earth explorer of ESA's Earth observation programme. *Acta Astronautica*, 54(3):167–175, 2004.

[22] Enrico Canuto and Luca Massotti. All-propulsion design of the drag-free and attitude control of the European satellite GOCE. *Acta Astronautica*, 64(2):325–344, 2009.

[23] Sara Cresto Aleina, Matteo Dentis, Simona Ferraris, Paolo Maggiore, Nicole Viola, and Maria Antonietta Viscio. Reusable Space Tug concept and mission. In *PROCEEDINGS OF THE INTERNATIONAL ASTRONAUTICAL CONGRESS*, volume 11, pages 8860–8872. International Astronautical Federation, IAF, 2015.

[24] Neil F Palumbo, Ross A Blauwkamp, and Justin M Lloyd. Basic principles of homing guidance. *Johns Hopkins APL Technical Digest*, 29(1):25–41, 2010.

[25] Matt Hawkins, Bong Wie, and Yanning Guo. Spacecraft guidance algorithms for asteroid intercept and rendezvous missions. *International Journal Aeronautical and Space Sciences*, 13(2):154–169, 2012.

[26] Matt Hawkins, Alan Pitz, Bong Wie, and Jesús Gil-Fernández. Terminal-Phase Guidance and Control Analysis of Asteroid Interceptors. In *AIAA Guidance, Control, and Navigation Conference, Toronto, Canada*, 2010.

[27] Neil F Palumbo, Ross A Blauwkamp, and Justin M Lloyd. Modern homing missile guidance theory and techniques. *Johns Hopkins APL Tech. Dig*, 29(1):42–59, 2010.

[28] Matt Hawkins, Yanning Guo, and Bong Wie. ZEM/ZEV feedback guidance application to fuel-efficient orbital maneuvers around an irregular-shaped asteroid. In *AIAA Guidance, Navigation, and Control Conference*, page 5045, 2012.

[29] Yanning Guo, Matt Hawkins, and Bong Wie. Applications of generalized Zero-Effort-Miss/Zero-Effort-Velocity feedback guidance algorith. *Journal of Guidance, Control, and Dynamics*, 2013.

[30] Qinghai Gong, Zhengyu Song, and Xinguang Lv. Application of optimal Zero-Effort-Miss/Zero-Effort-Velocity feedback guidance in hazard avoidance for planetary landing. In *Guidance, Navigation and Control Conference (CGNCC), 2014 IEEE Chinese*, pages 1835–1841. IEEE, 2014.

[31] Matteo Dentis, Elisa Capello, and Giorgio Guglieri. A Novel Concept for Guidance and Control of Spacecraft Orbital Maneuvers. *International Journal of Aerospace Engineering*, 2016, 2016.

[32] Capello Elisa and Dentis Matteo. Quasi-flyable Predictive Guidance Algorithms for orbital RVD missions. In *$12^{th}$ International AIRTEC Congress*, 2017.

[33] Torre Sangrà, Elena Fantino, et al. Review of Lambert's problem. In *ISSFD 2015: 25th International Symposium on Space Flight Dynamics, 19-23 October, Munich, Germany*, pages 1–15, 2015.

[34] ER Lancaster. Solution of Lambert's problem for short arcs. *Celestial Mechanics and Dynamical Astronomy*, 2(1):60–63, 1970.

[35] Dario Izzo. Revisiting Lambert's Problem. *Celestial Mechanics and Dynamical Astronomy*, 2014(1):1–15, 2014.

[36] Sara Jean MacLellan. *Orbital rendezvous using an Augmented Lambert Guidance scheme*. PhD thesis, Massachusetts Institute of Technology, 2005.

[37] Ossama Abdelkhalik and Daniele Mortari. N-impulse orbit transfer using genetic algorithms. *Journal of spacecraft and rockets*, 44(2):456–460, 2007.

[38] Richard H Battin. Lambert's problem revisited. *AIAA Journal*, 15(5):707–713, 1977.

[39] Richard H Battin and Robin M Vaughan. An elegant Lambert algorithm. *Journal of Guidance, Control, and Dynamics*, 7(6):662–670, 1984.

[40] Steven L. Nelson and Paul Zarchan. Alternative approach to the solution of Lambert's problem. *Journal of Guidance, Control, and Dynamics*, 15(4):1003–1009, 1992.

[41] Giulio Avanzini. A simple Lambert algorithm. *Journal of guidance, control, and dynamics*, 31(6):1587, 2008.

[42] Wigbert Fehse. *Automated rendezvous and docking of spacecraft*, volume 16. Cambridge university press, 2003.

[43] Hari B Hablani, Myron L Tapper, and David J Dana-Bashian. Guidance and relative navigation for autonomous rendezvous in a circular orbit. *Journal of Guidance, Control, and Dynamics*, 25(3):553–562, 2002.

[44] Renato Zanetti. Optimal glideslope guidance for spacecraft rendezvous. *Journal of Guidance, Control and Dynamics*, 34(5):1593–1597, 2011.

[45] Yassine Ariba, Denis Arzelier, Laura Sofia Urbina, and Christophe Louembet. V-bar and R-bar Glideslope Guidance Algorithms for Fixed-Time Rendezvous: A Linear Programming Approach. *IFAC-PapersOnLine*, 49(17):385–390, 2016.

[46] David P. Dannemiller. Multi-Maneuver Clohessy-Wiltshire Targeting. In *AAS Astrodynamics Specialist Conference*, 2011.

[47] Gilberto Arantes and Luiz S Martins-Filho. Guidance and control of position and attitude for rendezvous and dock/berthing with a noncooperative/target spacecraft. *Mathematical Problems in Engineering*, 2014, 2014.

[48] Riccardo Bevilacqua, Marcello Romano, Fabio Curti, Andrew P Caprari, and Veronica Pellegrini. Guidance navigation and control for autonomous multiple spacecraft assembly: analysis and experimentation. *International Journal of Aerospace Engineering*, 2011, 2011.

[49] R Bevilacqua, T Lehmann, and M Romano. Development and experimentation of LQR/APF guidance and control for autonomous proximity maneuvers of multiple spacecraft. *Acta Astronautica*, 68(7):1260–1275, 2011.

[50] Edward N Hartley. A tutorial on model predictive control for spacecraft rendezvous. In *Control Conference (ECC), 2015 European*, pages 1355–1361. IEEE, 2015.

[51] S Di Cairano, H Park, and I Kolmanovsky. Model predictive control approach for guidance of spacecraft rendezvous and proximity maneuvering. *International Journal of Robust and Nonlinear Control*, 22(12):1398–1427, 2012.

[52] Edward N Hartley, Paul A Trodden, Arthur G Richards, and Jan M Maciejowski. Model predictive control system design and implementation for spacecraft rendezvous. *Control Engineering Practice*, 20(7):695–713, 2012.

[53] Josep Virgili-Llop, Costantinos Zagaris, Hyeongjun Park, Richard Zappulla, and Marcello Romano. Experimental evaluation of model predictive control and inverse dynamics control for spacecraft proximity and docking maneuvers. *CEAS Space Journal*, pages 1–13, 2016.

[54] Hyeongjun Park, Marcello Romano, Costantinos Zagaris, Josep Virgili-Llop, and Richard II Zappulla. Nonlinear model predictive control for spacecraft rendezvous and docking with a rotating target. In *27th AAS/AIAA Spaceflight Mechanics Meeting*, 2017.

[55] Hyeongjun Park, Costantinos Zagaris, Josep Virgili-Llop, Richard Zappulla, Ilya Kolmanovsky, and Marcello Romano. Analysis and Experimentation of Model Predictive Control for Spacecraft Rendezvous and Proximity Operations with Multiple Obstacle Avoidance. In *AIAA/AAS Astrodynamics Specialist Conference*, page 5273, 2016.

[56] S Vromen, FJ de Bruijn, and Erwin Mooij. Guidance for Autonomous Rendezvous and Docking with Envisat Using Hardware-in-The-Loop Simulations. *IWSCFF 2015*, 2015.

[57] Ming Ge, Min-Sen Chiu, and Qing-Guo Wang. Robust PID controller design via LMI approach. *Journal of process control*, 12(1):3–13, 2002.

[58] John Doyle. Guaranteed margins for LQG regulators. *IEEE Transactions on Automatic Control*, 23(4):756–757, 1978.

[59] Brian L Stevens, Frank L Lewis, and Eric N Johnson. *Aircraft control and simulation: dynamics, controls design, and autonomous systems*. John Wiley & Sons, 2015.

[60] Ernest Bauer. Standard Atmospheres for Engagement Modeling. Technical report, INSTITUTE FOR DEFENSE ANALYSES ALEXANDRIA (VA), 1993.

[61] Samuel Schweighart and Raymond Sedwick. A perturbative analysis of geopotential disturbances for satellite cluster formation flying. In *Aerospace Conference, 2001, IEEE Proceedings.*, volume 2, pages 2–1001. IEEE, 2001.

[62] M Jones, E Gomez, A Mantineo, and UK Mortensen. Introducing ECSS software-engineering standards within ESA. *ESA bulletin*, pages 132–139, 2002.

[63] Veit Oehler, Jan M Krueger, Tanja Beck, Michael Kirchner, Hans L Trautenberg, Jörg Hahn, and Daniel Blonski. Galileo system performance status report. In *Proceedings of the 22th International Technical Meeting of the Satellite Division*, pages 2956–2966, 2009.

[64] Benjamin W Remondi. Global Positioning System carrier phase: description and use. *Journal of Geodesy*, 59(4):361–377, 1985.

[65] Geoffrey Blewitt. Carrier phase ambiguity resolution for the Global Positioning System applied to geodetic baselines up to 2000 km. *Journal of Geophysical Research: Solid Earth*, 94(B8):10187–10203, 1989.

[66] Patrick Henkel and Chen Zhu. Carrier phase integer ambiguity resolution with inequality constraints for GPS and Galileo. In *Statistical Signal Processing Workshop (SSP), 2011 IEEE*, pages 409–412. IEEE, 2011.

[67] A Nardo, B Li, and PJG Teunissen. Partial ambiguity resolution for ground and space-based applications in a GPS+ Galileo scenario: a simulation study. *Advances in Space Research*, 57(1):30–45, 2016.

[68] Dennis Odijk, Peter JG Teunissen, and Amir Khodabandeh. Galileo IOV RTK positioning: standalone and combined with GPS. *Survey Review*, 46(337):267–277, 2014.

[69] Robert Odolinski, Peter JG Teunissen, and Dennis Odijk. Combined BDS, Galileo, QZSS and GPS single-frequency RTK. *GPS solutions*, 19(1):151–163, 2015.

[70] Nandakumaran Nadarajah, Peter JG Teunissen, and Noor Raziq. Instantaneous GPS-Galileo attitude determination: single-frequency performance in satellite-deprived environments. *IEEE Transactions on Vehicular Technology*, 62(7):2963–2976, 2013.

[71] Ublox Company. *NEO-M8 - u-blox M8 concurrent GNSS modules - Datasheet*, August 2016.

[72] Ublox Company. *NEO/LEA-M8T - u-blox M8 concurrent GNSS timing modules - Datasheet*, June 2016.

[73] Ublox Company. *EVK-M8T - Evaluation Kit - User Guide*, March 2016.

[74] Ublox Company. *u-blox 8 / u-blox M8 - Receiver Description - Including Protocol Specification*, May 2016.

# Appendix A

# Simulator Test and Validation

As mentioned in Section 3.2, all the functions which concur in the definition of the orbital simulator has been implemented in C language, tested and validated. The *testing* process is required to check that the implemented functions are not affected by code bugs, i.e. the functions are producing coherent results. Instead, the *validation* process ensures that the coherent results obtained during the testing phase are satisfying the requirements allocated to the function under exam. In other words, the validation process tests the function with different input and check that the output is within a limited range with respect to the theoretical expected results.

For this purpose, it has been defined a validation process to be applied to all the *Level 2* and *Level 1* functions, since the *Level 0* function is consisting the complete orbital simulator, validated at the level of Simulator and GNC integration. The validation process consist in defining a set of *Test Cases* required to test and validate each function, defining a set of *Test Scripts* used to run and store input, state, and output data of each function, and compiling a document which includes Test Cases description and results and which collect results of each test, including Test Scripts. In the following it will be described the test cases adopted during the validation process of *Level 2* functions. Test cases description of *Level 1* function are not included in the present dissertation since they are a combination of more test cases of *Level 2* function, and so validation of *Level 1* function is limited to a debugging test.

| Name of the tested function | |
|---|---|
| *Version* | Version of the tested function. |
| *Test Case Name* | ID and name of the test case. |
| *Description* | Description of the test case, what the test case should do, initial condition, external input, etc. |
| *Scope* | The scope of the test case, i.e. the expected result of the test case. |
| *Success Criteria* | Detailed description of success criteria, including numerical expected results if applicable. |
| *Result* | Test case results: passed or failed. |
| *Attachments* | List of applicable documents, source files, results files, etc. |

Table A.1 Test cases template.

## A.1 Validation of Level 2 Functions: Test Scripts Description

In this section there are summarized all the test cases defined for validation of *Level 2* functions. The template used for the definition of the test cases is reported in Table A.1. Different test cases have been defined for each function. Detailed description of test cases defined for each function is not provided to simplifying the readability of the present dissertation, but a brief overview will be provided.

**Position Dynamics** This function has been extensively tested in order to verify that the propagated position dynamics produce results according to theoretical ones prescribed by Clohessy-Wiltshire equations. Particular case study has been: null origin shift, free drift at different altitudes, execution of tangential and radial impulsive and continuous maneuvers. The function has been completed tested up to the version *2.0*.

**Chaser Attitude Dynamics** This function propagates the attitude of the chaser spacecraft, hence it has been verified that the output quaternion and angular velocity had been coincident with theoretical results. Test cases for this function are: null motion, single axis constant angular velocity, single axis applied torque, gyroscopic effect of reaction wheels. The function has been completed tested up to the version *2.1*.

**Target Attitude Dynamics** This function, differently from the Chaser Attitude Dynamics, propagates a three-axial forced oscillating motion with fixed amplitude and constant frequency. Hence, it has been tested only the correct execution of the single axis forced motion. The function has been completed tested up to the version *1.1*.

**Reaction Wheels** This function has been defined to model correctly the reaction wheels assembly. Test cases have been the single axis reaction wheels command to check the correctness of torque and speed saturation. The function has been completed tested up to the version *3.0*.

**Thrusters** This function models the reaction control thrusters system. Test cases verified the single thruster force and torque generation and the double thruster force and torque generation inlcuding thrust errors. The function has been completed tested up to the version *1.2*.

**GPS** The function models the GPS sensor. Test cases verified the position and velocity measurement and the GPS time model, including the time shift of the GPS week. The function has been completed tested up to the version *1.1*.

**IMU** Function related to the IMU sensor. Test cases verified the acceleration measurement activating one specific error each time, bias, scale factor, bias stability and noise, including verification of the IMU counter. The function has been completed tested up to the version *1.0*.

**Camera** Model of the Camera sensor. Test cases check the range, elevation and azimuth measurement with and without errors, and check errors reducing with relative range. In addition, roll, pitch and yaw measurements have been validated, including the counter and the acquisition flag. The function has been completed tested up to the version *1.2*.

**Lidar** This is the function which models the Laser/Lidar sensor. As for the Camera, test cases check the range, elevation and azimuth measurement with and without errors, and checked errors reducing with relative range, including the counter and the acquisition flag. The function has been completed tested up to the version *1.1*.

**Radio Finder** This is the function which models the Radio Finder sensor. As for the Lidar, test cases check the range, elevation and azimuth measurement with and without errors, and checked errors reducing with relative range, including the counter and the acquisition flag. The function has been completed tested up to the version *1.1*.

**Star Tracker** This function models the Star Tracker sensor. Test cases validated the quaternion measurement, including errors, and the angular rate limitation, as well as acquisition flag and counter. The function has been completed tested up to the version *1.0*.

**Drag** The function related to the atmospheric drag disturbance. Test cases are related to evaluation of the drag force acting on a spacecraft with unitary frontal area orbiting at different altitudes. The function has been completed tested up to the version *1.0*.

**Gravity Gradient** The function related to the gravity gradient disturbance. Test cases checked the correctness of the computation of the disturbance torque at different attitude and altitude. The function has been completed tested up to the version *1.0*.

**J2 Effect** The function which models the J2 disturbance. Test cases verify the correctness of the computation of Eq. 3.45. The function has been completed tested up to the version *1.0*.

**Solar Pressure** Model of the solar radiation pressure disturbance. Test cases aimed to verify the correctness of computation of the solar radiation disturbance and the correct computation of the eclipse period. The function has been completed tested up to the version *1.0*.

Validation status of all the *Level 2* functions is depicted in Fig. A.1.

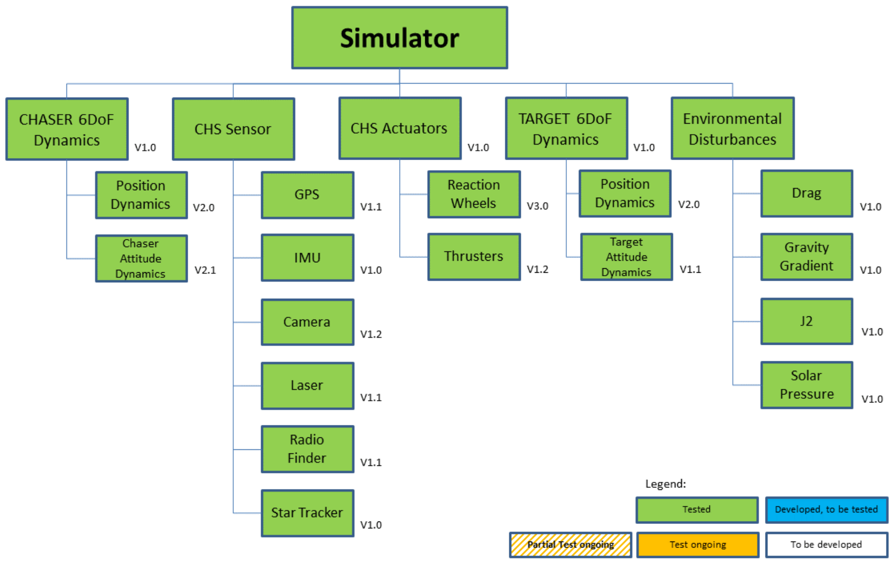


Fig. A.1 Level 2 functions validation status.

# Appendix B

# Summary of Activities at University of Sevilla

As planned in the present PhD project, a visiting period at a foreign institution has been scheduled. It has been chosen to spend three month at the University of Sevilla, from September 2016 to December 2016. Activities planned during this period aimed to improve the knowledge of the author in the field of Navigation, especially in satellite navigation based on different GNSS constellations. In the specific, it has been studied advanced navigation techniques based on Differential GPS, and detailed studies have been carried out in order to investigate the adoption of such navigation techniques supported by the Galileo GNSS.

Motivations which drove the choice of investigating differential GNSS techniques were to study the possible future implementation of such navigation techniques, based on navigation signal of the upcoming Galileo constellation, in a rendezvous and docking mission scenario, evaluating performance of such system during the complete RVD mission. Before introducing GNSS techniques in a simulated space environment, since the Galileo constellation was no fully operative at the time of executing such activities, a practical terrestrial application had been studied in order to collect data about availability and performance of the Galileo constellation as well as setting up relative navigation algorithm based on real observation. A detailed resume of the activities is extensively described in the present appendix, including details about relative navigation algorithms investigated and about the realization of the two electronic GNSS receivers built and used to collect Galileo navigation data.

## B.1 Advanced Navigation Techniques

During the vising period at the University of Sevilla, some advanced navigation techniques have been studied, focusing on GNSS relative navigation. In particular has been studied a possible implementation of a differential GNSS navigation involving the upcoming *Galileo Constellation*. The Galileo GNSS program was born on May 23, 2003, by an agreement between European Union (EU) and European Space Agency (ESA), with the goal to develop an accurate and efficient GNSS based on *civilian application*, differently from GPS and GLONASS which have *military purposes*. In addition, Galileo will improve localization performances, improving signal availability in urban areas and geographic regions above 75 deg of latitude. Moreover, Galileo will provide a number services[1]:

- *Open Service (OS)*: mass market service, providing position, velocity and timing information for any user equipped with a compatible Galileo receiver. Positioning accuracy is meter-level. It will have a global coverage but it will not provide the integrity signal.
- *Commercial Service (CS)*: it will provide higher performance than OS by paying a fee to the Galileo Service Provider. The signal is protected by commercial encryption. Positioning accuracy is centimeter-level. The integrity signal is not provided.
- *Public Regulated Service (PRS)*: this service is used by governmental institution. It will ensure a full operativity in all conditions and it will be provided by anti-jamming systems. Integrity signal is provided.
- *Search and Rescue Service (SAR)*: support service to SAR operations. The emergency signal emitted by a beacon is detected and used to track and drive SAR operations.

Currently, the Galileo constellation is still far from full operability: at the present moment of writing this dissertation, 12 Galileo satellites are operative and available, while at the moment of the study on December 2016, only 8 satellites was operative. A performance status report of Galileo Giove-A and Giove-B satellites can be found in [63].

[1] http://www.navipedia.net

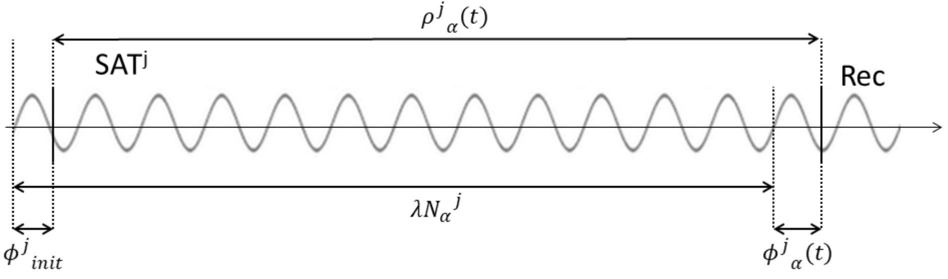


Fig. B.1 Carrier phase signal model.

## B.1.1 Basics of Differential GNSS Navigation

Differential GNSS (DGNSS) navigation techniques allows achievement of position accuracy of the order of centimeters. DGNSS positioning is supported by fixed GNSS ground stations, which is known the position with high accuracy. Such ground stations are able to compute errors from their known position and the position computed by processing received GNSS signals. Since position errors are constant within an area of tens to hundreds of square kilometers, corrections are broadcast locally and received by DGNSS compatible receivers which apply such corrections during the processing of the GNSS signal obtaining a improvement in position accuracy up to centimeter level. An alternative technique to compute relative position is the *real time kinematic (RTK)*. RTK uses information about the phase of the signal carrier to compute relative position between two receivers, or more. Relative positioning using carrier phase is deeply described in [12] and [64]. A scheme of the carrier phase signal can be found in Fig. B.1. The carrier phase observed by the receiver $\alpha$ of the signal emitted by the satellite $j$ can be modeled as

$$\phi_\alpha^j(t) = \rho_\alpha^j(t) - \lambda N_\alpha^j - c(dt^j + dT_\alpha) + \phi_{init}^j + c\,d_{ion}^j - c\,d_{trop}^j + \varepsilon(\phi) \qquad \text{(B.1)}$$

where $\phi_\alpha^j(t)$ is the carrier phase detected by the receiver, $\rho_\alpha^j(t)$ is the geometric range between the satellite and the receiver, $\lambda$ is the wavelength of the selected signal (1575.42 MHz for the L1/E1 band of GPS/Galileo satellites), $N_\alpha^j$ is the integer carries phase ambiguity, $c$ is the vacuum speed of light, $dt^j$ and $dT_\alpha$ are respectively the satellite and receiver clock errors, $d_{ion}^j$ and $d_{trop}^j$ are transmission delays of ionosphere and troposphere respectively, and $\varepsilon(\phi)$ is the carrier phase measurement error due to noise and multipath effects. Since the signal transmitted by a satellite is affected by different errors due to the paht the signal is travelling, some errors can be canceled using the *single difference* technique to process signal in the form of Eq. B.1 detected by two receivers $\alpha$ and $\beta$ as

$$\begin{aligned}\Delta\phi^{j}_{\beta\alpha}(t) &= \phi^{j}_{\beta}(t) - \phi^{j}_{\alpha}(t) = \\ &= \Delta\rho^{j}_{\beta\alpha}(t) - \lambda\Delta N^{j}_{\beta\alpha} - c\Delta dT_{\beta\alpha} + \Delta c\, dt^{j}_{ion_{\beta\alpha}} + \\ &+ \Delta c\, dt^{j}_{trop_{\beta\alpha}} + \varepsilon(\Delta\phi_{\beta\alpha})\end{aligned} \tag{B.2}$$

in which it has been used the symbol $\Delta$ to identify differential terms between the receiver $\beta$ and the receiver $\alpha$. With the single difference techniques, the initial phase and the satellite clock error are canceled each other, while other terms are still present, even if ionosphere- and troposphere-induced delays are negligible with respect multipath errors, since the two receiver which are defining the so called *baseline* are very close with respect to their distance from the satellites. To formally cancel such remaining terms, it is possible to apply the *double difference* technique, which is an extension of the single difference technique by computing differential terms with respect a second satellite $i$

$$\nabla\Delta\phi^{ji}_{\beta\alpha}(t) = \nabla\Delta\rho^{ji}_{\beta\alpha}(t) + \lambda\nabla\Delta N^{ji}_{\beta\alpha} + \varepsilon(\nabla\Delta\phi_{\beta\alpha}) \tag{B.3}$$

where the symbol $\nabla$ has been used to identify difference between satellite $j$ and satellite $i$. As it can be noticed in Eq. B.3, only the multipath error is still present. A third technique is based on *triple difference*, as the double difference but computing observables at epoch $t$ and $t+1$. Even if the triple differences is a more robust technique, especially in the case of loss of phase lock of a satellite signal, it will not improve dramatically the position accuracy, while it cannot be exploited the integer nature of the integer ambiguities even if a more complex resolution scheme is implemented [64]. For this reason, this technique will not be further investigated.

A critical aspect in determining relative position is the resolution of the integer ambiguity $N$. This integer number corresponds to the integer number of cycles which are contained in the LOS vector between the satellite and the receiver. This vector is also called *pseudorange* and it can be defined by

$$\rho^{j}_{\alpha}(t_0) = \phi^{j}_{\alpha}(t_0) + \lambda N \tag{B.4}$$

where $t_0$ is the initial *lock* time of the satellite by the receiver, while $\phi^{j}_{\alpha}(t_0)$ is the phase between the received carrier signal and the carrier signal generated by the receiver. The value of $N$ is computed at the lock time $t_0$ and it remains constant if the satellite is continuously locked by the receiver, and so the phase $\phi^j_\alpha(t)$ is varying with time and it is computed by evaluating the number of cycles from $N$

$$\rho^j_\alpha(t) = \phi^j_\alpha(t) + \lambda N \tag{B.5}$$

If the receiver loss the link with the satellite, the value of $N$ must be computed again: this event is called *cycle slip*. Integer ambiguity resolution is a critical aspect for determination of relative position using carrier phase and studies dates back to the beginning of GPS navigation [65], while more recent and advanced techniques can be found in literature, including application studies to the Galileo constellation [66], [67].

To complete the implementation of carrier phase relative GNSS it is necessary to compute the value of $\Delta\rho^j_{\beta\alpha}$. In terrestrial applications and also in some space applications, the length of the baseline is usually smaller than the distance between each receiver and the satellite, hence a planar wave can be assumed. According to this assumption, the unitary vectors from each receiver to the satellite $j$ are parallel, and consequently the the relative geometric range can be expressed by

$$\Delta\rho^j_{\beta\alpha}(t) = r^e_{\beta\alpha} \cdot e^j \tag{B.6}$$

where $r^e_{\beta\alpha}$ is the baseline vector in ECEF frame and $e^j$ is the unitary vector between each receiver and the satellite, as depicted in Fig. B.2. Hence, the baseline expressed in body frame is in the form

$$r^b_{\beta\alpha} = \begin{bmatrix} x_\beta - x_\alpha \\ y_\beta - y_\alpha \\ z_\beta - z_\alpha \end{bmatrix} \tag{B.7}$$

and the rotation in ECEF frame occurs trough the rotation matrix $R_{eb}$ from Body to ECEF frame, such as

$$r^e_{\beta\alpha} = R_{eb} \cdot r^b_{\beta\alpha} \tag{B.8}$$

Assuming the receiver $\beta$ as reference receiver, the resulting geometric relative range is found for the single and double difference schemes, $\Delta\rho^j_{\beta\alpha}(t)$ and $\nabla\Delta\rho^{ji}_{\beta\alpha}(t)$, and the desired scheme can be implemented.

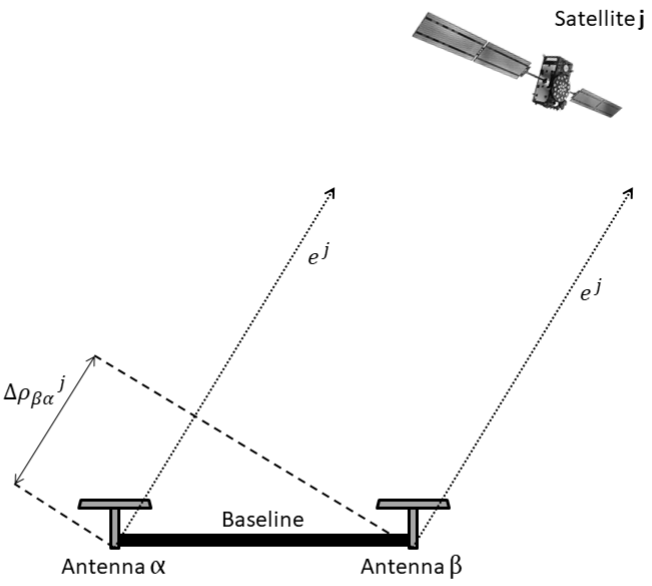


Fig. B.2 Interferometric model for single difference resolution.

## B.1.2 Design and Testing of a Galileo Receiver

The leading motivation of activities conducted at the University of Seville was the design and realization of a GNSS receiver compatible with the Galileo constellation to investigate the potential implementation of relative positioning algorithms, described in the previous section, in a real RTK test bench. Currently, it has been found in literature case studies which involve the Galileo constellation for the development of RTK networks, using Galileo alone [68] or combined with other GNSS constellation, such as GPS, GLONASS, or Beidu, and other GNSS augmentation systems [69], [70]. Expected innovation introduced by the present study is related to the real implementation of such RTK network, taking into account factors as satellite availability, real errors, and more, since previous studies are based on simulations.

For this purpose, it has been selected a GNSS receiver produced by *u-blox*: the u-blox NEO-M8. This receiver is able to track almost all the most used GNSS constellation, such as GPS, GLONASS, Beidu and Galileo, as well as GNSS augmentation systems such as SBAS, QZSS and IMES[2] [71], [72]. In addition, this receiver is the only one of the M8 series which is designed to provide raw data,

[2]SBAS - Satellite-Based Augmentation System; QZSS - Quasi-Zenith Satellite System; IMES - Indoor MEssaging System

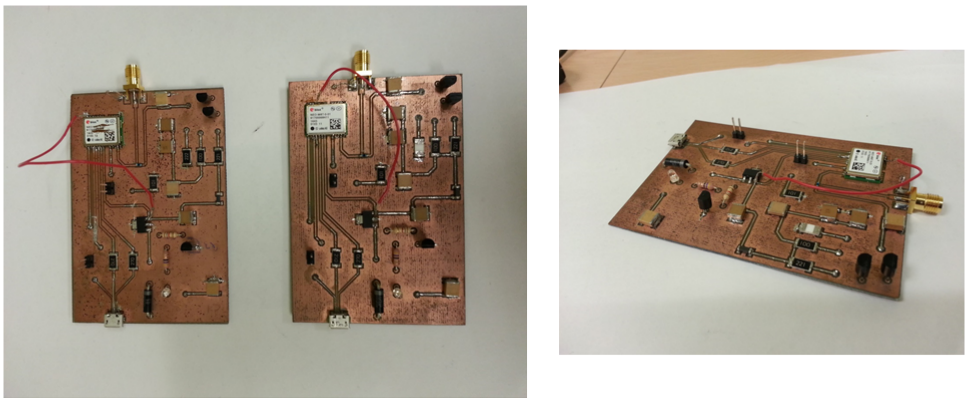
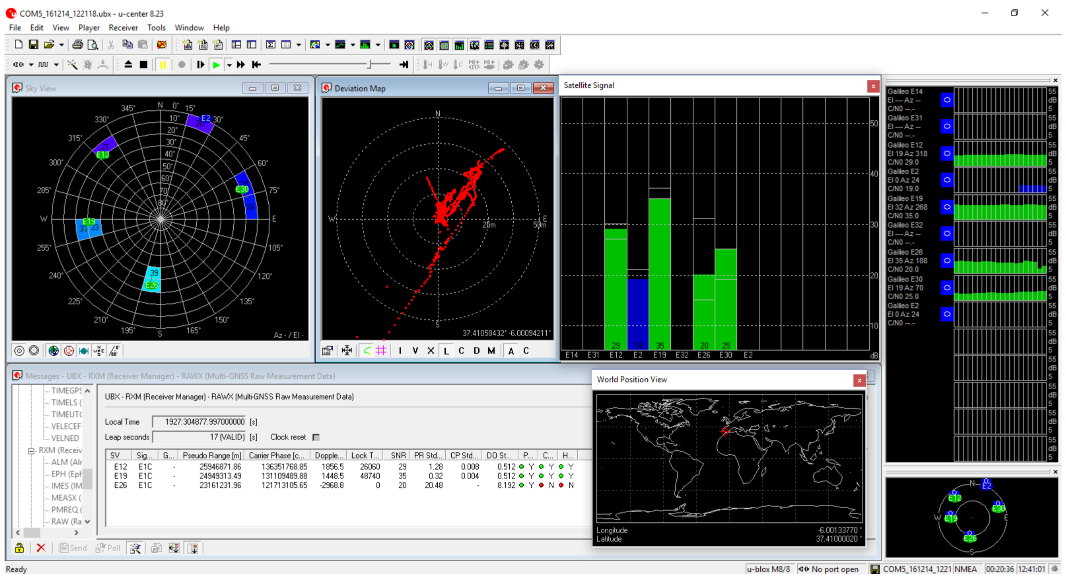


Fig. B.3 u-blox NEO-M8T GNSS receivers.

including carrier phase and integer ambiguity data. The design of the electronic board, supported by the team of the Department of Electronic Engineering of the University of Sevilla, has been based on the evaluation kit EVK-M8T [73], but the PCB has been completed re-designed with different components. In total, two complete Galileo receivers based on the NEO-M8T chip has been realized, as in Fig. B.3, with the purpose of test single and double difference techniques off-line, once the data availability provided by the chip had been confirmed. The electronic boards has been realized with the photoengraving technique.

The scheduled activities have been planned in order to preliminary tests the two receiver and consequently check the availability of Galileo constellation, which included only 8 satellites theoretically available at the scheduled date of test, on the beginning of December 2016. The second planned activity has been to get a measurement campaign in order to get data from both receivers and successively test single and double difference algorithms off-line by processing data of the measurement campaign, with the future purpose to implement such RTK network on real time platforms.

In Fig. B.4 is depicted a screenshot related to a tracking test of Galileo satellites. A total of five Galileo satellites has been tracked by the receiver on December 14, 2016, 12:41:01 CET, and four of them have been used to get the position fix. The computed position is reported in a map in Fig. B.5. Measurements have been get with the software *u-center v8.32* provided by u-blox. The software is able to display data computed by the receiver, connected via USB, and messages related to three different protocol specifications:



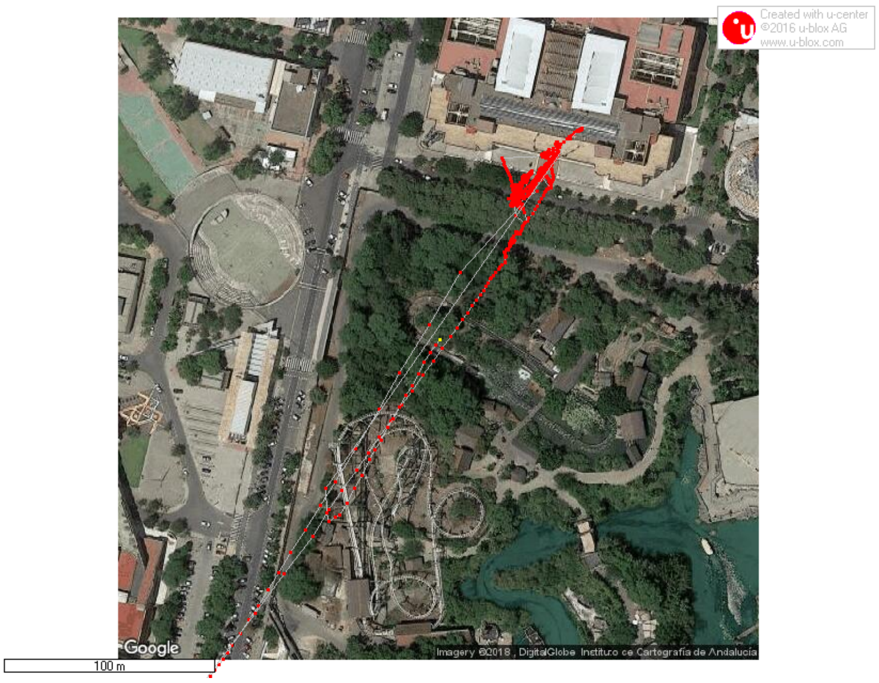
UBX - RXM (Receiver Manager) - RAWX (Multi-GNSS Raw Measurement Data)

Local Time 1927:304877.997000000 [s]
Leap seconds 17 (VALID) [s] Clock reset

| SV | Sig... | G... | Pseudo Range [m] | Carrier Phase [c... | Dopple... | Lock T... | SNR | PR Std... | CP Std... | DO St... | P... | C... | H... |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| E12 | E1C | - | 25946871.86 | 136351768.85 | 1856.5 | 26060 | 29 | 1.28 | 0.008 | 0.512 | ● Y | ● Y | ● Y |
| E19 | E1C | - | 24949313.49 | 131109489.88 | 1448.5 | 48740 | 35 | 0.32 | 0.004 | 0.512 | ● Y | ● Y | ● Y |
| E26 | E1C | - | 23161231.96 | 121713105.65 | -2968.8 | 0 | 20 | 20.48 | - | 8.192 | ● Y | ● N | ● N |

```
Galileo E14
El --- Az --
C/N0 ---.-
Galileo E31
El --- Az --
C/N0 ---.-
Galileo E12
El 19 Az 318
C/N0 29.0
Galileo E2
El 0 Az 24
C/N0 19.0
Galileo E19
El 32 Az 268
C/N0 35.0
Galileo E32
El --- Az --
C/N0 ---.-
Galileo E26
El 35 Az 188
C/N0 20.0
Galileo E30
El 19 Az 70
C/N0 25.0
Galileo E2
El 0 Az 24
C/N0 ---.-
```

Fig. B.4 Tracking of 5 Galileo satellites.



Created with u-center
©2016 u-blox AG
www.u-blox.com
Google
Imagery ©2018, DigitalGlobe, Instituto de Cartografía de Andalucía
100 m

Fig. B.5 Computed position in a map of Sevilla.

- *UBX*: proprietary protocol of u-blox.
- *NMEA*: protocol specification for naval and GPS applications.
- *RINEX*: data interchange format for raw data of navigation satellites.

Even if both NMEA and RINEX protocols provide raw data for all GNSS constellations, including Galileo, the firmware of the NEO-M8T receiver does not allows to output raw data with such protocols. In addition, the NEO-M8T was initially provided of the version 1.0 of the firmware (TIM1.02) which does not allow detection of Galileo navigation messages, hence the firmware of the receiver has been updated to a recent version. To receive Galileo signals is required the version 3.0 of the firmware, named TIM 1.10, but this firmware was not freely available at the moment of the test. The only freely available firmware version 3.0 was the firmware named SPG 3.01, which is not specifically designed for timing devices, such as the NEO-M8T, since it has been designed for standard precision devices, as NEO-M8N series [74]. By adoption of the firmware SPG 3.01 most capabilities has been lost, but position fix and raw data was available, even if only transmitted by the UBX protocol. In absence of an official TIM 1.10 firmware, the only solution found in official resources[3] has been the upgrade to SPG 3.01. Unfortunately, SPG 3.01 firmware and UBX protocol does not provide a precise position of each satellite, since the position of the satellite is known in terms of azimuth and elevation with accuracy of 1 deg, while the pseudorange is known and contained in the raw data message. For this reason, single or double differences schemes as in Eqs. B.2 and B.3 could not be applied to implement an accurate differential Galileo system, and consequently activities have been delayed. In addition, there were several difficulties to track more than four Galileo satellite during the whole measurement period due to the still spare constellation of only eight satellites, hence the quality of the position fix and other measurement was still poor compared with the expected theoretical performance, a further factor to postpone measurement campaigns.

[3]https://forum.u-blox.com