POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Path planning of redundant free-floating space robots with base attitude restoration and obstacle avoidance

Tesi di Laurea Magistrale in
Space Engineering - Ingegneria Spaziale

Author: **Andrea Bersani**

Student ID: 977341
Advisor: Prof. Mauro Massari
Academic Year: 2022-23

Abstract

The exploitation of autonomous satellite-mounted robotic manipulators for servicing purposes is a topic of interest for the sustainability and affordability of the space industry. These systems are able to efficiently perform activities that are either dangerous or impossible for astronauts, including repair of malfunctioning spacecrafts and removing space debris of different sizes.
However, while the technology has been demonstrated and multiple missions are under development, there are no currently active spacecrafts incorporating an autonomous robotic arm.
This work's aim is to propose a new approach that generates multiple possible trajectories that account for the key criticalities of Spacecraft-Manipulator Systems for the capture a target satellite.
The paths are generated incrementally exploiting the sampling of the joint space, following the scheme of Rapidly-Exploring Random Tree algorithms and including the minimization of a cost function assigned to each trajectory.

Keywords: On-orbit operations, Space robotics, Path planning, RRT algorithm

Sommario

L'uso di manipolatori robotici montati su satelliti per la manutenzione è un argomento di grande interesse per la sostenibilità e l'accessibilità del settore spaziale. Questi sistemi sono in grado di svolgere efficientemente attività che sono pericolose o impossibili per gli astronauti, come le riparazioni di veicoli spaziali malfunzionanti e la rimozione di detriti di diverse dimensioni.
Nonostante questa tecnologia sia stata dimostrata e ci siano più missioni in fase di sviluppo, non esistono al momento satelliti attivi che incorporano un braccio robotico autonomo.
L'obiettivo di questo lavoro è di proporre un nuovo approccio che generi più traiettorie che tengono in conto delle criticalità dei sistemi satellite-manipolatore per la cattura di un satellite obiettivo.
Le traiettorie sono generate in maniera incrementale sfruttando il campionamento dello spazio dei giunti, seguendo lo schema di un algoritmo RRT e includendo la minimizzazione di una funzione obiettivo, che viene assegnata ad ogni traiettoria.

Parole chiave: Operazioni spaziali, Robotica spaziale, Pianificazione del moto, Algoritmo RRT

Contents

1 | Introduction

The exploitation of space has created an endless amount of new possibilities and technologies, including telecommunications, Earth protection and monitoring, navigation and exploring missions that massively improved our understanding of the Universe and its origins.
However, the space sector features criticalities that have to be kept into account and, eventually, overcome: the environment is hazardous, missions and spacecrafts have very high costs, and the most commonly exploited orbits (e.g. Geosynchronous Orbits) are seeing an increase in the number of inhabiting spacecrafts.
For these reasons, spacecrafts are required to have a high degree of autonomy when launched, and accomplishing the mission goals is made difficult by many possible hazards of the space environments: radiation, oxidation, and impact with the growing number of debris. Besides, the biggest differences with respect to non-space industry are the limited possibility to remove non-functioning spacecrafts, the effort required in order to create larger orbital structures and the limited possibility of maintenance and repair of orbiting structures after launch, which causes many missions that could be saved if accessible to be indefinitely compromised.
On-Orbit Servicing (OOS), Active Debris Removal and On-Orbit Assembly attempt to remedy these problems, and employing robotics in order to accomplish them has received a high amount of attention over the past years.

1.1. On-Orbit Servicing and Active Debris Removal

On-orbit servicing refers to tasks such as inspecting, refuelling, upgrading, repairing or rescuing satellites that are employed to lengthen a mission's lifetime. Active Debris Removal refers to the activity of removing non-functioning satellites from their current economically and scientifically useful orbits to bring them towards disposal orbits [30]. On-Orbit Assembly is the set of activities to be performed to build larger orbital structures that cannot be launched altogether but require multiple smaller modules.
The concept of OOS has been proposed in the 1960s and it has been implemented in many

missions within the last century, with examples ranging from repairing the solar arrays and microwave antenna of Skylab in 1973, to the Space Shuttle program that successfully serviced the Solar Maximum Satellite (SMM), Palapa 2 and Westar 6, and ultimately, the Hubble Space Telescope, that has received a total of five servicing missions [20].
Most of these missions have been carried out either by remotely controlled devices, or by astronauts through what is typically referred to as *Extravehicular Activities* (EVA), that, however, present limitations due to the hazardous environment, typically require careful planning, and are sometimes unfeasible.
Besides, ADR cannot be performed through these methodologies, and thus robotics can be employed to perform all of these activities: it revolves around the usage of a *chaser* or *servicing* satellite operating on a *target* or *client* satellite or structure.
The most straightforward solution to the problem revolves around using satellite-mounted robotic arms, named *Spacecraft-Manipulator Systems* (SMS), to expand the available servicing missions to environments where manned operations are impossible, such as Geosynchronous Orbits, and to reduce the costs of these operations.
The currently operational robotic arms in space are teleoperated, and they are on board of the ISS.
The *Canadarm2* is a 7-Degrees of Freedom (DoF) teleoperated robotic manipulator that assists astronauts in their EVAs, aids in on-orbit assembly of the station itself, and it has also been utilized for On-Orbit Servicing demonstrations. It is based on the *Canadarm1* which has been used for Space Shuttle missions utilized to deploy, capture and repair satellites.
Other manipulators currently operated on board are the *Remote Manipulator System* (JEM-RMS) and the *European Robotic Arm* (ERA), tasked with managing payloads and assisting activities for astronauts.

1.1.1. Space Robots for OOS and ADR

While no completely autonomous robotic mission has yet been launched for On-Orbit Servicing, Active Debris Removal or On-Orbit Assembly, several proposed missions and demonstrators have attempted to show the potentiality of the usage of space robots for these tasks.
The pioneer mission in the field has been the *Engineering Test Satellite-VII* (ETS-VII), which was launched on 28th November 1997. It included an assembly of a client and servicer, and the latter was provided with a robotic manipulator arm to verify on-orbit technologies such as autonomous rendez-vous and docking, monitoring of the target satellite, replacement of its units, refuelling and assembly.

The *Demonstration of Autonomous Rendezvous Technology* (DART) included a secondary satellite that has been launched on 15th April 2005 and encountered failure during its operations. Its mission goal included an autonomous rendezvous, approach, and flyby of a target satellite.
Another successful demonstration of autonomous On-Orbit Servicing has been the *Orbital Express*, developed by DARPA and Boeing since 1999 and finally launched in 2007. The main goal of the mission was to establish on-orbit satellite servicing infrastructure for routine, cost-effective and autonomous resupply and reconfiguration. More in detail, the mission's operations included the launch, the far-field and near-field rendezvous, and ultimately the capture and mating operations, followed by release, separation, and disposal. The mission included two spacecrafts, ASTRO and NEXTSat. The former had the role of a servicer with a 3 meters length robotic arm, utilized to replace expendable components, while the latter functioned as the client satellite that included multiple features to ease its capture by the servicer.
DARPA has also developed the *Front-end Robotics Enabling Near-term Demonstration* (FREND), a 7-degrees-of-freedom robotic arm in the context of the *Spacecraft for the Universal Modification of Orbits* (SUMO), which successfully tested and evaluated critical challenges in the combination of stereophotogrammetric imaging with the FREND robotic manipulator arm on ground in the Naval Research Laboratory (NRL). The FREND arm and these evaluation will be key in the development of the future projects of DARPA, the *Robotic Servicing of Geostationary Satellites* (RSGS) which is aimed at developing a Robotic Servicing Vehicle (RSV), capable of capturing and manipulating targets through the usage of a manipulator arm; and the *Phoenix* program, aimed at the removal and eventual reuse of parts of decommissioned satellites in GEO orbits.
The *Deutsche Orbital Servicing* (DEOS) mission has been studied by Deutsche zentrum für Luft- und Raumfahrt (DLR), and it involved the design of an on-orbit servicing satellite with robots and related manipulation tools, with capability of capturing and servicing faulty satellites in multiple orbits, and it was expected to carry a servicer and a client in order to perform tests for series of key technologies, such as rendezvous and docking, manipulation, disposal to graveyard orbits or atmospheric reentry.
The *European Space Agency* (ESA) has also promoted the CleanSpace initiative, for which a first mission study, e.Deorbit, has been carried out first as a debris removal of ENVISAT, to then being renamed and modified to CleanSpace-1 to accomplish a wider variety of functions than ADR, including On-Orbit Servicing of multiple satellites. The spacecraft for this mission will capture targets utilizing multiple arms that will wrap around them.

1.2. Space Robotics: Review of key technologies

A *Space Robot*, or *Space Manipulator System* (SMS), is generally constituted by a base satellite on which one or multiple manipulator arms are mounted, that typically carry end-effectors, devices mounted at the free extremity of the arm and that will accomplish capture and manipulation tasks.
Many categorizations of space robots can be found in literature, and a brief summary is hereby given in order to more accurately describe the key technologies this work will refer to.
A first differentiation can be made between arms mounted on large space structures, such as the ISS, and manipulators mounted on spacecrafts of comparable size and inertia [35].

- *Manipulators mounted on large orbital structures*: Currently operational manipulators in space are on board of the ISS, and have been listed in previous paragraphs. They can be treated as fixed-base manipulators due to the fact that the mass and inertia of such structures are very high in comparison to mass and inertia of the manipulators, and that even when a heavy payload is transported by a manipulator, the influence on the state of the orbital structure can be in most cases neglected.

- *Manipulators mounted on small satellites*: Whenever the manipulators are of comparable mass and size with the satellites they are mounted on, the coupling dynamic between the motion of the manipulator and position and orientation of the base satellite becomes of greater effect. These systems are non-holonomic, and the coupling effect shall either be compensated by the Attitude and Orbital Control System (AOCS) of the satellite, or be minimized within the manipulator arm trajectory planning.

Within small satellite-mounted manipulators, a further division can be traced between scenarios in which the Attitude and Orbital Control System is acting during the capture, and other scenarios in which it is completely turned off [7].

- *Free-flying space robot*: Free-flying space robots are characterized by an active AOCS through the duration of the capture manoeuvre. In this way, thrusters or other actuators such as reaction wheels can compensate for the effect of the torques on the base induced by the motion of the manipulator. The satellite's position and orientation can follow a prescribed trajectory, and end-effectors can reach their target in pre-determined configurations, requiring the simultaneous control of the base and of the manipulator.

- *Free-floating space robot*: Free-floating space robots assume to have no external forces acting on the system (including the AOCS), thus since linear and angular momenta are conserved, they will feature a base that moves due to the reaction forces and torques imposed my the motion of the manipulator. It has the advantage of not utilizing thrusters during the operations, hence of reducing fuel consumption. The system shows nonholonomic behaviour due to the non-integrability of the momentum equations.

Differences can also be individuated in terms of the mission objective to be accomplished. In particular, the target satellites can be classified between cooperative and non-cooperative [33].

- *Cooperative targets*: cooperative targets are defined as those satellites that typically feature an AOCS that is at least partially functioning. This means that they can aid the servicer satellites by correcting their position and attitude. These targets are typical of OOS missions.
- *Non-cooperative targets*: these targets are either unable to control their attitude because of a damaged AOCS or they are at the end of their lifetime, thus showing a behaviour of uncontrolled tumbling. These are the common targets of ADR missions and require much more careful planning for the chaser operations.

Typically, the capturing process starts with far and close-range rendezvous manoeuvres, and then the operations for robotic arms can be subdivided into three or four main phases [9, 33].

1. *Observation and Planning Phase* is the initial required phase in which the chaser is put at a safe distance from the target such that it can estimate its state in terms of position and velocity and its parameters like its mass or its geometrical features with the highest accuracy possible, to then be able to plan effectively the subsequent phases of the mission.
2. *Pre-Grasping Phase* includes the final approach of the satellite-manipulator system towards the grasping point.
3. *Post-Grasping Phase* is the final phase, it includes the impact of the manipulator end-effector with the target satellite, the effective capture and finally the post-capturing stabilization, necessary to accomplish the mission goals of OOS or ADR.

1.2.1. Path Planning Problem

The path planning of space-manipulator systems constitutes the pre-grasping phase and it is the determination of the displacement, velocity and acceleration of the space robot to have its end-effector reach an identified grasping point on the target satellite.
Path planning can be performed in the *Cartesian space*, defining the end-effector trajectory, velocity and acceleration to then employ techniques such as inverse kinematics or dynamics [40] to compute the relative time evolution of the joint variables or the required torques; it can be performed in the *configuration space*, where the joint variables trajectories are defined to then calculate the resulting end-effector position and orientation [6].
The main complication for space robots with respect to Earth-based robots is that the latter are typically considered as fixed-base, while for space robots the base is unconstrained and sometimes uncontrolled, leading to 6 more Degrees of Freedom (DoF) for the problem [39].
For this reason, research for this environment had to not only take into account typical parameters to be maximized such as manipulability, or to be minimized such as control effort for the manipulator, but also a minimization on the impact of the robotic manipulator motion on the base attitude and translation.
Furthermore, Space is typically a dynamic environment, where multiple moving obstacles can be present in the workspace of the robot. Thus, the techniques for obstacle avoidance have to take this into account.

Obstacle Avoidance

Avoiding collisions with objects is key in achieving the goal of the missions for space robots. The problem with obstacles is twofold: first is modelling the obstacles in order to accurately represent them with a degree of safety with respect to model inaccuracy and the second is the dynamic nature of the obstacles for space applications.
Approximately accurate models of different obstacles with geometric primitives based on *super-quadric* approximation have been studied [24, 25]. Since most of the satellites and, in general, spacecrafts, are composed of standardized modules, objects can be modelled as a composition of surfaces defined by these curves, without which obstacles would be modelled as convex surfaces with loss of workspace.
Obstacles can be treated typically as hard constraints for optimization problems or sampling-based approaches, however *Artificial Potential Field* (APF) methods, first introduced in [15], typically define a repulsive force with respect to the obstacles and an attractive force towards reaching the goal, relaxing the problem in terms of constraints while still guaranteeing collision-free path planning. The APF methods were successfully applied

to free-floating space robots as standalone algorithms for avoiding obstacles of different dimension and shape in space, first by finding a collision-free path for the end-effector, and then applying inverse kinematics to apply a simple collision detection based on the depth and direction of the links [22]; APF methods employing Laplace potential fields have been investigated to avoid local minima within the obstacles themselves [50]. Due to the translation of constraints into parts of the objective function, the APF have been used in conjunction with other optimization methods, for example reinforcement learning [21] or A* algorithm [12].
Avoiding obstacles can also be based on *Relative Velocity* between the obstacles and the links and joints of the manipulator. Enhanced version of the *Velocity Damper Method* (VDM) [53] has been employed in order to damp the relative velocity that the manipulator has with obstacles, in particular only if the closest point of the manipulator to the obstacle, continuously computed, falls within a pre-defined safety distance from the object.
Optimization Techniques are available in order to include the distances between obstacles and manipulator as part of the objective function. In particular, various rationales can be utilized in order to find the closest point to the obstacle, and then attempt to maximize this distance while optimizing other quantities, such as manipulator manipulability and base attitude change. This approach has been employed with controlled base and manipulator for a close rendezvous manoeuvre [17, 18].
One of the most promising set of algorithms to avoid collisions is *Sampling-based path planning*, which has been extensively covered for Earth mobile robots, and it has been explored for space manipulators. These revolve around generating random samples through a pre-defined metric in order to sample the state space and find a feasible solution to reach the goal state.
In particular, the *A* algorithm*, which is originally a grid search algorithm, has been modified to use its rationale to develop a sampling-based method that bases its search on the currently most promising path towards the goal through an heuristic evaluation [34]. Other sampling-based algorithms that were utilized are the *Rapidly-exploring Random Trees* (RRT), first introduced in [16]. These methods generate a tree of sample states (either in the workspace or in the joint space) with a bias towards the unexplored regions of the state space and perform well with non-holonomic constraints and for dynamic obstacle avoidance. Their usage for space manipulators has been explored in [37], and modified methods such as the *Bidirectional RRT* (BiRRT), growing trees that eventually intersect from both the initial state and one or multiple final state, have shown great performance in avoding collisions [2, 38].
RRTs have also been applied to optimal motion planning (RRT*) [14], utilizing sampling

towards unexplored regions while minimizing a certain cost function. For space manipulators, nonlinear gradient-based optimizations can be embedded in a RRT*, for example minimizing the mechanical energy and the actuator actions for the trajectory optimization [41]. Typically, RRT methods require a smoothing technique for the trajectory that is initially jerky due to the nature of the sampling. This can be done in two different steps, finding waypoints through the RRT method to then apply spline fitting afterwards [52].

Base Reaction Minimization

The dynamical coupling between the base satellite and the robotic manipulator is typically troublesome as it makes the end-effector path harder to compute. Furthermore, if the reaction forces are too large in magnitude the AOCS of the satellite might lack the power to compensate them. Their minimization is of pivotal importance for Space Manipulator Systems.

A proposed solution is the Disturbance Map (DM) that has then been improved to an *Enhanced Disturbance Map* (EDM), constructed through a singular value decomposition of a matrix ruling the dynamic coupling that allows to determine "hot zones" and "cold zones" for which the robot manipulator motion has a greater or lower impact on the satellite attitude [8].

The *Virtual Manipulator* (VM) has been introduced in [43] as a way of modelling the system without external action on the base in order to produce reaction-less motion of the arm, and different models based on it have been proposed [54].

Other solutions are based on a *coordinated control* of the satellite base and of the manipulator [32]. Since both the attitude control system and the on-board computer have limited capabilities, the manoeuvre can be divided in two parts: a feed-forward attitude control against the robot's reaction forces and a control on the arm such that it produces forces that can be counteracted [31]. A later work uses feedback linearization of the dynamic model in order to introduce an optimization problem based on quaternions, yielding a final solution with asymptotical stability in terms of base orientation [1]. The combination of coordinated control and navigation through cameras has been studied, utilizing Proportional-Derivative controllers to optimize the trajectory [47, 48].

Another possibility is to use Near-Optimal path planning in order to enforce base attitude restoration from initial to final time. This has proven useful especially in bidirectional methods employing Lyapunov functions minimizing the difference between the initial and final manipulator states attitude [46].

The *Reaction Null Space* [28, 29] has received particular attention as it defines a set of joint velocities that are projected in the null space of the torque acting on the base. In particular, it has been defined both at the velocity level, in the equations for linear and angular momenta, and at acceleration level, in order to treat problems with external forces acting on the system. The method presents the limit of requiring space robots with a number of degrees of freedom at least equal to the number of component of the torque to be zeroed plus the number of end-effector coordinates to be controlled. A solution that has been proposed involves minimizing the torque acting on the base in the least-square sense, including constraints on the acceleration of the joints [5].

Optimization Methods

Various quantities can be optimized for the trajectory planning of a space robot. An example has been given [17, 18] that includes obstacle avoidance within the objective function, but collisions can be included as constraints to optimize other parameters.
Considering the contact force between the servicer's end-effector and the grasping handle is very important in the guidance of the robotic manipulator as it is a pseudo-instantaneous external force that will enter the dynamics equations. For this problem, two solutions are possible: one is to impose zero relative velocity between the end-effector and the grasping point at the final time, the other is to have the contact force pass through the center of mass of the servicing satellite system, or to minimize the angle that the force has with it. This way, an optimal capture is defined in a closed form where the final point of the Optimal Control Problem has been already optimized beforehand, leaving the possibility to optimize other quantities, such as the torque imposed by the manoeuvre on the spacecraft's base [10].
The energy demand of the system must also be kept as low as possible due to the limited possibilities of the electric power systems in space. Examples have been given in which the motion of the robot has been decomposed into primitives to then compute the power requested as the integral over the course of the manoeuvre of product between the joint torques and the joint velocities, and then applying an optimization method such as the Genetic Algorithm (GA) to minimize it [4].

1.3. Proposed Approach

This work's main aim is to present a procedure for the trajectory planning of a Spacecraft-Manipulator System in free-floating mode that can be adapted to multiple missions and capture scenarios.
The approach of choice includes general requirements for these manoeuvres such as obstacle avoidance and time minimization, and includes an overall minimization of the change in attitude between the initial and final conditions of the Spacecraft-Manipulator System.
A sampling-based path planning approach has been selected in order to accomplish the capture task prioritizing a robust obstacle avoidance, which this type of algorithm guarantee even in dynamic environments.
This RRT algorithm generates a path for the end-effector towards a goal condition by exploring the configuration space of the manipulator joint variables which will produce a rotation of the chaser base spacecraft.
The attitude variation is not minimized along the whole trajectory, leaving the system with a wider manoeuvre capability, but the final attitude configuration is brought to a desired value through optimized sample generation.

The path planning for the mounted robotic manipulator is based on a simplified orbital model: firstly, both the target and the chaser lie on circular orbits; secondly, the orbital perturbations such as atmospheric drag, zonal harmonics and solar radiation pressure are not included in the model; thirdly, the rendezvous manoeuvre of the chaser ensemble with the target satellite is considered completed beforehand by the Attitude and Orbital Control System; lastly, only manoeuvres of short duration are considered.
The model of both the target and the chaser spacecraft is also subject to hypotheses: both systems are assumed to be composed of rigid bodies only, and the physical properties of the rigid bodies, such as masses, inertia and dimensions to be completely known.
Finally, the navigation and sensor errors are not considered: the states of both the spacecraft-manipulator system and of the target are exactly identified at every time instant of the manoeuvre.

1.4. Thesis Structure

The thesis is divided into four main sections: first, in chapter 2, the overall model of the capture scenario is described: it includes an introduction of the definitions and notation utilized for the work, followed by a general description of the simplified orbital models the work is referring to.
This chapter also includes the detailed model for the kinematics and the dynamics of the spacecraft-manipulator system that will then be employed for the trajectory planning.

Then, in chapter 3 the path planning approach is discussed in detail, starting from a brief overview of the Rapidly-Exploring Random Tree algorithms and their variants.
Afterwards, the proposed path planning approach is described from the definition of each of the composing blocks, including the obstacle avoidance method and the optimized nodes generation.
Ultimately, the path interpolation with polynomial splines is discussed.

Chapter 4 describes the case study through which the proposed approach is validated, including a quantitative description of the chaser and the target.
The results of the case study simulation are then presented and discussed, including an analysis on the qualities and the limits of the procedure.

Finally, chapter 5 presents a summary of the work, the overall assessment on the performance of the approach and proposals on future additions for further validation.

2 | System Modelling

2.1. Definitions and Notation

The representation of the basic geometric entities follows [40], and is organized as follows:

- **Reference Frames** are identified by Σ_n and they are defined from their origin O_n and three orthonormal axes $\mathbf{i}, \mathbf{j}, \mathbf{k}$.
- **Vectors** are identified by $\mathbf{v}^k_{(P)}$ where the superscript indicates the reference frame in which said vectors are expressed. Whenever present, the subscripts indicate the object to which the vector is referring to (in terms of position, velocity, etc.)
- **Rotation Matrices** are identified by R^j_k, where subscripts indicate the reference frame from which the quantities are rotated, and the superscripts indicate the reference frame to which the quantities are referred after the rotation. These matrices are orthonormal, hence $R^{-1} = R^T$.

The general representation of a point P in space with respect to two different reference frames Σ_{i-1} and Σ_i is:

$$\mathbf{p}^i_P = \mathbf{o}^i_{i-1} + R^i_{i-1}\mathbf{p}^{i-1}_P \tag{2.1}$$

The quantities in eq. (2.1) are:

- $\mathbf{p}^i_P$: position of the point P in the reference frame Σ_i.
- $\mathbf{o}^i_{i-1}$: vector indicating the position of the origin of Σ_{i-1} expressed in Σ_i
- R^i_{i-1}: rotation matrix from Σ_{i-1} to Σ_i.
- $\mathbf{p}^{i-1}_P$: vector expressing the position of the point in Σ_{i-1}.

This transformation between reference frames is available in a more compact form in what is called *Homogeneous Transformation Matrix* which combines the translation and the rotation between two different reference frames. In this framework, *homogeneous vectors* are represented by their three spatial components as usual, and by a fourth unit component that is introduced for dimensional congruence in the transformations:

$$\tilde{\mathbf{p}} = \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix} \tag{2.2}$$

Then, the transformation from a reference frame to another is simply expressed as:

$$\tilde{\mathbf{p}}_P^i = T_{i-1}^i \tilde{\mathbf{p}}_P^{i-1} \tag{2.3}$$

Where the Homogeneous Transformation Matrix is:

$$T_{i-1}^i = \begin{bmatrix} R_{i-1}^i & \mathbf{o}_{i-1}^i \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{2.4}$$

The vector $\mathbf{0}^T$ represents a null row vector, and to these matrices the rules of consecutive transformations through consecutive matrix multiplication apply as if they were common rotation matrices. It is however important to highlight a major difference in the fact that since these matrices are not orthonormal, their inverse differs from their transpose ${T_{i-1}^i}^{-1} \neq {T_{i-1}^i}^T$.
A common definition in literature that will be utilized throughout the work to describe the kinematics and dynamics of the Space Manipulator System is the skew-symmetric matrix of a vector $\mathbf{a} = \begin{bmatrix} a_x & a_y & a_z \end{bmatrix}^T$, which can be utilized to symbolize the cross product between a two vectors $\mathbf{b} = \mathbf{a} \times \mathbf{c}$ in a vector-matrix notation $\mathbf{b} = \mathbf{a}^\times \mathbf{c}$:

$$\mathbf{a}^\times = S(\mathbf{a}) = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \tag{2.5}$$

2.1.1. Euler Angles Representation

Euler angles represent a 3D rotation as a combination of three consecutive rotations around arbitrarily chosen axes from the reference frames defined by the previous rotation. In this way, any rotation in plane can be represented by the three angles, $[\varphi, \vartheta, \psi]$.
The axes about which the rotation is performed modify the structure of the rotation matrices as in (2.6), and the rotation sequence defines the order in which they are combined

in order to find the overall rotation matrix.

$$R_i(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \tag{2.6a}$$

$$R_j(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \tag{2.6b}$$

$$R_k(\gamma) = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.6c}$$

Hereon the notation utilized is the Roll-Pitch-Yaw, or Z-Y-X series of rotations, in which to rotate quantities from a reference frame to another, the following sequence of rotations is utilized: a rotation ψ about i, a rotation ϑ about j and ultimately a rotation φ about k.
This generates the overall rotation matrix:

$$R(\varphi, \vartheta, \psi) = R_k(\varphi) R_j(\vartheta) R_i(\psi) \tag{2.7}$$

In order to retrieve the Euler parameters from the overall rotation matrix, the following relations apply. For $\theta \in \left(-\frac{\pi}{2}; \frac{\pi}{2}\right)$:

$$\varphi = \text{atan2}(r_{21}, r_{11}) \tag{2.8a}$$

$$\vartheta = \text{atan2}\left(-r_{31}, \sqrt{r_{32}^2 + r_{33}^2}\right) \tag{2.8b}$$

$$\psi = \text{atan2}(r_{32}, r_{33}) \tag{2.8c}$$

While for $\theta \in \left(\frac{\pi}{2}; \frac{3}{2}\pi\right)$:

$$\varphi = \text{atan2}(-r_{21}, -r_{11}) \tag{2.9a}$$

$$\vartheta = \text{atan2}\left(-r_{31}, -\sqrt{r_{32}^2 + r_{33}^2}\right) \tag{2.9b}$$

$$\psi = \text{atan2}(-r_{32}, -r_{33}) \tag{2.9c}$$

And they are undefined for $\cos\vartheta = 0$, case in which only the sum of ψ and φ can be obtained.
The advantage in using Euler angles is the immediate physical interpretation that can be given on their value and the minimal representation, meaning that three parameters

are utilized to describe the same number of rotations in space. The issues concerning the Euler angles are the representation singularities in $\vartheta = \pm\frac{\pi}{2}$.

2.1.2. Quaternion Representation

In order to avoid representation singularities, quaternions can be utilized for the representation of rotations.

Quaternions are unitary norm vectors of four elements based on the Euler axis and angle representation, and are thus non-minimal as four parameters are utilized to represent three rotations in space. Their elements are divided between what is typically referred to as the *scalar part* η and the *vector part* $\boldsymbol{\varepsilon}$. They are defined as:

$$\eta = \cos\frac{\vartheta}{2} \tag{2.10a}$$

$$\boldsymbol{\varepsilon} = \mathbf{r}\sin\frac{\vartheta}{2} \tag{2.10b}$$

Where $\mathbf{r}$ is the axis with respect to which the rotation is performed, and θ is the angle of the mentioned rotation.

The rotation matrix between a fixed and a rotated reference frame through quaternions is:

$$R(\eta, \boldsymbol{\varepsilon}) = \begin{bmatrix} 2(\eta^2 + \epsilon_x^2) - 1 & 2(\epsilon_x\epsilon_y - \eta\epsilon_z) & 2(\epsilon_x\epsilon_z + \eta\epsilon_x) \\ 2(\epsilon_x\epsilon_y + \eta\epsilon_z) & 2(\eta^2 + \epsilon_y^2) - 1 & 2(\epsilon_y\epsilon_z + \eta\epsilon_x) \\ 2(\epsilon_x\epsilon_z + \eta\epsilon_y) & 2(\epsilon_y\epsilon_z + \eta\epsilon_x) & 2(\eta^2 + \epsilon_z^2) - 1 \end{bmatrix} \tag{2.11}$$

The inverse transformation is always possible, and defining the elements of the rotation matrix as r_{ij} where i is the row and j is the column they belong to, it is possible to obtain:

$$\eta = \frac{1}{2}\sqrt{r_{11} + r_{22} + r_{33} + 1} \tag{2.12a}$$

$$\boldsymbol{\varepsilon} = \frac{1}{2}\begin{bmatrix} sign(r_{32} - r_{23})\sqrt{r_{11} - r_{22} - r_{33} + 1} \\ sign(r_{13} - r_{31})\sqrt{r_{22} - r_{33} - r_{11} + 1} \\ sign(r_{21} - r_{12})\sqrt{r_{33} - r_{11} - r_{22} + 1} \end{bmatrix} \tag{2.12b}$$

Complementary to the Euler angles description, the unit quaternions avoid representation singularities but lose an immediate physical interpretation.

2.2. Reference Scenario

A typical scenario for an on-orbit capture is analyzed. A target satellite is orbiting Earth, and the chaser satellite has completed the far rendezvous manoeuvre in order to reach a position from which the manipulator capture sequence can be started.

2.2.1. Reference Frames Definition

In order to accurately model the system, reference frames to which physical quantities of either the target satellite or the servicer refer to are defined.

- The *Earth-Centered Inertial* reference frame Σ_{ECI}, is used to define a reference frame to which quantities of both satellites can be referred to.
  Its origin is in the center of the Earth, and its axes are: X_{ECI} on the celestial equatorial plane, pointing to the Vernal equinox direction, Z_{ECI} pointing towards the Celestial North Pole and Y_{ECI} as to form a right-handed frame.
  Any vector that is expressed in an inertial reference frame will have its superscript omitted to lighten the notation.

- The *Local Vertical Local Horizontal* reference frame can be defined individually both for the chaser and for the target $\Sigma^T_{LVLH}, \Sigma^C_{LVLH}$, it is a moving reference frame with respect to the inertial reference frame.
  It has its origin in the center of mass of the satellite, and its axes are: x_{LVLH} in the direction of its position vector in the inertial reference frame, z_{LVLH} perpendicular to the spacecraft's orbital plane and directed as its angular momentum, and finally y_{LVLH} as to form a right-handed frame.

- The spacecraft *Body Axes* frame, that will be defined both for the chaser Σ^C_B and for the target Σ^T_B, is a frame attached to the satellite body itself.
  Its origin is in the Center of Mass (CoM) of the satellite and its axes are directed as the *principal axes of inertia* for simplicity, as common in literature.

The two spacecrafts are treated in the same framework for what concerns the orbital mechanics of the two systems, whereas the kinematic and dynamic model differs between the target, presented in section 2.3, and the chaser, for which the presence of the robotic arm adds terms in the computation of the attitude kinematics and dynamics, treated in detail in section 2.4.

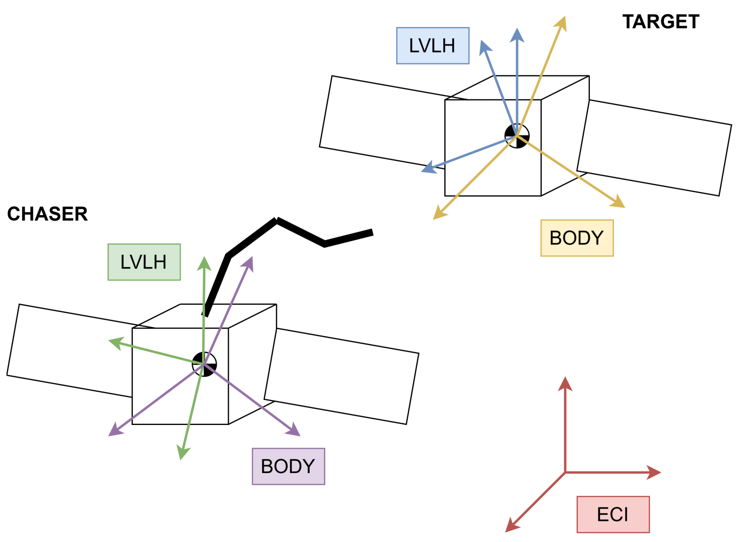


Figure 2.1: Capture Scenario with Reference Frames

Reference Frames Transformations

The defined reference frames will have a relative distance and orientation. In order to refer quantities that are defined with respect to a certain reference frame to another, it is possible to apply *Homogeneous Transformation Matrices* to describe the transformation of the quantities between different frames.

The inertial reference frame is considered to be the frame to which every other frame must refer to, as it is fixed and the transformations from and to it are simply defined.

LVLH frames are rotating with the satellite's progress along its orbit. For this reason, their orientation depends on the spacecrafts' *mean motion* n.

Furthermore, they are centered in the satellite (or satellite base for the chaser) CoM, thus a translation with respect to the center of the ECI frame is necessary. The homogeneous transformation matrix defining the relationship between Σ_{LVLH} and Σ_{ECI} frame is:

$$T_{ECI}^{LVLH}(t) = \begin{bmatrix} c_\Omega \cos nt - s_\Omega c_i \sin nt & c_\Omega \sin nt + s_\Omega c_i \cos nt & s_\Omega \sin i & r_x(t) \\ -s_\Omega \cos nt - c_\Omega c_i \sin nt & -s_\Omega \sin nt - c_\Omega c_i \cos nt & c_\Omega s_i & r_y(t) \\ s_i \sin nt & -s_i \cos nt & c_i & r_z(t) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.13}$$

Where c_Ω, s_i are respectively the cosine and sine of constant angles Ω, which is the Right Ascension of the Ascending Node (RAAN), and i, which is is the orbit's inclination. t is the explicit dependence on time and $\mathbf{r}_S = \begin{bmatrix} r_x(t) & r_y(t) & r_z(t) \end{bmatrix}$ is the satellite's position vector in the *ECI* frame.

The transformation between the Σ_{ECI} and the body axes Σ_B of the spacecrafts depends on the representation of the orientation of the latter with respect to the former. Utilizing Euler angles that are grouped in a vector $\boldsymbol{\varphi} = \begin{bmatrix} \varphi & \vartheta & \psi \end{bmatrix}^T$, having defined the rotation matrix between the two frames in section 2.1.1 it is possible to write the following homogeneous transformation matrix:

$$T^B_{ECI}(\boldsymbol{\varphi}(t)) = \begin{bmatrix} R^B_{ECI}(\boldsymbol{\varphi}(t)) & \mathbf{r}_S(t) \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{2.14}$$

A combined rotation can be utilized in order to directly obtain the transformation from the body frame to the respective object's $LVLH$ frame, that share the same origin.

$$T^B_{LVLH}\left(t, \boldsymbol{\varphi}(t)\right) = \begin{bmatrix} R^{ECI}_{LVLH}(t) R^B_{ECI}(\boldsymbol{\varphi}) & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{2.15}$$

Where $R^{ECI}_{LVLH} \in \mathbb{R}^{3\times3}$ is the inverse of the submatrix composed by the first three rows and columns of T^{LVLH}_{ECI}. Since this is a rotation matrix, it is orthonormal and $R^{-1} = R^T$. Under the assumption of manoeuvres of short durations, it is possible to assume that the motion of the satellites with respect to Earth is negligible, thus the effects of orbital mechanics can be completely ignored such that Σ_{LVLH} features the same orientation for the chaser and the target, and can thus considered fixed in time, becoming a target-centered Inertial Reference Frame, hereon called $\Sigma_{\mathcal{I}}$.
Not modelling the orbital mechanics, the relative target position $\mathbf{r}^{\mathcal{I}}_C$ depends only on the coupling mechanism within the chaser satellite.
The homogeneous transformation matrix used to describe the chaser relative motion and orientation is:

$$T^{B,C}_{\mathcal{I}} = \begin{bmatrix} R^{B,C}_{\mathcal{I}} & \mathbf{r}^{\mathcal{I}}_C \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{2.16}$$

2.3. Target Motion Model

2.3.1. Target Attitude Kinematics

The attitude kinematics are defined as the time evolution of the orientation of a satellite's body frame with respect to the inertial reference frame.
Thus, they are expressed as the first derivative of the representation of choice, which is

in this case the quaternions, for the rotations:

$$\dot{\mathbf{q}} = \frac{1}{2}\begin{bmatrix} 0 & \omega_z & -\omega_y & \omega_x \\ -\omega_z & 0 & \omega_x & \omega_y \\ \omega_y & -\omega_x & 0 & \omega_z \\ -\omega_x & -\omega_y & -\omega_z & 0 \end{bmatrix}\mathbf{q} \tag{2.17}$$

It is also possible to define the same expression as explicitly depending from the angular velocity through a proportionality matrix depending on quaternions:

$$\dot{\mathbf{q}} = \frac{1}{2}\Xi(\mathbf{q})\boldsymbol{\omega} \tag{2.18}$$

The reason for which the quaternions are hereby utilized is detailed in section 3.2.4.

2.3.2. Target Attitude Dynamics

The attitude dynamics for the target are described by the Euler's equations for an uncontrolled satellite. They can be expressed in matrix form:

$$I_T\dot{\boldsymbol{\omega}} - I_T\boldsymbol{\omega} \times \boldsymbol{\omega} = \mathbf{0} \tag{2.19}$$

Where I_T is the inertia tensor of the target satellite, $\boldsymbol{\omega}$ is its angular velocity with respect to the target-centered $LVLH$ frame, expressed in body axes.

2.3.3. Grasping Point Motion

Lastly, combining the models for the description of the target's motion, it is possible to define the trajectory of the grasping point. Defining a reference frame Σ_{GP} that will identify the preferred capture orientation, and assuming every body to be rigid, a constant transformation matrix identifying position and orientation of the grasping point and its frame can be defined in $\Sigma_{B,T}$:

$$\mathbf{T}_{B,T}^{GP} = \begin{bmatrix} \mathbf{R}_{B,T}^{GP} & \mathbf{x}_{GP}^{B,T} \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{2.20}$$

Thus, it is simple to obtain its trajectory in time, as it depends on the time evolution of the transformation matrices, connected with the time evolution of the quaternions, which is known as the dynamics of the system are fully described.

2.4. Space Manipulator System Model

The servicing satellite is composed by a base satellite, with degrees of freedom corresponding to the translational and rotational variables, and by the manipulator arm, a chain series of rigid bodies.

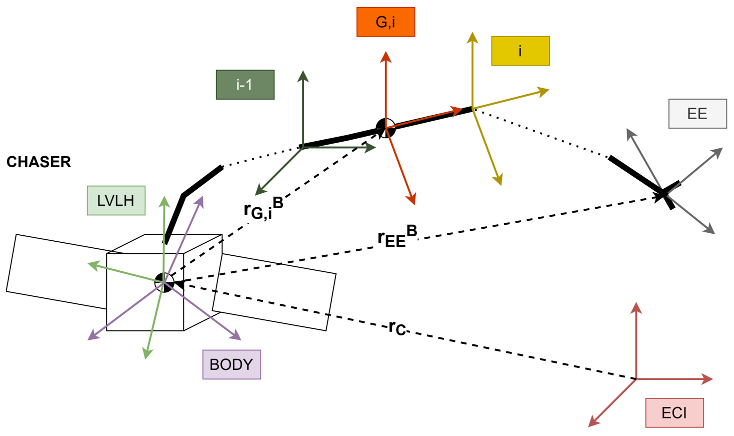


Figure 2.2: Chaser Model

2.4.1. Kinematics

For what concerns the kinematic and dynamic model of the system, the framework presented in [45] has been followed.
The kinematic formulation for the system is based on the division of the problem between the kinematics of the base satellite and the kinematics of the manipulator, both described in the target-centered $\Sigma_{\mathcal{I}}$ frame.
For the *base satellite kinematics* it is preferred to use Euler angles for the rotations representation as they are a minimal attitude representation and thus a minimum number of variables for the equations of motion. The angular velocity of the body and the derivatives of the Euler angles are related by summing the contributions of the angular rates of each Euler angle about its respective axis:

$$\boldsymbol{\omega} = \dot{\psi}\mathbf{i} + \dot{\vartheta}R(\psi)\mathbf{j} + \dot{\varphi}R(\vartheta)R(\psi)\mathbf{k} \tag{2.21}$$

These equations are available in matrix form and they can be specified for the examined case of selection of RPY angles. The relation that will be used here onward relates the rate of change of the Euler angles to the angular velocity of the satellite body:

$$\begin{bmatrix} \dot{\varphi} \\ \dot{\vartheta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & \frac{\sin\psi}{\cos\vartheta} & \frac{\cos\psi}{\cos\vartheta} \\ 0 & \cos\psi & -\sin\psi \\ 1 & \tan\vartheta\sin\psi & \tan\vartheta\cos\psi \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{2.22}$$

The Euler Angles can be integrated, thus it is possible to obtain the relative orientation of the chaser spacecraft base with respect to the LVLH frame knowing the angular velocity of the body by integrating (2.22).

2.4.2. Manipulator Direct Kinematics

The manipulator direct kinematics are used in order to find the position of the end-effector $\mathbf{r}_{EE}$ and its rotation matrix $R_{\mathcal{I}}^{EE}$ with respect to the base spacecraft, and the position of the various joints of the chain and of the links centers of mass.
The *Denavit-Hartenberg convention* is a systematic and general method that defines the relative position and orientation of two successive links, in order to compose the transformation matrices that are then used to transform the end-effector position and orientation to the satellite body frame.
Firstly, it is possible to define individual joint reference frames. They are found starting from the previous joint and link. In particular, the reference frame centered in the $i-th$ joint is denominated Σ_{i-1} and it is aligned with the link $i-1th$.
Thus, the definition of the reference frames is as follows:

1. The axis $\mathbf{z}_i$ is along the axis of joint $i+1$.
2. The origin O_i of the reference frame is at the intersection of z_i with the common normal to z_i and z_{i-1}. The common normal is the line containing the minimum distance segment between the two lines.
3. Axis x_i is along the common normal to axes z_i and z_{i-1} with positive direction from joint i to joint $i+1$.
4. Axis y_i is chosen as to form a right-handed frame.

According to this convention, some particular cases fall outside of its description:

- Σ_0 has its axis x_0 and its origin O_0 undefined. Typically, they are chosen arbitrarily in the simplest way possible (e.g. aligning x_0 with the body frame *x-axis* and choosing the origin of the reference frame at the midpoint of the physical axis of the first joint when the latter is revolute).

- Σ_n has its axis z_n undefined. Since this joint is revolute most of the time, the axis is taken parallel to z_{n-1}.
- When two consecutive axes intersect, the direction of x_i is arbitrary.

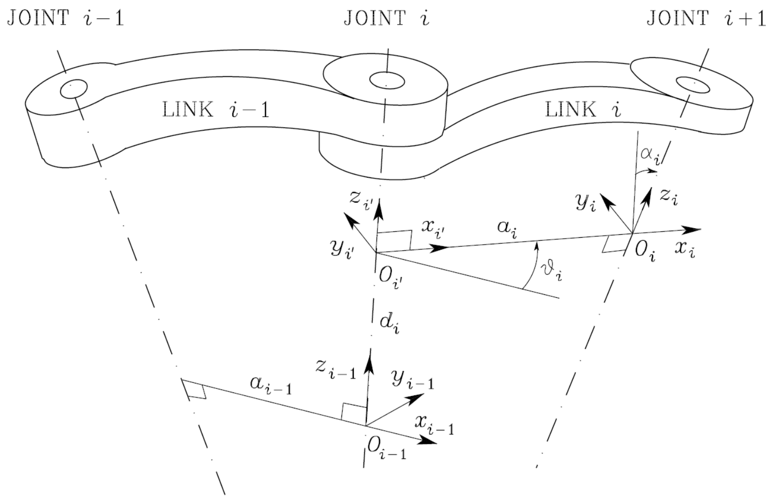


Figure 2.3: Denavit-Hartenberg Convention

The *Denavit-Hartenberg parameters* are obtained from the definition of two consecutive reference frames. They are four geometric parameters that are defined as follows:

- a_i is the distance between two origins of two consecutive reference frames.
- d_i is the coordinate of the origin of the $i-th$ reference frame as measured along z_{i-1}.
- α_i is the angle between axes z_{i-1} and z_i taken positive when the rotation is counter-clockwise.
- θ_i is the angle between axes x_{i-1} and x_i taken positive when the rotation is counter-clockwise.

Having defined these parameters, the Homogeneous Transformation Matrices between two reference frames can be identified by the sequence of rotations and translations performed to transform the two consecutive reference frames:

$$A_{i-1}^{i} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.23}$$

By identifying the end-effector reference frame with the $n - th$ reference frame, it is possible to utilize the rule of consecutive transformations to directly obtain its relative position and orientation with the first joint of the manipulator:

$$T_0^{EE} = \prod_{i=1}^{n} A_{i-1}^{i} \tag{2.24}$$

To ease the definition of the dynamics of the system (in section 2.4.4), it is necessary to highlight the dependencies of the motion of the center of mass of the various links with respect to the joint variables.
A simplification is hereby utilized: the center of mass of each link is coincident with the midpoint of the line between the two adjacent joints.
Thus, the position of the center of mass of the $i - th$ link with respect to the $i - th$ joint reference frame is:

$$\mathbf{x}_{G,i} = \begin{bmatrix} \frac{1}{2} a_i \cos\theta_i \\ \frac{1}{2} a_i \sin\theta_i \\ \frac{d_i}{2} \end{bmatrix} \tag{2.25}$$

It is also possible to define the reference frame $\Sigma_{G,i}$ which is based on the homogeneous transformation between two consecutive joint reference frames. In this way, the reference frame in the center of mass of each link is aligned with the one in the next joint and it depends on the previous joint variable.

$$A_{i-1}^{G,i} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & \frac{1}{2}a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & \frac{1}{2}a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & \frac{d_i}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.26}$$

This representation, in general, does not coincide with the exact center of mass of the link, especially whenever both a_i and d_i are non-zero. In this case, however, it is assumed (as is common in literature) that either variable is zero for each link.

Finally, the *complete direct kinematics equations* can be found: they are based on transforming each quantity to $\Sigma_{\mathcal{I}}$.
It is possible to define a constant transformation matrix from the satellite base body frame to the first joint of the manipulator, similarly to the grasping point in eq. (2.20):

$$T_B^0 = \begin{bmatrix} R_B^0 & \mathbf{x}_0^B \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{2.27}$$

This allows to write the complete transformation from $\Sigma_{B,C}$ to Σ_{EE}:

$$T_B^{EE} = T_B^0 \prod_{i=1}^{n} A_{i-1}^i \tag{2.28}$$

Finally, it is possible to transform every quantity to the Local Vertical Local Horizontal Frame, in order to define the framework connecting the end-effector position and orientation (and the relative velocities) with the base frame, highlighting the dependencies on the quantities. This is valid for the end-effector (2.29), for the joint positions (2.30) and for the links Centers of Mass positions (2.31).

$$T_{\mathcal{I}}^{EE} = T_{\mathcal{I}}^B(\boldsymbol{\varphi}) T_B^0 \prod_{i=1}^{n} A_{i-1}^i(q_i(t)) \tag{2.29}$$

$$T_{\mathcal{I}}^{k} = T_{\mathcal{I}}^B(\boldsymbol{\varphi}) T_B^0 \prod_{i=1}^{k} A_{i-1}^i(q_i(t)) \tag{2.30}$$

$$T_{\mathcal{I}}^{G,k} = T_{\mathcal{I}}^B(\boldsymbol{\varphi}) T_B^0 \prod_{i=1}^{k-1} A_{i-1}^i(q_i(t)) A_{k-1}^{G,k}(q_k(t)) \tag{2.31}$$

2.4.3. Manipulator Differential Kinematics

The differential kinematics individuate the relations existing at the velocity or acceleration level, named first order and second order differential kinematics respectively, between the end-effector and the joint variables, both for the linear and angular velocities and accelerations. The same framework can be followed in order to find the relationships between the CoM of the links and the joint variables.
This section computes the velocities of the links CoMs and of the end-effector by separating the effects of the motion of the chaser base and of the manipulator motion, while referring the end-effector velocity to the inertial reference frame $\Sigma_{\mathcal{I}}$.
In particular, it is possible to regroup the linear and angular velocities of the end-effector and of the base satellites in the 6 × 1 *generalized velocity* vectors $\mathcal{V}_b = \begin{bmatrix} \mathbf{v}_b & \boldsymbol{\omega}_b \end{bmatrix}^T$ and $\mathcal{V}_{EE} = \begin{bmatrix} \mathbf{v}_{EE} & \boldsymbol{\omega}_{EE} \end{bmatrix}^T$, obtaining the following relation.

$$\mathcal{V}_{EE} = J_B^{EE} \mathcal{V}_b + J^{EE} \dot{\boldsymbol{\theta}} \tag{2.32}$$

The first order differential kinematics for the end-effector of the manipulator is a linear relationship, and it can be directly expressed in $\Sigma_{\mathcal{I}}$, and it is indicated by (M):

$$\begin{bmatrix} \mathbf{v}^{(}_{EE}M) \\ \boldsymbol{\omega}^{(}_{EE}M) \end{bmatrix} = \begin{bmatrix} J_P(\mathbf{q}) \\ J_\omega(\mathbf{q}) \end{bmatrix} \dot{\mathbf{q}} \tag{2.33}$$

The *Geometric Jacobian* is defined exploiting the Denavit-Hartenberg description of the system, and can be computed separately for the linear and the angular portions. In particular, the linear velocity of the end-effector can be written as:

$$\dot{\mathbf{x}}^{(}_{EE}M) = \sum_{i=1}^{n} \frac{\partial \mathbf{x}_{EE}}{\partial q_i} \dot{q}_i = \sum_{i=1}^{n} J_{P,i} \dot{q}_i \tag{2.34}$$

The angular velocity can be similarly written as:

$$\dot{\boldsymbol{\omega}}^{(}_{EE}M) = \sum_{i=1}^{n} \mathcal{J}_{\omega,i} \dot{q}_i \tag{2.35}$$

Each of the terms of the summation multiplying a joint coordinate is a column in the 3-by-n linear or angular velocity Jacobian matrix, and their computation slightly differs depending on whether the joints are prismatic or revolute. Only the latter is hereby presented as most space manipulators use revolute joints rather than prismatic [33].
Thus, linear velocity Jacobian is computed as:

$$\mathcal{J}_{P,i} = \mathbf{z}_{i-1} \times (\mathbf{x}_{EE} - \mathbf{x}_{i-1}) \tag{2.36}$$

While the angular velocity Jacobian is computed as:

$$\mathcal{J}_{\omega,i} = \mathbf{z}_{i-1} \tag{2.37}$$

Each of the terms that compose these relations is depending on the joint variables, in particular:

- $\mathbf{z}_{i-1}$ is the z-axis of the reference frame Σ_{i-1}, attached to joint i, expressed in the inertial reference frame. Thus, it is simply the third column of the rotation matrix from the inertial reference frame to Σ_{i-1}:

$$\mathbf{z}_{i-1} = R_{\mathcal{I}}^{B} R_{B}^{0} \left(\prod_{j=1}^{i-1} R_{j-1}^{j} \right) \mathbf{z}_0 \tag{2.38}$$

Where $\mathbf{z}_0 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$ extracts the third column of the rotation matrix.

- $\mathbf{x}_{EE}$ is the position of the end-effector in the inertial reference frame. Having defined the homogeneous transformation between the latter and the end-effector frame, it is simple to define:

$$\mathbf{x}_{EE} = E_{3\times4} T_{\mathcal{I}}^{B} T_{B}^{0} T_{0}^{EE} \mathbf{p}_0 \tag{2.39}$$

  Where $\mathbf{p}_0 = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T$ selects the fourth column of the transformation matrix, and the extraction matrix E selects the first three rows of a $4 \times n$ matrix or vector (2.40).

$$E_{3\times4} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.40}$$

- $\mathbf{x}_{i-1}$ is the position vector in the inertial reference frame of the origin of Σ_{i-1}. Its calculation is the same as the position of the end-effector, selecting the fourth column of the overall homogeneous transformation matrix $T_{\mathcal{I}}^{i-1}$

Following this scheme employing Denavit-Hartenberg parameters, the computation of the Geometric Jacobian is straightforward.
It is worth mentioning that the Geometric Jacobian computes the end-effector velocity in terms of the end-effector frame, with a geometric technique employed in order to find its components. The *Analytical Jacobian* can be computed by directly differentiating the relations of the linear and angular velocities. This outputs the same Jacobian for the linear part, whereas for the angular velocities it is necessary to use a representation of the rotation (e.g. ZYZ Euler Angles) in order to express the relative orientation with the inertial frame, and this differs from the angular velocity of the end-effector itself. It is possible, however, to express the relation between the latter and the derivative of the variables for the representation of the rotation, giving the holding relation between the geometric and analytical Jacobian:

$$J_G = \begin{bmatrix} I_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & T(\boldsymbol{\phi}_{EE}) \end{bmatrix} J_A \tag{2.41}$$

The Jacobians relating each of the individual links CoM velocity and the joint velocities can be computed in a similar way to the end-effector's, dividing the linear and the angular velocity components. They are:

$$\dot{\mathbf{x}}_{G,i} = \mathcal{J}_P^{G,i} \dot{\mathbf{q}} \tag{2.42}$$

$$\dot{\boldsymbol{\omega}}_{G,i} = \mathcal{J}_{\omega}^{G,i}\dot{\mathbf{q}} \tag{2.43}$$

Once again, the Jacobians can be defined column-wise, considering that each link position and velocity depends on the motion of the previous joints:

$$J^{G,i} = \begin{bmatrix} \mathcal{J}_{P,1}^{G,i} & \dots & \mathcal{J}_{P,i}^{G,i} & 0 & \dots & 0 \\ \mathcal{J}_{O,1}^{G,i} & \dots & \mathcal{J}_{O,i}^{G,i} & 0 & \dots & 0 \end{bmatrix} \tag{2.44}$$

Finally, the non-zero columns composing the Jacobian matrices are computed differently for prismatic and revolute joints. Following the same reasoning as the previous section, the latter are defined:

$$\mathcal{J}_{P,j}^{G,i} = \mathbf{z}_{j-1} \times (\mathbf{x}_{G,i} - \mathbf{x}_{j-1}) \tag{2.45}$$

$$\mathcal{J}_{Of,j}^{G,i} = \mathbf{z}_{j-1} \tag{2.46}$$

Where the terms of the equations are:

- $\mathbf{z}_{j-1}$ is the $z-axis$ of the reference frame Σ_{j-1} obtained by the third column of the rotation matrix $R_{\mathcal{I}}^{j-1}$.
- $\mathbf{x}_{G,i}$ is the position vector of the CoM of the $i-th$ link, which is also the origin of the reference frame $\Sigma_{G,i}$. It is obtained by the fourth column of the matrix $T_{\mathcal{I}}^{G,i}$.
- $\mathbf{x}_{j-1}$ is the position vector of the origin of the reference frame Σ_{j-1}, thus the fourth column of the matrix $T_{\mathcal{I}}^{j-1}$.

These expressions relate the velocities of each link to each joint variable in $\Sigma_{\mathcal{I}}$. They will result particularly useful for defining the dynamics of the coupled base-manipulator system.

Finally, the differential kinematics can be found for the whole space manipulator system. The Jacobian J_b relating the end-effector generalized velocity with the base satellite's [3] can be defined knowing that the angular velocity of the end-effector is the same as the base's, whereas the linear velocity of the end-effector in $\Sigma_{\mathcal{I}}$ is the same as the base with an additive drift term due to the angular motion of the base with respect to the inertial frame:

$$J_b^{EE} = \begin{bmatrix} I_{3\times3} & x_{EE}^{\times} \\ 0_{3\times3} & I_{3\times3} \end{bmatrix} \tag{2.47}$$

Ultimately, the same transformation can be applied in order to find the differential kinematics for each center of mass of the links, utilizing the respective individual Jacobian matrix $J^{G,i} = \begin{bmatrix} J_P^{G,i} & J_O^{G,i} \end{bmatrix}^T$ rather than the end-effector's.

2.4.4. Dynamics

The dynamics were utilized in this work in order to validate the kinematic model that will be utilized for the path planning algorithm.
The definition of the dynamics cannot follow two independent paths for the base satellite and for the manipulator due to the strong mechanical coupling between the two systems. In order to keep a physical meaning to the variables introduced, the *Euler-Lagrange Formalism* is hereby used to obtain the dynamic equation of the system:

$$\frac{d}{dt}\left(\frac{\partial \mathcal{T}}{\partial \dot{\mathbf{q}}}\right) - \frac{\partial \mathcal{T}}{\partial \mathbf{q}} + \frac{\partial \mathcal{V}}{\partial \mathbf{q}} = \mathbf{Q_q} \tag{2.48}$$

A more specific formulation for the system involves the simplification of having null potential field. This basically means considering no gravitational forces acting on the system, and this assumption holds generally for space manipulator systems in literature. For the same reason, orbital perturbations can be considered null for low duration manoeuvres.
This means that the main contribution to the dynamics is due to the kinetic energy of the system itself. The *external forces* are the control forces and torques on the base satellite and the torques of the joint actuators that control the motion of the arm. These are all considered to be applied exactly to the center of mass of the base and to the exact point in which the joints are located in order to simplify the equations without loss of generality.
The generalized coordinates of the system are thus the base position and orientation, and the joint trajectories:

$$\mathbf{q} = \begin{bmatrix} \mathbf{x}_b \\ \boldsymbol{\varphi}_b \\ \boldsymbol{\theta} \end{bmatrix} \in \mathbb{R}^{(6+n)\times 1} \tag{2.49}$$

Kinetic Energy

The kinetic energy of the system of rigid bodies can be computed summing the contributions of the base satellite and of the manipulator:

$$\mathcal{T} = \frac{1}{2}\left[m_b \dot{\mathbf{x}}_b^T \dot{\mathbf{x}}_b + \boldsymbol{\omega}_b^T I_b \boldsymbol{\omega}_b + \sum_{i=1}^{n} \left(m_i \mathbf{v}_i^T \mathbf{v}_i + \boldsymbol{\omega}_i^T I_i \boldsymbol{\omega}_i \right) \right] \tag{2.50}$$

Each quantity within the equations has to be computed with respect to the same reference frame. In order to follow classical mechanics without neglecting any term, the quantities

will all be referred to the *Inertial reference frame.*
Hence, the *inertia tensors* in eq. (2.50) are rotated to $\Sigma_{\mathcal{I}}$ from their respective body axes, which are the satellite's and the ones of the $i-th$ link:

$$I_b = R^B_{\mathcal{I}} I^B_b R^{B^T}_{\mathcal{I}} \tag{2.51a}$$

$$I_i = R^B_{\mathcal{I}} R^{G,i}_B I^{G,i}_i R^{G,i^T}_B R^{B^T}_{\mathcal{I}} \tag{2.51b}$$

Where each quantity of these equations has been previously defined. Finally, recalling eq. (2.32) and its generalization to the CoMs of the various links, it is possible to find the separate formulations of linear and angular velocities in the inertial reference frame to obtain [23] the direct dependencies of the velocities of the centers of mass of the links with the generalized coordinates of the system:

$$\mathbf{v}_{G,i} = \dot{\mathbf{x}}_b + \boldsymbol{\omega}_b^{\times}\mathbf{x}_{G,i} + J_P^{G,i}\dot{\boldsymbol{\theta}} \tag{2.52a}$$

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_b + J_{\omega}^{G,i}\dot{\boldsymbol{\theta}} \tag{2.52b}$$

Having defined the various dependencies, the kinetic energy of the system is finally available in matrix form, expressing the direct dependency on the generalized coordinates of the system in order to use the Euler-Lagrange equations to find the equations of motion of the system. The derivative of the orientation of the base has been substituted with its angular velocity in generalized coordinates, and the conversion to the orientation is available through the differential relation in (2.22).
The inertia matrix is partitioned in block submatrices, with subscripts indicating the part of the mechanical system they refer to: translational (t), rotational (r), manipulator (m), and their combinations that will result in the strong *Coupling Effects* between the base and the manipulator.

$$\mathcal{T} = \frac{1}{2}\dot{\mathbf{q}}^T M(\mathbf{q})\dot{\mathbf{q}} = \frac{1}{2}\begin{bmatrix}\dot{\mathbf{x}}_b^T & \boldsymbol{\omega}_b{}^T & \dot{\boldsymbol{\theta}}^T\end{bmatrix}\begin{bmatrix} M_t & M_{tr} & M_{tm} \\ M_{tr}^T & M_r & M_{rm} \\ M_{tm}^T & M_{rm}^T & M_m \end{bmatrix}\begin{bmatrix}\dot{\mathbf{x}}_b \\ \boldsymbol{\omega}_b \\ \dot{\boldsymbol{\theta}}\end{bmatrix} \tag{2.53}$$

Finally, the analytical expressions of the matrices are available substituting eq. (2.51) and eq. (2.52) in eq. (2.50), and are hereby presented in compact form similarly to [11], specifying to which reference frame the quantities are referred to.

$$M_t = \left(m_B + \sum_{i=1}^{n} m_i\right) I_{3\times3} \in \mathbb{R}^{3\times3} \tag{2.54a}$$

$$M_{tr} = -\sum_{i=1}^{n} m_i \mathbf{x}_{G,i}^{\times} \in \mathbb{R}^{3\times3} \tag{2.54b}$$

$$M_r = I_b + \sum_{i=1}^{n} \left(I_i - m_i \mathbf{x}_{G,i}^{\times}\mathbf{x}_{G,i}^{\times}\right) \in \mathbb{R}^{3\times3} \tag{2.54c}$$

$$M_{tm} = \sum_{i=1}^{n} m_i J_P^{G,i} \in \mathbb{R}^{3\times n} \tag{2.54d}$$

$$M_{rm} = \sum_{i=1}^{n} m_i \mathbf{x}_{G,i}^{\times} J_P^{G,i} + I_i J_\omega^{G,i} \in \mathbb{R}^{3\times n} \tag{2.54e}$$

$$M_m = \sum_{i=1}^{n} (m_i J_P^{G,i^T} J_P^{G,i} + J_\omega^{G,i^T} I_i J_\omega^{G,i}) \in \mathbb{R}^{n\times n} \tag{2.54f}$$

Where each quantity has been defined in previous paragraphs. In this simplification, links masses and inertia take into account the presence of the joint motors, whereas in more accurate descriptions, they should be considered in their own relative reference frame centered at the joints themselves:

$$\begin{aligned} M_m = \sum_{i=1}^{n} (m_i J_P^{G,i^T} J_P^{G,i} + J_\omega^{G,i^T} I_i J_\omega^{G,i} + \\ + m_{m,i} J_P^{i-1^T} J_P^{i-1} + J_\omega^{i-1^T} I_{m,i} J_\omega^{i-1}) \end{aligned} \tag{2.55}$$

The subscripts m, i means the quantities are relative to the motor positioned in the $i-th$ joint, thus relative to the reference frame Σ_{i-1}.

Equations of Motion

Ultimately, it is possible to apply the derivatives of the Euler-Lagrange equations in eq. (2.48) to the matrix representation of the kinetic energy, and by using the *base generalized velocity* introduced in section 2.4.3 the equations of motion are obtained:

$$\begin{bmatrix} M_b & M_{bm} \\ M_{bm}^T & M_m \end{bmatrix} \begin{bmatrix} \dot{\mathcal{V}}_b \\ \ddot{\boldsymbol{\theta}} \end{bmatrix} + \begin{bmatrix} \mathbf{c}_b \\ \mathbf{c}_m \end{bmatrix} = \begin{bmatrix} \mathbf{F}_b \\ \boldsymbol{\tau} \end{bmatrix} \tag{2.56}$$

Where the terms $\mathbf{c}_B(\mathbf{q}, \dot{\mathbf{q}})$ and $\mathbf{c}_m(\mathbf{q}, \dot{\mathbf{q}})$ are the nonlinear components due to centrifugal and Coriolis forces that stem from the relative motions between the links and the chaser

satellite base.
These terms are defined from the Lagrange equations as:

$$\mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) = \dot{M}\dot{\mathbf{q}} - \frac{1}{2}\dot{\mathbf{q}}^T \frac{\partial M}{\partial q_k}\dot{\mathbf{q}} \tag{2.57}$$

The time derivatives of the various matrices can be computed applying the chain rule, thus the matrices $\frac{\partial M}{\partial q_k}$ are computed in the same framework. It is possible to highlight that none of the terms of the mass matrix depends from $\mathbf{x}_B$, while the derivatives with respect to $\boldsymbol{\varphi}_B$ and $\boldsymbol{\theta}$ are non-zero. The details on the computation of the derivatives are given in section 2.4.5.
The computation of the nonlinear terms of the equations of motion can also be obtained following [39], applying the derivatives with respect to each coordinate to each term of the mass matrix.

$$\mathbf{c}_i = \sum_j \sum_k \frac{\partial m_{i,j}}{\partial q_k} - \frac{1}{2}\frac{\partial m_{k,j}}{\partial q_i} \tag{2.58}$$

This is particularly useful in the bidimensional case.
The equations of motion in the inertial frame are hence fully defined, however it is possible to introduce further assumptions of no external torques or forces applied to the system ($\mathbf{F}_b = 0$), in order to simplify the equations of motion. This will hold true in the case of a completely free-floating space manipulator before the capture [23, 27].

2.4.5. Generalized Equations of Motion

The dynamical system expressed in (2.56), with $\mathbf{F}_b = \mathbf{0}$, is an underactuated system, as only the joint coordinates are related to an external acting torque, while the remainder of the Space Manipulator System moves as a consequence of the joints movements.
Then, considering no forces or torques acting on the overall system center of mass, it is possible to compute the linear and angular momenta of the system:

$$\mathcal{P} = (m_b + \sum_{i=1}^{n} m_i)\mathbf{v}_b + \sum_{i=1}^{n} m_i \mathbf{v}_i = \mathcal{P}_0 \tag{2.59a}$$

$$\mathcal{L} = I_b\boldsymbol{\omega}_b + \sum_{i=1}^{n} \left(I_i\boldsymbol{\omega}_i + m_i \mathbf{x}_{G,i}^{\times}\mathbf{x}_{G,i}^{\times}\right) = \mathcal{L}_0 \tag{2.59b}$$

Where $\mathcal{P}_0$ and $\mathcal{L}_0$ are the initial linear and angular momenta of the system. They can be expressed in a much more compact form:

$$\begin{bmatrix} \mathcal{P} \\ \mathcal{L} \end{bmatrix} = M_b \mathcal{V}_b + M_{bm} \dot{\boldsymbol{\theta}} = \mathcal{M}_0 \tag{2.60}$$

Since the system is performing a floating manoeuvre, the momenta are constant throughout the motion:

$$\frac{d}{dt} \begin{bmatrix} \mathcal{P} \\ \mathcal{L} \end{bmatrix} = M_b \dot{\mathcal{V}}_b + M_{bm} \ddot{\boldsymbol{\theta}} + \dot{M}_b \mathcal{V}_b + \dot{M}_{bm} \dot{\boldsymbol{\theta}} = \mathbf{0}_{6\times 1} \tag{2.61}$$

It is possible to assume an initial value for the momentum of the system, in particular it is reasonable to assume that the overall system momentum is initially zero, while in [26] the case of non-zero initial momentum is treated.
The physical interpretation of the overall momentum being zero is that the second term of the equations is the momentum that is imposed by the manipulator on the base (*coupling momentum*) and the first term is the *reaction momentum* of the base with respect to the manipulator motion [28].
With the usage of (2.60),(2.61) it is possible to obtain the generalized base velocity and acceleration's direct dependency from the joint coordinates, or it can be obtained from the first six equations of motion:

$$\ddot{\mathcal{X}}_b = -M_b^{-1}(M_{bm}\ddot{\theta} + \mathbf{c}_b) \tag{2.62}$$

Substituting this expression in the last n equations of motion gives the *equations of motion in generalized form*, including the dynamics of the base within the motion of the joints of the manipulator.

$$(-M_{bm}^T M_b^{-1} M_{bm} + M_m)\ddot{\boldsymbol{\theta}} - M_{bm}^T M_b^{-1} \mathbf{c}_b + \mathbf{c}_m = \boldsymbol{\tau} \tag{2.63}$$

It is possible to define the *generalized inertia matrix* and the generalized vector of non-linear terms, as in [49] and [1]:

$$\hat{M} = M_m - M_{bm}^T M_b^{-1} M_{bm} \in \mathbb{R}^{n\times n} \tag{2.64a}$$

$$\hat{\mathbf{c}} = \mathbf{c}_m - M_{bm}^T M_b^{-1} \mathbf{c}_b \in \mathbb{R}^{n\times 1} \tag{2.64b}$$

An important feature of these generalized matrices is that when substituting each quantity, these two generalized terms depend only on $\boldsymbol{\theta}$ [51], eliminating the dependence on $\boldsymbol{\varphi_B}$.

Typically, the behaviour of the generalized form of the equations of motion is obtained computing $\boldsymbol{\varphi_B}$ to then compute the mass matrix terms as in (2.54).
Finally, the generalized form of the equations of motion is:

$$\hat{M}\ddot{\boldsymbol{\theta}} + \hat{\mathbf{c}} = \boldsymbol{\tau} \tag{2.65}$$

An important aspect is that the same torques of the generalized form of the equations of motion are applied to the joints in the overall equations of motion, thus it is possible to apply the same torque control for the two systems, without transforming coordinates.

Mass Matrix Derivatives

The derivatives of the mass matrix components are here obtained individually to then utilize (2.57) in order to find the overall vector of nonlinear terms.
It is possible to utilize the equations of conservation of linear and angular momentum to obtain the terms of $\mathbf{c}_b$ as functions of $\boldsymbol{\theta}$ only, thus discarding the derivatives with respect to the base angular coordinates that would introduce complications and would be based on the representation of choice for the rotations.
Knowing that M_b^{-1} is symmetric [45], it is possible to rewrite the generalized equations of motion:

$$\hat{M}\ddot{\boldsymbol{\theta}} + \dot{\hat{M}}\dot{\boldsymbol{\theta}} - \frac{1}{2}\dot{\boldsymbol{\theta}}^T\frac{\partial \hat{M}}{\partial \boldsymbol{\theta}}\dot{\boldsymbol{\theta}} = \boldsymbol{\tau} \tag{2.66}$$

Time Derivatives The time derivative of the generalized mass matrix can be easily computed by knowing that for a square, invertible matrix such as M_b, the following property holds when the derivative with respect to any scalar is computed:

$$\frac{dA^{-1}}{dx} = -A^{-1}\frac{dA}{dx}A^{-1} \tag{2.67}$$

Thus, the overall time derivative of the generalized mass matrix is:

$$\dot{\hat{M}} = \dot{M}_m - (\dot{M}_{bm}^T M_b^{-1} M_{bm} + M_{bm}^T M_b^{-1} \dot{M}_{bm} - M_{bm}^T M_b^{-1} \dot{M}_b M_b^{-1} M_{bm}) \tag{2.68}$$

Hence, the derivatives of three different portions of the mass matrix must be computed. This translates in the derivation of the quantities that depend on time within the submatrices. The derivatives of the dynamic quantities are mostly based on knowing the time derivative of a rotation matrix [39]:

$$\dot{R}_{\mathcal{I}}^i = \boldsymbol{\omega}_i^{\times} R_{\mathcal{I}}^i \tag{2.69}$$

The individual angular velocity of each joint coordinate system is known in the inertial reference frame and it is given by:

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_B + J_\omega^i \dot{\boldsymbol{\theta}} \tag{2.70}$$

Finally, it is possible to compute the time derivatives for the components of the mass matrices that depend on time, in particular the quantities for which they need to be computed are:

- *Link i's CoM position with respect to the base spacecraft $x_{G,i}$*: This term is readily available as it is the velocity of the link's center of mass to which the linear velocity of the base is subtracted, available in (2.52):

$$\frac{d\mathbf{x}_{G,i}}{dt} = \boldsymbol{\omega}_b^\times \mathbf{x}_{G,i} + J_P^{G,i} \dot{\boldsymbol{\theta}} \tag{2.71}$$

- *Link i's Jacobian matrix for the positions $J_P^{G,i}$*: the time derivative of this term depends on the rotation of each joint previous to the link and on the position of the link's CoM velocity and on the previous link's velocity. The derivative relative to the i-th link is:

$$\frac{dJ_P^{G,i}}{dt} = \begin{bmatrix} \dot{\mathcal{J}}_{P,1}^{G,i} & \dots & \dot{\mathcal{J}}_{P,i}^{G,i} & \mathbf{0} & \dots & \mathbf{0} \end{bmatrix} \tag{2.72}$$

The generic column of the derivative of the Jacobian is:

$$\frac{d\mathcal{J}_{P,k}^{G,i}}{dt} = \left(\dot{R}_{\mathcal{I}}^{k-1} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right)^\times (\mathbf{x}_{G,i} - \mathbf{x}_{j-1}) + \mathbf{z}_{k-1}^\times \left(\frac{d\mathbf{x}_{G,i}}{dt} - \frac{d\mathbf{x}_{k-1}}{dt} \right) \tag{2.73}$$

- *Link i's Jacobian matrix for the rotations $J_\omega^{G,i}$*: the derivative of the Jacobian for the rotation depends only on the rotation of the previous joints axes.

$$\frac{dJ_\omega^{G,i}}{dt} = \begin{bmatrix} \dot{\mathcal{J}}_{\omega,1}^{G,i} & \dots & \dot{\mathcal{J}}_{\omega,i}^{G,i} & \mathbf{0} & \dots & \mathbf{0} \end{bmatrix} \tag{2.74}$$

The generic column of the derivative of the Jacobian is:

$$\frac{d\mathcal{J}_{\omega,k}^{G,i}}{dt} = \dot{R}_{\mathcal{I}}^{k-1} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{2.75}$$

- *Inertia Matrices of the Chaser Base, and of Link i in $\Sigma_{\mathcal{I}}$ I_b, I_i:* the inertia matrices are expressed in the inertial frame within the equations of motion. Their dependency on time is due to the derivatives of the rotation matrices:

$$\frac{dI_i}{dt} = \dot{R}_{\mathcal{I}}^{i\,T} I_i R_{\mathcal{I}}^{i} + R_{\mathcal{I}}^{i\,T} I_i \dot{R}_{\mathcal{I}}^{i} \tag{2.76a}$$

$$\frac{dI_b}{dt} = \dot{R}_{\mathcal{I}}^{BT} I_b R_{\mathcal{I}}^{B} + R_{\mathcal{I}}^{B\,T} I_b \dot{R}_{\mathcal{I}}^{B} \tag{2.76b}$$

Finally, the expression of the derivatives for each submatrix of the inertia matrix is available:

$$\dot{M}_{tt} = 0^{3\times3} \tag{2.77a}$$

$$\dot{M}_{tr} = -\sum_{i=1}^{n} m_i \frac{d\mathbf{x}_{G,i}}{dt} \tag{2.77b}$$

$$\dot{M}_{rr} = \dot{I}_b + \sum_{i=1}^{n} (\dot{I}_i - m_i(\dot{\mathbf{x}}_{G,i}^{\times}\mathbf{x}_{G,i}^{\times} + \mathbf{x}_{G,i}^{\times}\dot{\mathbf{x}}_{G,i}^{\times}) \tag{2.77c}$$

$$\dot{M}_{tm} = \sum_{i=1}^{n} m_i \dot{J}_P^{G,i} \tag{2.77d}$$

$$\dot{M}_{rm} = \sum_{i=1}^{n} \dot{I}_i J_{\omega}^{G,i} + I_i \dot{J}_{\omega}^{G,i} + m_i(\dot{\mathbf{x}}_{G,i} J_P^{G,i} + \mathbf{x}_{G,i} \dot{J}_P^{G,i}) \tag{2.77e}$$

$$\dot{M}_{mm} = \sum_{i=1}^{n} \dot{J}_{\omega}^{G,iT} I_i J_{\omega}^{G,i} + J_{\omega}^{G,i^T} I_i \dot{J}_{\omega}^{G,i} + J_{\omega}^{G,iT} \dot{I}_i J_{\omega}^{G,i} + m_i(\dot{J}_P^{G,iT} J_P^{G,i} + J_P^{G,i^T} \dot{J}_P^{G,i}) \tag{2.77f}$$

Joint Coordinates Derivatives The explicit joint coordinates derivative of the inertia matrix is obtained similarly to the time derivative in (2.68):

$$\begin{aligned}\frac{\partial \hat{M}}{\partial \theta_k} = \frac{\partial M_m}{\partial \theta_k} - \frac{\partial M_{bm}}{\partial \theta_k}^T M_b^{-1} M_{bm} - M_{bm}^T M_b^{-1} \frac{\partial M_{bm}}{\partial \theta_k} + \\ + M_{bm}^T M_b^{-1} \frac{\partial M_b}{\partial \theta_k} M_b^{-1} M_{bm}\end{aligned} \tag{2.78}$$

The inertia matrix components that depend directly on the joint coordinates are the same that depend on time, except for the derivative of the chaser base inertia matrix which depends only on the rotation of the chaser base with respect to the inertial frame.
In this case, however, the values of the derivatives of the quantities relative to the center of mass of link i with respect to θ_k depends on the relative values of k and i. The required derivatives to be computed are:

- *Link i's CoM position with respect to the base spacecraft* $x_{G,i}$

$$\frac{\partial \mathbf{x}_{G,i}}{\partial \theta_k} = E_{3\times 4}\, {}_4R_B^{\mathcal{I}} \prod_{l=1}^{k-1} {}_4R_{l-1}^{l} \frac{dA_{k-1}^{k}}{d\theta_k} \prod_{m=k+1}^{i-1} A_{m-1}^{m} A_{i-1}^{G,i}\, {}_4\mathbf{z}_0 \quad if\ k < i \tag{2.79a}$$

$$\frac{\partial \mathbf{x}_{G,i}}{\partial \theta_k} = E_{3\times 4}\, {}_4R_B^{\mathcal{I}} \prod_{l=1}^{k-1} {}_4R_{l-1}^{l} \frac{dA_{k-1}^{G,k}}{d\theta_k}\, {}_4\mathbf{z}_0 \quad if\ k = i \tag{2.79b}$$

$$\frac{\partial \mathbf{x}_{G,i}}{\partial \theta_k} = \mathbf{0}^{3\times 1} \quad if\ k > i \tag{2.79c}$$

Where ${}_4\mathbf{z}_0 = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T$ and $E_{3\times 4}$ is defined in (2.40), while the newly defined quantities are the matrices ${}_4R_k^{k-1}$, which are the rotation matrices transforming the coordinates from the next joint to the previous joint, modified in order to be 4-by-4 dimensional to be compliant with the homogeneous transformation matrices framework, which are defined as:

$${}_4\mathbf{R}_k^{k-1} = \begin{bmatrix} R_k^{k-1} & 0^{3\times 1} \\ 0^{1\times 3} & 1 \end{bmatrix} \tag{2.80}$$

The derivatives of the homogeneous transformation matrices are non-zero only for their specific joint angle:

$$\frac{dA_{i-1}^{i}}{d\theta_i} = \begin{bmatrix} -\sin\theta_i & -\cos\theta_i\cos\alpha_i & \cos\theta_i\sin\alpha_i & -a_i\sin\theta_i \\ \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.81}$$

- *Link i's Jacobian matrix for the rotations* $J_\omega^{G,i}$: The derivatives of the Jacobians are easy to find from their definitions in (2.45) and (2.46). In particular, the derivatives of the Jacobian relative to the $i-th$ link Center of Mass are, for what concerns the angular velocity:

$$\frac{dJ_\omega^{G,i}}{d\theta_k} = \begin{bmatrix} 0^{3\times 1} & \dots & 0^{3\times 1} & \frac{d\mathbf{z}_k}{d\theta_k} & \dots & \frac{d\mathbf{z}_{i-1}}{d\theta_k} & 0^{3\times 1} & \dots & 0^{3\times 1} \end{bmatrix} \tag{2.82}$$

In which the derivatives of the $j-th$ joint axis with respect to the $k-th$ variable is obtained from the consecutive rotations:

$$\frac{d\mathbf{z}_{k-1}}{d\theta_k} = \prod_{l=1}^{k-1} R_{l-1}^{l} \frac{dR_{k-1}^{k}}{d\theta_k} \prod_{m=k+1}^{j} R_{m-1}^{m}\mathbf{z}_0 \tag{2.83}$$

- *Link i's Jacobian matrix for the positions* $J_P^{G,i}$: For the linear velocity Jacobian, it is possible to repeat a similar discussion, knowing that the elements of the matrix are obtained from cross products between quantities that both depend on θ_k. The nonzero elements in the derivatives of the Jacobian relative to the $i-th$ link with respect to the $k-th$ joint angle are located in the columns from 1 to i, and are here described separately from the quantities in the columns relative to the joint previous to the $k-th$ and the ones comprised between the latter and the $i-th$.

$$\frac{d\mathcal{J}_{P,k}^{G,i}}{d\theta_j} = \mathbf{z}_{k-1}^B \times \left(\frac{d\mathbf{x}_{G,i}^B}{d\theta_j}\right) \quad if \quad k < j+1 \tag{2.84a}$$

$$\begin{aligned}\frac{d\mathcal{J}_{P,k}^{G,i}}{d\theta_j} = \mathbf{z}_{k-1}^B \times \left(\frac{d\mathbf{x}_{G,i}^B}{d\theta_j} - \frac{d\mathbf{x}_{k-1}^B}{d\theta_j}\right) + \\ +\frac{d\mathbf{z}_{k-1}^B}{d\theta_j} \times \left(\mathbf{x}_{G,i}^B - \mathbf{x}_{k-1}^B\right) \quad if \quad j+1 < k \leq i\end{aligned} \tag{2.84b}$$

Where the expressions for the derivatives of the links center of mass positions have already been obtained in (2.79), and the derivatives of the joints $k-1$ positions are computed following the same scheme presented for the various links center of mass.

2.4.6. Generalized Jacobian Matrix

The *Generalized Jacobian Matrix* for Space Manipulator Systems has been introduced in [42] and it follows the same reasoning behind the generalized equations of motion, to express the end-effector velocity and orientation.
It is indeed possible to solve eq. (2.59) for $\mathcal{V}_b$ to find the dependence of the base linear and angular velocities from the joint coordinates.

$$\mathcal{V}_b = -M_b^{-1} M_{bm} \dot{\boldsymbol{\theta}} \tag{2.85}$$

This way, it is possible to rewrite the complete end-effector generalized velocity eq. (2.32), which, considering null initial momentum, simplifies to:

$$\mathcal{V}_{EE} = \left(J^{EE} - M_b^{-1} M_{bm} J_b^{EE}\right) \dot{\boldsymbol{\theta}} \tag{2.86}$$

Thus, the overall matrix relating the joint coordinates with the end-effector velocity that includes the effects of the motion of the base produced by the joints motion is termed the

Generalized Jacobian Matrix (GJM), and it is expressed as:

$$\hat{J} = J^{EE} - M_b^{-1} M_{bm} J_b \tag{2.87}$$

The GJM is utilized in order to apply simple kinematic control to the robotic manipulator as would happen in Earth-based applications.

2.4.7. Overall Equations of Motion

Utilizing the generalized form of the equations of motion, it is possible to build the overall model of the Spacecraft-Manipulator System. The dynamics are computed only for the joint rotations, whereas the base coordinates' motion will depend on the joint coordinates according to eq. (2.85).
However, the kinematic equations for the rotation of the base must be integrated in order to obtain the rotation matrix from the inertial reference frame to the chaser base's axes at each time instant. The propagation of the chaser base's inertial position is not required to propagate the equations of motion.
The overall model for the Spacecraft-Manipulator System will thus be:

$$\dot{\boldsymbol{\varphi}} = f(\boldsymbol{\varphi}, \boldsymbol{\omega}_b) \tag{2.88a}$$

$$\hat{M}(\boldsymbol{\varphi}, \boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \hat{C}(\boldsymbol{\varphi}, \boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} = \hat{\boldsymbol{\tau}} \tag{2.88b}$$

Where the function $f(\boldsymbol{\varphi}, \boldsymbol{\omega}_b)$ depends on the choice of the representation of the rotation of the base with respect to the inertial reference frame, and φ is the orientation representation of choice.
The Coriolis Matrix $\hat{C}$ is defined as:

$$\hat{C} = \dot{\hat{M}} - \sum_{k=1}^{n} \frac{1}{2} \dot{\boldsymbol{\theta}}^T \frac{\partial \hat{M}}{\partial \theta_k} \dot{\boldsymbol{\theta}} \tag{2.89}$$

3 | Path Planning Algorithm

The typical path planning scenario this work refers to is the capture of a tumbling target of which the main parameters are fully known, in particular the dimensions and mass of both the target and the Spacecraft-Manipulator System are considered to be given and the state components of the target are assumed to have been accurately measured with no error. For real cases, this is not verified as the mass and dimensions of the satellites vary from the original values at launch and the state of the target is known with errors due to sensors sensitivity.

Furthermore, the far and close rendezvous approach of the chaser to the target are considered to be completed, and the relative linear velocity between the two systems centers of mass is considered to be null. The chaser is brought at rest before starting the capture manoeuvre, in order to more reasonably enforce the condition of zero initial momentum mentioned in section 2.4.5.

The obstacles within the Space Manipulator Systems are also fully known and characterized in their state and dimension.

The goal of the path planning is to provide trajectories for the joints, to make the end-effector reach a pre-identified grasping point on the target satellite, avoiding collisions with obstacles for the manipulator, which include the chaser base and any point differing from the grasping point for the target satellite, and further obstacles that can be present in the workspace, while being compliant with several constraints such as limits on joint angles, angular velocities and accelerations.

For space manipulator systems, it is as important to preserve telecommunications with Earth before and after the manoeuvre, thus it will be important to keep into account the minimization of the rotation of the base due to the motion of the joints. This chapter is organized as follows: in section 3.1 a general overview on Rapidly-Exploring Random Trees is given, including modified and improved versions that will be utilized in the construction of the proposed algorithm.

Then, in section 3.2, the proposed algorithm is covered in detail, from an overview on the scheme utilized to further explanation of each of its subparts.

3.1. Rapidly-Exploring Random Trees

Rapidly-Exploring Random Trees have been first introduced in [19]. They belong to the class of sampling-based algorithms for path planning, whose idea is to connect samples retrieved from the state space of the system to construct a tree that ultimately connects the initial state and the target state. RRTs have been widely utilized in robotics, and their usage has already been applied to Space Manipulator Systems. They have proven to be particularly suited for problems with differential constraints, nonlinear dynamics and for non-holonomic systems [13], while retaining the property of being probabilistically complete with an exponentially decaying rate of failure with the number of nodes that belong to the tree [19].

Definitions

- A graph G = (V,E) is a set of states and connections between them. It includes two subsets: the vertices V and the edges E.
- The vertices $V \subset X$ are a finite number of states belonging to the state space of the system.
- The edges $E \subset V \times V$ are a collection of vertices that are connected by a path.
- A directed path is a sequence of nodes such that each consecutive couple of nodes is contained in the edges set $(v_i, v_i + 1) \in E$ for $1 \leq i \leq n - 1$.
- Trees $T(V, E)$ are defined as graphs in which each of the vertices of the trees has one unique parent vertex except for the initial node (also denominated tree root) which has no incoming neighbor. The parent vertices to a certain node are defined as those vertices which are connected through a path to the node. For a node v the parent nodes are: $\{u \in V | (u, v) \in E\}$.

The general structure of all RRT algorithms is shown in algorithm 3.1, as introduced in [19].

Algorithm 3.1 Basic RRTs

```
1: Initialize T with $x_0$ first node
2: for $k = 1 : K$ do
3:     $x_{sample} \leftarrow$ Sample: Generate new random state
4:     $x_{nearest} \leftarrow$ Nearest$(T, x_{sample})$: Find nearest tree node
5:     $x_{new} \leftarrow$ Steer$(x_{nearest}, x_{sample})$: Generate new possible tree node
6:     if ObstacleFree$(x_{nearest}, x_{new})$ then
7:         AddVertex$(T, x_{new})$
8:         AddEdge$(E, x_{nearest}, x_{new})$
9:     end if
10: end for
11: Return T,E
```

The main components of a Rapidly-Exploring Random Tree are:

1. *Sampling*: the function `Sample`: $\rightarrow X_{free}$ Returns a state computed through an uniform distribution, belonging to the subset of the obstacle-free states within the specified boundaries.

2. *Nearest Neighbor*: takes as input the tree $T(V, E)$ and the sampled point $x \in X_{free}$ and returns a vertex of the tree that is closest to x according to a certain *metric*, typically a weighted Euclidean distance between the states.

$$\texttt{Nearest}(T(V, E), x) = \text{argmin}_{u \in V} ||x - v||$$

3. *Steering*: given two states $x, y \in X$, the function returns a third point $z \in X$ such that it minimizes the distance from the second node y while being reachable from the previous node x.

$$\texttt{Steer}(x, y) = \text{argmin}_{z \in V, \texttt{Feasible}(x,z)} ||z - y||$$

Multiple possibilities exist to determine whether the path between x and z is feasible. Original versions of the algorithm utilized an upper bound on value computed by the metric utilized in the nearest neighbor selection.

4. *Obstacle Collision Check*: given two states x and y, a Boolean-valued function `ObstacleFree` returns `True` if no obstacles are detected in the path between the two states. The most simple and straightforward form of collision check is done by connecting the two states through a segment and checking whether the segment

intersects any obstacle.

A newer version of the RRT algorithm has been introduced in [13], with the aim of finding an optimal path to which the simple RRT algorithm does not converge, and it is presented in algorithm 3.2.

Algorithm 3.2 RRT*

```
1: Initialize T with $x_0$ first node
2: for $k = 1 : K$ do
3:     $x_{sample} \leftarrow$ Sample: Generate random state
4:     $x_{nearest} \leftarrow$ Nearest$(T, x_{sample})$: Find nearest tree node
5:     $x_{new} \leftarrow$ Steer$(x_{nearest}, x_{sample})$: Generate new possible tree node
6:     if ObstacleFree$(x_{nearest}, x_{new})$ then
7:         AddVertex$(T, x_{new})$
8:         $x_{min} \leftarrow x_{nearest}$
9:         $X_{near} \leftarrow$ Near$(T, x_{new})$: Locate tree nodes in the vicinity of the new node
10:        for all $x_{near} \in X_{near}$ do
11:            if ObstacleFree$(x_{near}, x_{new})$ then
12:                $c =$ Cost$(x_{near})$ + Cost$(x_{near}, x_{new})$: Evaluate cost from each nearby node
13:                if $c <$ Cost$(x_{new})$ then
14:                    $x_{min} \leftarrow x_{near}$: Find minimum cost path
15:                end if
16:            end if
17:        end for
18:        AddEdge$(E, x_{near}, x_{new})$
19:        for all $x_{near} \in X_{near} \setminus x_{min}$ do
20:            if ObstacleFree$(x_{new}, x_{near})$ & Cost$(x_{near}) >$ Cost$(x_{new})$ + Cost$(x_{new}, x_{near})$
               then
21:                $x_{parent} \leftarrow$ Parent$(x_{near})$: Find the previous parent of nearby node
22:                $E \leftarrow$ RemoveEdge$(x_{parent}, x_{near})$: Remove edge from previous parent and
                   nearby node
23:                $E \leftarrow$ AddEdge$(x_{new}, x_{near})$: Generate new path from new node and nearby
                   node
24:            end if
25:        end for
26:    end if
27: end for
28: Return T,E
```

The RRT* algorithm follows the same steps of the simple RRT algorithm for what concerns the new node generation, but before adding the new edges to the tree two operations are carried out:

Cost Evaluation: After generating the new node and including it in the tree, RRT* utilizes the function `Near` in order to search among the nodes belonging to the tree which are in close proximity of the newly generated node, attempting to find a node which shows a lower cost path with respect to $x_{nearest}$. This is mainly done because the steering operation typically follows a simple metric that does not consider a more complex cost functional in the new node generation.

Rewiring: The new nodes in RRTs are generated sequentially, growing the tree and selecting the nearest node of the latter with a certain heuristic, repeating the process until the end condition or the maximum number of iteration is reached. One of the main issues with this approach is that while the state space is explored thoroughly, it is impossible to select shorter (or, in this case, less costly) paths between two already generated nodes. The function `Near` can be utilized in order to search in the neighborhood of the newly generated node for vertices that were reached with a higher cost when compared to a new connection attempt from x_{new} to $x_{near} \in X_{near}$. If this condition is verified, x_{new} is utilized as the new parent node, the older edge connecting x_{near} and its parent is eliminated from the edges and a new connection is established.

Near Vertices utilizes as input the tree $T(V, E)$ and a node $x \in V$ in order to find all the vertices near to x according to a certain metric, typically coincident with the metric utilized to find the nearest neighbor. For the weighted Euclidean distance metric case, for example, the near vertices can be located as:

$$\texttt{Near}(T(V,E), x, r) \rightarrow \{V' \subset V \mid \forall\, y \in V' \; ||y - x|| < r\}$$

Where r is a radius that should relate as closely as possible to the limits of reachable states from the current position.

In cases of nonholonomic systems, as are Space Manipulator Systems, various problems arise during path planning. In particular, the main issues these types of system experience in the path planning via RRTs are the modelling of obstacles, that are straightforward for what concerns Cartesian space planning, but that become difficult to obtain in the state space of systems, furthermore, many configuration states have to be discarded because

they would enter the region denominated of inevitable collision, inside which the collision with obstacle is unavoidable with any control action, a solution is the usage of artificial potential fields in order to circumvent these issues, leading to complex nonlinear control problems; another problem is the metric, which in Cartesian space can be designed and optimized while in the configuration space its formulation becomes much harder.
With reference to RRT*, the construction of the path is not straightforward, and it is subjected to the problem of straight line connection that the RRT* in Cartesian space typically generates, in fact due to the differential constraints it is impossible to connect two states through a straight line, which the RRT* algorithm tends to do during its rewiring phase.
For these specific problems, a specific algorithm has been formulated and denominated Kinodynamic RRT* [44]. The main path planning step in which these algorithms differ from their counterparts is in the rewiring mechanism. It would be in fact impossible to guarantee that any pair of states (belonging to X_{near}) can be connected through an *optimal* path. This algorithm, however, is specialized in system with linearized dynamics.

RRTs with Goal-Oriented Sampling is another class of Rapidly-Exploring Random Trees, in which what changes is the sampling mechanism rather than the new node generation. The simplest version of Goal-Biased RRT is shown in algorithm 3.3.

Algorithm 3.3 Goal-Biased RRT

```
1: Initialize T with $x_0$ first node
2: for $k = 1 : K$ do
3:     $p \leftarrow [0, 1]$
4:     if $p > p_{threshold}$ then
5:         $x_{sample} \leftarrow x_{goal}$
6:     else
7:         $x_{sample} \leftarrow$ Sample: Generate new random state
8:     end if
9:     ExtendRRT
10: end for
11: Return T,E
```

The peculiarity of these types of algorithms is that at the start of each iteration, a random number is generated. If this number is above a certain threshold, the goal state becomes the sample state, and the algorithm tends to explore the regions closer to the goal rather than deeply exploring the rest of the configuration space.

3.2. Proposed Algorithm

The proposed path planning algorithm is based on a simple Rapidly-Exploring Random Tree, modified in order to have a faster exploration of the configuration space. The path planning is executed at the kinematics level, and it shall output a path towards the goal that can be followed by a controller. The algorithm includes a robust obstacle avoidance, a consideration on the minimization of the chaser base rotation, and it attempts to minimize the time of the manoeuvre.

In order to generate kinematically feasible paths, connecting nodes simply and fast, the space manipulator system state for the execution of the RRT algorithm includes the joint positions $\boldsymbol{\theta}$ and velocities $\dot{\boldsymbol{\theta}}$, and the control in order to bring the system from one state to another is represented by the joint accelerations $\mathbf{a}_\theta$.

The simple state equations for the RRT expansion is:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\boldsymbol{\theta}} \\ \ddot{\boldsymbol{\theta}} \end{bmatrix} = \mathbf{f}(\mathbf{x}, t) = \begin{bmatrix} \dot{\boldsymbol{\theta}} \\ \mathbf{a}_\theta \end{bmatrix} \tag{3.1}$$

The generation of the trajectory for the joint variables is done independently from the motion of the spacecraft base as its linear and angular coordinates movement can be computed directly from the trajectory of the joint variables, as they are uncontrolled and thus depend on them. Their value depends on the value of the joint velocities as in (2.85), which is based on the equations of conservation of linear and angular momentum.

$$\mathcal{V}_b = -M_b^{-1} M_{bm} \dot{\boldsymbol{\theta}} \tag{3.2}$$

While these variables do not directly enter the state of the RRT and thus are not utilized in the generation of new nodes, their computation is of pivotal importance to retrieve the inertial position of the whole spacecraft-manipulator system, including its base, all the links and all the joints. These positions are then utilized for goal connection checks and obstacle avoidance.

The algorithm follows a scheme similar to the RRT algorithm in [37],[36] as the path planning is performed at a joint level, however here the goal connection is kinematics-based, the obstacle avoidance is performed by a dense sampling of the obtained trajectory and the minimization of the base rotation is obtained by assigning costs to nodes rather than using the bidirectional approach.

The sequence of operations of the algorithm is:

1. The algorithm generates new random nodes in the joint space $\begin{bmatrix} \boldsymbol{\theta} \end{bmatrix}$, then it applies

a distance metric in order to find the closest node belonging to the tree.

2. From the closest node, the path planning on the joint variables is initiated in order to find a suitable connection between the tree and the new sampled joint configuration.

3. After the connection is established between joint coordinates, the corresponding motion of the base and of the whole spacecraft-manipulator system is computed. This allows for the obstacle avoidance to be then performed.

4. Finally, following the approach of the RRT* algorithms, a cost is assigned to each newly generated node, and a better connection is sought to the new joint angles that feature a low joint space distance from the new node.

5. At the end of each iteration of the algorithm, similarly to [41], if the corresponding end-effector position is close to the one of the grasping point, a connect-to-goal step is attempted by utilizing a version of the inverse kinematic algorithm for free-floating space manipulator systems that includes a second task of an attempt of minimization of the base rotation [3].

The general algorithm flowchart is given in fig. 3.1, and each block of the algorithm will be treated in a specific section.

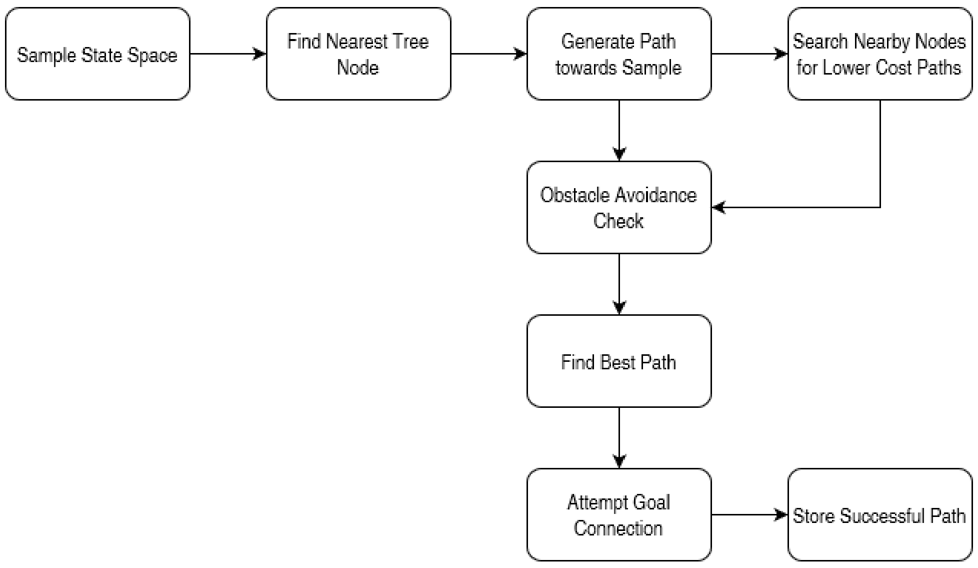


Figure 3.1: RRT-Based Algorithm

The overall primary goal for the algorithm is to produce a trajectory such that, at a certain time t_f, the following conditions are satisfied:

$$\mathbf{x}_{EE}(t_f) = \mathbf{x}_{GP}(t_f) \tag{3.3a}$$

$$\boldsymbol{\varphi}_{EE}(t_f) = \boldsymbol{\varphi}_{GP}(t_f) \tag{3.3b}$$

$$\mathbf{v}_{EE}(t_f) = \mathbf{v}_{EE}(t_f) \tag{3.3c}$$

$$\boldsymbol{\omega}_{EE}(t_f) = \boldsymbol{\omega}_{GP}(t_f) \tag{3.3d}$$

Where $\boldsymbol{\varphi}$ symbolizes the generic orientation of the reference frames of the two points that will need to be represented in a consistent way.

3.2.1. Sampling and Metric

The admissible joint variables are defined as the ones which respect the constraints on the minimum and maximum value, given as specifics of the robotic manipulator, including limits on angular positions, velocities and accelerations:

$$\boldsymbol{\theta}_{min} \leq \boldsymbol{\theta} \leq \dot{\boldsymbol{\theta}}_{MAX} \tag{3.4a}$$

$$\dot{\boldsymbol{\theta}}_{min} \leq \dot{\boldsymbol{\theta}} \leq \dot{\boldsymbol{\theta}}_{MAX} \tag{3.4b}$$

$$\ddot{\boldsymbol{\theta}}_{min} \leq \ddot{\boldsymbol{\theta}} \leq \ddot{\boldsymbol{\theta}}_{MAX} \tag{3.4c}$$

At the start of each iteration, a random sample of joint angles is generated from an uniform distribution of the admissible ones.

$$\boldsymbol{\theta}_{sampled} \in (\boldsymbol{\theta}_{min}, \boldsymbol{\theta}_{MAX}) \tag{3.5}$$

Then, following the approach of Rapidly-Exploring Random Trees, the nearest node belonging to the tree *T(V,E)* needs to be found as the node from which the connection towards the new sample state is started.
The nearest node is obtained through a *metric* that should represent as close as possible the cost-to-go to said sample [19]. In this case, since the control sequence to reach the new samples is straightforwardly defined (3.1), it is possible to use the simplest metric of the Euclidean distance between samples in order to find the nearest node.

$$d = ||\boldsymbol{\theta}_{sampled} - \boldsymbol{\theta}_{tree}|| \tag{3.6}$$

The selected tree node from which the steering is performed is the node for which the Euclidean distance from the new sample point is minimal.

3.2.2. Steering Method

The steering process is the generation of a new node of the tree starting from the nearest node of the tree $\mathbf{x}_{nearest}$ which has been found applying the previously defined metric, and the new sample obtained from the uniform distribution of admissible samples as in section 3.2.1.

The generation of a new node is non-trivial for nonholonomic systems. [37] and [36] use a method which involves creating 3^n, where n is the number of joints, combinations of available joint torques, where the torque applied to each joint assumes either the maximum value, the maximum negative value, or zero. After generating a set of 3^n points through the propagation of the state equations as many times, it selects the generated node closest to the sample according to the metric.

By tackling the problem at a kinematic level, it is possible to connect any pair of states whenever obstacles are not considered. The presence of obstacles can then be considered after the path between any couple of states has been generated.

Nodes are generated at zero joint velocity in order to be able to move towards the random sample generated at the next iteration through a trajectory that reaches the maximum allowed joint velocity starting from zero velocity, whereas starting from an initial velocity might greatly decrease the ability to reach the successive random sample in a low amount of time.

The nodes are generated starting from the computation of the difference between the two states $\Delta\boldsymbol{\theta}$. It is possible to distinguish between two cases once the distance is known:

1. In order to reach the new node, it is sufficient to apply a constant acceleration for a certain amount of time Δt, and an opposite constant deceleration for the same amount of time.

2. The distance between the two nodes is high in magnitude, and the connection cannot be granted by using constant acceleration, as it would either violate constraints on the maximum allowed joint acceleration or on the joint velocity. In this case, the path is divided in three subpaths: for a certain Δt_1 at a constant acceleration, then at a constant velocity for a different Δt_2, and finally for the same Δt_1 at the opposite constant deceleration. This allows to reach any state depending on Δt_2 while still enforcing null final joint velocity and being compliant with the joint velocity and acceleration limits.

Both cases make use of the integration of the simple kinematic equations in order to represent the path of the joint angles:

$$\boldsymbol{\theta}_f = \boldsymbol{\theta}_0 + \dot{\boldsymbol{\theta}}_0(t - t_0) + \frac{1}{2}\ddot{\boldsymbol{\theta}}(t - t_0)^2 \tag{3.7}$$

It is then possible to fully compute the final joint angles that would result from the application of a constant acceleration to arrive to an intermediate state i, and a constant opposite deceleration, consecutively to the acceleration, to bring the system to zero velocity and to the final state. This results in the following set of equations:

$$\boldsymbol{\theta}_i = \boldsymbol{\theta}_0 + \dot{\boldsymbol{\theta}}_0(t - t_0) + \frac{1}{2}\bar{\boldsymbol{a}}(t - t_0)^2 \tag{3.8a}$$

$$\boldsymbol{\theta}_f = \boldsymbol{\theta}_i + \dot{\boldsymbol{\theta}}_i(t - t_i) - \frac{1}{2}\bar{\boldsymbol{a}}(t - t_i)^2 \tag{3.8b}$$

Knowing that $t - t_i$ is the same as $t - t_0$, now denominated Δt, and knowing that $\dot{\theta}_0 = 0$ as all nodes are generated at zero velocity and that $\dot{\theta}_i = \bar{\boldsymbol{a}}\Delta t$, it is possible to compute the required acceleration depending on $\Delta\boldsymbol{\theta}$:

$$\bar{\boldsymbol{a}} = \frac{\Delta\boldsymbol{\theta}}{\Delta t^2} \tag{3.9}$$

To understand whether the node is reachable through the application of this acceleration through the fixed amount of time Δt, two checks are needed on whether the acceleration is compliant with the joint limits, and whether the maximum reached velocity during the motion is within the set of admissible joint velocity.
The constraint on the maximum and minimum acceleration component is:

$$\bar{\boldsymbol{a}}_{MAX} \leq \ddot{\boldsymbol{\theta}}_{MAX} \tag{3.10a}$$

$$\bar{\boldsymbol{a}}_{min} \geq \ddot{\boldsymbol{\theta}}_{min} \tag{3.10b}$$

While the constraints on the maximum and minimum joint velocity reached are:

$$\max(\dot{\boldsymbol{\theta}}_i) = \bar{\boldsymbol{a}}_{MAX}\Delta t \leq \dot{\theta}_{MAX} \tag{3.11a}$$

$$\min(\dot{\boldsymbol{\theta}}_i) = \bar{\boldsymbol{a}}_{min}\Delta t \geq \dot{\theta}_{min} \tag{3.11b}$$

Where $\bar{\boldsymbol{a}}_{MAX}$ and $\bar{\boldsymbol{a}}_{min}$ are the maximum and minimum components of the acceleration with their respective sign.

Overall, the constraints on the acceleration are:

$$\bar{\boldsymbol{a}}_{MAX} \leq \ddot{\boldsymbol{\theta}}_{MAX} \tag{3.12a}$$

$$\bar{\boldsymbol{a}}_{min} \geq \ddot{\boldsymbol{\theta}}_{min} \tag{3.12b}$$

$$\bar{\boldsymbol{a}}_{MAX} \leq \frac{\dot{\theta}_{MAX}}{\Delta t} \tag{3.12c}$$

$$\bar{\boldsymbol{a}}_{min} \geq \frac{\dot{\theta}_{min}}{\Delta t} \tag{3.12d}$$

Whenever any of these constraints is not satisfied, it is not possible to connect two states through this simple method. However, it is possible to define a new acceleration vector proportional to the current one, such that the trajectory is performed in the direction of the new sample, while being compliant with the constraints.
This is done by computing the ratio of the acceleration component that violates the constraints with the value of the violated constraints. If multiple constraints are unsatisfied, the highest ratio is taken such that it incorporates the others. Then, this ratio is utilized to multiply all acceleration components, in order to maintain the acceleration vector direction while changing its magnitude to be compliant with the constraints.

$$r = \max\left(\frac{\bar{\boldsymbol{a}}_{MAX}}{\ddot{\boldsymbol{\theta}}_{MAX}}, \frac{\bar{\boldsymbol{a}}_{min}}{\ddot{\boldsymbol{\theta}}_{min}}, \frac{\bar{\boldsymbol{a}}_{MAX}}{\frac{\dot{\theta}_{MAX}}{\Delta t}}, \frac{\bar{\boldsymbol{a}}_{min}}{\frac{\dot{\theta}_{min}}{\Delta t}}\right) \tag{3.13}$$

The new acceleration is thus:

$$\bar{\boldsymbol{a}}_{NEW} = \frac{1}{r}\bar{\boldsymbol{a}} \tag{3.14}$$

Since it is impossible to reach the new state with the newly computed acceleration, an intermediate step of motion with constant velocity is performed. The constant velocity is applied for an unknown time Δt_2, whose dependency with respect to $\Delta\boldsymbol{\theta}$ can be computed.
By applying (3.7) to the three-segment path, with the assumptions of same Δt_1 for acceleration and deceleration, and knowing that the constant velocity $\dot{\boldsymbol{\theta}}_{i,1}$ is $\bar{\boldsymbol{a}}_{NEW}\Delta t_1$, with null initial velocity it is possible to write:

$$\boldsymbol{\theta}_{i,1} = \boldsymbol{\theta}_0 + \frac{1}{2}\bar{\boldsymbol{a}}_{NEW}\Delta t_1^2 \tag{3.15a}$$

$$\boldsymbol{\theta}_{i,2} = \boldsymbol{\theta}_{i,1} + \bar{\boldsymbol{a}}_{NEW}\Delta t_1 \Delta t_2 \tag{3.15b}$$

$$\boldsymbol{\theta}_f = \boldsymbol{\theta}_{i,2} + \bar{\boldsymbol{a}}_{NEW}\Delta {t_1}^2 - \frac{1}{2}\bar{\boldsymbol{a}}_{NEW}\Delta {t_1}^2 \tag{3.15c}$$

By substituting (3.15a) and (3.15b) in (3.15c), the propagation time can be computed directly from $\Delta\boldsymbol{\theta}$, knowing the other parameters involved in the equations:

$$\Delta t_2 = \frac{\Delta\boldsymbol{\theta}}{\bar{\boldsymbol{a}}_{NEW}\Delta t_1} - \Delta t_1 \tag{3.16}$$

It is important to note that the position joint limits are never checked throughout the application of this approach, as the minimum and maximum for the joint positions will be at the extremes of each propagation, which belong necessarily to the admissible set since they are obtained through sampling.
After the variables are obtained, it is possible to integrate the overall space-manipulator system kinematic model in order to retrieve the linear and angular motion of the spacecraft base.

3.2.3. Obstacle Avoidance

Avoiding obstacles is of pivotal importance for the success of the path planning of the space manipulator. Obstacles in the workspace can refer to debris or satellite parts that float in the vicinity of the target, the target satellite body itself and its appendages, and also the chaser base and the self-collisions of the manipulator arm.
Rapidly-Exploring Random Trees are excellent algorithms for obstacle avoidance in a dynamic environment, and can include the avoidance of moving obstacles inside the workspace of the manipulator, provided they are modelled correctly.
For Space Manipulator System, the main problem arising during obstacle avoidance is the evaluation of the relative distance from the obstacle, as checking the distance between each point of the manipulator and the obstacle requires an optimization problem parametrizing each link with a continuous variable and the minimization of a distance function.
In order to simplify the process, key points are chosen on the manipulator in order to perform the computation of the distance from the obstacles. The minimal amount of points guaranteeing a safe obstacle avoidance has been identified in [25] to be the positions of the joints and the positions of the links midpoints.
The obtained path is discretized in small time intervals, and for each time sample t_i the direct kinematics are computed in order to find the key points positions in the inertial reference frame $\mathbf{x}_{G,i}(t_i)$ and $\mathbf{x}_i(t_i)$.
Obstacles such as the target satellite (excluding its grasping handle) show their own dynamic, thus their position and orientation change with time, and for this reason they will be evaluated at the time t_i.

To find the relative distance between the obstacles and the manipulator points, it is necessary to refer the position of the manipulator and the obstacle's envelope to the same reference frame. In this case, a transformation to the obstacle body axes is considered.
The position of the links centers of mass, assumed to be at the midpoints between two joints, and of the joints are available in the inertial reference frame through the relations in section 2.4.1 and are already available as they are utilized to compute the dynamic parameters of the system.
The rotation to the obstacle body frame from the inertial reference frame, which is in this case identified with the Local Vertical Local Horizontal frame due to the short duration of the overall manoeuvre, is performed through the relation (2.15), in which the orientation angles depend on time through the Euler equations, expressed in section 2.3.2.
The overall transformation of the position of the manipulator's joints and links in the body axes of the $j-th$ obstacle are:

$$\mathbf{x}_i^{B,O_j} = R_{\mathcal{I}}^{B,O_j}(\boldsymbol{\varphi}(t))\mathbf{x}_i \tag{3.17}$$

Where i represents the $i-th$ point of the manipulator, including the joints and the midpoints.
At this point, the distance from the obstacle is known to be:

$$\Delta\mathbf{x}_{i,O_j}^{B,O_j} = \mathbf{x}_i^{B,O_j} - R_{\mathcal{I}}^{B,O_j}(\boldsymbol{\varphi}(t))\mathbf{x}_{O_j} \tag{3.18}$$

Where the position of the obstacle's computed geometric center is expressed in its body axes to have the distance from the manipulator to be coherent with the framework.
After computing the distance between the manipulator's points and the obstacle center, the closest point to the obstacle can be found as the minimum Euclidean norm of the distance, and will be then utilized to compute the effective distance with the obstacle's walls.

$$d_{min} = \min||\Delta\mathbf{x}_{i,O_j}^{B,O_j}|| \tag{3.19}$$

Obtaining the minimum distance in each obstacle's body axes is useful within the framework described in [24], which defined the possibilities to model the obstacles in the manipulator workspace through continuous geometric primitives. Typically, in fact, the obstacles found in space such as satellites or nozzles should not be modelled through spheres that would either compromise the robot's workspace or underestimate the true envelope of the obstacles. For this reason, super-quadric functions can be used to approximate different-shaped obstacles. For cuboid-like objects, as the one considered in this work, the envelope is represented through the following relation:

$$S(x,y,z) = \left(\frac{x}{a}\right)^8 + \left(\frac{y}{b}\right)^8 + \left(\frac{z}{c}\right)^8 - 1 \tag{3.20}$$

Where a, b, c are the sides of the cuboid divided by 2. Thus, the spatial coordinates of the various manipulator points can be substituted inside the equations in order to compare the results, in fact the following hold:

$$\begin{cases} S(x_i, y_i, z_i) < 0 & \text{if} \quad \text{point i} \quad \text{is inside} \quad S \\ S(x_i, y_i, z_i) = 0 & \text{if} \quad \text{point i} \quad \text{is on the surface} \quad S \\ S(x_i, y_i, z_i) > 0 & \text{if} \quad \text{point i} \quad \text{is outside} \quad S \end{cases} \tag{3.21}$$

These conditions are rather optimistic, in fact the superquadric functions tend to underestimate the dimensions of the objects as shown in 3.2, particularly at their vertices. In this work, the satellites are approximated adding a safety distance to generate the envelopes of the cuboids: for safer obstacle avoidance, a larger envelope can be generated taking the vertices into account and using $a, b, c = \frac{\sqrt{2}}{2}(a_0, b_0, c_0)$ where the latter quantities are the sides of the cuboid. After the distance from the center of the obstacle, in its own

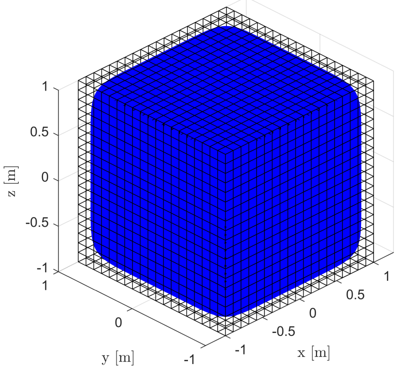


Figure 3.2: Super-quadric envelope and cube, using $a, b, c = \frac{1}{2}l$

body frame, has been substituted within the equation of the geometric primitive for each point of the path between two nodes, it is possible to understand whether any of the manipulator points has hit the obstacle, and if that is the case, the new node is completely discarded and a new iteration begins.

3.2.4. New Node Optimization

Following the scheme defined for RRT* algorithms, once a connection to a new state is performed and validated through obstacle avoidance, it is possible to query the surroundings of the newly found node, in order to find whether a different connection to the same state would have generated a better overall final state.
For this reason, a cost is assigned to each new node of the tree. Since the generation of new nodes regards the sole joint coordinates, different initial conditions modify the configuration of the overall Spacecraft-Manipulator System, including the rotation of the chaser base, the position and the orientation of the end effector. These quantities, that are the goals of the path planning, can be included in a cost function that is assigned to each new node, and due to the dependencies on the initial conditions, to each path between two nodes.

- **Distance from the goal**: The end-effector position and orientation for each node are given from the direct kinematics algorithm. In the case of a free-floating spacecraft-manipulator system, this depends on the joint coordinates angular position and on the base coordinates linear and angular position. The motion of the base is heavily dependent on the initial conditions, thus a different path to reach the same joint variables modifies the end-effector position:

$$\mathbf{x}_{EE} = E_{3\times4} T_{\mathcal{I}}^{B}(\boldsymbol{\varphi}) T_{B}^{0} \prod_{i=1}^{n} A_{i-1}^{i}(q_i(t)) \mathbf{z}_0 = E_{3\times4} T_{\mathcal{I}}^{EE} \mathbf{z}_0 \tag{3.22}$$

Where $\mathbf{z}_0$ is a column vector $\begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T$.
The orientation of the end-effector is expressed in quaternions in order to avoid representation singularities, since the minimal representation is considered less important than in the equations of motion for the orientation of the base. This choice also maintains the possibility of using the Geometric Jacobian rather than computing the Analytical Jacobian.
The rotation matrix of the end-effector frame with respect to the Local Vertical Local Horizontal (inertial) frame can be extracted from the homogeneous transformation matrix:

$$R_{\mathcal{I}}^{EE} = E_{3\times4} T_{\mathcal{I}}^{EE} E_{3\times4}^{T} \tag{3.23}$$

Where $E_{3\times4}$ is a matrix extracting the first three rows from a $4 \times n$ matrix, and it has been defined in (2.40).
By using the conversion formulae between the Direction Cosine Matrix and the quaternions in (2.11), it is possible to retrieve the unit quaternion of the relative

orientation of the end-effector in the LVLH frame.
The relative orientation between two frames is described through quaternions as $\Delta\mathbf{q} = \mathbf{q}_d * \mathbf{q}_e^{-1}$ [40], and the reference frames are aligned if $\Delta q = \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix}$, hence the relative orientation of the grasping point and the end-effector in the LVLH frame depends on the vector part of the quaternion measured error:

$$\Delta\mathbf{q} = \Delta\boldsymbol{\varepsilon} = \eta_e(\boldsymbol{\theta})\boldsymbol{\varepsilon}_d - \eta_d\boldsymbol{\varepsilon}_e(\boldsymbol{\theta}) - \boldsymbol{\varepsilon}_d^{\times}\boldsymbol{\varepsilon}(\boldsymbol{\theta}) \tag{3.24}$$

Where $\mathbf{q}_d$ is the orientation of the grasping point reference frame at time instant t_k, while $\mathbf{q}_e$ is the orientation of the end-effector reference frame at time t_k.
The portion of the cost function regarding the relative position and orientation of the end-effector and the grasping point is simply:

$$f_{EE} = w_x||\mathbf{x}_{GP}(t_k) - \mathbf{x}_{EE}(t_k)|| + w_\alpha||\boldsymbol{\varepsilon}_{GP}(t_k) - \boldsymbol{\varepsilon}_{EE}(t_k)|| \tag{3.25}$$

Where t_k is the time at which sample k is reached.

- **Base Rotation**: free-floating space manipulator systems must attempt to reach the final point by using manipulator trajectories that induce a low coupling momentum on the spacecraft base as to avoid unwillingly modifying its attitude and losing the telecommunications or tracking capabilities, or to require costly manoeuvres to restore the attitude.
In this framework, rather than attempting to execute manoeuvres that would fall into the reaction null space of the Spacecraft-Manipulator System, the generated nodes are assigned a cost that grows with the distance of the base attitude with respect to its desired value.
The cost of a sample relative to the base rotation is evaluated through the norm of the difference between the orientation of the two reference frames:

$$f_{\alpha_B} = w_B||\boldsymbol{\varphi}_b^d - \boldsymbol{\varphi}_b(t_k)|| \tag{3.26}$$

- **Time**: the minimization of the time of the manoeuvre is typically less important than the previously defined tasks. However, completing the manoeuvres in shorter times allows to introduce a lower magnitude error of the simplified model considering a fixed LVLH frame with respect to the real case.
Thus, the time at which a certain node is reached is computed as an additional term

to the weighted cost function.

$$f_t = w_t \cdot t_k \tag{3.27}$$

After the generation of a new node, its cost is evaluated through the overall cost function (3.28).

$$f = f_{EE} + f_{\alpha_B} + f_t \tag{3.28}$$

Then, a set of nodes in the proximity of the newly generated node is found as nodes that show a distance metric, which will be the same utilized to find the nearest sample, lower than a threshold $\bar{\theta}$, which is a free parameter for the algorithm, whose value will relate to a certain overall sum of the joint rotations:

$$\boldsymbol{\Theta}_{Near} = \left\{ V \in T(V,E) \quad | \quad ||\boldsymbol{\theta}_{new} - \boldsymbol{\theta}_{tree}|| < \bar{\theta} \right\} \tag{3.29}$$

For each point, a connection to $\boldsymbol{\theta}_{new}$ is generated through the same method utilized for the generation of new nodes of the RRT, in section 3.2.2.
After the connections have been found, a check for possible collisions with the obstacles is performed following the method described in section 3.2.3.
Among the paths with no collisions, the lowest cost trajectory is selected. It is possible that the best path was corresponding to the initially found path towards the new goal starting from $\boldsymbol{\theta}_{Nearest}$, otherwise a state $\boldsymbol{\theta}_{Near}$ will be the new parent node to the node that will be added to the tree.

3.2.5. Connection to Goal

At the end of each iteration, the position, orientation, and velocity of the end effector are computed for the last generated node of the tree.
Whenever the weighted sum of the distance between the current position of the end-effector and the current position and orientation of the grasping point, of the orientation difference between the end-effector and the grasping point, and of the chaser base attitude with its desired value, are below a certain threshold, which is a free parameter for the algorithm, a solution is seeked through the inverse kinematics algorithm for free-floating space manipulators, which has been developed in [3].

$$d_{CG} = w_x^{GC}||\mathbf{x}_{GP} - \mathbf{x}_{EE}|| + w_\varphi^{GC}||\boldsymbol{\varepsilon}_{\boldsymbol{GP}} - \boldsymbol{\varepsilon}_{EE}|| + w_{\varphi_b}^{GC}||\boldsymbol{\varphi}_b^d - \boldsymbol{\varphi}|| < k \tag{3.30}$$

The inverse kinematic algorithm for free-floating space manipulators is similar to the Jacobian pseudo-inverse algorithm for fixed-base redundant manipulators, with the difference that base variables computations will essentially be part of the algorithm as their changes affect the position and orientation of the end-effector, but they themselves depend only on the motion of the joints.
The algorithms are based on the simple relation between the velocity of the end-effector and the joint velocities, that is:

$$\dot{\boldsymbol{\theta}} = J_A^{\dagger} \mathcal{V}_{EE} \tag{3.31}$$

Where the analytical Jacobian needs to be used in order to correctly express the relation between the end-effector angular velocity and the Euler angles-represented rotation. With a different representation, such as unit quaternion, for the orientation of the end-effector, it is possible to avoid the usage of the analytical Jacobian, using the Geometric Jacobian. In this specific case, using a representation such as the unit quaternion, it is possible to use the relation between the Generalized Jacobian Matrix, the end-effector velocity and the joint velocities, by using the relation in (2.86):

$$\dot{\boldsymbol{\theta}} = \hat{J}^{\dagger} \mathcal{V}_{EE} \tag{3.32}$$

The † indicates the Moore-Penrose pseudo-inverse of the non-square Generalized Jacobian Matrix.
Then, in order to iterate the algorithm and match the end-effector position and orientation along a trajectory, the joint coordinates are integrated through a simple forward Euler numerical scheme:

$$\boldsymbol{\theta}(t_{k+1}) = \boldsymbol{\theta}(t_k) + \dot{\boldsymbol{\theta}} \Delta t \tag{3.33}$$

Substituting (3.32) in (3.33), the joint coordinates variation over time relative to the trajectory defined by the end-effector velocity can be obtained:

$$\boldsymbol{\theta}(t_{k+1}) = \boldsymbol{\theta}(t_k) + \hat{J}^{\dagger} \mathcal{V}_{EE} \Delta t \tag{3.34}$$

This solution, however, includes a numerical drift in the integration of the equations, thus it is necessary to resort to a closed-loop solution involving the definition of the operational space error between the end-effector desired position and orientation and its current position and orientation $\mathbf{e} = \begin{bmatrix} \mathbf{e}_x & \mathbf{e}_\varphi \end{bmatrix}^T$:

$$\mathbf{e}_x = x_{GP} - x_{EE}(\boldsymbol{\theta}) \tag{3.35}$$

The orientation error expression depends on the representation of the orientation that has been chosen. In this case, the unit quaternion representation has been utilized as it is consistent with the directions of $\boldsymbol{\omega}_{EE}$.
The relative orientation of two reference frames is available from (3.24):

$$\mathbf{e}_{\varphi} = \varepsilon_{GP} - \varepsilon_{EE}(\boldsymbol{\theta}) \tag{3.36}$$

The time derivative of the overall error is:

$$\dot{\mathbf{e}} = \dot{\mathbf{x}}_d - \mathcal{V}_{EE} = \dot{\mathbf{x}}_d - \hat{J}\dot{\boldsymbol{\theta}} \tag{3.37}$$

It is then possible to substitute the expression for the time derivative of the joint coordinates in order to obtain the error evolution over time.
To generate an asymptotically stable system, the typical choice of inverse kinematic algorithms is:

$$\dot{\boldsymbol{\theta}} = \hat{J}^{\dagger}(\dot{\mathbf{x}}_d + K\mathbf{e}) \tag{3.38}$$

Such that, after the substitution and if K is a positive-definite matrix, the following generated system is asymptotically stable:

$$\dot{\mathbf{e}} + K\mathbf{e} = 0 \tag{3.39}$$

The error will tend to zero with a growing number of iterations, with a convergence rate that depends on the eigenvalues of K. However, the values for the elements of K are limited by the joint limits in terms of velocity. The check on whether these are violated is done right after the algorithm has completed the goal connection, and if there are violations, the path to the goal is discarded. Likewise, the obstacle avoidance is performed after the check on the violation of the joint limits, and paths can be discarded if they impact the target at any point different from the grasping point.
The main advantage in using a redundant manipulator for the inverse kinematics is the possibility to modify the joint velocity vector for the algorithm with the inclusion of an additional task that is performed through the unused degree of freedom of the manipulator. This task is typically the minimization of the chaser base spacecraft rotation, and it can be added to the algorithm through a projection on the null space of the Generalized Jacobian Matrix as to not interfere with the main task of following the end-effector trajectory:

$$\dot{\boldsymbol{\theta}} = \hat{J}^{\dagger}(\dot{\mathbf{x}}_d + K\mathbf{e}) + (I_{n\times n} - \hat{J}^{\dagger}\hat{J})J_C^T K_C \mathbf{e}_C \tag{3.40}$$

Where $I_{n\times n} - \hat{J}^{\dagger}\hat{J}$ projects the joint velocities computed by the error on the second task to the null space of the GJM, to avoid that the minimization of the latter error interferes with the accomplishment of the main task.
The Jacobian Matrix J_C is the matrix relating the vector of the joint velocities with the variables relative to the second task, in this case it represents the relation between the joint velocity and the base angular velocities, as in (2.85), thus it is the last three rows of the matrix:

$$J_C = -M_b^{-1} M_{bm} \tag{3.41}$$

The error $\mathbf{e}_C$ is defined as the distance from the reference attitude of the current orientation of the base, and since it is expressed in Euler angles and a typical desired orientation corresponds with the $LVLH$ frame, its value is:

$$\mathbf{e}_C = \varphi_{B,d} - \varphi_B(\boldsymbol{\theta}) = -\varphi_B(\boldsymbol{\theta}) \tag{3.42}$$

As the base orientation is represented through Euler angles.
The closed-loop inverse kinematic algorithm is stopped whenever a certain tolerance on the position and orientation of the end-effector is reached.

3.2.6. Spline Smoothing

The path generated through Rapidly-Exploring Random Trees is typically jerky, as it discretizes the state space in a number of points. While the generation in this case has been done continuously, but with sudden bumps in acceleration and velocity, it is necessary to generate an overall continuous path with a continuous variation of acceleration and velocity to provide a smooth trajectory.
The position and orientation of the base and on the end-effector does not depend on the path taken but only on the initial and final overall states, thus it is possible to utilize the RRT points as waypoints for the approximation of the state variables $\boldsymbol{\theta}(t)$ as polynomials [52].
It is possible to show that the polynomial interpolation of the joint variables and of the joint velocity is sufficient in order to match the two states, as the base linear and angular position will not depend on the path taken to get to a certain $\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}$ but rather only on the initial conditions. Thus, it is possible to define the following conditions in order to obtain a smooth path between two states:

$$\boldsymbol{\theta}(t_0) = \boldsymbol{\theta}_0 \tag{3.43a}$$
$$\boldsymbol{\theta}(t_f) = \boldsymbol{\theta}_f \tag{3.43b}$$
$$\dot{\boldsymbol{\theta}}(t_0) = \mathbf{0} \tag{3.43c}$$
$$\dot{\boldsymbol{\theta}}(t_f) = \mathbf{0} \tag{3.43d}$$
$$\ddot{\boldsymbol{\theta}}(t_0) = \mathbf{0} \tag{3.43e}$$
$$\ddot{\boldsymbol{\theta}}(t_f) = \mathbf{0} \tag{3.43f}$$

Given these six conditions, the simplest function that can be used to approximate each $\theta(t)$ is a fifth-order polynomial of the form:

$$\theta(t) = a\frac{t-t_0}{t_f-t_0}^{5} + b\frac{t-t_0}{t_f-t_0}^{4} + c\frac{t-t_0}{t_f-t_0}^{3} + d\frac{t-t_0}{t_f-t_0}^{2} + e\frac{t-t_0}{t_f-t_0} + f \tag{3.44}$$

This means that the joint velocities are the time derivative of this function:

$$\dot{\theta}(t) = 5a\frac{(t-t_0)^4}{(t_f-t_0)^5} + 4b\frac{(t-t_0)^3}{(t_f-t_0)^4} + 3c\frac{(t-t_0)^2}{(t_f-t_0)^3} + 2d\frac{t-t_0}{(t_f-t_0)^2} + e\frac{1}{t_f-t_0} \tag{3.45}$$

While the joint accelerations will be:

$$\ddot{\theta}(t) = 20a\frac{(t-t_0)^3}{(t_f-t_0)^5} + 12b\frac{(t-t_0)^2}{(t_f-t_0)^4} + 6c\frac{(t-t_0)}{(t_f-t_0)^3} + 2d\frac{1}{(t_f-t_0)^2} \tag{3.46}$$

The conditions in $t = t_0$ can be enforced in a simple manner:

$$\theta(t_0) = f = \theta_0 \tag{3.47a}$$
$$\dot{\theta}(t_0) = e = 0 \tag{3.47b}$$
$$\ddot{\theta}(t_0) = d = 0 \tag{3.47c}$$

The final conditions, corresponding to $t = t_f$, will result in a set of simple equations in the three remaining unknowns:

$$\theta(t_f) = a + b + c = \theta_f - \theta_0 \tag{3.48a}$$
$$\dot{\theta}(t_f) = 5a + 4b + 3c = 0 \tag{3.48b}$$
$$\ddot{\theta}(t_0) = 10a + 6b + 3c = 0 \tag{3.48c}$$

Where the remaining denominator terms have already been simplified. This system of linear equations can be rewritten in matrix form and solved easily:

$$\begin{bmatrix} 1 & 1 & 1 \\ 5 & 4 & 3 \\ 10 & 6 & 3 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} \Delta\theta \\ 0 \\ 0 \end{bmatrix} \tag{3.49}$$

After the path smoothing, it is necessary to perform a final check on the values of the joint variables, their velocities and their accelerations in order to find whether they are admissible.

Then, the path is sampled densely in order to repeat the obstacle avoidance check since it will differ from the paths for which it has already been performed.

4 | Simulation Results

In this chapter, the algorithm proposed in section 3.2 is validated, applied to a problem with a series of simplifying assumptions, and the tree is expanded in order to find multiple solutions to be compared. In section 4.1, the case scenario and the parameters of the simulation are introduced and discussed. Then, the results of the simulation are presented and discussed.

4.1. Simulation Setup and Parameters

The algorithm is applied to a numerical simulation of a capture of an uncontrolled target satelllite, utilizing a 4-Degrees of Freedom Manipulator mounted on a microsatellite that attempts to capture a small CubeSat.
The simulation has been carried out in the planar case, with a Space Manipulator System similar to the one described in [45].
The main task of the Space Manipulator System is to capture the grasping handle on the target satellite. This task does not require all the Degrees of Freedom of the manipulator to be performed, as it requires the end-effector to match the position and the orientation of the grasping point.
This leaves an additional degree of freedom of redundancy with respect to the main task, and it can be utilized in order to perform the restoration of the attitude of the base of the servicing satellite.
The rendez-vous manoeuvre is hypothesized to be completed before the start of the manipulator trajectory planning and the Attitude and Orbital Control System to be completely turned off during the remainder of the manoeuvre.
If the manoeuvre under these hypotheses is of sufficiently short duration, the effects due to orbital mechanics can be ignored and thus the Space Manipulator System can effectively be considered in a pure free-floating case, as no external force is applied to it.
As already discussed in previous paragraphs, in this case the target-centered LVLH reference frame will be aligned with the LVLH frame of the chaser, and it is typically assumed in literature to be fixed and thus aligned with the ECI reference frame.

4.1.1. Physical Parameters

The Spacecraft-Manipulator System's 3D model is shown in fig. 4.1, in its body axes reference frame.
The manipulator arm lies in the xy-plane of the body axes reference frame, and the joints are highlighted and rotated by an angle of 30°.
The figure also shows the body axes, the end-effector reference frame and the joint-centered reference frames, each aligned to the previous link except for the first joint for which the orientation of the reference frame is arbitrary. The x-axes of each frame are in red, the y-axes in green, and the z-axes are in blue.

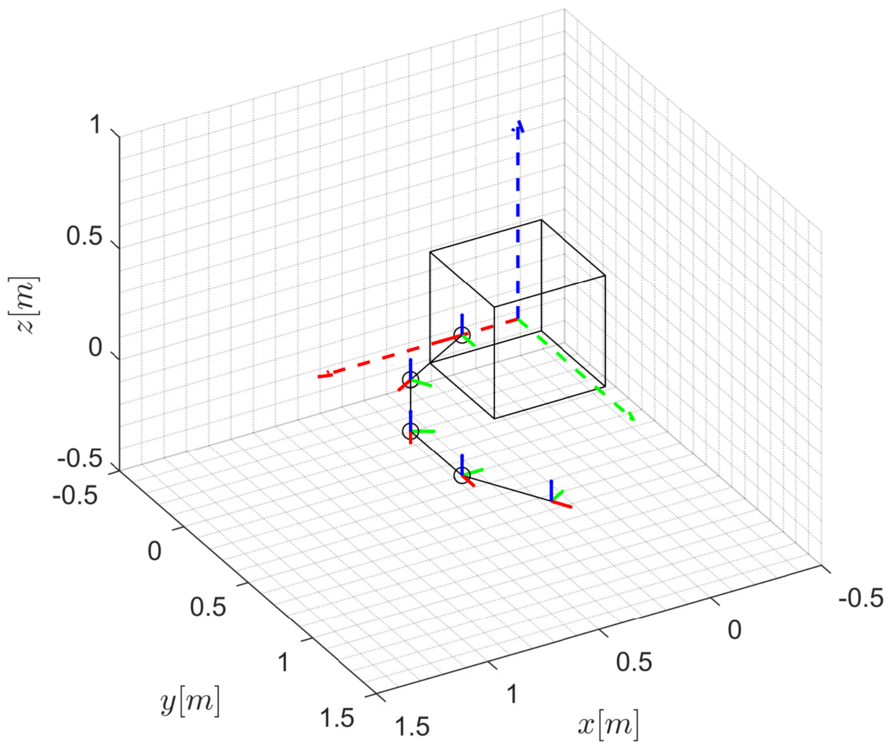


Figure 4.1: 3D Model of the SMS

The key physical features, such as the mass, the dimensions, and the inertia of both the chaser satellite base and of the robotic of the manipulator arm are resumed in table 4.1. The thickness for the manipulator arms are not considered, and the inertia is given only for the out-of-plane component, as it is the only component that will feature in the equations of motion for the planar case.

Parameter	**Value**
Base Dimension $[l_{B,x}\ l_{B,y}]\ [m]$	[0.5 0.5]
Mass $m_B\ [kg]$	10
Inertia $I_{B,zz}\ \left[\frac{kg}{m^2}\right]$	0.8333
Links Length $l_L\ [m]$	[0.4 0.4 0.4 0.4]
Links Masses $m_L\ [kg]$	[2 2 2 2]
Links Inertias $I_{L,zz}\ \left[\frac{kg}{m^2}\right]$	[0.0267 0.0267 0.0267 0.0267]

Table 4.1: Space Manipulator System Physical Parameters

The first joint is located on the center of the positive x-body axis face of the cube.
The centers of mass of each link are assumed to be at the respective link midpoint, while the end-effector is coincident with the last link's endpoint. The orientation of the end-effector will be the same as the last link's center of mass. Their values are presented in table 4.2.

Parameter	**Value**
Joint 1 Base Frame Coordinate $\mathbf{x}_0^B\ [m]$	[0.25 0]
Links Centers of Mass Location $\mathbf{x}_{G,i}^{i-1}\ [m]$	$\left[\frac{l_{L,i}}{2}\ 0\right]$
End-Effector Position $\mathbf{x}_{EE}^3\ [m]$	$[l_{L,4}\ 0]$
End-Effector Orientation $\theta_{EE}^3\ [rad]$	0

Table 4.2: Space Manipulator System Configuration Parameters

Physical joint limits are defined in terms of joint angular position, velocity, and acceleration. The first joint has a reduced joint rotation span compared the latter joints in the kinematic chain to guarantee that the first link never makes contact with the chaser base. The joint position, velocity and torque limits are presented in table 4.3 and they have been selected in order to allow the maximum movement for each of the manipulator joints while rendering effectively impossible for two consecutive links to collide, since the self-collisions of the joints of the robotic arm has not been considered in the obstacle avoidance.

Parameter	**Minimum**	**Maximum**
Joint Angular Position Limits $\boldsymbol{\theta}_{min}, \boldsymbol{\theta}_{MAX}\,[deg]$	$\begin{bmatrix}-85\\-175\\-175\\-175\end{bmatrix}$	$\begin{bmatrix}85\\175\\175\\175\end{bmatrix}$
Joints Velocity Limits $\dot{\boldsymbol{\theta}}_{min},\ \dot{\boldsymbol{\theta}}_{MAX}\ [deg]$	-30	30
Joint Acceleration Limits $\ddot{\boldsymbol{\theta}}_{min},\ \ddot{\boldsymbol{\theta}}_{MAX}\ \left[\frac{deg}{s^2}\right]$	-10	10

Table 4.3: Space Manipulator System Joint Limits

The target satellite's model is shown in fig. 4.2. It is important to note that, in contrast with fig. 4.1, the body axes shown here are rotated but follow the same color code: the x-axis is in red, the y-axis is in green and the z-axis is in blue. The presence of solar panels has been taken into account to be out-of-plane with respect to the plane in which the manipulator will move, as the z-axis of the target's body frame will be aligned with the one of the target-centered LVLH frame. This is a further requirement imposed to the close approach manoeuvre by the chaser satellite base. The grasping point position and orientation is highlighted in fig. 4.2.

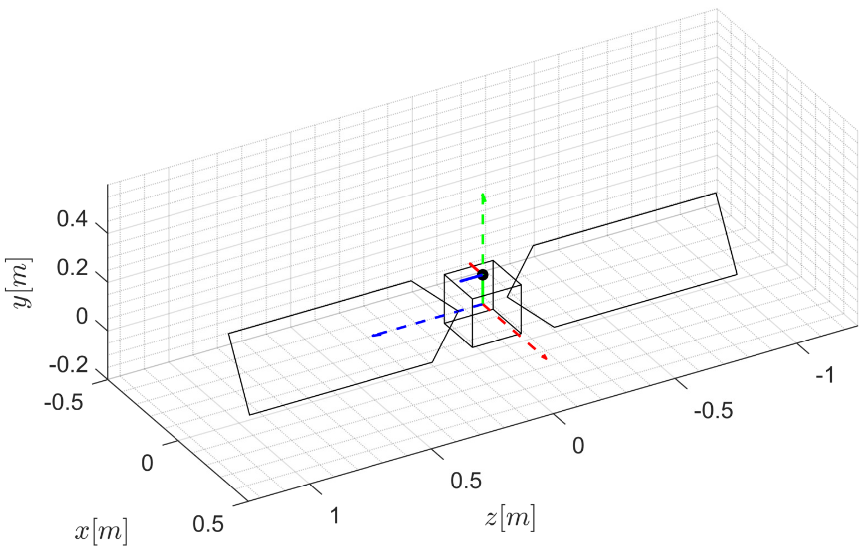


Figure 4.2: 3D Model of the Target Satellite

Parameter	Value
Base Dimension $[l_{T,x}\ l_{T,y}]\ [m]$	[0.2 0.2 0.2]
Mass $m_T\ [kg]$	1.25
Inertia $I_{T,zz}\ \left[\frac{kg}{m^2}\right]$	0.00833
Grasping Handle $[x_{GP}^{B,T}\ y_{GP}^{B,T}\ z_{GP}^{B,T}][m]$	[0 0.12 0]
Grasping Handle Orientation $R_{B,T}^{GP}$	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Table 4.4: Target Satellite Physical Parameters

4.1.2. Simulation Parameters

The capture scenario is described in the Local Vertical Local Horizontal reference frame centered on the target satellite. This reference frame, according to the hypotheses illustrated in section 2.2, is fixed in time and can be approximated to be an inertial reference frame.
The target satellite will be rotating with respect to its Local Vertical Local Horizontal reference frame, and the grasping point moves accordingly.
In this simplified environment, the target satellite has been considered to rotate at a constant angular velocity, but this feature will not affect the effectiveness of the algorithm, since it utilizes instantaneous values for the position and velocity of the grasping point at each discrete time instant t_k.

Parameters	Values
Target Position $\mathbf{x}_T\ [m]$	$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$
Target Rotation $\theta_T\ [rad]$	0
Target Velocity $\mathbf{v}_T\ \left[\frac{m}{s}\right]$	$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$
Target Angular Velocity $\omega_T\ \left[\frac{rad}{s}\right]$	$2 \cdot 10^{-2}$

Table 4.5: Target Satellite State Parameters

The initial state of the chaser satellite with respect to the target shows the manipulator in a semi-retracted position, and the joints to be initially still, to enforce the condition of zero initial linear and angular momentum for the whole Space Manipulator System.

	Parameters Values
Initial Base Position $\mathbf{x}_B$ $[m]$	$\begin{bmatrix}-1.1\\0\end{bmatrix}$
Initial Base Rotation θ_B $[deg]$	0
Initial Joint Position Vector $\boldsymbol{\theta}$ $[deg]$	$\begin{bmatrix}60\\-150\\0\\150\end{bmatrix}$
Initial Base Velocity $\mathbf{v}_B$ $\left[\frac{m}{s}\right]$	$\begin{bmatrix}0\\0\end{bmatrix}$
Initial Base Angular Velocity θ_B $\left[\frac{rad}{s}\right]$	0
Initial Joint Velocity $\dot{\boldsymbol{\theta}}$ $\left[\frac{rad}{s}\right]$	$\begin{bmatrix}0\\0\\0\\0\end{bmatrix}$

Table 4.6: Space Manipulator System State Parameters

The simulation scenario is shown in fig. 4.3, depicting the chaser satellite and the target satellite relative position and orientation at the initial time instant, highlighting the position of the grasping fixture on the target.

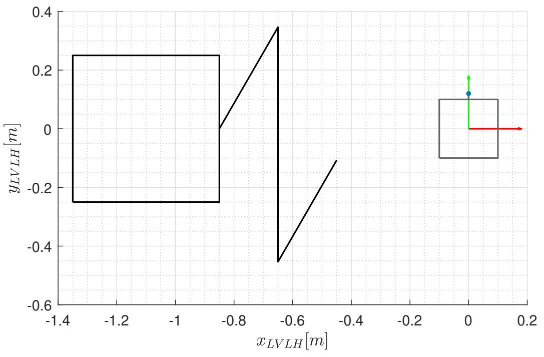


Figure 4.3: Simulation Initial Scenario

The choice for the parameters of the algorithm has been driven by the attempt to generate as many probable connections to the goal as possible, while limiting the computational time for the algorithm. Parameters for the RRT algorithm and for the Inverse Kinematics Algorithm are given in table 4.7 and table 4.8 respectively.

RRT	
Parameter	Value
# Iterations	10000
New node cost weights $\begin{bmatrix} w_x \\ w_\alpha \\ w_{\varphi_b} \\ w_t \end{bmatrix}$	$\begin{bmatrix} 1 \\ 1 \\ 1 \\ 10^{-3} \end{bmatrix}$
Connect-to-goal threshold d_{GC}	0.3
Connect-to-goal weights $\begin{bmatrix} w_x^{GC} \\ w_\varphi^{GC} \\ w_{\varphi_b}^{GC} \end{bmatrix}$	$\begin{bmatrix} 1 \\ \frac{0.2}{\pi} \\ 0 \end{bmatrix}$
Near Search Radius $\bar{\theta}$	0.5

Table 4.7: RRT Simulation Parameters

IK	
Parameter	Value
Maximum # Iterations	10000
Δt	0.001 s
Gain matrix for the end-effector task K	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
Gain matrix for the attitude task K_c	$\begin{bmatrix} 1 \end{bmatrix}$
Final Error Tolerance for the position e_x	10^{-3}
Final Error Tolerance for the orientation e_φ	1°

Table 4.8: IK Simulation Parameters

The parameters for the Inverse Kinematics are defined in order to have a final completion manoeuvre lasting a short amount of time (maximum $t = 10$ seconds), while the matrices values are kept low in order to obtain joint velocities that respect the joint limits as solutions.

4.2. Results

The results of the path planning algorithm are hereby displayed: first, a trajectory that could be obtained with the algorithm is shown in terms of end-effector and manipulator motion in time, including the time evolution of joint angles, velocity, and accelerations compared to the joint limits.
Then, the joint space exploration and the operational space exploration are analyzed, showing that the tree structure grows fast towards unexplored areas and that it is able to cover densely the space even with a small amount of nodes, and thus in a short time.
Finally, the quality of the generated trajectories with differing numbers of iterations is discussed.
One trajectory generated by the algorithm is shown in fig. 4.4 in terms of end-effector trajectory and chaser-manipulator configurations at the various nodes and throughout the inverse kinematics solution.

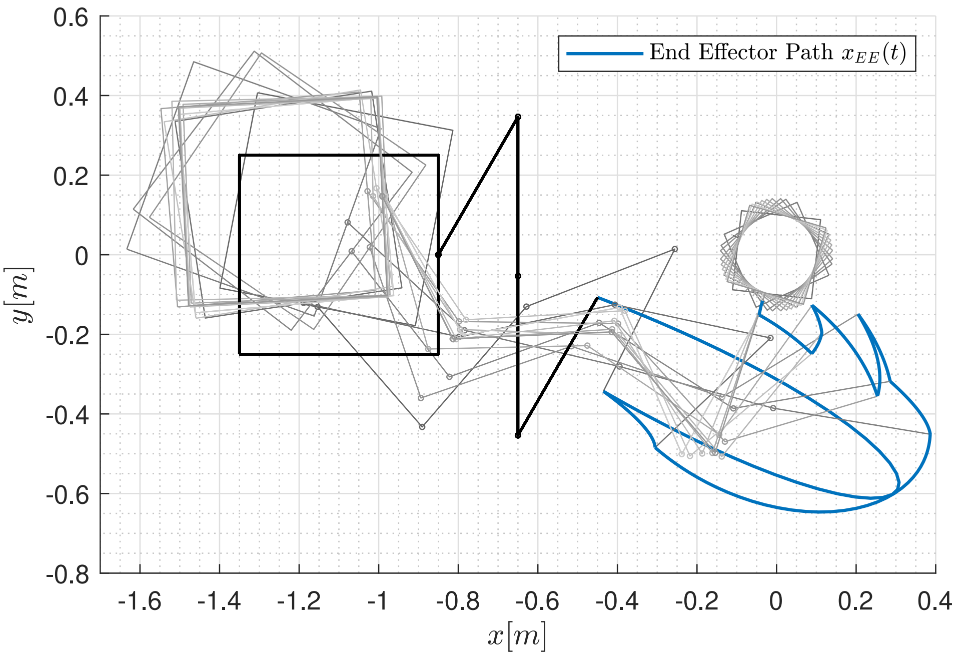


Figure 4.4: Manoeuvre

The joint rotations, velocities and accelerations are computed for the trajectory which has been interpolated through splines, and their trajectories over time are shown in fig. 4.5,fig. 4.6,fig. 4.7 and fig. 4.8.

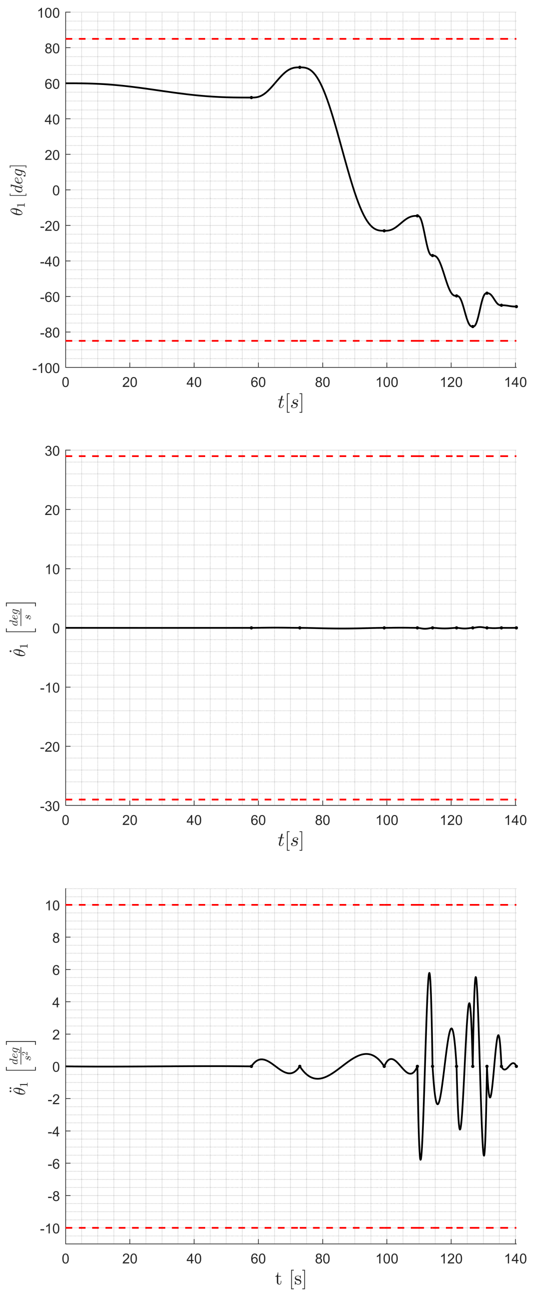


Figure 4.5: Joint 1

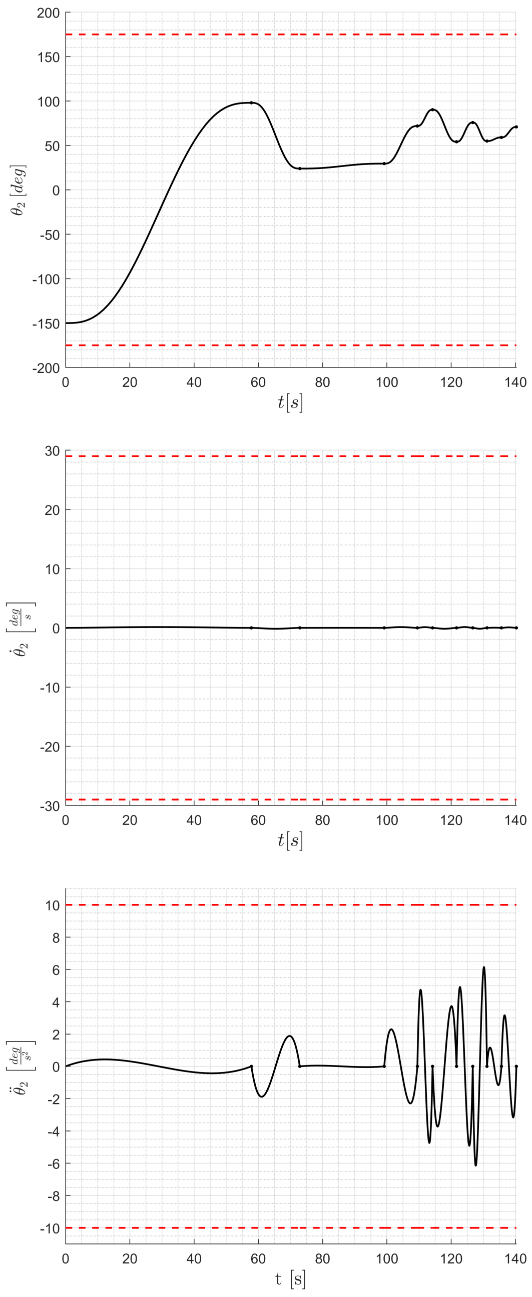


Figure 4.6: Joint 2

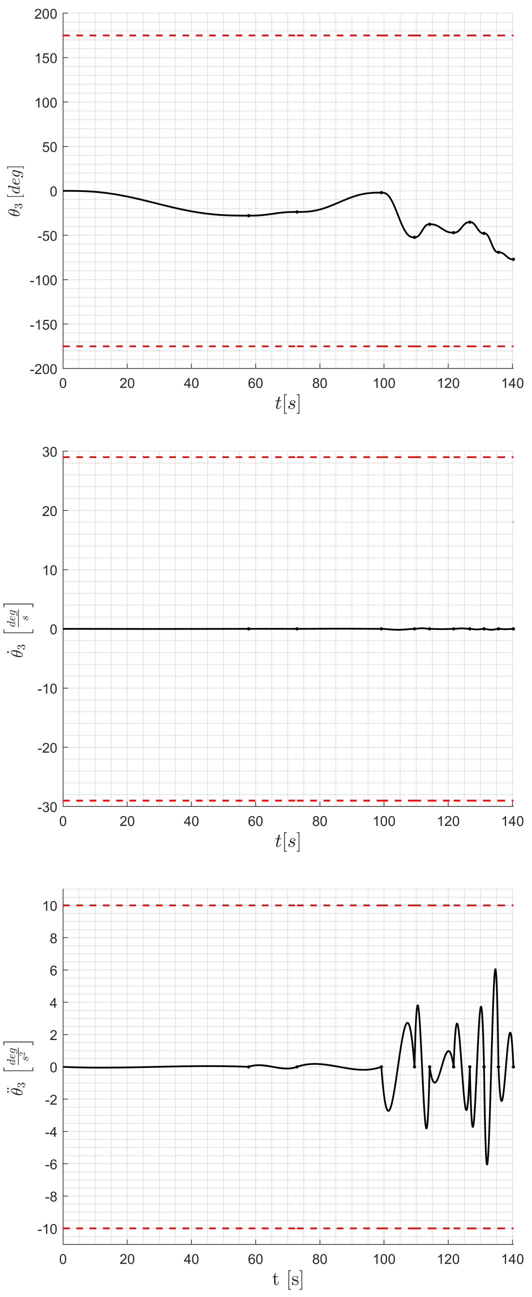


Figure 4.7: Joint 3

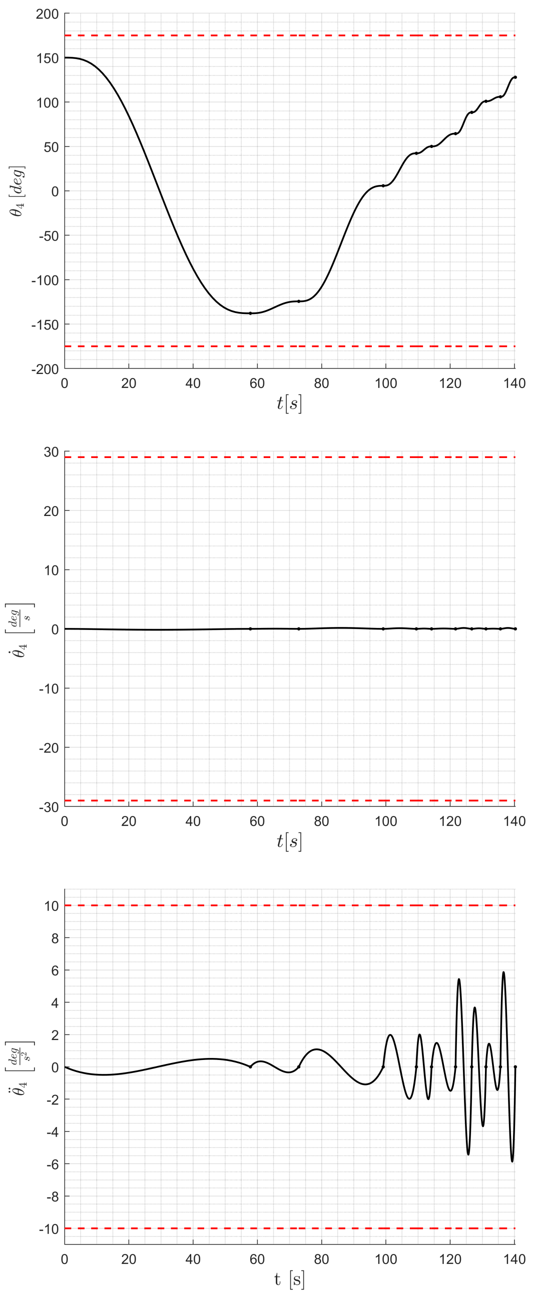


Figure 4.8: Joint 4

The results clearly indicate that there are no violations of the constraints of either joint variable throughout the path computed through the RRT. Moreover, the joint velocities are kept at very low values throughout the path, whereas joint acceleration feature low values for the initial nodes and increase for the successive nodes.
This is mainly due to the way in which they are generated: in fact, initial nodes are generated through paths of longer duration, with a constant maximum acceleration that has been applied for a short amount of time in comparison to the time required to reach the node. As the joint space is explored more densely, the paths found between two

nodes become shorter, and thus the acceleration applied to generate them becomes more impactful.

Furthermore, fig. 4.4 shows that the generated path does not collide with the target satellite along the trajectory.

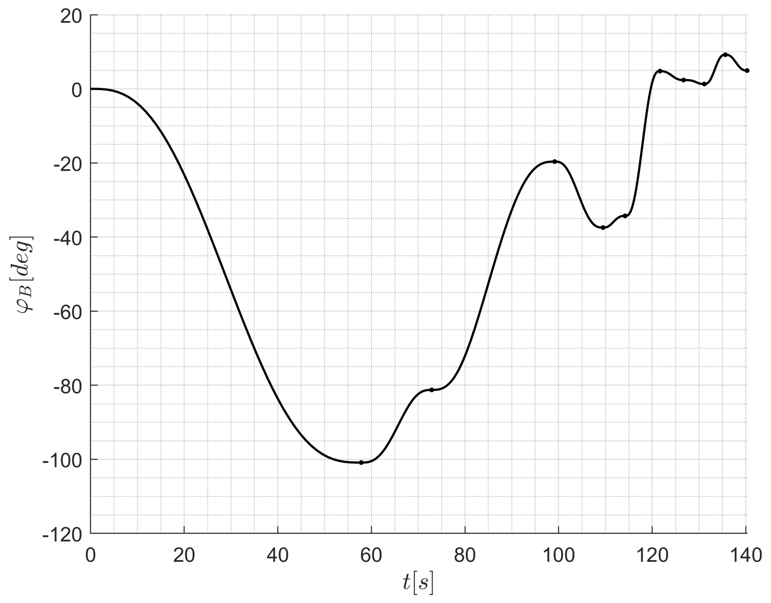


Figure 4.9: Chaser Base Rotation

As stated previously, the condition on the chaser base rotation is not to maintain a precise Earth pointing throughout the manoeuvre duration, but rather to communicate with Earth at the start of manoeuvre and at the capture instant.

The behaviour of the base rotation coordinate φ_B is shown in fig. 4.9, and it approaches the zero value at the end of the trajectory described by the RRT.

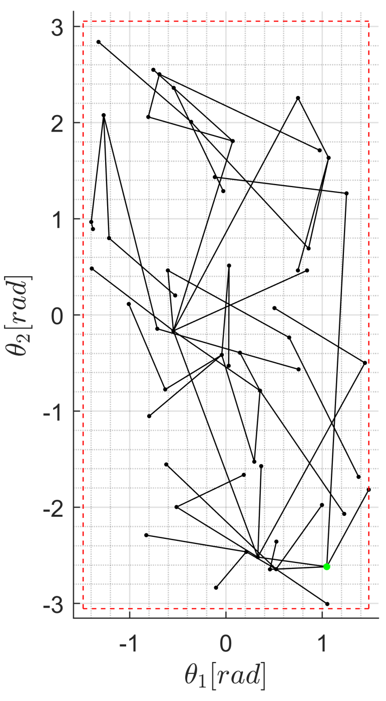
The final error on the end-effector position and orientation, the time of completion of the manoeuvre and the final base rotation are given in table 4.9.

Parameter	**Value**
$x_{EE}[m]$	$\begin{bmatrix} -0.0346 \\ -0.1159 \end{bmatrix}$
$\phi_{EE}[deg]$	72.5953
$\phi_b[deg]$	2.9538
$e_x[m]$	$9.94 \cdot 10^{-4}$
$e_\phi[deg]$	0.8985
t_f	$2m37s$

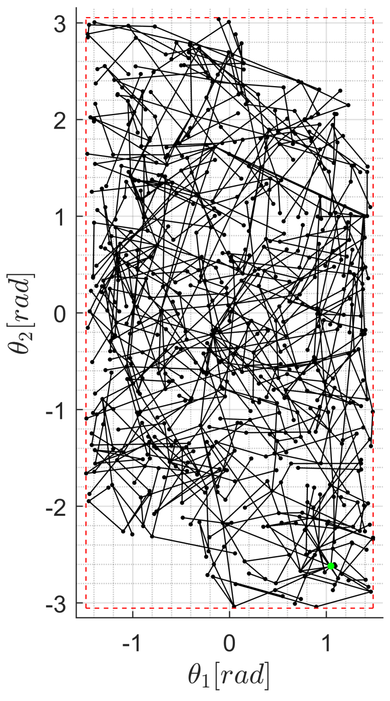
Table 4.9: End Conditions

The manoeuvre completion time is below 10 minutes, which is the typical threshold above which the assumptions on the orbital mechanics model are considered no longer valid.

The proposed algorithm features a rapid exploration of the joint space even with a low number of nodes, whereas typically RRTs generate nodes in the proximity of the tree and expand the tree structure slowly towards unexplored areas.
The following figures show the combination of the joint variables as the tree progresses its path, connecting the nodes through simple line edges rather than through the trajectory that has been defined among two nodes. The starting node of the tree, relative to the inital position of the joints for the Space Manipulator System, has been highlighted in green.



(a) 100 Iterations

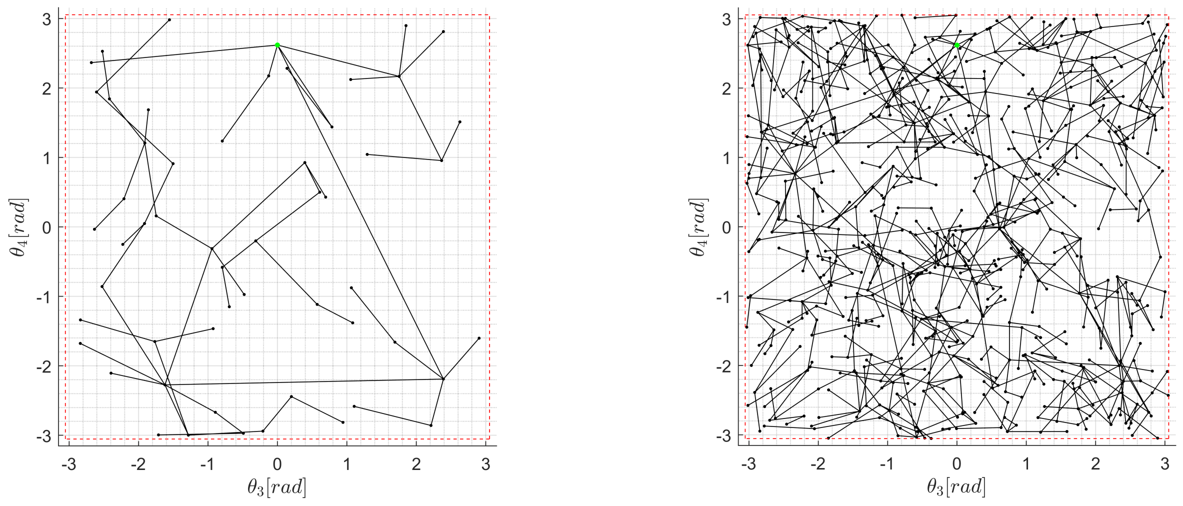


(b) 1000 Iterations

Figure 4.10: Generated Tree: First and Second Joint

In fig. 4.10a the propagation of the tree for the first and second joint is shown.
Even with a low amount of vertices, the explored combinations of joints cover a wide range of the possible combination, then, as the number of iterations progresses, the new nodes explore the state space more densely and generate new paths towards the areas with a lesser amount of nodes.
These figures feature an unexplored area, corresponding to a combination of high mag-

nitude values for both joint angles, which correspond to the manipulator configurations that display an impact with the chaser base.



(a) 100 Iterations

(b) 1000 Iterations

Figure 4.11: Generated Tree: Third and Fourth Joint

The latter two joints feature no forbidden areas within their limits, and similarly to the previous case, show a fast exploration of the state combinations.

The proposed approach generates multiple possible trajectories, that are then evaluated based on the success of the Inverse Kinematics algorithm, on the violation of the joint limits, on the eventual impacts with the target, and on the final rotation of the chaser satellite base.
The obtained trajectories that resulted in an attempted goal connection are thus classified into categories:

1. *Failed connections*: These trajectories reached the proximity of the grasping point location, and the Inverse Kinematics Algorithm attempted a connection to the goal. However, the algorithm could not establish a successful path to the goal by limiting the error under the tolerance values. This is mainly due to major initial condition differences between the node from which the connection was attempted and the grasping point location.

2. *Constraints violations*: The trajectories belonging to this set reached the goal condition effectively, but through the path generated by the Inverse Kinematics Algorithm, since no joint constraints are applied, one or multiple violations have been detected. These paths are discarded as there is no guarantee that the SMS could follow them.

3. *Target Collision*: These paths reached the goal conditions but, during the iterations of the Inverse Kinematics Algorithm, one or multiple collision with the target satellite have been detected.

4. *Valid Paths*: These trajectories successfully reached the goal location while being compliant with the path constraints on the joint variables and not impacting the target satellite or other obstacles in the workspace in the path generated by the Inverse Kinematics Algorithm.

5. *Optimal Paths*: The trajectories belonging to this set reached the goal condition without impacting the target or violating joint constraints, and moreover these paths feature a final rotation of the base $\varphi_b < 5°$, that requires a very low subsequent Attitude and Orbital Control System effort to restore the attitude for telecommunications or other operations.

The number of iterations for the algorithm coincides with the amount of new nodes that are attempted to be added to the tree.
A higher amount of nodes implies a better exploration of the joint space, a higher number of available connections to the goal, and a larger sample of states for each Θ_{Near} to find an optimized path.

1000 Iterations	
Attempted Connections	19
Failed Connections	9 (47.37%)
Constraint Violations	7 (36.84%)
Target Collision	0 (0.00%)
Valid Paths	3 (15.79%)
Optimal Paths	2 (10.53 %)

Table 4.10: Path Categories: n = 1000

2500 Iterations	
Attempted Connections	61
Failed Connections	34 (55.74%)
Constraint Violations	20 (32.79%)
Target Collision	2 (3.28%)
Valid Paths	5 (8.20%)
Optimal Paths	5 (8.20%)

Table 4.11: Path Categories: n = 2500

5000 Iterations	
Attempted Connections	117
Failed Connections	57 (48.72%)
Constraint Violations	52 (44.44%)
Target Collision	2 (1.71%)
Valid Paths	6 (5.13%)
Optimal Paths	5 (4.27%)

Table 4.12: Path Categories: n = 5000

10000 Iterations	
Attempted Connections	179
Failed Connections	116 (64.80%)
Constraint Violations	44 (24.58%)
Target Collision	0 (0.00%)
Valid Paths	19 (10.61%)
Optimal Paths	14 (7.82%)

Table 4.13: Path Categories: n = 10000

The path categories amount for different numbers of iterations are shown in table 4.10,

table 4.11, table 4.12 and table 4.13.
The amount of goal connections increases with the number of nodes, but the amount of successful paths towards the goal does not show a definite behaviour that would suggest using a high number of nodes to be generated.
However, even though an excessive number of iterations is deemed unnecessary, the amount of optimal paths for fewer iterations is very low, and due to the random nature of the generation of the nodes, this might coincide with an overall failure of the algorithm in finding an optimal path.

5 | Conclusions

The main goal of this work was to formulate a path planning approach for a Space Manipulator System that could generate a trajectory towards the grasping handle of a target satellite while avoiding collisions with obstacles and restoring the initial attitude of the chaser satellite base.
The path planning formulation is based on a common sampling-based approach for robots in the Rapidly Exploring Random Trees, combined with obstacle avoidance and goal connection methods that have already been proposed and validated in literature.
The proposed approach succeeded in finding multiple trajectories that accomplished the goals of the mission while being compliant with the limits for the joint variables of the manipulator, avoding obstacles in the manipulator workspace, under certain assumptions on the environment.
In order to validate and expand the algorithm, it is necessary to drop some hypotheses and add features that would increase the success rate of the algorithm:

- *3D Simulation*: The work hereby presented shows the possibility of solution of the trajectory planning problem for a planar case typical scenario. The implementation of a 3D simulation would better clarify the applicability to real cases.
- *Add rewiring*: Typically, RRT* algorithms converge to an optimal solution when the number of iterations increases. The presented approach does not converge to an optimal solution given that the rewiring step, checking whether a new tree node can be employed as an intermediate node towards another tree node, lowering the path cost, is not performed. This step can be performed through the solution of a 2-Point Boundary Value Problem.
- *Add Other Minimization Parameters*: This approach computed the cost of each of the generated nodes including the distance to the goal and the rotation of the base. Other useful variables, such as the manipulability index, can be added within the cost function of each node. Additionally, the distance from the obstacles can be included to add robustness to obstacle modelling errors.

Bibliography

[1] F. Aghili. Coordination control of a free-flying manipulator and its base attitude to capture and detumble a noncooperative satellite. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2365–2372. IEEE, 2009.

[2] J. R. Benevides and V. Grassi. Autonomous path planning of free-floating manipulators using rrt-based algorithms. In *2015 12th Latin American Robotics Symposium and 2015 3rd Brazilian Symposium on Robotics (LARS-SBR)*, pages 139–144. IEEE, 2015.

[3] F. Caccavale and B. Siciliano. Kinematic control of redundant free-floating robotic systems. *Advanced robotics*, 15(4):429–448, 2001.

[4] W. K. Chung and Y. Xu. Path planning algorithm for space manipulator with a minimum energy demand. In *2012 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 1556–1563. IEEE, 2012.

[5] S. Cocuzza, I. Pretto, and S. Debei. Reaction torque control of redundant space robotic systems for orbital maintenance and simulated microgravity tests. *Acta Astronautica*, 67(3-4):285–295, 2010.

[6] J. J. Craig. *Introduction to robotics*. Pearson Educacion, 2006.

[7] S. Dubowsky and E. Papadopoulos. The kinematics, dynamics, and control of free-flying and free-floating space robotic systems. *IEEE Transactions on robotics and automation*, 9(5):531–543, 1993.

[8] S. Dubowsky and M. A. Torres. Path planning for space manipulators to minimize spacecraft attitude disturbances. In *Proceedings of IEEE international conference on robotics and automation*, volume 3, pages 2522–2528. Citeseer, 1991.

[9] A. Flores-Abad, O. Ma, K. Pham, and S. Ulrich. A review of space robotics technologies for on-orbit servicing. *Progress in aerospace sciences*, 68:1–26, 2014.

[10] A. Flores-Abad, L. Zhang, Z. Wei, and O. Ma. Optimal capture of a tumbling object

in orbit using a space manipulator. *Journal of Intelligent & Robotic Systems*, 86: 199–211, 2017.

[11] A. M. Giordano, D. Calzolari, and A. Albu-Schäffer. Workspace fixation for free-floating space robot operations. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 889–896. IEEE, 2018.

[12] C. Ju, Q. Luo, and X. Yan. Path planning using artificial potential field method and a-star fusion algorithm. In *2020 global reliability and prognostics and health management (PHM-Shanghai)*, pages 1–7. IEEE, 2020.

[13] S. Karaman and E. Frazzoli. Incremental sampling-based algorithms for optimal motion planning. *Robotics Science and Systems VI*, 104(2), 2010.

[14] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *The international journal of robotics research*, 30(7):846–894, 2011.

[15] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. In *Proceedings. 1985 IEEE international conference on robotics and automation*, volume 2, pages 500–505. IEEE, 1985.

[16] J. J. Kuffner and S. M. LaValle. Rrt-connect: An efficient approach to single-query path planning. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 2, pages 995–1001. IEEE, 2000.

[17] R. Lampariello. Motion planning for the on-orbit grasping of a non-cooperative target satellite with collision avoidance. *i-SAIRAS 2010*, 2010.

[18] R. Lampariello. On grasping a tumbling debris object with a free-flying robot. *IFAC Proceedings Volumes*, 46(19):161–166, 2013.

[19] S. M. LaValle and J. J. Kuffner Jr. Randomized kinodynamic planning. *The international journal of robotics research*, 20(5):378–400, 2001.

[20] W.-J. Li, D.-Y. Cheng, X.-G. Liu, Y.-B. Wang, W.-H. Shi, Z.-X. Tang, F. Gao, F.-M. Zeng, H.-Y. Chai, W.-B. Luo, et al. On-orbit service (oos) of spacecraft: A review of engineering developments. *Progress in Aerospace Sciences*, 108:32–120, 2019.

[21] Y. Li, D. Li, W. Zhu, J. Sun, X. Zhang, and S. Li. Constrained motion planning of 7-dof space manipulator via deep reinforcement learning combined with artificial potential field. *Aerospace*, 9(3):163, 2022.

[22] S. Liu, Q. Zhang, and D. Zhou. Obstacle avoidance path planning of space manipu-

lator based on improved artificial potential field method. *Journal of The Institution of Engineers (India): Series C*, 95:31–39, 2014.

[23] Y. Masutani and F. Miyazaki. Sensory feedback control for space manipulators. *Journal of the Robotics Society of Japan*, 7(6):647–655, 1989.

[24] Z. Mu, W. Xu, X. Gao, L. Xue, and C. Li. Obstacles modeling and collision detection of space robots for performing on-orbit services. In *2014 4th IEEE International Conference on Information Science and Technology*, pages 461–466. IEEE, 2014.

[25] Z. Mu, W. Xu, and B. Liang. Avoidance of multiple moving obstacles during active debris removal using a redundant space manipulator. *International Journal of Control, Automation and Systems*, 15(2):815–826, 2017.

[26] K. Nanos and E. Papadopoulos. On the use of free-floating space robots in the presence of angular momentum. *Intelligent Service Robotics*, 4:3–15, 2011.

[27] D. Nenchev, Y. Umetani, and K. Yoshida. Analysis of a redundant free-flying spacecraft/manipulator system. *IEEE Transactions on Robotics and Automation*, 8(1): 1–6, 1992.

[28] D. N. Nenchev. Reaction null space of a multibody system with applications in robotics. *Mechanical Sciences*, 4(1):97–112, 2013.

[29] D. N. Nenchev, K. Yoshida, P. Vichitkulsawat, and M. Uchiyama. Reaction null-space control of flexible structure mounted manipulator systems. *IEEE Transactions on Robotics and Automation*, 15(6):1011–1023, 1999.

[30] S.-I. Nishida and S. Kawamoto. Strategy for capturing of a tumbling space debris. *Acta Astronautica*, 68(1-2):113–120, 2011.

[31] M. Oda and Y. Ohkami. Coordinated control of spacecraft attitude and space manipulators. *Control Engineering Practice*, 5(1):11–21, 1997.

[32] E. Papadopoulos and S. Dubowsky. Coordinated manipulator/spacecraft motion control for space robotic systems. In *ICRA*, pages 1696–1701, 1991.

[33] E. Papadopoulos, F. Aghili, O. Ma, and R. Lampariello. Robotic manipulation and capture in space: A survey. *Frontiers in Robotics and AI*, page 228, 2021.

[34] S. M. Persson and I. Sharf. Sampling-based a* algorithm for robot path-planning. *The International Journal of Robotics Research*, 33(13):1683–1708, 2014.

[35] T. Rybus. Obstacle avoidance in space robotics: Review of major challenges and proposed solutions. *Progress in Aerospace Sciences*, 101:31–48, 2018.

[36] T. Rybus. Point-to-point motion planning of a free-floating space manipulator using the rapidly-exploring random trees (rrt) method. *Robotica*, 38(6):957–982, 2020.

[37] T. Rybus and K. Seweryn. Application of rapidly-exploring random trees (rrt) algorithm for trajectory planning of free-floating space manipulator. In *2015 10th International Workshop on Robot Motion and Control (RoMoCo)*, pages 91–96. IEEE, 2015.

[38] T. Rybus, J. Prokopczuk, M. Wojtunik, K. Aleksiejuk, and J. Musiał. Application of bidirectional rapidly exploring random trees (birrt) algorithm for collision-free trajectory planning of free-floating space manipulator. *Robotica*, 40(12):4326–4357, 2022.

[39] B. Siciliano, O. Khatib, and T. Kröger. *Springer handbook of robotics*, volume 200. Springer, 2008.

[40] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. *Robotics: Modelling, Planning and Control*. Springer, 2009.

[41] S. Stoneman and R. Lampariello. Embedding nonlinear optimization in rrt for optimal kinodynamic planning. In *53rd IEEE Conference on Decision and Control*, pages 3737–3744. IEEE, 2014.

[42] Y. Umetani, K. Yoshida, et al. Resolved motion rate control of space manipulators with generalized jacobian matrix. *IEEE Transactions on robotics and automation*, 5 (3):303–314, 1989.

[43] Z. Vafa. Space manipulator motions with no satellite attitude disturbances. In *Proceedings., IEEE International Conference on Robotics and Automation*, pages 1770–1775. IEEE, 1990.

[44] D. J. Webb and J. Van Den Berg. Kinodynamic rrt*: Asymptotically optimal motion planning for robots with linear dynamics. In *2013 IEEE international conference on robotics and automation*, pages 5054–5061. IEEE, 2013.

[45] M. Wilde, S. Kwok Choon, A. Grompone, and M. Romano. Equations of motion of free-floating spacecraft-manipulator systems: an engineer's tutorial. *Frontiers in Robotics and AI*, 5:41, 2018.

[46] Z. Xie, X. Zhao, Z. Jiang, H. Yang, and C. Li. Trajectory planning and base attitude restoration of dual-arm free-floating space robot by enhanced bidirectional approach. *Frontiers of Mechanical Engineering*, 17(1):2, 2022.

[47] W. Xu, B. Liang, C. Li, and Y. Xu. Autonomous rendezvous and robotic capturing of non-cooperative target in space. *Robotica*, 28(5):705–718, 2010.

[48] W. Xu, Y. Liu, B. Liang, X. Wang, and Y. Xu. Unified multi-domain modelling and simulation of space robot for capturing a moving target. *Multibody System Dynamics*, 23:293–331, 2010.

[49] Y. Xu and T. Kanade. *Space robotics: dynamics and control*, volume 188. Springer Science & Business Media, 1992.

[50] Y. Yanoshita and S. Tsuda. Space robot path planning for collision avoidance. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 2. Citeseer, 2009.

[51] K. Yoshida and D. N. Nenchev. A general formulation of under-actuated manipulator systems. In *Robotics Research: The Eighth International Symposium*, pages 33–44. Springer, 1998.

[52] M. Yu, J. Luo, M. Wang, and D. Gao. Spline-rrt$*$: Coordinated motion planning of dual-arm space robot. *IFAC-PapersOnLine*, 53(2):9820–9825, 2020.

[53] L. Zong, J. Luo, M. Wang, and J. Yuan. Obstacle avoidance handling and mixed integer predictive control for space robots. *Advances in Space Research*, 61(8):1997–2009, 2018.

[54] L. Zong, M. R. Emami, and J. Luo. Reactionless control of free-floating space manipulators. *IEEE Transactions on Aerospace and Electronic Systems*, 56(2):1490–1503, 2019.

List of Figures

List of Tables