DESIGN OF A VERSATILE SPACE MANIPULATOR FUNCTIONAL ENGINEERING SIMULATOR FOR IOS

Lorenzo Capra[1] **and Michèle Lavagna**[2]

[1]*Politecnico di Milano, Italy, lorenzo.capra@polimi.it*
[2]*Politecnico di Milano, Italy, michelle.lavagna@polimi.it*

ABSTRACT

The work discusses the rising interest in In Orbit Servicing activities for repairing and maneuvering satellites in space, and an approach to simulate these scenarios. A simulator is created to address the challenges of designing and analyzing space manipulators systems for these tasks. The simulator uses a comprehensive approach, integrating mechanical engineering, robotics, control systems, and simulations. It's built using Simulink, MATLAB's library SPART, and Simscape Multibody, and considers complex interactions, including orbital motion, flexibility dynamics, joint constraints, external forces, and contact dynamics. It also features a guidance and control block for closed-loop simulations. The simulator's flexibility allows for an easy integration of new components and algorithms. It's validated through various tasks and scenarios, aiding in understanding manipulator behavior in simple benchmark problems. The simulator also offers rendering and visualization capabilities through the Simscape Multibody features. The development of a versatile functional engineering simulator aims to provide a powerful tool for the analysis and evaluation of space manipulators in in-orbit servicing scenarios.

Key words: In Orbit Servicing; Simulator; Space Manipulator.

1. INTRODUCTION

The significance of in-orbit servicing (IOS) technologies and methodologies within the realm of satellite and space system development has remarkably grown in recent years. Central to the success of IOS missions is the precise control and manipulation of spaceborne robotic systems, often referred to as space manipulators.
To effectively address the complex challenges associated with designing and analyzing space manipulators for in-orbit servicing scenarios, a versatile and functional engineering simulator is introduced, to assess the performance of these systems. The development of this simulator hinges on a multidisciplinary approach, merging fundamental principles from mechanical engineering, robotics, control systems, and simulation technologies. An engineering simulator is essential for studying robotics in orbit servicing scenarios as it enables the accurate emulation of complex space environments, intricate mechanical interactions, and dynamic orbital conditions, providing a controlled and risk-free platform for testing and refining robotic systems and strategies.
The recent push towards IOS activities by leading space agencies, and more specifically the interest in space robotics, has developed in a multitude of studies and many missions on the subject are in plan. The e.Deorbit program by ESA focused on reducing space debris by actively removing a defunct satellite using specialized robotic spacecraft [1]. It aimed to safely rendezvous with a target satellite, attach to it, and perform a controlled deorbit maneuver to burn up both spacecrafts upon reentry into Earth's atmosphere. Following the same ethos, ESA is presently funding the ClearSpace-1 mission, which employs a concept of caging capture involving several robotic arms that encircle the target and capture it. The ESA-supported COMRADE study instead, investigated different control strategies with the support of a simulator to simulate active debris removal (ADR) and refuelling scenarios [2].
This work presents the implementation of a general-purpose simulator for robotics IOS activities, highlighting all its components and testing its effectiveness against a common mission operating phase, that is motion synchronization with the target.

2. SIMULATOR STRUCTURE

This section introduce to the structure of the simulator, which is presented in Fig. 1. Notably from the block diagram the simulator can be divided in two main parts: the first is related to the space manipulator modeling and it encompass its kinematics, dynamics and the equations of motion solver; the second is related to the guidance, navigation & control (GNC) of this multi-body system. The scheme in Fig. 1 reports just the main skeleton of the functional engineering simulator, which is then augmented by several features which are presented later in Sec. 2.1. A detailed description of the blocks in the simulator scheme is now provided.

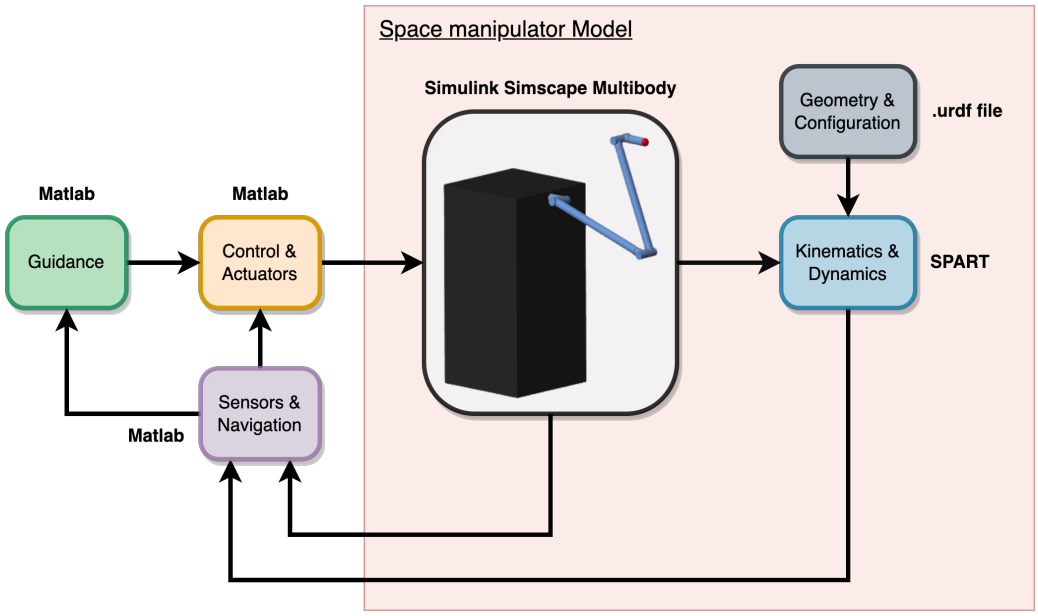

Figure 1. Space manipulator simulator architecture.

Geometry & Configuration. The first block is used to initialize the configuration of the kinematic tree that is the space manipulator. It entails a URDF (Unified Robot Description Format) file, which follows XML specifications, storing in a data structure fashion the characteristics of the multi-body system. These information are then parsed and rearranged. The file is written as a series of dedicated tags for physical features of the robotic model, defining the geometry, the global configuration of the overall system and the connection order between different parts of the system. The mass, inertia, dimensions and orientation of each element are specified here.

Kinematics & Dynamics. The MATLAB library SPART (SPAce Robotics Toolkit) [3] is employed to retrieve the kinematics and dynamics parameters through a direct-path approach [4], which considers the spacecraft centre of mass as reference, and recursive algorithms. It is a modeling and control software package for mobile-base robotic multi-body systems, based on a Newton-Euler approach, which makes use of the Decoupled Natural Orthogonal matrix to obtain the generalized inertia matrix (GIM) and the convective inertia matrix (CIM) in efficient and recursive fashion. The complete description of the spacecraft-manipulator system's state involves the joint variables q along with their first and second time derivatives, $\dot{q}$ and $\ddot{q}$, respectively. Notably, the generalized joint variables q encompass not only the base-spacecraft's position and attitude, q_0, but also the manipulator's joint states, q_m, resulting in $q = [q_0, q_m]$. The equation of motion can be concisely represented as Eq. 1, where the τ vector signifies the generalized forces exerted on the system within joint space. Here, H represents the generalized inertia matrix, while C corresponds to the generalized matrix of convective inertia, comprising influences such as Coriolis and centrifugal forces.

$$H(q)\ddot{q} + C(q,\dot{q})\dot{q} = \tau \tag{1}$$

Such equation can then be expanded to make the contributions of the base-spacecraft-link, of the manipulator and of their coupling explicit, as in Eq. 2.

$$\begin{bmatrix} H_0 & H_{0m} \\ H_{0m}^T & H_m \end{bmatrix} \begin{bmatrix} \ddot{q}_0 \\ \ddot{q}_m \end{bmatrix} + \begin{bmatrix} C_0 & C_{0m} \\ C_{m0} & C_m \end{bmatrix} \begin{bmatrix} \dot{q}_0 \\ \dot{q}_m \end{bmatrix} = \begin{bmatrix} \tau_0 \\ \tau_m \end{bmatrix} \tag{2}$$

Space Manipulator Model. To solve the equations of motion, a model of the space manipulator is implemented with the Simulink Sim- scape Multibody library, which provides a simulation environment for 3D mechanical systems, using blocks to represent bodies (links and joints), elements, con- nections, forces, and so on. This model is then validated against numerical integration of the equations of motion. Key features of the simulator include a versatile modelling framework that accommodates a diverse range of space manipulator architectures, allowing for the representation of various configurations and kinematic structures. The base space robot configuration consists of a 6-DOF base spacecraft equipped with a 7-DOF redundant robotic manipulator. From this, more complex topologies have been tested, to check the validity of the simulator and its independence from the robotic model uploaded. In particular multiple appendages and different end effector mechanical systems can be implemented and interchanged according to mission specifications.

GNC chain. The sensors reading of the state and derived measurements, together with the navigation estimation algorithms are provided, at the current stage of the

simulator implementation, by performance models that should reliably represent the errors on the data. The reconstructed state is then fed to the guidance and control blocks, which computes the desired space manipulator evolution in time and output the control actions entering the Simscape model. The modularity of the simulator architecture enables easy replacement or substitution of components, promoting flexibility, adaptability, and experimentation. This is a key factor in the simulator's ability to integrate new functionalities and incorporate/support different guidance and control algorithms, according to the scenario analyzed. The default control algorithm implemented is a state-of-the-art Computed Torque Control (CTC). It is a model-based feedback linearization controller, consisting of an inner nonlinear compensation loop and an outer loop with an exogenous control signal employing a proportional-derivative (PD) strategy. The output control expression is reported in Eq. 3, while a block diagram of how CTC works is in Fig. 2.

$$\tau = H(q)(\ddot{q} + K_d\dot{e}_q + K_p e_q) + C(q,\dot{q})\dot{q} \tag{3}$$

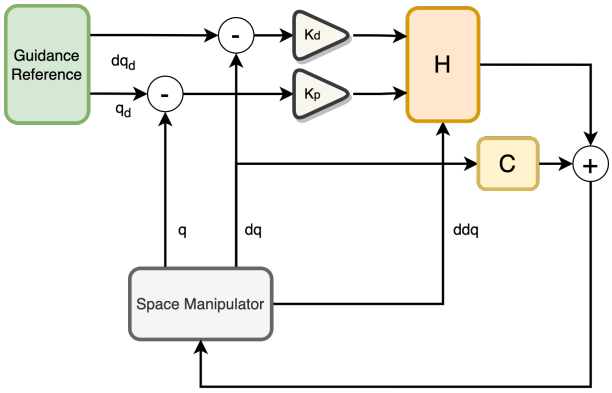

Figure 2. Computed Torque Control block diagram.

2.1. Simulator Features

The additional features of the simulator are now addressed. Relative orbital motion is explicitly considered as a crucial factor often overlooked in existing literature on space manipulator analysis. Recognizing the significance of this dynamic phenomenon, the simulator incorporates the relative orbital motion between the manipulator and its target, according to the nonlinear model expressed in cartesian space in Eq. 4.

$$\begin{cases} \ddot{x} - 2\dot{\theta}_0\dot{y} - \ddot{\theta}_0 y - \dot{\theta}^2 x = \frac{\mu(r_0+x)}{[(r_0+x)^2+y^2+z^2]^{\frac{3}{2}}} + \frac{\mu}{r_0^2} \\ \ddot{y} + 2\dot{\theta}_0\dot{x} + \ddot{\theta}_0 x - \dot{\theta}^2 y = \frac{\mu y}{[(r_0+x)^2+y^2+z^2]^{\frac{3}{2}}} \\ \ddot{z} = \frac{\mu z}{[(r_0+x)^2+y^2+z^2]^{\frac{3}{2}}} \end{cases} \tag{4}$$

This set of equations is completed by the expressions in Eq 5 and Eq. 6, to constitute a 10-dimensional system of nonlinear differential equations.

$$\ddot{r}_0 = r_0\dot{\theta}_0^2 - \frac{\mu}{r_0^2} \tag{5}$$

$$\ddot{\theta}_0 = -\frac{2\dot{r}_0\dot{\theta}_0}{r_0} \tag{6}$$

To clarify the meaning of a few variables: $\mu = 398600 km^3/s^2$ is the Earth's gravitational constant, θ is the true anomaly of the target orbit, r_0 is the radial distance of the target from Earth. By accurately capturing this relative motion, the simulator provides a more realistic and comprehensive representation of IOS scenarios, allowing for a deeper understanding of the challenges and complexities associated with manipulating objects in this dynamic space environment. This aspect is particularly relevant in the free-floating operating case, where the spacecraft is not controlled and subject to the acceleration field in Eq. 4, causing a simulation error in the order of centimeters, if neglected.

The capability to model the flexibility dynamics of the robotic arm is a valuable feature in the proposed simulator for analysing space manipulators in IOS scenarios. The main advantages are a more accurate and realistic representation on the manipulator's behaviour, which is essential for understanding the effects of flexibility on the system's performance, stability, and control; a detailed analysis of how the arm's mechanical deformations affect its motion, accuracy, and overall performance; the possibility to account for deviations from an ideal trajectory, because of the effect of the manipulator flexibility dynamics on the end effector path. This feature enhances the precision and reliability of the manipulator's movements during critical operations, such as docking, grasping, or manipulation tasks. Flexibility is included in the robotic manipulator Simscape model by modeling the links as beam elements capable of elastic deformation, including contributions due to extension, bending and torsion.

The simulator also allows to simulate and evaluate the effects several other aspects, like joint constraints, external forces, and contact dynamics. The latter is incorporated taking advantage of the Simscape's Spatial Contact Force block, which models the contact between geometries associated with a pair of bodies, by exporting the geometries of the elements as convex hull representations. Depending on the investigated scenario, this analysis provides valuable insights into operations like grasping, docking and collision avoidance to name a few. Finally, to facilitate a comprehensive understanding of the simulated scenarios, the simulator also offers rendering and visualization capabilities, providing visual feedback and enhancing the user's perception of the manipulator's behaviour and performance.

3. IOS EXAMPLE SCENARIO

The proposed simulator is adopted to analyze a common phase of IOS activities, that is motion synchronization. This operation focuses adjusting the spacecraft platform and robotic end effector relative position and orientation with respect to the target, so to be stationary as seen from the target point of capture perspective. That proper relative state acquisition at the end of the closing phase is crucial for the success of the following activities. The scenario is illustrated in Fig. 3, where a cylindrical shaped uncooperative target is randomly tumbling and the space manipulator end effector shall move inside the desired tracking pose expressed by the green sphere. To successfully perform this operation, the error pose in terms of position and orientation shall be inside the thresholds reported in Eq. 7

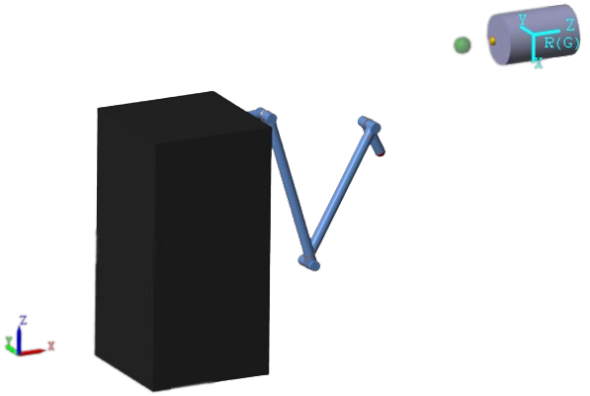

Figure 3. Motion synchronization example scenario.

$$e_{pos} < 5cm \tag{7a}$$

$$e_{\alpha} < 5^{\circ} \tag{7b}$$

The guidance algorithm employed in this case simply tracks the desired pose and compute the error of the current space manipulator state with respect to it. This is then passed to the CTC algorithm which generates at each time step the control action τ.
A Montecarlo analysis is performed, randomizing the initial conditions of the space manipulator, and the error evolution in terms of end effector cartesian position, divided in axial and normal, and attitude, is reported in Fig. 4. The controller is capable of tracking the desired position very well, with a reactive settling time. Also the attitude is tracked correctly, although some particular initial conditions have the space manipulator respond not as fast as compared to the position convergence.

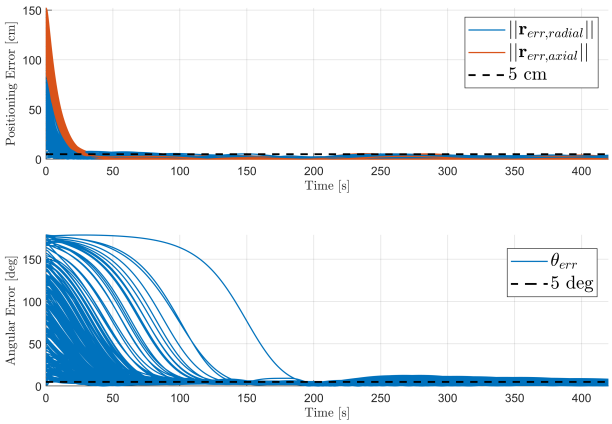

Figure 4. Motion synchronization position and attitude errors.

4. CONCLUSIONS

The development of a versatile functional engineering simulator aims to provide a powerful tool for the analysis and evaluation of space manipulators in in-orbit servicing scenarios. By leveraging its multidisciplinary capabilities and realistic simulation environment, the proposed simulator offers valuable insights into the performance, behaviour, and feasibility of manipulator systems, thereby supporting the advancement of in-orbit servicing technologies and missions in the ever-evolving space industry. Several useful features are implemented and tested to verify the validity of the simulator as a tool for the analysis of IOS activities, from simple benchmarks to more complex and demanding scenarios.

REFERENCES

[1] Jaekel S., Lampariello R., Rackl W., et al, 2018, Design and Operational Elements of the Robotic Subsystem for the e.deorbit Debris Removal Mission, Frontiers in Robotics and AI.

[2] Colmenarejo P., Branco J., Santos N., et al, 2018, Methods and outcomes of the COMRADE project - Design of robust Combined control for robotic spacecraft and manipulator in servicing missions: comparison between between Hinf and nonlinear Lyapunov-based approaches.

[3] Virgili-Llop J., Drew II J., Romano M., 2016, SPART SPAcecraft Robotics Toolkit: an Open-Source Simulator for Spacecraft Robotic Arm Dynamic Modeling And Control.

[4] Moosavian S., Papadopoulos E., 1998, On the kinematics of multiple manipulator space free-flyers and their computation, Journal of Robotic Systems.