DOI: 10.13009/EUCASS2025-540
11TH EUROPEAN CONFERENCE FOR AERONAUTICS AND SPACE SCIENCES (EUCASS)

EXPERIMENTAL EVALUATION OF ROBOTIC GRASPING STRATEGIES FOR ON-ORBIT SERVICING

Matteo Carattoli,[1] Marco Sabatini,[2] David Paolo Madonna,[3] Victor Joannic,[4] Mauro Pontani,[5] Paolo Gasbarri,[6]

[1]*Department of Mechanical and Aerospace Engineering, Sapienza University of Rome, Rome, Italy*
[2] *School of Aerospace Engineering, Sapienza University of Roma, Rome, Italy*
[3]*Department of Mechanical and Aerospace Engineering, Sapienza University of Rome, Rome, Italy*
[4]*Ecole de l'Air et de l'Espace, Salon-de-Provance, France*
[5]*Department of Astronautical, Electrical, and Energy Engineering, Sapienza University of Rome, Rome, Italy*
[6]*School of Aerospace Engineering, Sapienza University of Roma, Rome, Italy*
Marco.sabatini@uniroma1.it, https://sia.web.uniroma1.it

Abstract

On-orbit servicing (OOS) is a key technology for sustainable space operations, enabling satellite maintenance, refueling, and upgrades. This paper presents an experimental investigation of robotic grasping strategies using a six-DOF robotic arm and a granite-based planar testbed simulating microgravity. A custom end-effector and metrology system support accurate tracking and capture. An enhanced RRT-based trajectory planner ensures safe maneuvering. Experimental results confirm the feasibility of autonomous grasping, highlighting the system's strengths and limitations, and laying the groundwork for future extensions involving cooperative and uncooperative target capture in dynamic scenarios.

1. Introduction

On-orbit servicing (OOS) has emerged as a pivotal capability for enhancing the longevity, resilience, and sustainability of satellite missions. It encompasses a wide range of operations—including refueling, repair, repositioning, and hardware upgrades—that collectively aim to extend the functional lifetime of spacecraft beyond their original design parameters [1]. Despite its strategic value, the autonomous execution of such operations remains a formidable challenge. This is due to the complexities of operating in a microgravity environment, the unpredictable behavior of target satellites, and the intricacies of robotic manipulation in space [2].

This study presents an experimental campaign focused on investigating robotic grasping within a planar microgravity analog environment. A six-degree-of-freedom collaborative robotic arm was integrated with a low-friction air-bearing platform to replicate the free-floating dynamics of a satellite in orbit. A central innovation of this work is the development and deployment of a custom-designed end-effector, paired with a precision metrology system for real-time tracking and feedback. Additionally, the system incorporates an optimized trajectory planning and control architecture based on an enhanced Rapidly-exploring Random Tree (RRT) algorithm, enabling efficient path planning in dynamic scenarios.

The overarching objective is to evaluate the system's capacity to autonomously grasp a cooperative target under conditions representative of those encountered in orbit. The insights gained from this investigation aim to inform the development of more advanced and adaptable robotic systems for future on-orbit servicing missions involving increasingly complex and dynamic operational demands.

2 Scenario

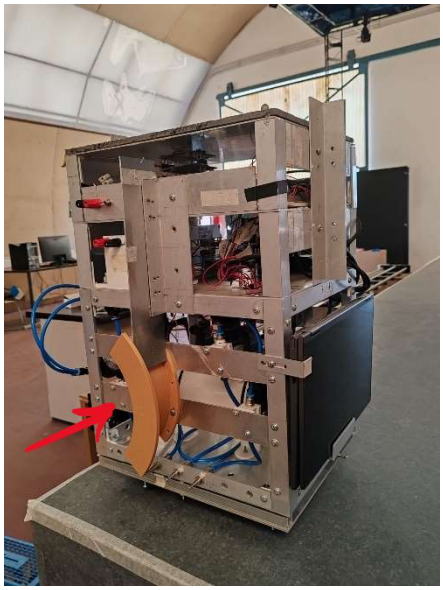
The objective of this work is to study a grasping maneuver between a chaser and a target platform. The chaser in the present configuration is equipped with a robotic arm on a fixed base, with possibility in future experiments to move on a linear rail. The target is a partially cooperative platform, that controls is position and attitude while floating frictionless on a flat granite table, so that even if not perfectly still, it does not move or rotate away; however, it does not take any action to facilitate grasping. This means that we are simulating a scenario in which the synchronization problem between target and chaser is solved, as well as the relative navigation problem.

Copyright © 2025 by First Author. Posted online by the EUCASS association with permission.

DOI: 10.13009/EUCASS2025-540

2.1 The target platform

The experiments were carried out using the PINOCCHIO platform (Figure 1a), a floating testbed developed at the GNLab of School of Aerospace Engineering of Sapienza University of Rome, with the goal of emulating the dynamics of an orbiting satellite. Designed for testing satellite guidance, navigation, and control (GNC) algorithms, the platform uses a pneumatic suspension system comprising three air pads to achieve nearly frictionless planar motion. It provides two translational degrees of freedom and one rotational degree of freedom in the horizontal plane.
The platform is equipped with cold-gas thrusters controlled by solenoid valves and an onboard inertial measurement unit (IMU). Translation and rotation are actuated through eight micro-nozzles distributed around the chassis. A detailed description of the platform's design and operation is available in [7].
The mechanical interface that is available for grasping is designed to represent (a quarter of) launch adapter ring, or LAR (gold-colored structure in Figure 1a), which is supposed to be grasped from a direction that is perpendicular the platform side.
For autonomous target capture, accurate determination of the relative position between the robot and the platform is essential. This is accomplished using an infrared (IR) optical tracking system. Specifically, a Vicon Vero camera system—comprising eight IR cameras—tracks objects fitted with retro-reflective markers. Each camera provides 2D marker positions, and 3D coordinates are computed through multi-view triangulation. The cameras are mounted on an elevated iron frame to maximize coverage of the experimental area (Figure 1b).



(a) PINOCCHIO platform



(b) Set-up with granite flat table and Vicon tracking system

Figure 1: Experimental set-up

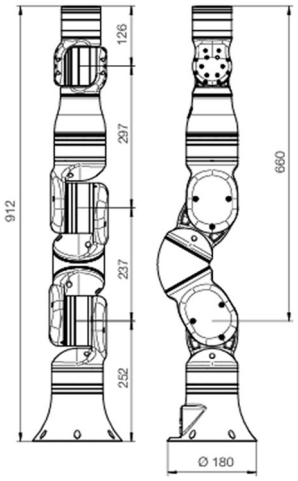
2.2 Chaser robotic arm

The chaser platform is represented by a six DOF manipulator, specifically an Igus ReBeL, a collaborative robotic arm featuring six degrees of freedom (DoF), as illustrated in Figure 2. The joint configuration, shown in Figure 1a, follows a serial chain with joints numbered from 1 to 6, starting at the base. Joints 1, 4, and 6 rotate about their respective local z-axes, while joints 2, 3, and 5 rotate about their local y-axes, enabling a versatile range of motion for complex manipulation tasks. Figure 1b details the length of each individual link, which defines the robot's kinematic structure and workspace. Each joint is capable of a maximum angular velocity of 45°/s, providing sufficient responsiveness for dynamic interactions. To ensure safe and reliable operation, each joint is also subject to predefined angular constraints, which prevent overextension or mechanical interference. These joint-specific angular limits are summarized in Table 1 and were strictly adhered to during all experimental procedures to maintain system integrity and repeatability. The end-effector has been designed and realized in-house, for the specific purpose of grasping mechanical interfaces like the LAR on the target, and reported in Figure 2c.

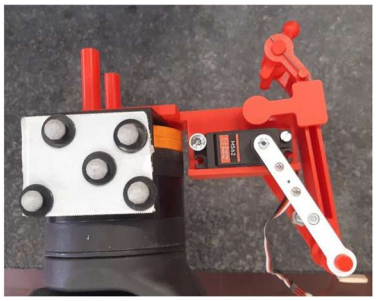
DOI: 10.13009/EUCASS2025-540



(a) ReBeL joint axis



(b) ReBeL link length



(c) End effector

Figure 2: ReBeL joints, links and end-effector

Table 1: Allowed joint angles

Joint	1	2	3	4	5	6
Max negative angle [°]	-179	-80	-80	-179	-95	-179
Max positive angle [°]	179	140	140	179	95	179

2.2.1 Direct kinematics

To define the direct kinematics of the ReBeL robot, we introduce the vector of joint variables **q** and the vector of link lengths **l**:

$$q = \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 & \theta_4 & \theta_5 & \theta_6 \end{bmatrix}^T \tag{1}$$

$$l = \begin{bmatrix} l_1 & l_2 & l_3 & l_4 & l_5 & l_6 \end{bmatrix}^T \tag{2}$$

Here, θ_i denotes the rotation angle between link *i–1* and link *i*, and l_i is the length of link *i*, following the Denavit-Hartenberg convention [3].
The relative orientation of each link's local reference frame with respect to the previous one is described using elementary rotation matrices, which depend on the corresponding joint's rotation axis:

$$R_i^{i-1}(\theta_i) = R_y(\theta_i) = \begin{bmatrix} \cos(\theta_i) & 0 & \sin(\theta_i) \\ 0 & 1 & 0 \\ -\sin(\theta_i) & 0 & \cos(\theta_i) \end{bmatrix} \; i = 2,3,5 \quad R_i^{i-1}(\theta_i) = R_z(\theta_i) = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) & 0 \\ \sin(\theta_i) & \cos(\theta_i) & 0 \\ 0 & 0 & 1 \end{bmatrix} \; i = 1,4,6 \tag{3}$$

where R_i^{i-1} indicates the rotation matrix from the local frame i to the local frame i-1.
Using these transformation matrices, both the rotation matrix from the base frame (Frame 0) to the end-effector frame (Frame 6), and the position vector of the end-effector relative to the base frame, can be computed through successive multiplication of the appropriate homogeneous transformations. Specifically:

$$R_6^0(\theta) = R_z(\theta_1) R_y(\theta_2) R_y(\theta_3) R_z(\theta_4) R_y(\theta_5) R_z(\theta_6) \tag{4}$$

$$r_{Q_0E_6}(l,\theta) = r(l)_{Q_0Q_1} + \sum_{i=1}^{5} \left(R_i^0(\theta)\, r(l)_{Q_iQ_{i+1}} \right) + R_6^0(\theta)\, r(l)_{Q_6E_6} \tag{5}$$

DOI: 10.13009/EUCASS2025-540

where the rotation matrix R_i^0 can be obtained from the first i products of the equation (4) and the vectors $r_{Q_iQ_{i+1}}$ represents the vector from the joint Q_i to the joint Q_{i+1} expressed in the local reference frame of the link i. According to the chosen convention:

$$r_{Q_iQ_{i+1}} = \begin{bmatrix} 0 & 0 & l_{i+1} \end{bmatrix}^T \tag{6}$$

Based on the selected kinematic convention, the complete transformation from base to end-effector includes both rotational and translational components, computed recursively.

3. Trajectory planning

Trajectory planning is a fundamental aspect of robotic grasping, enabling the manipulator to move safely and efficiently from its initial configuration to a target grasp pose. A well-planned trajectory must ensure collision avoidance, respect the robot's kinematic and dynamic constraints, and maintain task precision, particularly in environments where accuracy and reliability are critical. In grasping operations—especially those emulating on-orbit servicing (OOS)—trajectory planning contributes not only to motion feasibility but also to the overall robustness and success of the task.
In this study, trajectory generation is performed using the Rapidly-exploring Random Trees (RRT) algorithm, a sampling-based approach that is particularly effective in navigating high-dimensional configuration spaces. RRT incrementally constructs a tree of collision-free configurations, allowing the robotic arm to efficiently find viable motion paths while avoiding obstacles and adhering to joint limitations. Its adaptability and computational efficiency make it especially suitable for scenarios involving dynamic environments and uncertain conditions, such as those modeled in this experimental campaign.

3.1 RRT algorithm

The RRT algorithm is a widely used sampling-based motion planning technique, originally introduced by Steven M. LaValle in 1998 [4]. Designed to efficiently explore high-dimensional state spaces, RRT can generate feasible paths that connect an initial state to a goal state while avoiding collisions and satisfying system constraints. Its stochastic nature makes it especially valuable in environments where traditional deterministic planners may struggle due to complexity or unpredictability.
The algorithm incrementally builds a tree-like structure in the configuration space. At each iteration, a random sample is drawn from the state space, and the nearest node in the existing tree is identified. A short motion—referred to as steering—is then executed in the direction of the sample to create a new candidate node. If this new segment is verified to be collision-free, the node is added to the tree. This iterative process allows for rapid coverage of the configuration space and convergence toward the goal.
RRT is particularly advantageous for space robotics applications, where it must account for intricate structures such as solar panels, limited maneuverability, and multi-DoF planning challenges. Operating directly in joint space, RRT handles high-dimensional planning problems effectively and can be adapted to dynamic environments representative of on-orbit conditions [5]. An illustration of the RRT algorithm is provided in Figure 3.

Algorithm 1 Typical RRT expansion

Input: x_{init}, goal region $\mathcal{X}_{\text{goal}}$, obstacle set $\mathcal{O}$
Output: Tree $\mathcal{T}$ containing a path from x_{init} to $\mathcal{X}_{\text{goal}}$, if it exists
$\mathcal{T} \leftarrow \{x_{\text{init}}\}$ **while** iteration $\leq N_{\max}$ **do**
  $x_{\text{rand}} \leftarrow$ SampleFree() $x_{\text{near}} \leftarrow$ Nearest$(\mathcal{T}, x_{\text{rand}})$ $x_{\text{new}} \leftarrow$ Steer$(x_{\text{near}}, x_{\text{rand}})$ **if** CollisionFree$(x_{\text{near}}, x_{\text{new}}, \mathcal{O})$ **then**
    $\mathcal{T}$.add_node(x_{new}) $\mathcal{T}$.add_edge$(x_{\text{near}}, x_{\text{new}})$ **if** $x_{\text{new}} \in \mathcal{X}_{goal}$ **then**
      return $\mathcal{T}$
return *FAILURE*

Figure 3: RRT algorithm [5]

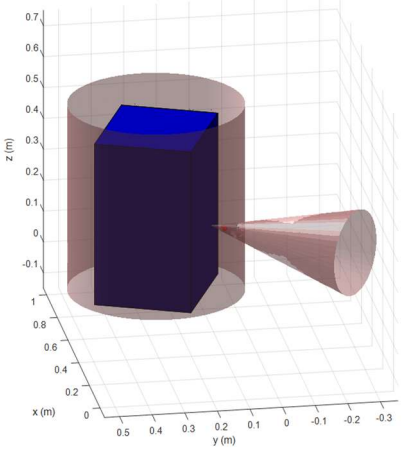
DOI: 10.13009/EUCASS2025-540

In the proposed experimental scenario, the primary obstacle is modeled in order to protect the target from being accidentally hit by the end-effector, and at the same time ensuring a safe path of the end effector towards the mechanical interface. Therefore, a forbidden area is designed around the platform, as the subtraction of a cylinder and an approach cone starting from the interface (see Figure 8)

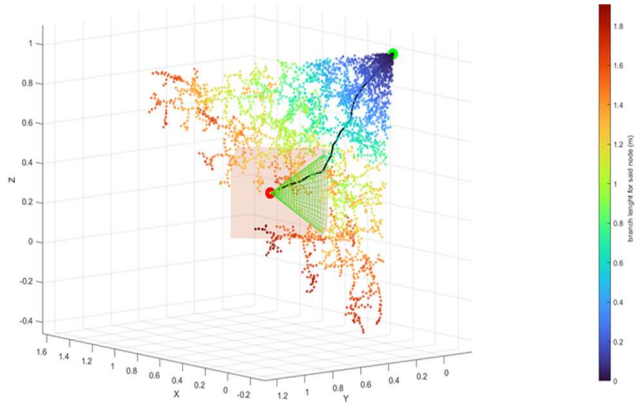
To guide the motion planning process, the RRT algorithm is configured with a specific set of parameters, which are detailed in Table 2. These parameters were selected to balance planning efficiency with path feasibility. A representative trajectory generated under these conditions—demonstrating successful navigation through the conical region while respecting all constraints—is illustrated in Figure 4b.

Table 2: RRT settings

State Space (C)	$[0 , L_{max}]$ x $[-L_{max} , L_{max}]$ x $[0 , L_{max}]$
Obstacle Set (O)	Cone
Sampling Distribution	Uniform
Distance Metric	Euclidean
Parent Selection	Closest existing node
Steering Function	Direct linear



(a) Obstacles



(b) RRT planned trajectory

Figure 4: Planned trajectory with obstacles

3.2 Improved RRT

The performance of the RRT algorithm is assessed using two primary metrics:
i) **Convergence** – the algorithm's ability to successfully identify a valid, collision-free path from the start to the goal configuration.
ii) **Path Cost Minimization** – the reduction in total path length, which directly impacts the efficiency and smoothness of the resulting trajectory.

To enhance performance with respect to these metrics, three key modifications were implemented:

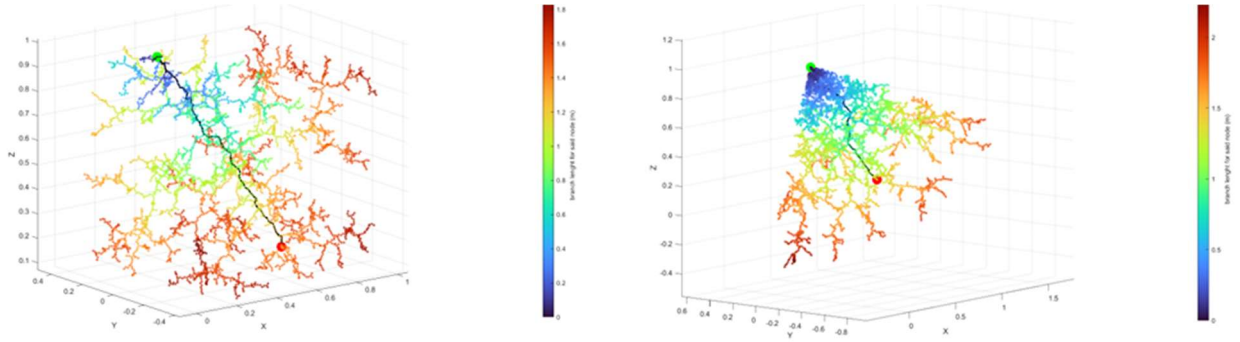
DOI: 10.13009/EUCASS2025-540

1) Probabilistic Goal Bias

To improve convergence speed, a **goal bias** strategy was introduced. In each iteration, there is a 5% probability that the random sample is replaced by the goal configuration itself. This biases tree expansion toward the goal, promoting faster connection while still maintaining sufficient exploration of the configuration space to avoid local minima.

2) Direct Sampling in a Constrained Domain

To further accelerate convergence, the sampling domain is restricted to a **conical region** extending from the initial position of the end-effector toward the goal. This focused sampling strategy significantly improves efficiency by guiding the tree within a task-relevant region of the space, rather than sampling uniformly. Experimental trials demonstrated a 30% reduction in median convergence time compared to standard uniform sampling. The difference is illustrated in Figure 5.



(a) Indirect sampling (b) Direct sampling

Figure 5: Direct vs indirect sampling

3) Cost-Based Rewiring for Path Optimization

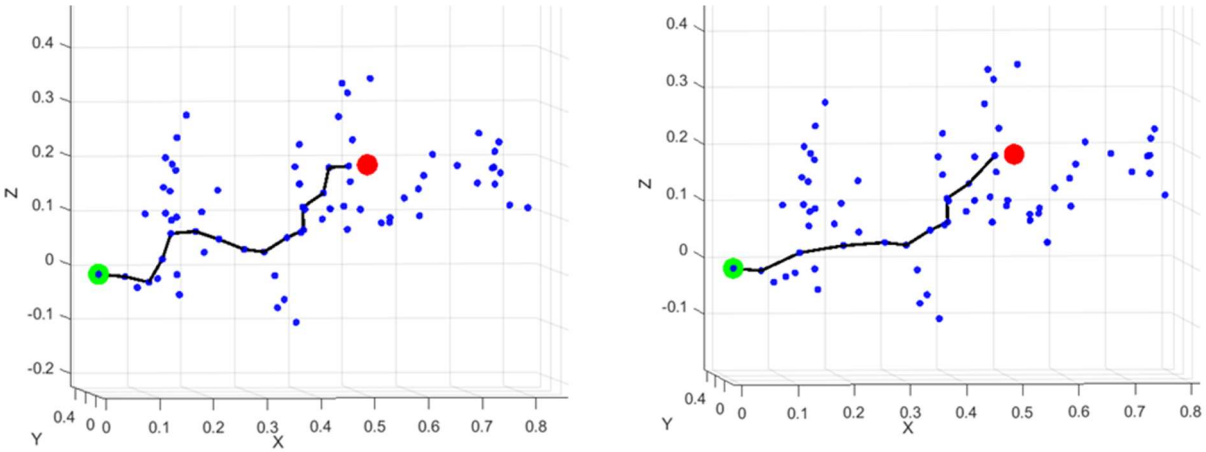
After identifying an initial valid path, a cost-based rewiring phase is performed to refine the solution and reduce total path cost. This process involves three steps:

i) Parent Optimization: For each new node, the parent is selected based on the lowest cumulative cost from within a local neighborhood, rather than simply choosing the nearest node.

ii) Neighbor Rewiring: If the path through the new node provides a lower-cost route to any of its neighbors, their parent and associated cost values are updated accordingly.

iii) Cost Propagation: When a node's parent is updated, the new cost and timing information are propagated downstream along the tree to maintain consistency across the entire path structure.

This post-processing step led to an average 25% reduction in final path length, as illustrated in Figure 6.



(a) Final path without rewiring (b) Final path with rewiring

Figure 6: Rewiring operation

DOI: 10.13009/EUCASS2025-540

3.3 Polynomial smoothing

RRT-generated trajectories often exhibit **abrupt changes in velocity** at node transitions, leading to discontinuous and jerky motion when executed directly on robotic hardware. Such non-smooth paths can degrade the performance and stability of the control system, especially in precision-critical tasks like robotic grasping.

To mitigate this issue, each trajectory coordinate is re-approximated using a polynomial smoothing technique based on a polynomial of degree N_{deg}. The smoothed trajectory is computed to satisfy two key criteria:

1. It adheres to the prescribed boundary conditions, as defined in Equation (7).
2. It minimizes the deviation from the original RRT-generated path, using an optimization objective described in Equation (8).

This approach ensures a smooth, continuous trajectory that preserves the general structure of the RRT path while significantly improving motion quality. A visual comparison between the raw and smoothed trajectories is presented in Figure 7, highlighting the improvements in path continuity and smoothness.

$$\begin{cases} X_{fit}(0) = X_{RRT}(0) \ , \ X_{fit}(t_f) = X_{RRT}(t_f) \\ \dot{X}_{fit}(0) = \dot{X}_{RRT}(0) \ , \ \dot{X}_{fit}(t_f) = \dot{X}_{RRT}(t_f) \end{cases} \tag{7}$$

$$J = \left\| \sum_{i=0}^{N_{deg}} c_i \, t^i - X_{RRT}(t) \right\|_2 \tag{8}$$

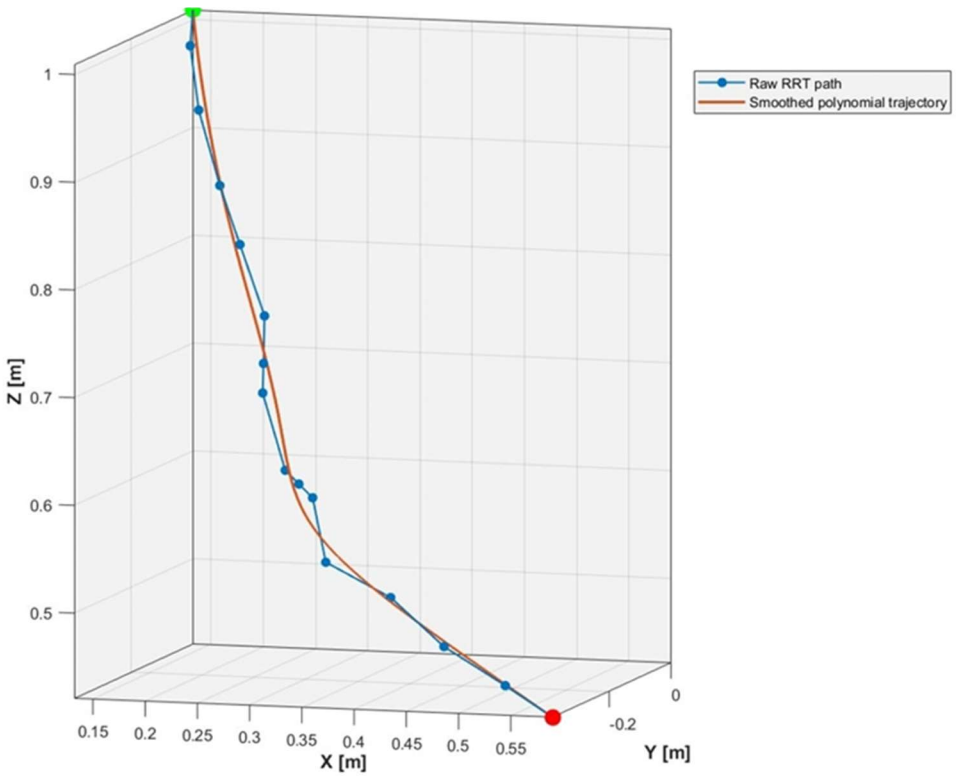


Figure 7: Resulting smoothed polynomial trajectory

3.4 Commanded attitude

Once the end-effector trajectory is fully defined in terms of position, its orientation over time must also be determined to complete the motion plan. The initial orientation is obtained from the known joint configuration using the forward

DOI: 10.13009/EUCASS2025-540

kinematics relation described in Equation (4). The final desired orientation is such that the end-effector perfectly match the mechanical docking interface (resembling a launch payload adapter) on the target platform.

To interpolate smoothly between the initial and final orientations, an axis-angle representation is employed. The rotation axis is defined as the fixed eigenaxis obtained from the axis-angle decomposition of the relative rotation between the two orientations. The rotation angle is then interpolated using a cubic spline, which ensures a smooth transition with continuous angular velocity throughout the trajectory.

This approach guarantees a physically plausible and dynamically consistent orientation profile, avoiding discontinuities that could compromise the accuracy or stability of the grasping maneuver.

4. Robot control

The Igus ReBeL robot allows for low-level control by directly commanding the angular velocities of its individual joints. Consequently, the control objective becomes the computation of the appropriate joint velocity vector that enables the end-effector to accurately follow the planned trajectory—both in position and orientation—and to reach the target configuration with the desired final pose.

To achieve this, the system relies on differential kinematics, which relates the end-effector's linear and angular velocity to the joint velocities through the robot's Jacobian matrix. By inverting or pseudo-inverting the Jacobian at each time step, the required joint rates can be calculated to ensure the end-effector progresses smoothly along the desired trajectory. This control strategy is well-suited to dynamic environments, as it provides real-time adaptability and responsiveness. Moreover, constraints such as joint velocity limits and singularity avoidance can be incorporated into the control scheme to maintain system stability and ensure safe operation throughout the motion.

4.1 Control architecture

To establish the relationship between the Cartesian linear and angular velocities of the end-effector and the joint angular velocities, we differentiate the kinematic equations (4) and (5) with respect to time. Let's define the vector x_e that contains the position p_e and the orientation ϕ_e of the end-effector.

$$x_e = \begin{bmatrix} p_e \\ \phi_e \end{bmatrix} \tag{9}$$

From the rule of the derivative of a composite function, the time derivative of x_e can be computed as:

$$\dot{x}_e = \frac{\partial x_e}{\partial q}\,\dot{q} = J(q)\,\dot{q} \tag{10}$$

Here, $\dot{x}_e$ is the spatial velocity of the end-effector, $\dot{q}$ is the vector of joint angular velocities, and J(q) is the Jacobian matrix, which maps joint velocity space to task space. The Jacobian is a $6xN_{dof}$ where N_{dof} is the number of degrees of freedom (DoF) of the robot—in this case, six.
The spatial Jacobian J(q) is computed following the method presented in [6], which is specifically formulated for anthropomorphic robotic arms with a spherical wrist configuration. This formulation accounts for both the translational and rotational contributions of each joint to the motion of the end-effector, ensuring accurate mapping across the full workspace of the manipulator.

$$J_{lin} = R_6^0 \begin{bmatrix} -R_0^6 \tilde{r}_{Q_0E_6} G_1 & -R_1^6 \tilde{r}_{Q_1E_6} G_2 & \dots & -\tilde{r}_{Q_5E_6} G_6 \end{bmatrix} \tag{11}$$

$$J_{ang} = \begin{bmatrix} R_0^6 G_1 & R_1^6 G_2 & \dots & G_6 \end{bmatrix} \tag{12}$$

$$J = \begin{bmatrix} J_{lin} \\ J_{ang} \end{bmatrix} \tag{13}$$

DOI: 10.13009/EUCASS2025-540

where the symbol $\tilde{r}$ indicates the skew-symmetric matrix associated with the vector r and G_i represents the rotation axis of the joint *i.*

Since the desired time evolution of the end-effector's velocity is known—both in terms of linear and angular components—the corresponding joint velocities can be computed using the inverse of the Jacobian matrix.

- The linear velocity profile of the end-effector is obtained by differentiating the polynomially smoothed trajectory, which approximates the original RRT-generated path.
- The angular velocity profile is derived from the time derivative of the commanded orientation, as defined in Section 3.4.

Together, these define the complete desired end-effector velocity vector in task space. To map this motion into joint space, the inverse kinematic relationship is applied:

$$\dot{q}_{com} = J^{-1}(q)\,\dot{x}_e \tag{14}$$

where J^(-1) (q) is the inverse of the Jacobian (used when the Jacobian is non-square or near-singular), $\dot{x}_e$ is the 6D end-effector velocity vector, and $\dot{q}$ is the resulting vector of joint angular velocities. This method ensures that the robot's motion closely tracks the desired trajectory while respecting the kinematic structure of the manipulator.

In the case of a redundant manipulator, the Jacobian matrix J is not square (i.e., it has more columns than rows), and its direct inverse does not exist. To address this, the Moore–Penrose pseudo-inverse is used to compute the joint velocity command:

$$J^* = J^T \left(J\,J^T\right)^{-1} \tag{15}$$

where J*(q) denotes the pseudo-inverse of the Jacobian.

However, trajectories derived purely from pseudo-inverse-based solutions can sometimes lead the robot into undesirable configurations, particularly in terms of orientation. In the current experimental context, strict orientation tracking is only required at the final phase of the maneuver, where accurate alignment between the end-effector and the graspable feature on the satellite platform is critical.

To reflect this priority—favoring positional accuracy throughout the trajectory while enforcing attitude constraints only near the goal—the computation of the joint angular velocities can be modified accordingly. A weighted or selectively damped inverse kinematics approach may be adopted to prioritize translational motion over rotational accuracy during intermediate steps, while transitioning to full pose control near the final grasping pose.
This prioritization can be formally introduced by redefining the end-effector velocity command vector or adjusting the Jacobian structure to emphasize positional components. The modified control formulation enhances trajectory smoothness and reduces the likelihood of encountering joint limits or singular configurations before reaching the target.
According to this the computation of the commanded joint angular rates is modified as follows:

$$\dot{q}_{com} = J^*_{lin}\,v_e + \left(I - J^*_{lin}\,J_{lin}\right) J^*_{ang}\,\omega_{des} \tag{16}$$

In this formulation, the commanded joint velocity vector is designed to accomplish **two distinct objectives**:

1. **Primary Task – Trajectory Tracking**
   The first term in Equation (16) ensures that the end-effector follows the desired linear velocity trajectory v_e using a minimum-norm solution. This guarantees a smooth and accurate path for the gripper while minimizing joint accelerations and avoiding unnecessary motion. It prioritizes positional accuracy, which is critical during most of the approach phase.

2. **Secondary Task – Orientation Correction**
   The second term leverages the robot's kinematic redundancy to refine the end-effector's orientation during the final stage of the maneuver. The projection operator $(I - J^*_{lin}\,J_{lin})$ ensures that the secondary task does not interfere with the primary trajectory tracking. Instead, it exploits the null space of the linear Jacobian to apply orientation corrections—achieving the desired rotational alignment without compromising the positional objective.

DOI: 10.13009/EUCASS2025-540

This hierarchical control strategy enables the robot to efficiently manage multiple objectives, enforcing strict orientation constraints only when necessary, while maintaining smooth and collision-free motion throughout the rest of the trajectory.

5. Experimental results

5.2 Experimental procedure

The experiment is organized into three main functional blocks:

1)Perception

A Simulink model acquires real-time data from the Vicon tracking system, which provides the positions of both the robot and the target within the global Vicon reference frame. A dedicated transformation subsystem then converts these positions into the robot's local base frame for subsequent processing.

2)Planning

A MATLAB function uses the initial end-effector position and the detected target position to execute an RRT-based trajectory planner, as detailed in Section 3. The resulting trajectory is smoothed, and joint velocity commands are derived using Eq. (16). These commands are then normalized to comply with the robot's maximum joint angular velocity limits.

3)Execution

A separate Simulink model transmits the normalized joint velocity commands to the robot and controls the gripper state (open/close) via a finite-state machine.

5.3 Results

A number of experiments have been conducted; a single, representative test case is presented here. Table 3 summarizes the key parameters of this successful experiment:

Table 3: Experiment data

Initial joint angles [°]	Target position [mm]	Target attitude [°]
$q = [0\ ,45\ ,90\ ,0\ ,-45\ ,0]$	$x_{goal} = 275\ ,\ y_{goal} = 300$	$\theta_{goal} = -23$

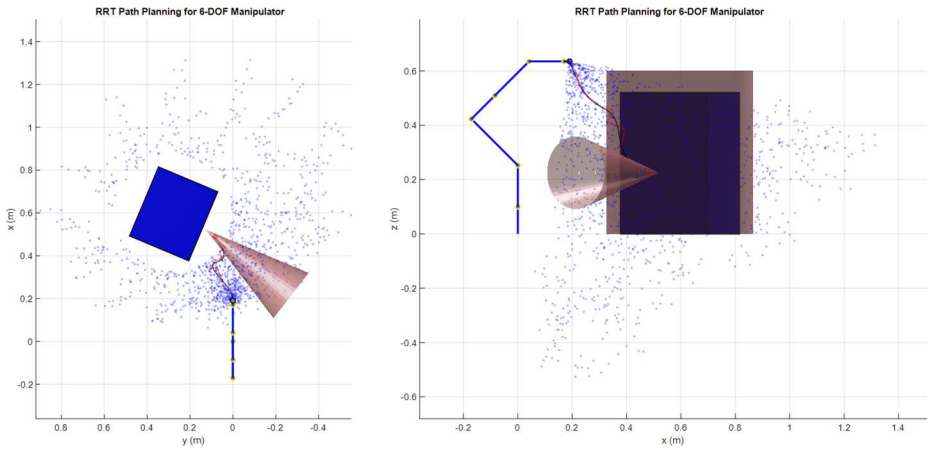
Different views of the end-effector trajectory found by the RRT algorithm are shown in Figure 10, where the platform is represented as a blue box, the obstacles in pale red and the initial and final manipulator configurations are represented in blue and red, respectively. The blue dots represent nodes of the RRT algorithm that did not lead to the goal.

In this test, the RRT algorithm generated a trajectory with a final grasping time of $t_f = 36\ s$. Figure 9 shows the joint angles and angular velocities that are computed by applying the Eq. (16) to the smoothed trajectory. Notice (Figure 9a) that all the joint angles remain in the range reported in Table 1. Since attitude tracking is a secondary task, a residual error affects the actual end-effector attitude, as illustrated in Figure 10. However, its magnitude remains within the end-effector's alignment tolerance, allowing for a successful grasp. The nominal trajectory generated with RTT algorithm is precisely tracked by the manipulator, as reported in Figure 11. There, an initial 5-second delay is introduced at the start of the simulation to ensure proper synchronization between the robotic arm's software and the Simulink-based control architecture. This delay prevents any mismatch between the commanded motion and the arm's physical response. The close overlap between the red and black lines in Figure 11 confirms the effectiveness of the controller in tracking the desired trajectory.

Figure 12 captures the dynamic response of the PINOCCHIO platform to the grasp event, occurring at t = 41s, as measured by the Vicon tracking system. Finally, several frames depicting the grasping of the platform can be found in Figure 13.

Overall, the experiment demonstrates the system's capability to autonomously execute a precise grasp on a cooperative, floating target, starting from a known initial configuration and target pose.

DOI: 10.13009/EUCASS2025-540



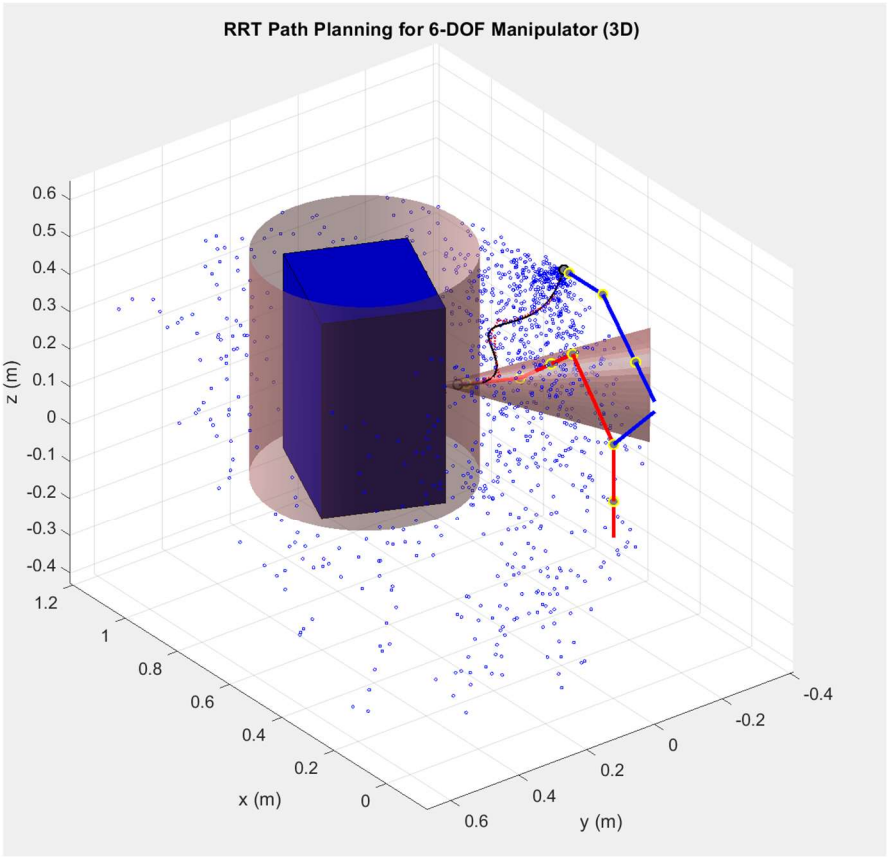
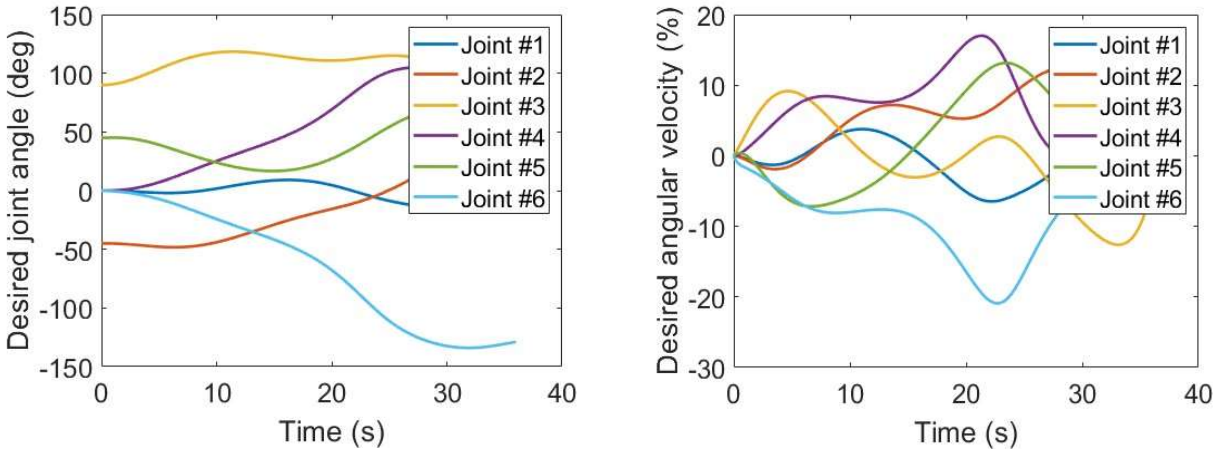


Figure 8: Top view (upper-left frame), side-view (upper-right frame) and 3D view of the trajectory computed with RRT algorithm, including obstacles.

DOI: 10.13009/EUCASS2025-540



(a) Desired joint angles [°] (b) Desired joint angular velocities [°/s]

Figure 9: Desired joint angles and rates

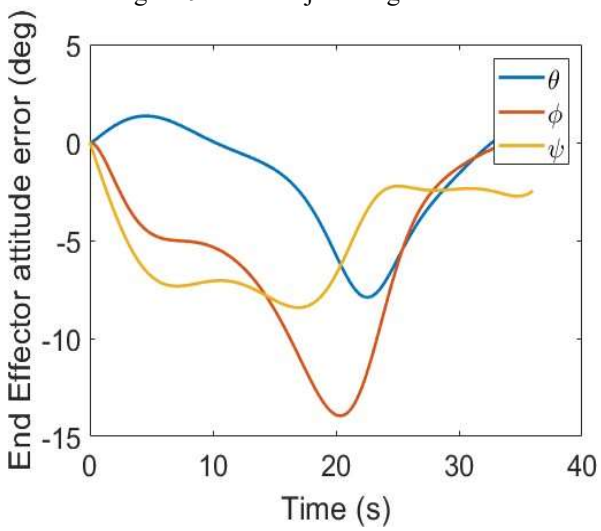


Figure 10: Attitude error

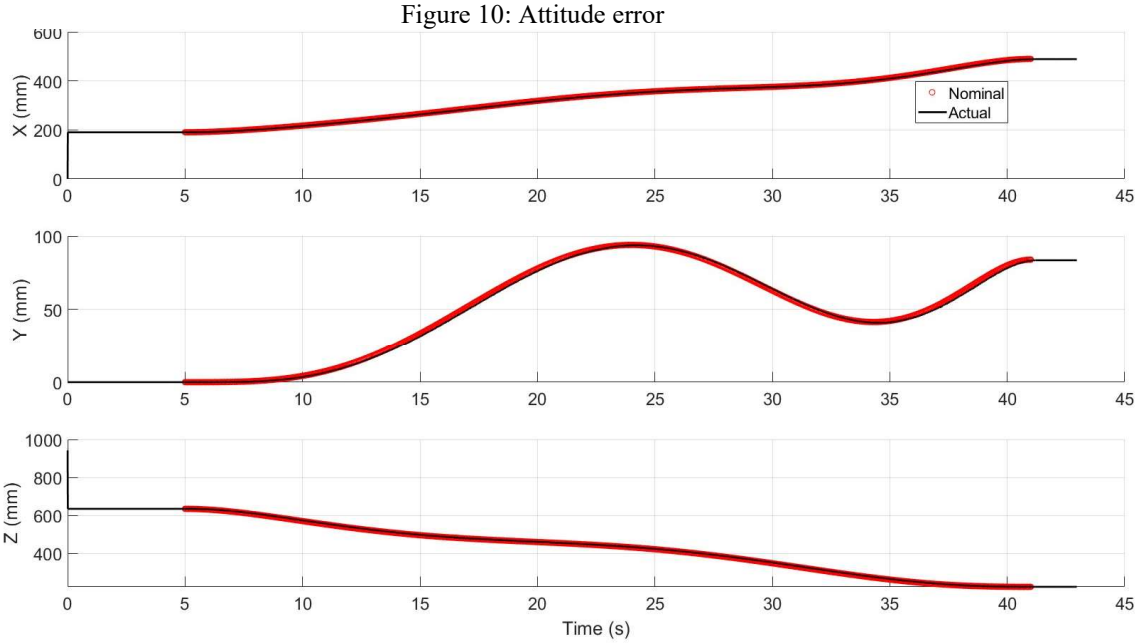


Figure 11: Actual vs nominal end-effector trajectory

DOI: 10.13009/EUCASS2025-540

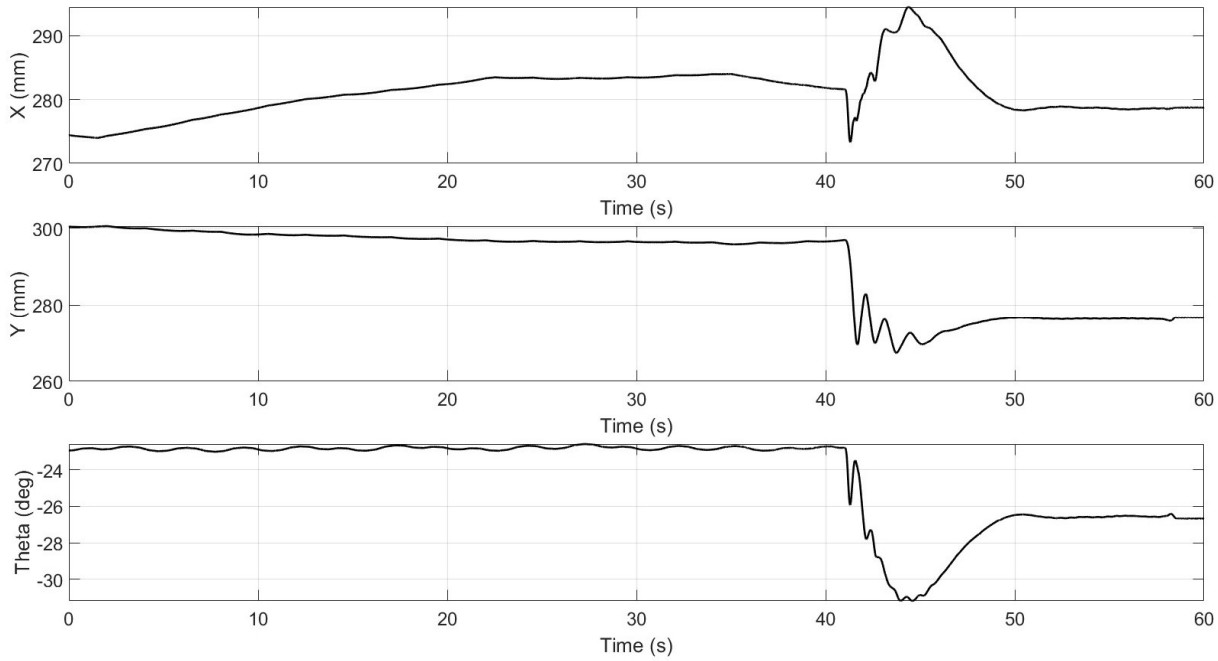


Figure 12: PINOCCHIO state

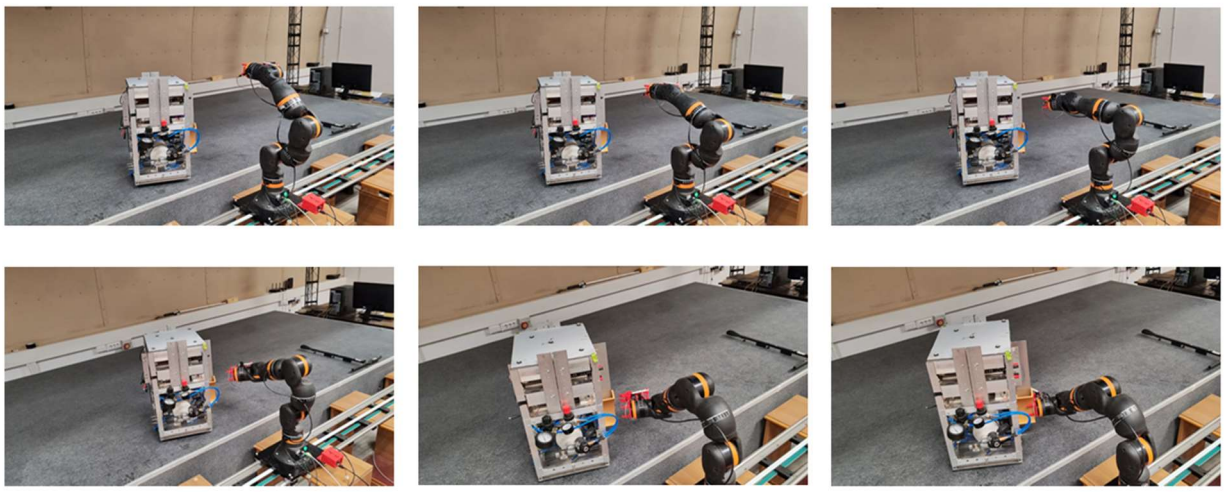


Figure 13: Successful experiment

6. Future developments

The results validate the robot's capability to plan and execute smooth trajectories from a known initial state to a slowly moving target, while successfully avoiding obstacles, adhering to joint constraints, and achieving a final orientation suitable for grasping. To increase the realism of the simulation environment and extend its applicability to more complex and representative on-orbit servicing scenarios, the following enhancements are proposed:

i) **Refine trajectory planning algorithms** to enable access to theoretically reachable configurations that are currently excluded due to suboptimal path generation leading to joint limit violations.

ii) **Extend the framework to handle uncooperative targets** moving along known trajectories. This introduces time-dependent planning, requiring the planner to account for the relative motion of the target and to update trajectories in real time.

iii) **Incorporate real-time iterative guidance**, enabling continuous updates of both robot and target states. The planner must adapt the trajectory dynamically at each time step to address the target's unpredictable behavior.

DOI: 10.13009/EUCASS2025-540

iv) **Add an additional degree of freedom** by mounting the robotic arm on a linear rail, allowing motion along a horizontal axis. This would significantly expand the robot's operational workspace and more accurately reflect conditions encountered in satellite servicing missions.

7. Conclusions

This work presented an experimental investigation of robotic grasping for in-orbit servicing applications, using a six-degree-of-freedom robotic manipulator operating in a laboratory environment that emulates microgravity conditions. A comprehensive trajectory planning strategy, based on an enhanced Rapidly-exploring Random Trees (RRT) algorithm, was developed and integrated into a control framework combining forward and inverse kinematics for guidance and grasp execution.

Experimental validation on the PINOCCHIO air-bearing platform confirmed the feasibility and accuracy of autonomous grasping maneuvers, demonstrating the effectiveness of the proposed approach. However, the study also highlighted limitations of the current planning method, particularly in accessing the full range of reachable configurations due to joint constraints and suboptimal path generation.

Future work should focus on developing advanced orientation planning techniques and incorporating additional degrees of freedom—such as activating a linear rail system—to extend the manipulator's operational workspace. Furthermore, expanding the experimental framework to include dynamic scenarios with moving targets will be a critical step toward transforming the current setup into a fully functional academic ground-testing facility for on-orbit servicing technologies.

.

References

[1] Donald J. Kessler et Burton G. Cour-Palais. "Collision Frequency of Artificial Satellites : The Creation of a Debris Belt". In : Journal of Geophysical Research : Space Physics 83.A6 (juin 1978), p. 2637-2646. doi : 10.1029/JA083iA06p02637

[2] Alizadeh M.; Zhu Z. H.; "A comprehensive survey of space robotic manipulators for on-orbit servicing", Front. Robot. AI, 2024, Vo. 11

[3] "Forward Kinematics : The Denavit-Hartenberg Convention". In : Introduction to Robotics : Mechanics and Control. Chap. 3, p. 71-102.

[4] Steven M. LaValle. Rapidly-Exploring Random Trees : A New Tool for Path Planning. Rapp. tech. TR 98-11. Dept. of Computer Science, Iowa State University, 1998.

[5] Tomasz Rybus et Karol Seweryn. "Application of Rapidly-exploring Random Trees (RRT) Algorithm for Trajectory Planning of Free-Floating Space Manipulator". In : Proceedings of the 10th International Workshop on Robot Motion and Control (Ro MoCo). Poznań, Poland : IEEE, 2015, p. 91-96. isbn : 978-1-4799-7043-8. doi : 10.1109/RoMoCo.2015.7219719.

[6] Bruno Siciliano et al. Robotics : Modelling, Planning and Control. Springer, 2010. isbn : 978-1-84628-641-4. doi : 10.1007/978-1-84628-642-1.

[7] Marco Sabatini, Paolo Gasbarri et Giovanni B. Palmerini. "Design, Realization and Characterization of a Free-Floating Platform for Flexible Satellite Control Experiments". In : Acta Astronautica 210 (mai 2023). Open-access article, published 6 May 2023 under CC BY license, p. 576-588. doi : 10.1016/j.actaastro.2023. 05.007. url : https://doi.org/10.1016/j.actaastro.2023.05.007.