POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Dynamics-Consistent Trajectory Mapping and Emulation in the RAFFAELLO Robotic Testbed: Methods with Digital Twin Validation and Dataset Generation

Laurea Magistrale in Space Engineering - Ingegneria Spaziale

Author: Stefano Belletti

Advisor: Prof. Paolo Panicucci

Co-advisors: Alban Beshaj, Fabio Ornati

Academic year: 2024-2025

1. Introduction

Missions targeting deep space objects face several criticalities: regarding the Guidance, Navigation and Control (GNC) subsystem of the spacecraft, navigation plays an important role, since visual-based algorithms have to be reliable. These algorithms are tested on ground to verify Verification and Validation (V&V), since in situ tests are prohibitive or simply not possible.
For this scope, datasets are extremely valuable, since they are composed of a recreated Field Of View (FOV) of the navigation camera. However, these synthetic images can be recreated with different levels of detail and they often lack photorealism. It is important to note that the lower this level of fidelity, the higher the sim-to-real gap is, indicating how poor the generated images are with respect to the real scenario [1]. Facilities can range from being completely digital, where all simulations and camera acquisitions are performed through software, to being completely physical, in the case of a full hardware simulation, which typically achieves the highest fidelity and the lowest sim-to-real gap.

The Robotic Arm Facility For Autonomous cubesat ExpLoration in cLose proximity Operations (RAFFAELLO) facility is a kinematic testbed: the image acquisitions are performed by a camera, capturing a physical target body mockup. These elements are controlled via high-torque robotic arms, which are designed to follow a predetermined path. Spacecraft dynamics can be reproduced indirectly through precomputed trajectories, but cannot be directly tested, as that would require a different configuration (e.g., low-friction surface).
This work investigates the extent to which a robotic manipulator testbed can generate accurate, dynamically consistent trajectory data that mimics real-life motion profiles, addressing the challenge of mapping such trajectories into this type of facility.

2. Methodology

The mapping problem has been tackled from different aspects: starting from a robotic viewpoint, analyzing the most suitable cost function, and arriving at the complete nonlinear programming problem formulation. Digital twin modeling is used to simulate the physical environ-

ment, allowing verification of constraints violation and validate the results. Lastly, Hardware-in-the-Loop (HIL) tests are carried out to generate datasets and verify the correctness of the solutions.

Figure 1: RAFFAELLO facility.

2.1. Robotics

The RAFFAELLO facility consists of two Yaskawa GP12 robotic manipulators, which face each other and are enclosed in a safety cage. The robotic analysis and development part of the thesis is crucial, since the Forward Kinematics (FK) and Inverse Kinematics (IK), as well as the Velocity Kinematics (VK) and Inverse Velocity Kinematics (IVK) are used to bound the task and joint spaces. Task space refers to the tridimensional space of the facility, where positions and velocities are defined; while the joint space refers to the angle and angular velocity solutions of the GP12 joint deflections.

FK consists in defining the orientation of the end effector knowing the joint solution, both product of exponentials (PoE) and Denavit-Hartenberg (D-H) [2] formulations can be used: the latter is chosen for its simplicity and lower computational costs, allowing to define each joint pose with homogeneous matrices comprising both attitude and positions. These matrices are multiplied in sequence to produce the FK result.

Regarding IK, the GP12 is a 6R PUMA-type robotic manipulator, which presents the so called *spherical wrist*: the intersection of the last three joints is fixed in space for any configuration of them. This enables decoupling of the IK problem in the *position problem* and the *orientation problem*, solving the first three joints and the last three independently. This simplifies the implementation, allowing for an analytical solution.

However, the IK solution is usually not unique. A unique solution must therefore be selected to ensure that continuity in joint space matches continuity in task space. This problem is easily solved for the first three joints (from the base), restricting to only *righty* and *elbow-up* solutions, in accordance with joint angle limits and implementation constraints. In contrast, the last three joints (the spherical wrist) exhibit periodicity due to their wide rotation ranges and inherent solution symmetries: this is solved by imposing a single direction of rotation, which can be tweaked with input parameters.

VK and IVK relate the joint and space velocities of the end effector: this relationship is established through the Jacobian matrix, which completes the maps between the two spaces and implements the singularity proximity analysis. Avoiding singularities is essential during trajectory mapping to prevent rapid joint movements that would compromise smoothness and dataset quality [3].

To ensure singularity free and smooth robotic trajectories, the manipulability ellipsoid and its associated scalar indicators are analyzed: quantifying the shape and size of the ellipsoid:

$$\begin{aligned} \mu_1(\boldsymbol{A}) &= \sqrt{\frac{\lambda_{max}(\boldsymbol{A})}{\lambda_{min}(\boldsymbol{A})}} \geq 1 \\ \mu_2(\boldsymbol{A}) &= \frac{\lambda_{max}(\boldsymbol{A})}{\lambda_{min}(\boldsymbol{A})} \geq 1 \\ \mu_3(\boldsymbol{A}) &= \sqrt{\lambda_1\lambda_2\lambda_3} = \sqrt{\det(\boldsymbol{A})} \end{aligned} \tag{1}$$

The indicators were computed separately for the angular-velocity sub-Jacobian ($\boldsymbol{J}_\omega$, with $\boldsymbol{A} = \boldsymbol{A}_\omega(\boldsymbol{\theta}) = \boldsymbol{J}_\omega(\boldsymbol{\theta})\boldsymbol{J}_\omega^T(\boldsymbol{\theta})$) and linear-velocity sub-Jacobian ($\boldsymbol{J}_v$ with $\boldsymbol{A} = \boldsymbol{A}_v(\boldsymbol{\theta}) = \boldsymbol{J}_v(\boldsymbol{\theta})\boldsymbol{J}_v^T(\boldsymbol{\theta})$), yielding μ_ω and μ_v. During optimization, nonlinear constraints and monitoring of μ ensured that mapped trajectories remained far from singularities, avoiding rapid joint accelerations and discontinuities.

2.2. Optimization

The optimization begins with the analysis of the problem from an astrodynamics perspective. The input trajectories consist of the spacecraft's

and target body's positions, velocities and attitudes, along with the Sun direction: important for the photorealism of the image acquisition.
These elements are defined in a target centered, inertial, reference frame, where the Sun direction varies over time. It can be observed that, at any given instant, rotating the target body together with the spacecraft's position, velocity and attitude matrices around the Sun direction maintains the same relative attitude between the two bodies, as well as the same target illumination. This holds because the phase angle and the relative spacecraft-target position remain unaltered. The same conclusions can be drawn for the Sun distance: if the whole system translates along the Sun direction, the same illumination conditions are met. Changes in the illumination intensity of the Sun is not relevant, since it can be adjusted in the facility by changing the camera or lamp settings.
The RAFFAELLO facility, however, uses a static lamp to simulate the Sun illumination: a shift from the inertial reference frame to the facility reference frame is achieved through a rotating reference frame, defined by the angular velocity of the Sun direction in the inertial frame.
The optimization problem is formulated based on these two main characteristics: rotation and translation along the Sun direction, the two free Degrees Of Freedom (DOF) of the problem.
This reference frame change, along with position and time scaling, allows defining a facility path, which is macroscopically equivalent to the original trajectory.
Rotation matrices are constructed using Rodrigues' formula [4] (Equation 2), based on a unit rotation axis $\hat{\boldsymbol{k}}$ and angle β computed using vector trigonometry:

$$\boldsymbol{R}_{\hat{\boldsymbol{k}}}(\beta) = \boldsymbol{I} + \sin\beta\ [\hat{\boldsymbol{k}}]_\times + (1-\cos\beta)\ [\hat{\boldsymbol{k}}]_\times^2 \quad (2)$$

For small angles, a second-order Taylor expansion provides a more stable evaluation.
Position vectors and attitude matrices are transformed via pre-multiplication by the rotation matrix. The transport theorem, instead, is used to transform velocity vectors between rotating frames, involving the angular velocity of the rotating frame $\boldsymbol{\omega}(t)$. This angular velocity is computed from the relative rotation matrix: $\boldsymbol{R}_{\text{rel}} = \boldsymbol{R}(t_{k+1})\boldsymbol{R}(t_k)^T = e^{[\hat{\boldsymbol{\omega}}]\times\psi}$ [3], by extracting the rotation angle $\psi = \cos^{-1}\left(\frac{\text{tr}(\boldsymbol{R}_{\text{rel}})-1}{2}\right) \in [0,\pi]$.
The angular velocity vector components are then recovered via:

$$\begin{aligned}
\boldsymbol{\omega}_x(t_k) &= \frac{\psi}{2\sin\psi\cdot\Delta t}(R_{32}-R_{23}) \\
\boldsymbol{\omega}_y(t_k) &= \frac{\psi}{2\sin\psi\cdot\Delta t}(R_{13}-R_{31}) \\
\boldsymbol{\omega}_z(t_k) &= \frac{\psi}{2\sin\psi\cdot\Delta t}(R_{21}-R_{12})
\end{aligned} \quad (3)$$

where R_{ij} are the elements of $\boldsymbol{R}_{\text{rel}}$. The same formulation can be extended to a centra finite differences approach, which is used for the internal nodes, whereas first-order forward/backward differences are employed at the endpoints.
The problem now reduces to an optimization of the rotation and translation of the system along the Sun (lamp) direction. MATLAB *fmincon* function is used for this scope, using nonlinear inequality constraints to verify the robotic feasibility of the mapped trajectory. The solution consists in $\boldsymbol{x} = [x_1\ x_2\ x_3]$: the initial angle rotation, constant angular velocity and initial position magnitude (scalar going from 0.2 to 1, depending on the distance from the lamp, measured between a minimum and a maximum distance).
It is important to remark that the optimization does not alter the *natural* constraints (phase angle, relative attitude, relative positions and velocities) of the problem: nonlinear constraints are only used to ensure the mapped solution can be reproduced by the robotic manipulators. Furthermore, for any given solution of the optimization, even if not robotically feasible, the mutual constraints of the trajectories (*natural* constraints) are preserved.

2.2.1 Preprocessing

The given trajectory is first processed to separate individual legs, by detecting velocity discontinuities, caused by maneuvers, and target body hemisphere changes. The latter is required since the target body requires reorientation on the end effector. When this procedure is complete, the optimization can start either for a single leg, or for the complete set: the solutions are expected to converge rapidly, but a *slicer* is introduced to split the analyzed leg in half

whenever the solution cannot be found. This procedure can be applied recursively as needed. Figure 2 shows the segmented trajectory for a custom mission around 25143 Itokawa, in the inertial reference frame. Northern hemisphere legs are shown in red and southern in blue, with diamond markers indicating the dividing points where hemisphere-change or maneuvers occur.

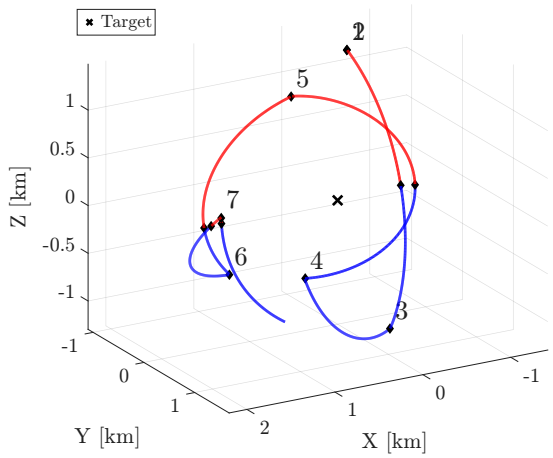

Figure 2: Trajectory preprocessing.

The pose of the lamp is not considered during the optimization and it is treated as static. The challenge consists in optimizing its position outside the single leg optimization: its values should change as little as possible, since the lamp positioning is done manually and is time consuming. However, this would require a significantly more complex implementation and has not been addressed in this work.

2.2.2 Nonlinear constraints

The nonlinear constraints consist of a set of scalar inequality equations designed to guarantee the safety and feasibility of the solution in terms of the robots' physical limits and collision avoidance. 161 equations are generated for each point of the analyzed leg, resulting in a total of $161N$ scalar quantities, where N is the length of the leg (typically 150-200). This leads to 24 000 - 32 000 constraints per leg, making this the most computationally intensive part of the entire optimization process. While some constraints could be relaxed, others are critical and must be strictly enforced.
For this reason and to prevent small numerical violations from invalidating otherwise valid solutions, conservative safety margins are introduced by shifting the bounds toward safer values. The main nonlinear inequality constraints categories include:

- $\boldsymbol{\theta}$ **and** $\boldsymbol{\dot{\theta}}$ **limits**: these ensure that the Yaskawa GP12 joint angles and angular velocities remain within the manufacturer's maximum allowable values;
- **Reachability**: ensuring the end-effector positions stay within the feasible workspace of each robot;
- **Collision avoidance with facility and between robots**: preventing self-collisions between the two arms and collisions with the cage or other facility components;
- **Collision avoidance and non-shadow condition with the lamp**: avoiding collisions with the static lamp and ensuring no shadowing of the target body (to preserve illumination consistency): a verification based on the phase angle is performed for this last task.

Description	Symbol	Value
θ margin	$\boldsymbol{\delta}_{\theta}$	5°
$\dot{\theta}$ margin	$\boldsymbol{\delta}_{\dot{\theta}}$	$1\,\mathrm{rad\,s^{-1}}$
Distance margin	$\boldsymbol{\delta}_{\mathrm{distance}}$	0.2 m
Robot distance margin	$\boldsymbol{\delta}_{\mathrm{robot}}$	0.3 m
Phase margin	$\boldsymbol{\delta}_{\mathrm{phase}}$	20°

Table 1: Nonlinear constraints margins.

Table 1 highlights the additional safety margin variables, used for the nonlinear constraints: these ensures a feasible solution, even when the optimization converges with a minimum constraint violation.

2.3. Validation

Validation is the key process of reproducing the real trajectories starting from the optimization solution. This is crucial to verify fidelity and dynamic consistency of the map.
The procedure is simple and consists in generating the poses of the two payloads of the robots, starting from the joint space solution: these quantities are then scaled back to real dimensions. The problem reduces to applying frame rotations and transportation to reproduce the inertial reference frame. The same procedure can also be carried out for the target body frame. Finally, each quantity is compared to the

reference values.
Jacobian validation is performed to check for its correctness, while camera sensor simulation is implemented to generate a synthetic FOV of the camera and verify pointing accuracy.
Jacobian validation is straightforward and involves creating a fictitious trajectory using waypoints and tridimensional spline interpolations: these values are then used to derive the velocities. The two task space quantities are mapped into joint space, gathering angles and angular velocities of the robot (respectively with IK and IVK): these values are then checked by comparing the integrated angular velocities with the angles of the IK result, which are characterized by extremely small errors and represent the reference.
Regarding the camera sensor emulation, the target body position is projected in the camera reference frame and altered by adding camera lens distortion effects. This produces a synthetic image featuring any desired LED pattern to be mounted on the target end effector: these objects are typically used for calibration purposes.

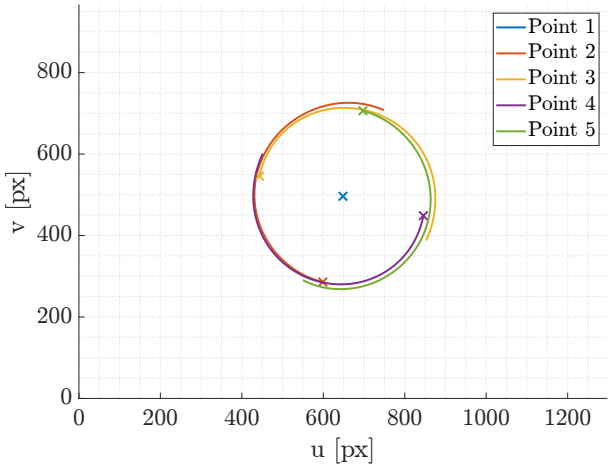

Figure 3: Camera sensor emulation.

3. Results

This section presents the results of the methodology, with emphasis on validation and error evaluation. Small errors are expected, given the geometrical approach of the optimization.

3.1. Kinematic Validation

The validation of the kinematic implementation has two main objectives: confirming the correctness of FK and IK, and verifying the Jacobian construction through IVK.
For the first objective, a custom fictitious trajectory is mapped from task space to joint space using IK, yielding joint angles $\boldsymbol{\theta}$. These are then transformed back to task space via FK, and the resulting position and attitude vectors are compared with the original. The errors are extremely small (1×10^{-15} both for position and attitude), consistent with floating-point numerical precision and confirming the algorithms as exact: no approximations are introduced in the closed-form trigonometric solutions of the IK.

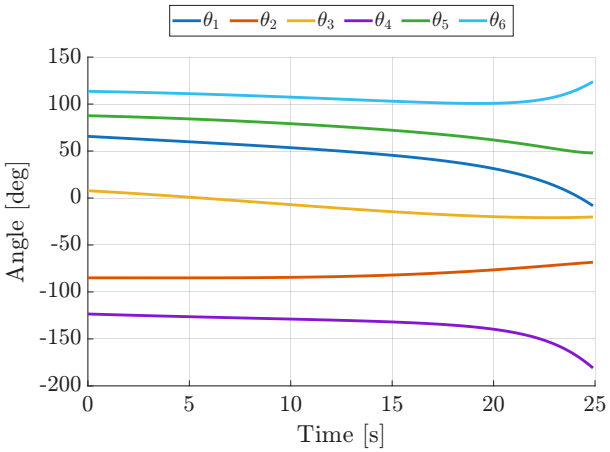

Figure 4: $GP12_1$ solution.

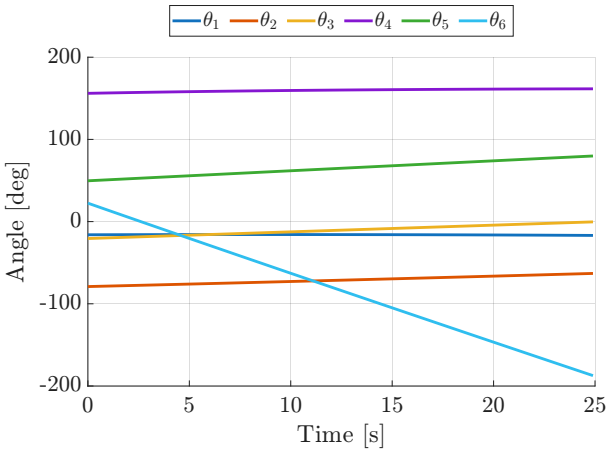

Figure 5: $GP12_2$ solution.

Furthermore, the continuity of the IK can be checked in the joint space results of a single leg optimization, shown in Figure 4 and Figure 5.
Regarding the Jacobian validation, IVK is used to generate $\dot{\boldsymbol{\theta}}$ solution which is integrated in time and compared with IK ($\boldsymbol{\theta}$). Errors remain in the range of 1×10^{-5} deg, which is acceptable and attributable to finite-difference approxima-

tions of the rotation matrix derivative, used for the twist evaluation, or in the integration step. This procedure confirms both the accuracy of the task-joint mapping and the correctness of the Jacobian construction.
The validated Jacobian plays a fundamental role in the overall mapping analysis, as it is used to compute the manipulability ellipsoid and μ_1 (ratio of largest to smallest eigenvalue), which serves as the cost function in the optimization. The smoothness and continuity of μ_1 ensure reliable gradient-based convergence and effective singularity avoidance during trajectory mapping.

3.2. Optimization Results

Analyzing now the results, it can be seen how the errors related to position vectors and attitude vectors are so low, they can be attributed to numerical precision errors, deriving from MATLAB internal floating point accuracy (magnitudes of 1×10^{-15}). Other behavior can instead be seen in the velocity error plots: the magnitude here is bigger by few orders of magnitude, but this result was expected.

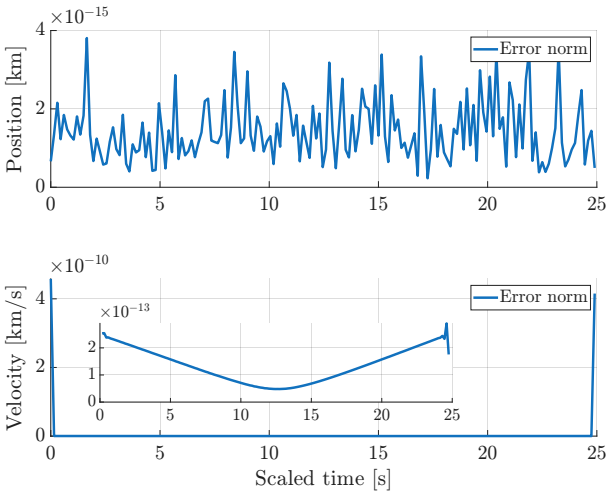

Figure 6: ${}^N\boldsymbol{r}_C$ and ${}^N\boldsymbol{v}_C$ error.

The reason behind this discrepancy lies in how velocities are transported between the rotating reference frames: the transport formula needs the reference frame's angular velocity vector, but only discrete data is available and a finite element approach must be implemented, leading to errors.
These relative errors, fortunately, are extremely small and well within margins for the validation of the method.
Lastly, the results show that the errors are larger at the boundaries (Figure 6): this is a clear evidence of the contribution of finite differences methods, where the boundary values have been modeled with forward and backward finite difference approaches, whereas the internal nodes use a more precise central scheme.
Regarding the result of the single leg optimization, Figure 7 shows the facility configuration at the initial timestep, where both GP12 robots reach near their workspace limits (extended backward from home positions), indicating challenging trajectory sizes, given by the current scale factor $\delta_L = 1/669$.

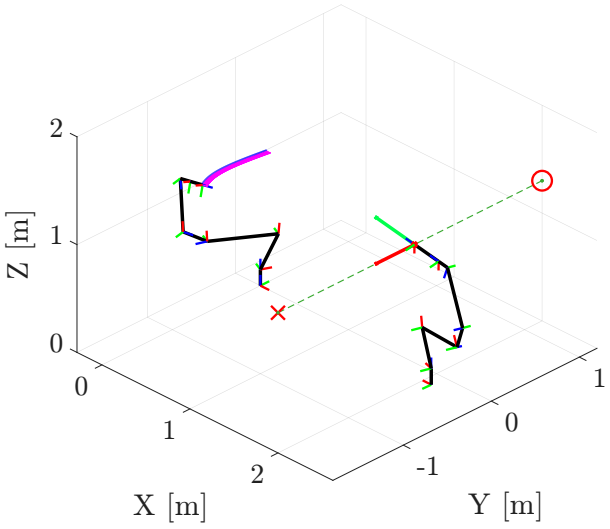

Figure 7: Facility solution.

Convergence of the solutions is highly influenced by the length scale, since feasible solutions are limited due to workspace constraints. Preliminary tests at $\delta_L = 1/2000$ consistently achieved full trajectory mapping with rapid convergences. This behavior is evident in the nonlinear constraints domains, reported for any combination of x_1 and x_3, while $x_2 = 0\,\mathrm{rad\,s^{-1}}$ for the $\delta_L = 1/2000$ case, and $x_2 = 0.0186\,\mathrm{rad\,s^{-1}}$ for the $\delta_L = 1/669$ case.

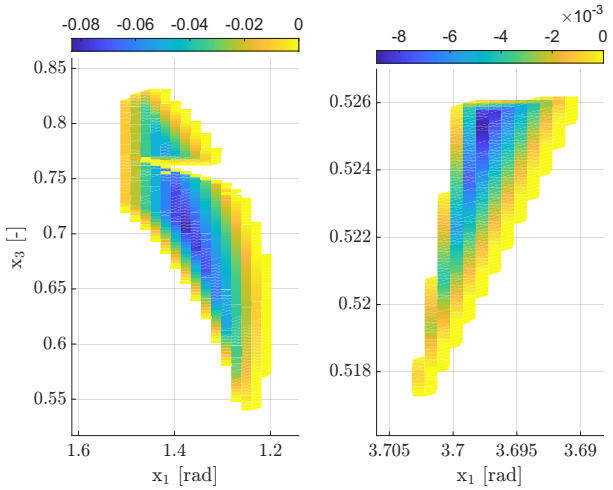

Figure 8: Constraints: feasible domains of $\delta_L = 1/2000$ (left) and $1/669$ (right).

Figure 8 shows how little the feasible region for the current facility scale is. This phenomenon makes solution finding much harder, but the developed *initial guess finder* algorithm proved to be effective enough to make the optimization converge.
Following with the cost function analysis, the smaller scale values are reported below:

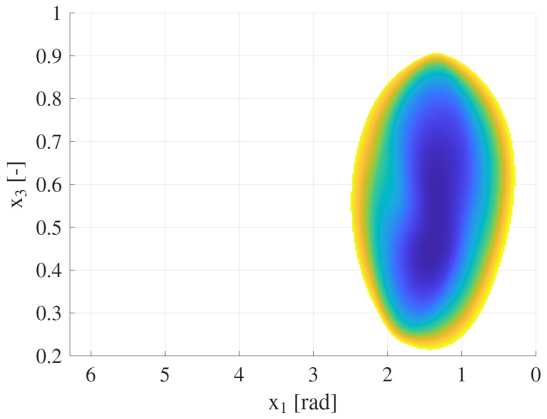

Figure 9: Cost function domain ($\delta_L = 1/2000$).

It can be seen how the cost function appears smooth and the two discernible minimums are close in terms of values. Figure 9 indicates that μ_1 is the perfect candidate for the cost function, and convergence is expected to be fast, for feasible solutions.

3.3. Downsampling

Additional tests were conducted using a Downsampling (DS) technique, which consists in selecting a smaller number of points from the original (OG) trajectory without interpolation and preserving the boundary points. Results of some $\delta_L = 1/2000$ legs are reported below:

Leg (points)	Time (s) OG	DS	
2,1 (211 → 54)	28.220	6.941	-75.40%
2,2 (213 → 54)	15.891	5.377	-66.16%
5,2 (106 → 54)	6.385	3.313	-48.11%
6,1 (328 → 56)	29.856	4.823	-83.84%

Table 2: Downsampling results.

Table 2 demonstrates significant time reductions with downsampling. Solutions from DS were validated on the full original trajectory, yielding complete feasibility in all cases. The last three legs show identical solutions to the original, while the first leg exhibits only minor deviations, preserving the overall feasibility and dynamic consistency.

3.4. Dataset acquisition

Figure 10 presents the final outcome of the thesis: an image acquired from the RAFFAELLO HIL testbed during dataset generation, with the $\delta_L = 1/669$ mapped solution.

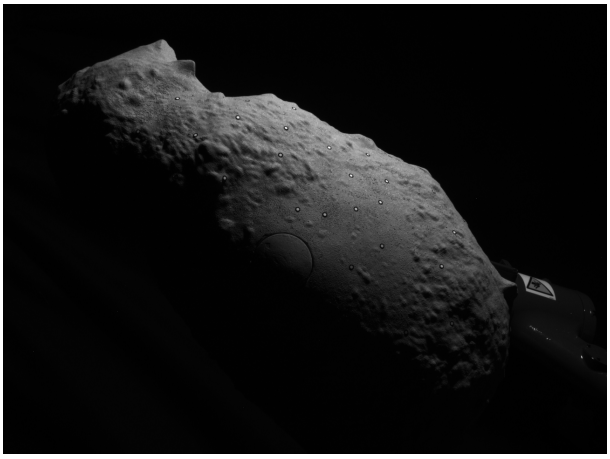

Figure 10: Dataset acquisition: frame 0075.

Future postprocessing will be required to remove the mockup plug, landmarks reflective dots and the robotic arm, enabling the use of the images for algorithm training and V&V.

4. Conclusions and future work

The proposed mapping methodology proved to be well-suited for the scope, allowing the correct reproduction of high-fidelity mission profiles with dynamic consistency. This approach is general in nature, since no assumptions were done on the mission or target body specific characteristics, while also being facility-specific, since the available degrees of freedom and constraints depend on the hardware characteristics of the RAFFAELLO testbed.

The core concept of the method is the identification of the two DOF of the problem: rotation and translation along the Sun (lamp) direction. These two values are exploited to map the original inertial trajectory in the testbed, inherently preserving the *natural* constraints.

The solution vector of the proposed method consists of defining the orientation with a constant angular velocity and an initial value, while considering the position as static. These are the three optimization parameters. This formulation corresponds to a first-order control of the orientation and a zero-order control of the position, since otherwise, the illumination of the target would not have been uniform throughout the HIL tests (regarding the current implementation).

The solution is not unique: starting from two variables corresponding to the initial conditions (similar to x_1 and x_3), the evolution of the orientation and translation can also be determined via parametric curves (e.g., spline interpolation), by relating time to rotation and/or translation or specifying initial and final derivatives. As long as the *natural* orbital constraints are not altered by the optimization variables, any desired exploitation of the two facility DOF remains valid.

Regarding the physical limits of the facility, the length scale δ_L proved to strongly influence the convergence of the optimization. For the current machined asteroid mockup ($\delta_L = 1/669$), only a limited number of feasible solutions were found, due to the relatively large scale constraining the available workspace. In contrast, preliminary tests conducted with a finer scale ($\delta_L = 1/2000$) consistently enabled successful mapping of the entire trajectory, with rapid convergence across all legs.

This high sensitivity to scale was expected, given the purely geometrical nature of the mapping approach and the fixed hardware dimensions. The bigger scale of 1/669 was in fact introduced when the $GP12_1$ was mounted on a linear rail, allowing it to move further away from the target and enabling the mapping of larger trajectory legs.

In summary, the proposed method successfully demonstrates that a kinematic robotic testbed can generate dynamically consistent trajectory data mimicking real motion profiles. The downsampling technique demonstrates further how the*geometrical* approach is well suited to generate consistent results with smaller problem dimensions.

In conclusion, future HIL tests will aim at reproducing the dynamically consistent trajectory solutions, as the current implementation of the RAFFAELLO facility supports only the static generation of individual poses.

References

[1] Emily Bates, Zahra Ahmed, Aviad Golan Peretz, Pol Francesch Huc, Antonio Rizza, Samuel Y. W. Low, Toby Bell, Gregory Zin, and Simone D'Amico. Digital and robotic twinning for validation of proximity operations and formation flying. In *Proceedings of the 48th Annual AAS Guidance, Navigation & Control Conference*, AAS 26-148, Breckenridge, Colorado, January 30–February 4 2026. American Astronautical Society.

[2] J. Denavit and R.S. Hartenberg. Notation for lower-pair mechanisms based on matrices. *A. Kinematic, ASME Journal of Applied Mechanics*, 22:215–221, 01 1995.

[3] Kevin M Lynch and Frank C Park. *Modern robotics*. Cambridge University Press, 2017.

[4] Richard M Murray, Zexiang Li, and S Shankar Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 2017.