

# Failure Management via Deep Reinforcement Learning for Robotic In-Orbit Servicing

Matteo D'Ambrosio <sup>\*1</sup>, Lorenzo Capra<sup>1</sup>, Andrea Brandonisio<sup>1</sup>, Michèle Lavagna<sup>1</sup>

<sup>1</sup>Politecnico di Milano, Aerospace Science and Technology Dept, Via la Masa 34, 20156 Milan, Italy

The recent applications of Artificial Intelligence (AI) and Reinforcement Learning (RL) to enhance spacecraft autonomy, reactivity, and adaptability in future In-Orbit Servicing (IOS) and Active Debris Removal (ADR) activities have shown potential towards overcoming some of the main difficulties that are encountered in orbital robotics missions, and could eventually provide practical solutions to strengthen their current capabilities. In highly critical missions such as in the capture of a spacecraft or debris through a robotic arm, having a system with heightened fault tolerance when conditions such as single-joint manipulator failures are encountered is of particular interest, and could aid in overcoming disastrous collision events. To this extent, this work evaluates the ability of an autonomous agent trained through meta-RL, providing the guidance of a space manipulator in real time to synchronize the end-effector to a desired state fixed to the mission target, to adapt to unexpected failures in the joints. These preliminary results show that the agent indeed improves the ability of the system to autonomously overcome manipulator failure events, as long as the goal end-effector position remains within the manipulator's workspace subsequently to the joint failure.

## 1 Introduction

The recent urges to develop new robotics technologies for future IOS and ADR mission scenarios have created large demands for the development of innovative Guidance, Navigation, and Control (GNC) approaches, that can aid in surpassing the main shortcomings of more classic alternatives and that can increase the overall robustness of current architectures, for example in degraded conditions like actuator failures. Among these methods, RL emerges as a potent strategy [1] to aid with spacecraft GNC. In [2–4], Deep Reinforcement Learning (DRL) agents are trained to conduct pinpoint planetary and moon landings and demonstrate high adaptability and fault-tolerance capabilities when subjected to thruster failures. In [5, 6], DRL is employed to control a spacecraft's trajec-

tory to reconstruct the shape of a tumbling object; the resulting agent is robust to losses of thrust capabilities in one of the directions, with only a small reduction in terms of performance. In the context of space robotics, agents have been trained to guide manipulators to a desired goal autonomously and can be used to overcome problems such as ill-posed inverse kinematics problems, and the satisfaction of secondary constraints into the trajectories without the need for real-time optimization [7–12]. Additionally, in works like [13], such agents are found to aid the system's overall robustness by enhancing failure management capabilities when one of the manipulator's 6 joints has failed, at reduced performance with respect to nominal conditions. The goal of this work is to extend this concept to a 7-Degree of Freedom (DoF) manipulator, to preliminarily evaluate whether a trained agent is capable of repurposing the redundant DoFs of the robotic arm to achieve its goal with the end-effector.

### 1.1 Dynamics

A free-flying [14] space robot model is adopted in this work, such that the base can be actively controlled to stay in its desired state. The space robot model is implemented through Matlab's Simscape Multibody library, and explicit kinematic and dynamic quantities are retrieved through the SPART toolkit [15]. The equations of motion of the space robot are found in Equation (1).

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

where  $\mathbf{H}$ ,  $\mathbf{C}$  are the generalized inertia and convective inertia matrices [15], and  $\boldsymbol{\tau}$  is the vector of external torques. The state vector  $\mathbf{q}$  collects the position and attitude of the space robot base, and the joint angles of the manipulator. Regarding the target spacecraft, its dynamics are represented through the unforced Euler Equations reported in Equation (2).

$$\mathbf{I}\dot{\boldsymbol{\omega}}_T = (\mathbf{I}\boldsymbol{\omega}_T) \times \boldsymbol{\omega}_T \quad (2)$$

<sup>\*</sup>Corresponding author. E-Mail: matteo.dambrosio@polimi.it

where  $I$  is the target’s inertia matrix and  $\omega_T$  is the target’s angular velocity vector.

### 1.2 GNC architecture

The adopted GNC architecture (Figure 1) is made up of an autonomous agent providing the guidance of a 7-DoF space manipulator, to align the end-effector with a desired position and attitude fixed to the tumbling target. Then, the space robot is controlled through the nonlinear feedback linearization controller in Equation (3), where the resulting linearized system is actuated by two PD regulators, respectively for the base and manipulator.

$$\tau = \begin{Bmatrix} \tau_0 \\ \tau_m \end{Bmatrix} = H \begin{Bmatrix} PD(\dot{q}_0^* - \dot{q}_0) \\ PD(\dot{q}_m^* - \dot{q}_m) \end{Bmatrix} + C\dot{q} \quad (3)$$

where quantities labeled “0” and “m” respectively refer to the base and manipulator, and  $q^* - q$  represents the error between set-point and the current state.

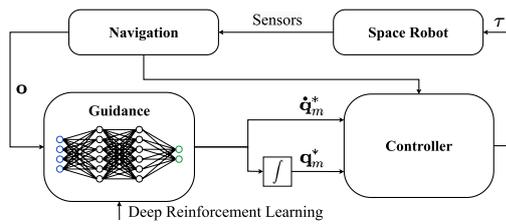


Figure 1: SR GNC architecture [11].

#### 1.2.1 DRL guidance

The autonomous DRL agent receives 32 observations  $o$  (Equation (4)) at each timestep and generates 7 actions  $a$  (Equation (5)) corresponding to the desired joint rates of the manipulator, that are integrated such that both position and velocity set-points can be provided to the controller.

$$o = [q_m, \dot{q}_m, \tilde{r}, \tilde{\theta}, \tilde{v}, \tilde{\omega}]^T \quad (4)$$

where  $q_m$  collects the joint angles of the manipulator, and all terms  $\tilde{\square}$  represent errors between the desired and current end-effector states.

$$a = \dot{q}_m^* = [\dot{\phi}_1, \dot{\phi}_2, \dot{\phi}_3, \dot{\phi}_4, \dot{\phi}_5, \dot{\phi}_6, \dot{\phi}_7]^T \quad (5)$$

The agent’s policy, i.e. the guidance law that is applied in the environment, is implemented through a Feedforward Neural Network (FNN) that is trained through the Proximal Policy Optimization (PPO) DRL algorithm [16], selected for its greater simplicity and higher sample efficiency, considering the complexity of the implemented environment; the hyperparameters of the FNN are reported in Table 1.

Layers	Actor size	Critic size
Input	32	32
1st hidden	300	300
2nd hidden	300	300
3rd hidden	300	300
Activation	<i>tanh</i>	<i>tanh</i>
Learning rate	1e-5	1e-5

Table 1: FNN hyperparameters.

The Artificial Potential Field (APF) based reward function [8, 11] is reported in Equation (6), and rewards the agent when either the position error or attitude error of the end-effector decreases.

$$U_k = -\tilde{r} + \frac{10}{1 + \tilde{r}_{ax}} + \frac{10}{1 + \tilde{r}_{tx}} + \frac{10}{1 + \tilde{\theta}} \quad (6)$$

$$\Delta U = U_k - U_{k-1} \quad (7)$$

$$R_k = \begin{cases} \Delta U & \text{if } \Delta U \geq 0 \\ 1.5\Delta U & \text{if } \Delta U < 0 \end{cases} \quad (8)$$

where  $U_k$  is the APF,  $\tilde{r}$ ,  $\tilde{r}_{ax}$ ,  $\tilde{r}_{tx}$  are the end-effector’s position errors opportunely projected in the space robot’s body frame, and  $\tilde{\theta}$  is its attitude error.

### 1.3 Training process

The reference agent applied in this work has been pre-trained in an environment where the following conditions are changed in each episode: target’s spin rate, initial manipulator configuration, end-effector goal position, and distance between space robot and target. Note that such agent has never been subjected to joint failures, and is similar to the one found in [11], apart from the collision avoidance objective which has been removed. The PPO hyperparameters that have been used to train the agent are referenced in Table 2, and are taken within the typical intervals found in the literature for the PPO algorithm.

Hyperparameter	Value
Clipping factor	0.2
GAE* factor	0.99
Discount factor	0.95
Entropy loss weight	0.01
Agent sample time	0.3 s

\* Generalized Advantage Estimation [17].

Table 2: PPO hyperparameters.

For clarity, the joints are numbered from 1 to 7 (see Figure 2), moving from the base of the space robot towards the tip.

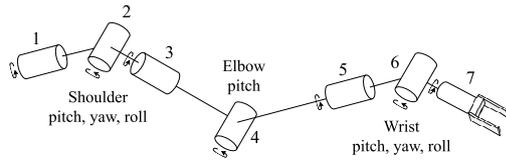


Figure 2: Joint numbering.

Each episode is considered successful if both the position and attitude errors of the end-effector stay below thresholds of (5 cm, 5 deg) for at least 30 s consecutively. The reference agent achieves a 100% success rate in guaranteeing such thresholds in nominal conditions without joint failures. An example of the scenario is reported in Figure 3.

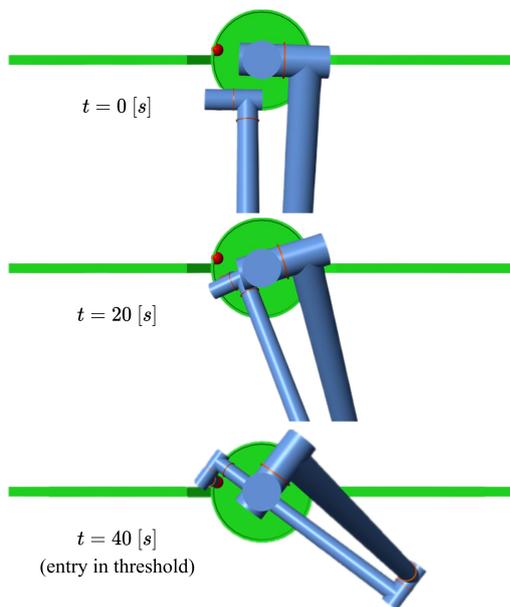


Figure 3: Autonomously generated trajectory (Chaser POV).

The scope of this work is to evaluate the agent’s behavior when a failure in any of the joints is encountered, and the joint results fixed at its initial angle. In these conditions, using a redundant 7-DoF manipulator is advantageous as it could still provide the workspace and dexterity to successfully complete the mission. The agent is evaluated in three conditions:

- (a) No training: The reference agent is directly tested in the environment with a failure in each of the joints, to see in what cases it can adapt, and how its success rate is impacted.
- (b) Training 1: The agent is trained for an additional 2000 episodes (applying transfer learning), with a 50% chance that a random failure is encountered in any of the joints. It is then tested as in point a).

- (c) Training 2: The agent is trained for an additional 2000 episodes (applying transfer learning), with a 50% chance that a random failure is encountered in joints [1, 5, 6, 7]. It is then tested as in point a).

## 2 Results

The results from cases (a), (b), and (c) are reported in Table 3. As can be seen, there is a drastic drop in terms of success rate from the nominal value of 100% in all cases, showing that the agent’s performance is generally impacted quite heavily when it encounters a failure in any of the manipulator’s joints.

Joint	1	2	3	4	5	6	7
(a)	62%	9%	13%	0%	42%	58%	54%
(b)	53%	0%	14%	0%	33%	72%	66%
(c)	67%	-	-	-	38%	74%	56%

\* Success rate averaged over 200 episodes.

Table 3: Success rate results from tests on cases a), b), c).

The reason for these results is twofold: firstly, the agent needs to adapt in real-time to completely new manipulator dynamics, with respect to what it has seen throughout the majority of its training. Additionally, the failures of some joints (specifically 2, 3, 4) create large reductions of the manipulator’s workspace, to the point that the end-effector rarely reaches the goal position, regardless of the trajectory requested by the agent. From Table 3 itself, it is not clear whether the additional training from cases b) and c) has had any effects on the agent’s performance as the achieved success rates only slightly differ from case a); from these results alone it could be argued that no improvements result from these trainings. To this extent, the agent’s performance is further evaluated by plotting the location of the grasping position on the target in each episode, against the episode’s success (see Figures 4 to 7).

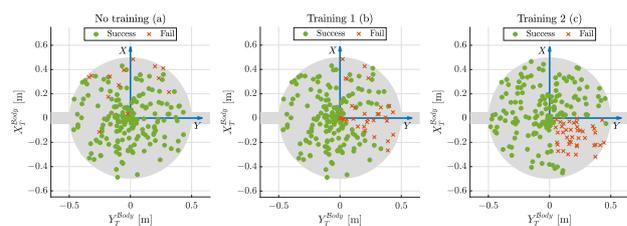


Figure 4: Case comparison: Joint 1

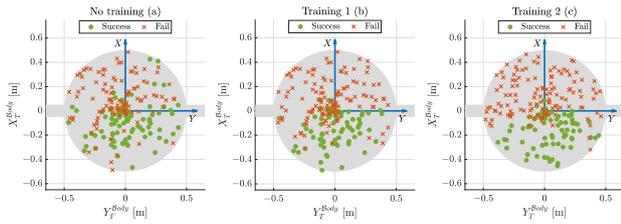


Figure 5: Case comparison: Joint 5

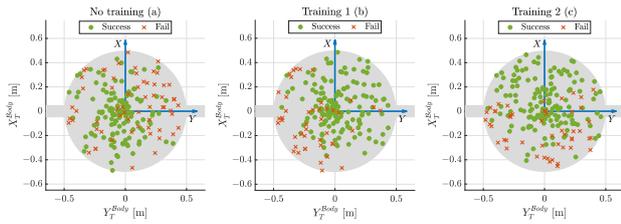


Figure 6: Case comparison: Joint 6

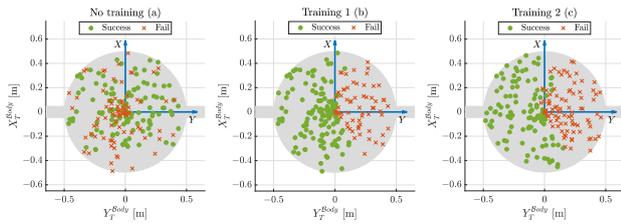


Figure 7: Case comparison: Joint 7

### 3 Discussion

Overall, the autonomous agent demonstrates different characteristics based on the specific manipulator joint that has failed in the simulation. As stated previously, the failure of joints (2-4) has too much of an impact on the manipulator’s workspace for the agent to adapt and still accomplish its objective, apart from a minimal set of initial conditions. Additional training does not have any real benefits in the cases with failures on joints (2-4) (shoulder pitch and yaw, and elbow pitch), and the agent does not provide sufficient robustness to the system to proceed with the mission. Indeed, more detailed results from these conditions are omitted for their small relevance at this stage of this study. Differently, condition (a) (No training) shows that without having been trained to do so, the agent autonomously adapts to failures in joints (1, 5, 6, 7), and can repurpose the operative joints to achieve its end-effector objective, at reduced performance. As shown in case (a) of Figures 4 to 7, the main shortcoming under these conditions is that the episode failures are encountered

quite randomly, regardless of where the grasping position is located on the target. The main benefit of conducting additional training through meta-RL, that is subjecting the agent to a distribution of different environments throughout the training process, is that the episode failures are encountered less randomly with respect to the state of the grasping position. For example, in the case of joint (1) (see Figure 4), the episode failures are mainly encountered when the grasping position falls in the fourth quadrant of the target. Similar behavior can also be noticed in the case of joint (7) (see Figure 7), where prior to training the agent fails in all quadrants, and after training no episode failures are encountered in quadrants 2 and 3.

The benefits shown by the recent literature on DRL methodologies for high-DoF manipulator guidance and control prove that similar strategies are worth investigating further, in view of developing architectures that can be applied outside of simulation environments. Indeed, DRL has been applied to overcome some of the main problems that emerge in more traditional redundant manipulator GNC strategies, such as the ill-posed inverse kinematics and difficulties in implementing motion constraints or secondary objectives in the manipulator trajectories, especially when these need to be satisfied in real-time. The results of this study shed light on an additional benefit that autonomous DRL agents could provide, demonstrating that more robust and fault-tolerant architectures can be achieved through meta-RL training approaches, to develop systems that can quickly adapt to changing environments and dynamics; in this work, this behavior is demonstrated by subjecting the agents to joint failures in the manipulator. Such architectures are of high interest for the foreseen orbital robotics tasks, and the extension of similar approaches would significantly enhance current capabilities of space robots. A few possible extensions to this work are summarized in the points below:

1. Provide information on the failed joint in the observations, to give the agent more context on how it could adapt the manipulator trajectories to achieve its goal.
2. Evaluate and optimize the neural network architectures to ones that have typically provided greater robustness on other problems, such as Recurrent Neural Networks (RNNs).
3. Tune the chance of joint failure during the training process, to a value that further increases the agent’s resulting performance.

## References

1. Silvestrini, S. *et al.* in, 819–981 (Elsevier, Jan. 2023). ISBN: 9780323909167.
2. Gaudet, B., Linares, R. & Furfaro, R. Adaptive guidance and integrated navigation with reinforcement meta-learning. *Acta Astronautica* **169**, 180–190. ISSN: 00945765 (Apr. 2020).
3. Gaudet, B., Linares, R. & Furfaro, R. Terminal adaptive guidance via reinforcement meta-learning: Applications to autonomous asteroid close-proximity operations. *Acta Astronautica* **171**, 1–13. ISSN: 0094-5765 (June 2020).
4. Scorsoglio, A., Furfaro, R., Linares, R. & Gaudet, B. *Image-based deep reinforcement learning for autonomous lunar landing in. 1 Part F* (American Institute of Aeronautics and Astronautics Inc, AIAA, 2020). ISBN: 9781624105951.
5. Brandonisio, A., Capra, L. & Lavagna, M. Deep reinforcement learning spacecraft guidance with state uncertainty for autonomous shape reconstruction of uncooperative target. *Advances in Space Research*. ISSN: 02731177 (July 2023).
6. Capra, L., Brandonisio, A. & Lavagna, M. Network architecture and action space analysis for deep reinforcement learning towards spacecraft autonomous guidance. *Advances in Space Research*. ISSN: 0273-1177 (Apr. 2023).
7. Wu, Y. H. *et al.* Reinforcement learning in dual-arm trajectory planning for a free-floating space robot. *Aerospace Science and Technology* **98**. ISSN: 12709638 (Mar. 2020).
8. Li, Y. *et al.* Constrained Motion Planning of 7-DOF Space Manipulator via Deep Reinforcement Learning Combined with Artificial Potential Field. *Aerospace* **9**. ISSN: 22264310 (3 Mar. 2022).
9. Huang, Z. *et al.* An Obstacle-Avoidance Motion Planning Method for Redundant Space Robot via Reinforcement Learning. *Actuators* **12**. ISSN: 20760825 (2 Feb. 2023).
10. Blaise, J. & Bazzocchi, M. C. Space Manipulator Collision Avoidance Using a Deep Reinforcement Learning Control. *Aerospace* **10**. ISSN: 22264310 (9 Sept. 2023).
11. D'Ambrosio, M., Capra, L., Brandonisio, A., Silvestrini, S. & Lavagna, M. Redundant Space Manipulator Autonomous Guidance for In-Orbit Servicing via Deep Reinforcement Learning. *Aerospace* **11**, 341. ISSN: 2226-4310. <https://www.mdpi.com/2226-4310/11/5/341> (5 Apr. 2024).
12. Ali, A. A. & Zhu, Z. H. Reinforcement learning for path planning of free-floating space robotic manipulator with collision avoidance and observation noise. *Frontiers in Control Engineering* **5**. ISSN: 2673-6268. <https://www.frontiersin.org/articles/10.3389/fcteg.2024.1394668/full> (May 2024).
13. Wang, S., Zheng, X., Cao, Y. & Zhang, T. *A Multi-Target Trajectory Planning of a 6-DoF Free-Floating Space Robot via Reinforcement Learning in* (Institute of Electrical and Electronics Engineers Inc., 2021), 3724–3730. ISBN: 9781665417143.
14. Moghaddam, B. M. & Chhabra, R. *On the guidance, navigation and control of in-orbit space robotic missions: A survey and prospective vision* July 2021.
15. Virgili-Llop, J., II, J. V. D. & Romano, M. *SPART SPACe-craft Robotics Toolkit: an Open-Source Simulator for Spacecraft Robotic Arm Dynamic Modeling And Control in* (2016).
16. Schulman, J., Wolski, F., Dhariwal, P., Radford, A. & Klimov, O. Proximal Policy Optimization Algorithms (PPO). *arXiv:1707.06347* (2017).
17. Schulman, J., Moritz, P., Levine, S., Jordan, M. & Abbeel, P. High-Dimensional Continuous Control Using Generalized Advantage Estimation. *4th International Conference on Learning Representations*. <http://arxiv.org/abs/1506.02438> (June 2015).