The Journal of the Astronautical Sciences (2021) 68:1145–1169
https://doi.org/10.1007/s40295-021-00288-7

ORIGINAL ARTICLE

Reinforcement Learning for Uncooperative Space Objects Smart Imaging Path-Planning

Andrea Brandonisio[1] · Michèle Lavagna[1] · Davide Guzzetti[2]

Accepted: 1 October 2021 / Published online: 2 November 2021

Abstract

Leading space agencies are increasingly investing in the gradual automation of space missions. In fact, autonomous flight operations may be a key enabler for on-orbit servicing, assembly and manufacturing (OSAM) missions, carrying inherent benefits such as cost and risk reduction. Within the spectrum of proximity operations, this work focuses on autonomous path-planning for the reconstruction of geometry properties of an uncooperative target. The autonomous navigation problem is called active Simultaneous Localization and Mapping (SLAM) problem, and it has been largely studied within the field of robotics. Active SLAM problem may be formulated as a Partially Observable Markov Decision Process (POMDP). Previous works in astrodynamics have demonstrated that is possible to use Reinforcement Learning (RL) techniques to teach an agent that is moving along a pre-determined orbit when to collect measurements to optimize a given mapping goal. In this work, different RL methods are explored to develop an artificial intelligence agent capable of planning sub-optimal paths for autonomous shape reconstruction of an unknown and uncooperative object via imaging. Proximity orbit dynamics are linearized and include orbit eccentricity. The geometry of the target object is rendered by a polyhedron shaped with a triangular mesh. Artificial intelligent agents are created using both the Deep Q-Network (DQN) and the Advantage Actor Critic (A2C) method. State-action value functions are approximated using Artificial Neural Networks (ANN) and trained according to RL principles. Training of the RL agent architecture occurs under fixed or random initial environment conditions. A large database of training tests has been collected. Trained agents show promising performance in achieving extended coverage of the target. Policy learning is demonstrated by displaying that RL agents, at minimum, have higher mapping performance than agents that behave randomly. Furthermore, RL agent may learn to maneuver the spacecraft to control target lighting conditions as a function of the Sun location. This work, therefore, preliminary demonstrates the applicability of RL to autonomous imaging of an uncooperative space object, thus setting a baseline for future works.

✉ Andrea Brandonisio
andrea.brandonisio@polimi.it

Extended author information available on the last page of the article.

Keywords Reinforcement learning · Autonomous spacecraft · Guidance and control · Artificial neural networks · On-orbit servicing

Introduction

Since the launch of the first satellite, Sputnik, on 4 October 1957, satellites have been typically equipped with everything they need to accomplish their mission, from launch to end of life. In the space environment, satellites are isolated entities. Full isolation leads to the design of self-contained architectures that are more complex and require additional redundancy to survive the space environment. Currently, ability to autonomously refuel, upgrade or repair satellites once they are on orbit is not yet available.

In recent years, the space sector has begun to move towards the development of autonomous servicing technologies with the goal of reducing to a minimum human intervention. Up to now, on-orbit servicing (OOS) activities are mostly performed and studied by government agencies, e.g. NASA or DARPA. DARPA Orbital Express (OE) in 2007, has demonstrated a full end-to-end robotic satellite servicing mission characterized by autonomous docking, fuel transfer and orbital replacement unit, all accomplished without human intervention. The demand of orbital servicing has increased in the recent years, pushing more and more companies towards the study of complete autonomous satellite services [1].

According to NASA and American Institute of Aeronautics and Astronautics (AIAA), the term *on-orbit servicing* is referred to all on-orbit activities that a space vehicle intentionally conducts on another resident space object (RSO). Such definition includes several activities: non-contact support, relocation and orbit modification, maintenance, refueling, commodities replenishment, upgrade, repair, assembly, and debris mitigation. The vehicle able to perform one or more of these activities is called *servicer*, while the vehicle that receives the service is called *client*. The client is called *cooperative client*, if it transfers useful information for acquisition, tracking, rendezvous or other servicing activities to the servicer via direct communication link or ground link; otherwise it is defined *uncooperative* or *non-cooperative client*. The ISS is the cooperative client of the resupply vehicle designed to mate it, while an orbital debris is considered a non-cooperative client. In most of the cases, a strong distinction is not possible.

This work focuses on the problem of reconstructing the geometry properties and the shape of an unknown and uncooperative space object, optimizing the acquisition of all the information needed to create a map of the target. Considering the pose as the position and attitude state of an object, the pose, motion and inertia matrix estimation is one of the first steps to develop a complete autonomous system for proximity operations with uncooperative clients. Several works have been developed to understand problems such as the reconstruction of geometric properties, inertia or dynamics of a space object or a small body, including asteroids and comets. Different from previous contributions, we have introduced a reinforcement learning agent that, to accomplish a shape reconstruction task, directly translate environment information to a thrust action that controls the spacecraft motion. Machine learning techniques are becoming increasingly popular in astrodynamics applications, thanks to their optimization

capabilities and very high versatility. Some notable works include applications for planetary landing [2], planetary navigation [3] and studies of small bodies [4, 5]. Instead, concerning space proximity operations, the most relevant works treated 3-degree-of-freedom and 6-degree-of-freedom closed-loop control applied to satellite rendezvous missions with cooperative clients [6, 7].

The autonomous exploration of an unknown environment has been studied especially in the field of robotics. The most popular formulation is called Simultaneous Localization and Mapping (SLAM) problem, that has been firstly developed in 1986. In SLAM problems an agent continuously estimates the map of an unknown environment while simultaneously localizing itself [8]. After the SLAM problem has been fully characterized, in the late '90s, Feder et al. have added the task of exploration planning creating the *active* SLAM problem. In active SLAM, localization, mapping and planning are strictly coupled [9]. Active SLAM applications aim to automate data collection that is needed to create the highest quality map with the least time and cost possible. In astronautics, active SLAM is considered in planetary rover path-planning [10]; the literature on on-orbit SLAM problems is more scarce: some innovative works have been developed for small bodies autonomous mapping in the last few years. One of the first works developed treats a Structure From Motion algorithm along with a real-time optimization to retrieve the body rotation and center of mass trajectory, assuming a complete knowledge of spacecraft pose [11]. All the works try to overcome ground-in-the-loop operations, but require high computational effort. Recent articles have proposed mathematical formulations based on Partially Observable Markov Decision Process (POMDP) to solve the active SLAM problem [12].

Reinforcement Learning (RL) and Deep Reinforcement Learning (DRL) techniques have been widely used to design agent's policies in planning problems [13]. In the robotic field, the first authors to propose RL to optimize a trajectory for map exploration have been Kollar et al. [14]. Tools such as RL and DRL are very useful especially when combined with the capabilities of neural networks: they allow to deal with complex planning problems that present wide and continuous state spaces. A recent work applies DRL to the problem of cars end-to-end driving [15]. In the space exploration field, in the last two years, two works by Chan et al. [4] and Piccinin [5] have proposed to use DRL for mapping small bodies. Both works are based on the work by Pesce et al. [12] and design a neural network to optimize the collection of the image data. These kind of DRL applications are well suited for spacecraft autonomy problems; this is particularly true in scenarios when human intervention may result to be too slow (for example, because of communication gaps) and the control action must be fast, e.g. rendezvous, proximity operations, actuators failures.

Relative Dynamics

Linearized Eccentric Model

Assuming that the relative distance ρ between the spacecraft and the target object is much lower than the distance between the target and the main attractor, r_T, that is $\rho/r_T << 1$, then it is possible to linearize the equations of motion [16]. Common use

is to express the equations in the Local Vertical Local Horizontal (LVLH) reference frame centred in the target object center of mass [16]. The resulting equations of motion are shown in Eq. 1.

$$\ddot{x} = \left(\frac{2\mu}{R^3} + \frac{h^2}{R^4}\right) x - \frac{2(\mathbf{V}\cdot\mathbf{R})h}{R^4} y + \frac{2h}{R^2}\dot{y} \tag{1a}$$

$$\ddot{y} = \left(\frac{h^2}{R^4} - \frac{\mu}{R^3}\right) y + \frac{2(\mathbf{V}\cdot\mathbf{R})h}{R^4} x - \frac{2h}{R^2}\dot{x} \tag{1b}$$

$$\ddot{z} = -\frac{\mu}{R^3} z \tag{1c}$$

$\mathbf{R}$, $\mathbf{V}$, R and h are respectively the position vector, velocity vector, position magnitude distance and angular momentum of the object in its Earth-centered inertial reference frame; x, y and z are the Cartesian components of the relative position of the spacecraft with respect to the target in LVLH frame.

Rotational Model

We express rotational dynamics in the LVLH reference frame centered in the target object center of mass, maintaining the same frame of the translational motion. Rotational dynamics may be expressed through the Euler's equations of motion, that are written in body-reference frame [17]. Applying the small angles approximation and the transpose theorem to the Euler's equations, the differential equations for the rotational dynamics of the target body in LVLH frame can be written as in Eq. 2. The products between infinitesimal terms can be neglected, leading to the final model in Eq. 3.

$$\begin{cases} I_x(\ddot{\alpha}_x - \dot{\alpha}_y n) + (I_z - I_y)(\dot{\alpha}_z + n)(\dot{\alpha}_y + \alpha_x n) = 0 \\ I_y(\ddot{\alpha}_y + \dot{\alpha}_x n) + (I_x - I_z)(\dot{\alpha}_z + n)(\dot{\alpha}_x - \alpha_y n) = 0 \\ I_y\ddot{\alpha}_z + (I_y - I_x)(\dot{\alpha}_x - \alpha_y n)(\dot{\alpha}_y + \alpha_x n) = 0 \end{cases} \tag{2}$$

$$\begin{cases} I_x\ddot{\alpha}_x + n(I_z - I_y - I_x)\dot{\alpha}_y + n^2(I_z - I_y)\alpha_x = 0 \\ I_y\ddot{\alpha}_y + n(I_x + I_y - I_z)\dot{\alpha}_x + n^2(I_z - I_x)\alpha_x = 0 \\ I_z\ddot{\alpha}_z = 0 \end{cases} \tag{3}$$

I_x, I_y and I_z are the principal moments of inertia, α_x, α_y and α_z are the target object rotational angles in LVLH frame, n is the target mean motion.

Similarly to other studies, we assume that the spacecraft camera always points to the object center of mass [4, 5]. This assumption leads to neglect the attitude dynamics of the spacecraft and therefore the attitude control.

Visibility Model

During training of the RL agent, equations of motion in Eq. 1 are numerically propagated. A *visibility model* generates, at each time step, the portion of the target object that the spacecraft can observe. The model of the target object is rendered with a set of faces, so the *visibility* model locates the faces that are in direct line of sight (LoS) with the spacecraft camera and under favorable light conditions. For example a face in LoS with the camera is not visible if fully in shadow. While the model of a target uncooperative object would be unknown, we may use randomization of the target object during training to induce generalization of the policy learned by the agent.

Figure 1 shows an object composed by N faces: the faces colored in red are in the field of view (FOV) of the camera (represented by the light blue cone) and with a good Sun illumination; the faces in yellow are still in the camera FOV but not fulfilling the requirement of the sun incidence angle, as will be later defined. Instead, the faces not colored are those ones outside the FOV of the camera.

Quality of illumination is determined as a function of the incidence angles between the face normal, the Sun and the camera directions. Incidence angles are considered in the formulation of the reward model and are explained in the next section.

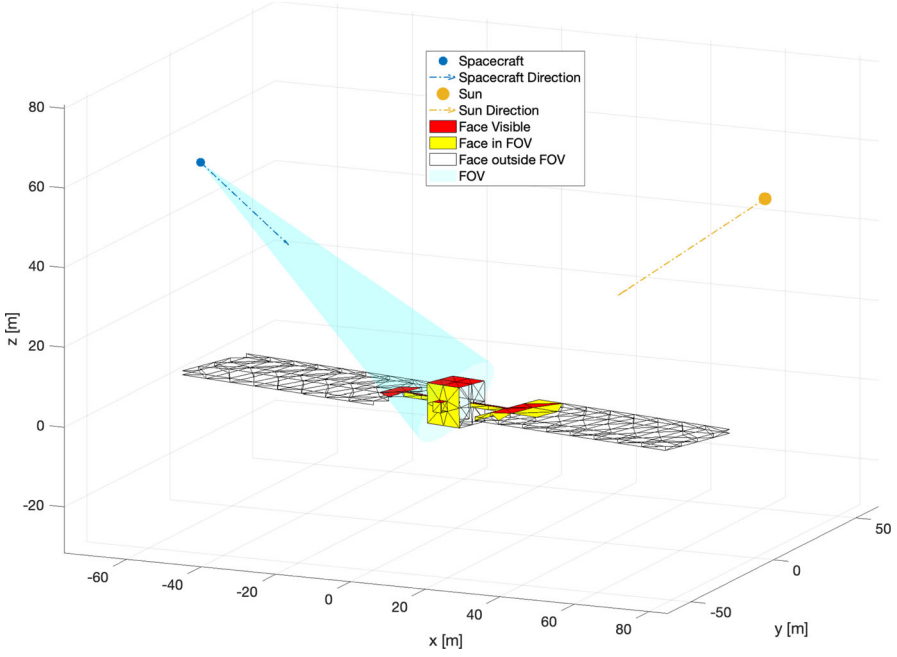


Fig. 1 Visibility model

Image Processing

There are different means to reconstruct the geometry of a small object from images. High fidelity topography is generally possible only when the images are taken very close to the object. *Stereophotoclinometry* (SPC) and *Stereophotogrammetry* (SPG) are state of the art methods that guarantee high resolution models given a sufficient number of observations. We assume SPC as the image processing reference method. SPC is an evolution of shape-from-shading approaches and links photometry to stereoscopy. In other words, from maplets initialization and images with navigation data, brightness is extracted to create a photometric model. Then, slopes, defined as the inclination of the maplets with respect to the ideal plane in which the photo lays, and albedo are estimated from the photometric model [18]. Stereoscopy defines the camera model.

The requirements to obtain high-quality SPC models are derived from the studies done by Gaskell et al. [19]. Such requirements, which inform the definition of the reward model for our RL agent, include:

- A minimum of three images per face (typically > 40).
- Emission angle, defined as the angle between the camera direction and the normal direction to the face, should be maintained around $\sim 45°$ (optimal range 35°-48°, limit range 5°-60°).
- Incidence angle, defined as the angle between the Sun direction and the normal direction to the face, should be maintained around $\sim 45°$ (optimal range 30°-50°, limit range 0°-70°).
- Variation in illumination conditions (optimal range 40°-90°, limit range 10°-120°). Variation in illumination conditions is a direct function of the variation of the Sun incidence angle. This parameter is difficult to track in a reward function, thus adding significant complexity in the agent training; for this reason it is not considered in the reward model.

In the *optimal*, we expect the best image possible. In the *limit* range, instead, we expect a lower quality image, yet usable for the image process. In the reward model the emission angle is called *camera incidence angle* and the incidence angle is called *Sun incidence angle*. It will be noticed that the *optimal* ranges considered in the reward model are wider than the ones here presented, while the *limit* ranges have been maintained equal. That was a design choice aimed to accelerate the learning through higher rewards for a wider space of possible states.

Autonomous Navigation

Reconstructing the geometry of the target body and the relative dynamics may require planning spacecraft trajectories within a uncertain environment. The problem of constructing and updating a map of an unknown object or environment while simultaneously tracking the spacecraft location is called Simultaneous Localization and Mapping (SLAM) [20, 21]. The SLAM problem may also include trajectory planning, which is known as active SLAM [22]. Active SLAM may be solved as

a Partially Observable Markov Decision Process (POMDP). Applications of active SLAM theory have been recently studied within the context of small body exploration missions [4, 5, 12].

Partially Observable Markov Decision Process

POMDP derives from the simpler Markov Decision Process (MDP), when part of the information characterizing the agent state is unknown [23]. In this case, the decision maker is only aware of partial information about the environment. A POMDP problem can be described as a six-tuple, $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \Omega, \mathcal{O})$, where: $\mathcal{S}$ is the space of all possible states in the environment, $\mathcal{A}$ is the space of all possible actions that can be taken in all the states of the environment, $\mathcal{R}_{\mathcal{A}}(s, s')$ is the reward space, function of all the pairs between two following states, $\mathcal{T}_{\mathcal{A}}(s, s')$ is the transition probability related to all the state-action pairs, Ω is the space of possible observations and $\mathcal{O}(o' \mid a, s')$ is the probability of making a particular observation, taking an action that leads to a particular new state.

An exact solution to a POMDP problem means obtaining an optimal action for each possible observation over the world states. The optimal action maximizes the expected reward of the agent over a possibly infinite horizon. Exact solutions exist only for a very thin class of POMDP problems. This complexity leads to an approximation of the general problem in order to make it more tractable: a POMDP problem can be reduced to a MDP problem exploiting a new formulation called *belief-space* MDP. This new formulation is characterized by a four-tuple $(\mathcal{B}, \mathcal{A}, \mathcal{R}_{\mathcal{A}}, \mathcal{T}_{\mathcal{A}})$.

- $\mathcal{B}$ is the belief space, where the *belief* is defined as $b = p(s \mid h)$, that is the probability of being in a state s after the history h of all the previous states.
- $\mathcal{A}$ is the original action space.
- $\mathcal{R}_{\mathcal{A}}(b, b')$ is the reward space in the new formulation.
- $\mathcal{T}_{\mathcal{A}}(b, b')$ is the belief transition function, that is the probability of reaching a new belief b' from a belief b taking an action a.

As before, the goal is always the maximization of the long-term reward. Two different optimal policies can be defined depending on whether the problem episode is finite or infinite. The first case is called *finite horizon* problem and the optimal policy is defined in Eq. 4.

$$\pi_\star = \arg\max_{\pi} \mathbb{E}_\pi \left[\sum_{k=0}^{T} \mathcal{R}_{\mathcal{A}}(b_k, b_{k+1}) \right] \tag{4}$$

The second case is named as *infinite horizon* problem and its optimal policy is defined exploiting a discount factor $\gamma \in [0, 1]$, as in Eq. 5.

$$\pi_\star = \arg\max_{\pi} \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k \mathcal{R}_{\mathcal{A}}(b_k, b_{k+1}) \right] \tag{5}$$

In our work the belief state space, the action space and the reward model are defined and described in *Guidance Architecture* Section.

Deep Reinforcement Learning

In this work two DRL methods have been compared: Deep Q-learning Network (DQN) and Advantage Actor-Critic (A2C).

DQN - Deep Q-Learning Network

The DQN is a deep reinforcement learning method proposed by DeepMind [24]. It combines classic reinforcement learning techniques with deep neural networks to solve decision problems with wide state space. In DQN, experiences are collected by playing several episodes, in which actions are chosen following an $\varepsilon - greedy$ policy. This policy consists in selecting an action randomly with probability ε or choosing one that maximizes the action-value function $Q(s, a)$ with probability (1-ε). The ε value is varied between an initial and final value, usually 1 and 0, with a linear or non-linear function, in order to exploit the action-value function more often when the network starts to approximate it better. Two mechanisms of novelty introduced by DQN are the *experience replay* and the *target network*. The first is the procedure of storing selected agent experiences in a replay memory R. Then the action-value function is updated by randomly sampling from the memory a defined number T of experiences. Such a randomized sampling is useful to reduce the correlation between the updates and the experiences. The target network is an additional network used to approximate the Q-value, different from the trained network that computes the actions. This mechanism is aimed to stabilize the learning process.

A2C - Advantage Actor-Critic

A2C methods represents an improvement of the basic Actor-Critic algorithm. Actor-Critic methods merge the advantages of value-based methods (i.e. DQN) and policy-based methods (i.e. Policy gradient) [13]. Actor-Critic methods split the agent model into two parts, one that computes the action and the other that computes the action-value function[1]. The two components are then called:

- *Actor*. The actor function takes as input the state and generates as output the *perceived* best action, thus, controlling the agent behaviour.
- *Critic*. The critic function evaluates the action taken by the actor network computing the corresponding action-value function.

Both the actor and the critic are function approximators, one for the action policy and the other for the Q-value function. When using Artificial Neural Networks (ANN) as function approximator, the training of the actor and critic network is performed separately and exploits the gradient ascent for parameter update. During the training process the actor learns to produce better actions while the critic learns to evaluate better those actions. The update may occur both at the end of each time step

[1] S. Karagiannakos, "The idea behind Actor-Critics and how A2C and A3C improve them", Nov 17, 2018, https://theaisummer.com/Actor_critics/

or at the end of an episode. Among actor-critic methods, a recent algorithm has displayed promising performance on continuous control problems: A2C - Advantage Actor-Critic. A2C employs an *advantage* value function $A(s_t, a_t)$, which is created by subtracting from the action-value function $Q(s_t, a_t)$ the value function $V(s_t, a_t)$, as in Eq. 6. All these quantities depend on the state s and the action a at the time step t.

$$A(s_t, a_t) = Q(s_t, a_t) - V(s_t, a_t) \tag{6}$$

The advantage function represents how better an action is compared to all other actions at a given state; it is different from the value function that instead captures how good it is to be in a particular state. Exploiting the Bellman equation for $Q(s, a)$, it is possible to write the advantage function $A(s, a)$ only in dependence of the value-function $V(s, a)$, the reward, r, and a discount factor, γ.

$$A(s_t, a_t) = \mathbb{E}[r_{t+1} + \gamma V(s_{t+1})] - V_v(s_t) = r_{t+1} + \gamma V_v(s_{t+1}) - V_v(s_t) \tag{7}$$

In A2C, the critic network is used to approximate the advantage value function in Eq. 7. Instead of learning directly the Q-values, the A2C critic learns the A-values; the evaluation of an action is not only based on how good the action is but also based on how better it can be. The direct consequence is the reduction of the high variance of the policy at each update and, therefore, the stabilization of the learning model. The update of the parameters may happen in two moments. TD-0 defines an A2C algorithm that updates the network parameters at each time step; TD-0 is an *on-policy* learning, because the behaviour policy is equal to the update policy. Instead, TD-n defines an A2C algorithm in which the network update step is taken only at the end of each episode. In this case, the learning is *off-policy*, because the agent learns when the episode is concluded, and therefore the update and behaviour policies are different.

Guidance Architecture

Figure 2 displays the guidance architecture considered in this work. The core of the architecture - the *Autonomous Decision Process* - is preceded by a pre-processing phase, which depends on the relative pose, the object geometry, the camera characteristics and the external environment conditions, i.e. the instantaneous Sun direction. The pre-processing feeds the autonomous decision block, that at each time step produces a control policy aimed to maximize the mapping of the target. The planning policy is based on DRL and A2C.

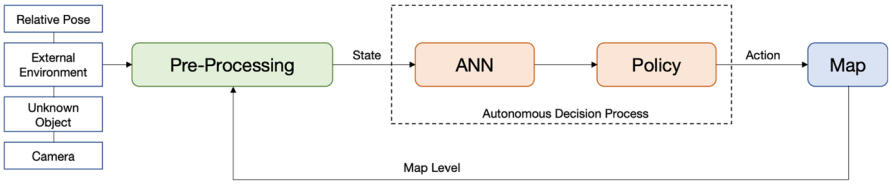


Fig. 2 On-board planning architecture scheme

Environment Model

In our work, the environment is characterized by three main components: the spacecraft, the unknown and uncooperative target object and the Sun. The Sun's relative dynamics depends only on the initial epoch; instead, the spacecraft-object relative dynamics is directly influenced by the actions taken by the agent. The spacecraft and the target object orbit around the Earth in orbits that may be eccentric. Two main assumptions are introduced: the spacecraft camera constantly points toward the target center of mass and the Earth dependence of the illumination condition is neglected, e.g. eclipses and reflections are ignored. Environment information is employed to create the state space, action space and reward models needed to define the autonomous decision process algorithm, as explained in the following.

Reward Model

One of the most significant part of a RL architecture is the definition of the reward model. RL agents typically learn a policy with the goal of maximizing the cumulative reward. Multiple objectives may be present in the reward model. In general terms, our goal is to define an agent that achieves high quality of the map of the target object together with a fast and safe process. Quality of the map depends on the adopted mapping technique. In the present work, reward is defined to ease SPC, in terms of Sun and camera exposition. In addition, the agent has to perform mapping operations in the shortest time possible, avoiding regions of the space that are considered dangerous and may end the episode. First, a *basic* reward model is developed. The *basic* reward model gives the same importance to the goodness of the image taken at each time step and to the survival of the spacecraft. In this way, the real goal of the mission - achieving the total map of the target object - *may not be* fully accomplished. *Then*, an *improved* reward model has also been analysed *placing* more *emphasis* on quality of the map.

Basic Reward Model

This model defines the total reward computed at each time step as function of the camera incidence angle, the Sun incidence angle, the spacecraft position and the time of flight.

- **Sun incidence score.** The Sun incidence angle η is the angle between the Sun direction relative to the target object and the normal to the face considered. The Sun incidence angle should be between 10° - 60°, to avoid shadows or excessive brightness. Values outside that interval may correspond to conditions that degrade the quality of the image. A unitary score is assigned for each of the faces in correct exposition among the n faces that are in the FOV of the spacecraft camera. Then, the overall score is given by the average expressed in Eq. 8.

$$R_\eta = \frac{1}{n} \sum_{j=0}^{n} r_{\eta,j} \tag{8}$$

where $r_{\eta,j}$ is the Sun incidence score for a single face and is defined in Eq. 9.

$$r_\eta = \begin{cases} 1 & \text{if } 10^\circ \leq \eta \leq 60^\circ \\ \frac{1}{10}\eta & \text{if } 0^\circ \leq \eta \leq 10^\circ \\ 7 - \frac{1}{10}\eta & \text{if } 60^\circ \leq \eta \leq 70^\circ \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

- **Camera incidence score.** The camera incidence angle ε is the angle between the normal to the face and the camera direction. This angle should be maintained between 10° - 50°. A unitary score is assigned to each face in the FOV. Then, the overall score is given by the average expressed in Eq. 10.

$$R_\varepsilon = \frac{1}{n}\sum_{j=0}^{n} r_{\varepsilon,j} \tag{10}$$

where $r_{\varepsilon,j}$ is the camera incidence score for a single face and is defined in Eq. 11.

$$r_\varepsilon = \begin{cases} 1 & \text{if } 10^\circ \leq \varepsilon \leq 50^\circ \\ \frac{1}{5}\varepsilon - 1 & \text{if } 5^\circ \leq \varepsilon \leq 10^\circ \\ 6 - \frac{1}{10}\varepsilon & \text{if } 50^\circ \leq \varepsilon \leq 60^\circ \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

- **Position score.** Negative scores are given when the spacecraft escapes from the region defined by a minimum and maximum distance, D_{min} and D_{max}. The position score is defined in Eq. 12.

$$R_d = \begin{cases} -100 & \text{if } d \leq D_{min} \text{ or } d \geq D_{max} \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

- **Time of flight score.** This score rewards the agent for obtaining the best map within a definite time window, identified by bounding the time of flight between T_{min} and T_{max}.

$$R_t = \begin{cases} 0 & \text{if } \Delta t < T_{min} \\ 1 & \text{if } T_{min} \leq \Delta t \leq T_{max} \end{cases} \tag{13}$$

There is no score for $\Delta t > T_{max}$ because the simulation is stopped if reaches T_{max} and therefore the episode ends.

At each time step, the total reward is computed as in Eq. 14. No weights are assigned to each of the different scores: their reward values are already tuned to better match the agent objectives and we preferred to avoid placing further importance on one score over the others. Later, we will demonstrate how to guide the agent towards an objective by directly changing the reward model.

$$R_k = R_\eta + R_\varepsilon + R_d + R_t \tag{14}$$

Improved Reward Model

The *improved* model is developed to give more importance on mapping level and fuel consumption during reward attribution.

- **Mapping level score.** The faces that have both good Sun and camera exposition are the ones that generate an improvement in the quality of the map. The maximum level defined for the map consists in having each faces photographed $N_{accuracy}$ times. Therefore, at each time step the mapping level, $Ml_{\%}$, can be computed considering how many good photos of each face have been taken until that moment. The corresponding reward score is defined in Eq. 15 and at each time step k rewards the agent for increasing the mapping level over the current value, $Ml_{\%,k-1}$.

$$R_m = \begin{cases} 1 & \text{if } Ml_{\%,k} > Ml_{\%,k-1} \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

- **Thrust score.** This score considers the number of times that thrusters are fired, n_f. Two thresholds are assigned: the first is a medium level threshold, l_{mid}, beyond which is still possible to fire the thrusters; the second threshold, l_{max}, defines the maximum number of firings, beyond which the thrusters cannot be used again. The score is defined in Eq. 16.

$$R_f = \begin{cases} -10 & \text{if } l_{mid} \leq n_f < l_{max} \\ -100 & \text{if } n_f \geq l_{max} \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

At each time step the total reward is given by Eq. 17.

$$R_k = R_\eta + R_\varepsilon + R_d + R_t + R_m + R_f \tag{17}$$

Note that, again, no weights have been assigned to the different scores.

Action Model

The action space is the space from which the agent chooses its actions. The *control interval*, Δt=20s, is the time interval between two state transitions. The agent interacts with the environment through a direct control of the spacecraft thrusters. We assume that the spacecraft can thrust in each of the six reference frame directions, namely $x, -x, y, -y, z, -z$. The option of null action is also available to the agent. This formulation renders a direct and continuous control on the trajectory, consistent with an active SLAM problem. The action space at each time interval is defined in Eq. 18. Therefore, at each time step, a choice among the seven possible actions is requested to the agent.

$$\mathcal{A} = [T_{x_+}, T_{x_-}, T_{y_+}, T_{y_-}, T_{z_+}, T_{z_-}, 0] \tag{18}$$

The control action values are fixed in terms of acceleration, $\bar{a} = 0.001\text{m/s}^2$; therefore, the actions directly affect the translational equations of motion of the spacecraft defined in Eq. 1. For example, if the agent selects the following action,

$a_k = [0, 0, 1, 0, 0, 0, 0]$, the relative translational equations of motion will become:

$$\ddot{x} = \left(\frac{2\mu}{R^3} + \frac{h^2}{R^4}\right) x - \frac{2(\mathbf{V}\cdot\mathbf{R})h}{R^4} y + \frac{2h}{R^2}\dot{y} \tag{19a}$$

$$\ddot{y} = \left(\frac{h^2}{R^4} - \frac{\mu}{R^3}\right) y + \frac{2(\mathbf{V}\cdot\mathbf{R})h}{R^4} x - \frac{2h}{R^2}\dot{x} + a_{y_+} \tag{19b}$$

$$\ddot{z} = -\frac{\mu}{R^3} z \tag{19c}$$

Similar equations may be written if a different action is selected.

State Model

The state space is designed to synthesize essential information needed by the agent to decide what is the best action to take. The state space is shown in Eq. 20.

$$\mathcal{S} = \begin{bmatrix} \boldsymbol{x} \\ \dot{\boldsymbol{x}} \\ \boldsymbol{\alpha} \\ \dot{\boldsymbol{\alpha}} \end{bmatrix} \tag{20}$$

In Eq. 20, $\boldsymbol{x}$ and $\dot{\boldsymbol{x}}$ are the relative position and velocity between the spacecraft and the target object, while $\boldsymbol{\alpha}$ and $\dot{\boldsymbol{\alpha}}$ are the relative angular position and velocity (assuming small rotations).

Preliminary Analysis

We conducted a series of preliminary analyses to inform the selection of the hyperparameters that define a RL agent as well as the learning process. In particular, we explored fixed versus random initial conditions, different control actions, reward model parameters, object shape definition and the DRL hyperparameters selection. The episode stopping criteria are four: the spacecraft exits the safe region delimited by D_{min} and D_{max}, the agent fully completes the map, the simulated episode reaches a maximum time of 6h or (only for the *improved reward*) the maximum number of firings is reached.

Training in a simulated environment requires to virtually reconstruct the shape of the target object. Target object geometry is rendered by a polyhedron shaped with a triangular mesh. The training simulations are performed on a rectangular parallelepipedon made up of 344 triangular faces with a whole length of 15m, shown in Fig. 3.

DRL Methods Trade-Off

The methods tested are: A2C with temporal difference TD-0, A2C with temporal difference TD-n, and DQN with batch optimization. Tests are performed assuming the *basic* reward model and starting from fixed initial state. Table 1 summarizes the results of the test campaign. A simulation is considered *promising* when displaying a

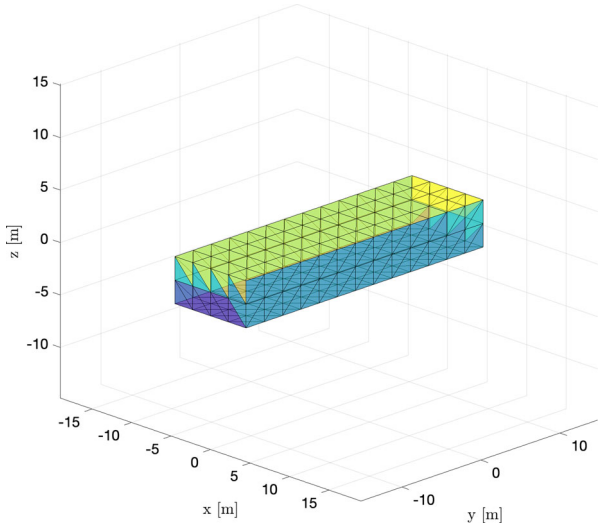


Fig. 3 Shape of target object in simulations

growing score with the number of episodes. The results presented are in function of the learning rate and the two hidden layers; the input and output layers are obviously maintained constant with 12 and 7 elements, which is the length of the state vector - Eq. 20 - and action vector - Eq. 18 - respectively. The choice to select two hidden layers with equal number of neurons is inspired to the ANN formulation of other DRL applications for space SLAM problems [3–5].

The sizes of the networks and the learning rates tested for each of the methods are comparable. Within our simple space, the most performing networks are the 100-100 or the 500-500, while large networks as the 1000-1000 seem to perform poorly on the selected number of 10000 episodes [25]. All of the methods seem to perform better with a smaller learning rate, i.e. 10^{-5} works better than 10^{-3}. Among the three methods, DQN performs the worst with the selected network architecture. Such outcome is expected, as, for continuous state-action problems, A2C generally surpasses the performance of DQN in terms of computational time and simulation length [25]. A2C with TD-n, hence an off-policy methodology, represents the best option among the trades explored.

Following the analysis in Table 1, learning capabilities of the most promising methods are further examined. To do so, the number and the length of the simulations is increased. Table 2 summarizes the results, noting the following characteristics.

- The most promising learning rates are of the order of $10^{-5} - 10^{-6}$.
- The selected number of episodes - 10000 - is insufficient to achieve a high mapping level.
- The trend of the TD-0 policy is worse trend than the trend of the TD-n policy.

Based on the trades presented in Tables 1 and 2, the A2C method with TD-n policy optimization step and 12-100-100-7 configuration is selected to continue the analysis.

Table 1 First trade-off analysis

Method	Optimization	ANN	Learning Rate	Promising
A2C	TD-0	12-10-10-7	$10^{-3} - 10^{-5}$	No
		12-100-100-7	$10^{-3} - 10^{-5}$	No
		12-500-500-7	$10^{-3} - 10^{-5}$	Yes
		12-1000-1000-7	$10^{-3} - 10^{-5}$	No
A2C	TD-n	12-100-100-7	$10^{-3} - 10^{-5} - 10^{-6}$	Yes
		12-500-500-7	$10^{-3} - 10^{-5} - 10^{-6}$	Yes
		12-1000-1000-7	$10^{-3} - 10^{-5} - 10^{-6}$	No
DQN	Batch	12-100-100-7	$10^{-3} - 10^{-5}$	No
		12-500-500-7	$10^{-3} - 10^{-5}$	No
		12-1000-1000-7	$10^{-3} - 10^{-5}$	No

Test Results

The workflow of the following test campaign is divided into four different but interconnected test cases listed in Table 3, each as a function of the initial condition and the reward model.

The definition of a baseline policy may facilitate understanding the quality of the training results. Our selected baseline policy is produced by a randomly initialized neural network without performing any optimization step. For the baseline policy, the expected result is a moving average score that remains constant during all the episodes. Two variables have been tracked:

1. The average score obtained by a randomly generated network. This quantity is strictly correlated to the size of the network and to the weights and biases randomly generated to initialize the network. A drawback is that different weights and biases can generate different average values.

Table 2 Second Trade-off Analysis

Method	Optimization	ANN	Learning Rate	Episodes	Promising
A2C	TD-0	12-500-500-7	10^{-3}	10000	No
			10^{-5}	10000	Yes
A2C	TD-n	12-100-100-7	10^{-3}	10000	No
			10^{-5}	10000	Yes
		12-500-500-7	10^{-3}	10000	No
			10^{-5}	10000	Yes
			10^{-6}	10000	Yes

Table 3 Test Campaign Workflow

Test Case	Initial Condition	Reward Model
1	Fixed (FIC)	Basic (BR)
2	Random (RIC)	Basic (BR)
3	Fixed (FIC)	Improved (IR)
4	Random (RIC)	Improved (IR)

2. The average mapping level. This quantity may complement the information provided by the average score, when the latter is difficult to interpret.

The study of the no-learning average score and mapping level is performed using the configuration selected after the trade-off analysis (A2C-TDn with 12-100-100-7). In Table 4, the results for all the cases are summarized.

Test Case 1: Fixed Initial Conditions - Basic Reward

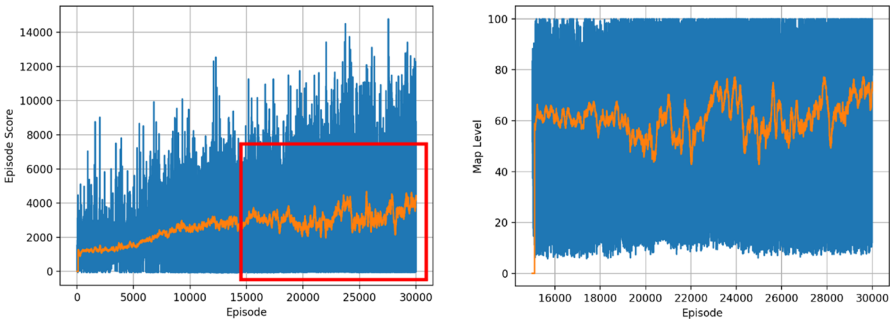
We first trained the agent on the *basic* reward function. The agent configuration used to train the agent is the A2C-TDn. The network configuration is composed by an input layer of 12 elements, two hidden layers of 100 elements each and the output layer that is composed by the 7 possible actions. This is the first of the four steps of the workflow in Table 3.

In Fig. 4 the average score and mapping level, referred to the last 15000 episodes in the red box, are displayed. The growth of the average score remains almost gradual and constant, even if, particularly in the last 10000, it presents more instability. Even if the basic reward puts more value on spacecraft survival, the mapping level trend follows the score trend. We may, then, infer that higher scores corresponds to higher mapping levels. Although an overall growing trend, the result still presents strong fluctuations in the mapping level trend. Such fluctuations may be attributed to multiple factors: the balance between *exploration* and *exploitation* in the learning algorithm, the fact that there is no score directly connected to the mapping level in the reward model and also the possible limited capability of the ANN architecture that could induce loss function saturation.

The validity of these results is confirmed by comparison to the no-learning results. For the FIC-BR agent structured with a 12-100-100-7 network configuration, the

Table 4 No-learning average scores

Problem	ANN	Average Score	Average Mapping Level
Fixed IC - BR	12-100-100-7	~ 450	$\sim 30\%$
Random IC - BR	12-100-100-7	~ 100	$\sim 16\%$
Fixed IC - IR	12-100-100-7	~ 40	$\sim 12\%$
Random IC - IR	12-100-100-7	~ -40	$\sim 10\%$



(a) Average Score (b) Average Mapping Level

Fig. 4 Basic Reward - Fixed IC. ANN: 12-100-100-7. Learning Rate: 10^{-5}

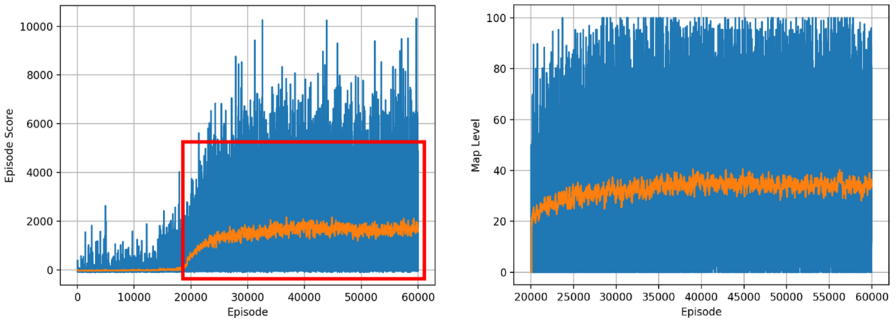
baseline score is $\sim$ 450 and the baseline mapping level is $\sim$ 28%. After training, the agent reaches a score of about 4000 and a mapping level greater than the 75%. In Appendix an example of a trajectory is shown.

Test Case 2: Random Initial Conditions - Basic Reward

The second step of the work consists in training an agent with the *basic* reward on a randomly selected initial state. The learning method is always the A2C-TDn, exploiting an ANN configuration of 12-100-100-7.

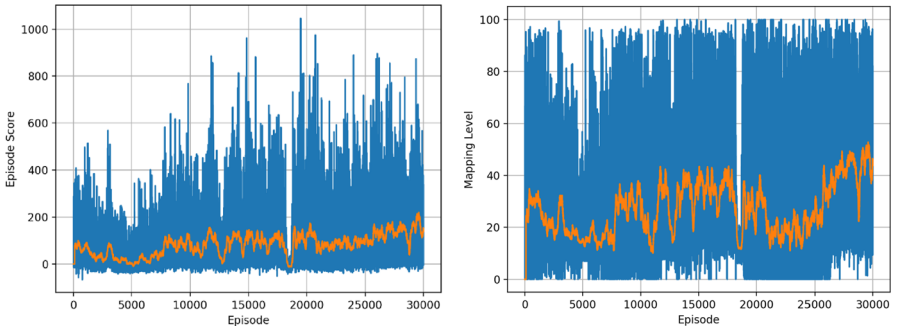
When task complexity increases, finer policy update may be required; hence a smaller learning rate may perform better than a higher one. In our tests this hypothesis is confirmed and hence we decreased the learning rate from 10^{-5} to 10^{-6}. Figure 5 plots the resulting average score and mapping level, referred to the last 40000 episodes in the red box.

In Fig. 5, score and mapping level are not as high as the simpler case of fixed initial condition (see Fig. 4). This is expected since the problem is more complex due to the wider space of the possible initial states. Nevertheless, the average score and mapping level values reached through the training are significantly greater relative to



(a) Average Score (b) Average Mapping Level

Fig. 5 Basic Reward - Random IC. ANN: 12-100-100-7. Learning Rate: 10^{-6}



(a) Average Score (b) Average Mapping Level

Fig. 6 Improved Reward - Fixed IC. ANN: 12-100-100-7. Learning Rate: 10^{-6}

those obtained by the no-learning agent in the same configuration. For the no-learning case, the average score value is $\sim$ 100 and the average mapping level is $\sim$ 16%. Instead, for trained agent the score and mapping level are about 2000 and 37%.

Test Case 3: Fixed Initial Conditions - Improved Reward

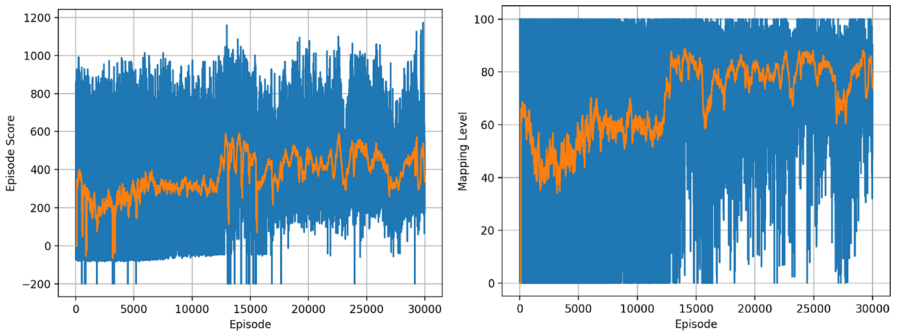
Training on the *basic* reward model preliminary demonstrates meaningful A2C agent behaviour that enables larger survival in the environment.

Training on the *improved* reward model aims to increase the average mapping level. Results are obtained with the A2C-TDn architecture, exploiting a network configuration of 12-100-100-7. In this case, the learning rate is 10^{-6}. Figure 6 portrays the history of the average score and mapping level during training.

Note that is not possible to directly compare the average score between the basic and improved reward models, because the scoring scale is different. For the fixed initial conditions with improved reward, the baseline for the no-learning network is about 40 as average score and about 10% of mapping level. The score level settles down to almost 180, that corresponds to a mapping level greater then 40%. Even if the initial conditions are the same of the *Test Case 1*, the fact that the reward model is different determines a learning curve that is unable to reach the same final results after the same amount of episodes. It may be, then, useful to exploit transfer learning to improve the test case results.

Transfer Learning

Transfer learning (TL) may be applied to easy training on more complex tests. One of the transfer learning techniques consists in pre-training the RL agent on a simpler task before training on the main task. As pre-task for FIC-IR we select FIC-BR. An RL agent, first trained on FIC-BR, is transferred to FIC-IR without resetting the architecture. Pre-training on FIC-BR enables the agent to learn three main skills: how to survive, what is the best quality of the image and how the environment evolves from



(a) Average Score (b) Average Mapping Level

Fig. 7 Improved Reward - Fixed IC with TL. ANN: 12-100-100-7. Learning Rate: 10^{-5}

that initial condition. All these skills are needed by the agent that aims to optimize policy within the improved reward model.

Figure 7 portrays the evolution of training on FIC-IR after transfer from FIC-BR. In the same fixed number of episodes, the pre-trained agent is capable of achieving a significantly higher average level than the one obtained by the agent without pre-training. The score value reaches about 400 and the mapping level surpasses the 80%, that represents the highest mapping level obtained in this work.

Test Case 4: Random Initial Conditions - Improved Reward

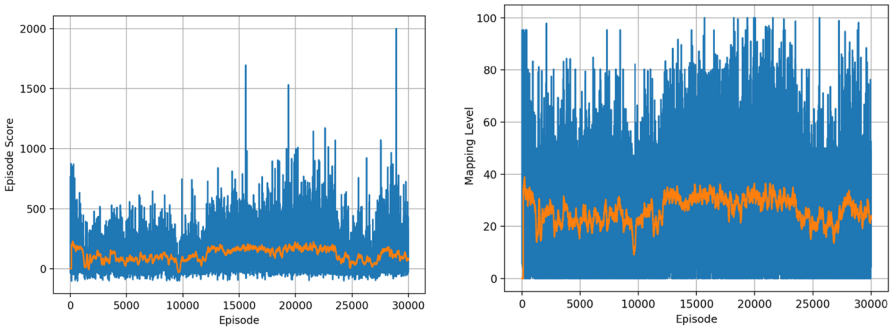
A2C-TDn method that implements a 12-100-100-7 network and learning rate 10^{-5} is trained on RIC-IR. Given the promising results obtained in FIC-IR, transfer learning is also adopted for RIC-IR. Two different pre-trained networks are employed: one pre-trained on FIC-IR with TL (Fig. 7) maintaining the learning rate of 10^{-5}; one pre-trained on FIC-IR, but reducing the learning rate to 10^{-6}. The no-learning baseline average score is about -40 and the average mapping level is about 10% , as shown in Table 4. In the first case, the average score reaches almost 150 and the average mapping level is slightly lower than 30%.

Observing the plots in Fig. 8, we note that the average remains nearly constant throughout most of the training duration. This may be caused by a high learning rate. Therefore, the learning rate is reduced from 10^{-5} to 10^{-6}.

As observable in Fig. 9, although the average levels do not shown a major improvement, oscillations are lowered, reaching an average mapping level slightly higher than 30%.

Sensitivity Analysis

Sensitivity tests help understanding the robustness of the trained agents. Two variations of the environment model are considered.

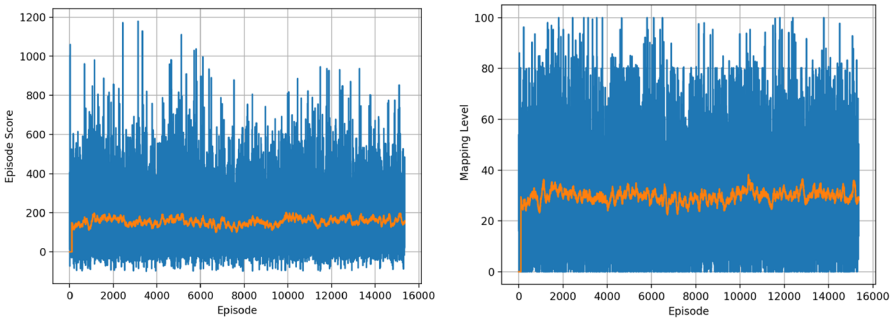


(a) Average Score (b) Average Mapping Level

Fig. 8 Improved Reward - Random IC with TL. ANN: 12-100-100-7. Learning Rate: 10^{-5}

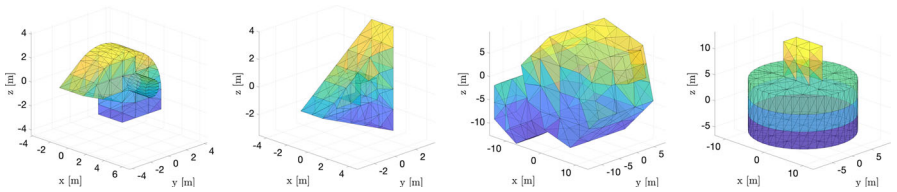
Object Shapes

Sensitivity analysis about the shape of the target object is performed by creating a set of objects, different in shapes, symmetry and dimensions. Then, using the agent obtained from the FIC-IR case trained with TL, a simulation is performed for each of the shapes generated. Based on our analysis, mapping level values ranges from 10% to 85% depending on the shape variation. Since very low mapping level are possible, the trained agent is not robust to large changes in target object shapes. We note that the variable that mostly affects the performance is the maximum dimension of the target object. Therefore, for future studies further analysis on the importance of target object dimension is warranted; in particular, the random shape may be added



(a) Average Score (b) Average Mapping Level

Fig. 9 Improved Reward - Random IC with TL. ANN: 12-100-100-7. Learning Rate: 10^{-6}



(a) Shape A (b) Shape B (c) Shape C (d) Shape D

Fig. 10 Random target object shapes

in the environment model for training. In this way, the agent may be less sensitive to the change of the object shape after roll-out. In Fig. 10, some of the different object shapes used for the sensitivity analysis are shown.

Control Frequency

This type of sensitivity analysis is devoted to understand if the trained agents are dependent on the frequency of control actions. The step between two possible control actions has been fixed at 20s throughout this work. Using the agent trained through FIC-IR with TL, the trend of the mapping level is recorded for increasing time intervals. The mapping level decreases in average with the increasing of the time between two control actions (decreasing control frequency). This means that the policy learnt by the agents is sensible to the value set as time interval between actions [25].

Final Remarks

The present work proposes an autonomous path-planning architecture for uncooperative and unknown target object exploration, aimed to reconstruct the target shape through image processing. The proposed methodology exploits DRL techniques to design an autonomous policy to control the spacecraft trajectory to obtain the highest map quality of the target. The environment is mainly characterized by the relative motion between the spacecraft and the object, Sun illumination conditions and the relative camera-body pose. The large variability of target objects characteristics in terms of shape leaves no space for the definition of a simple and general environment model. Sun illumination and viewing conditions of the body emerge as the key aspects for the realization of the algorithm to optimally retrieve image information. The extensive numerical tests obtained through A2C present promising results. The sensitivity analysis has underlined the need for more extensive training to facilitate policy generalization to a large set of environment configurations. Further comparison analysis with existing methodologies exploited in space objects mapping missions is recommended for future studies. DRL confirms to be a valid approach for solving the decision process problem, merging the advantages of RL and ANNs.

In conclusion, preliminary evidence collected through this work reveals that DRL is a promising approach to autonomous proximity operations.

Appendix: Spacecraft Trajectory Example

In Figs. 11, 12 and 13, three frames of a simulation episode that reaches a mapping level of 100% in the FIC-BR case are shown. In the graphs, the spacecraft, the object and the Sun direction are represented. The mapping level continues to increase throughout the episode. The number of control actions increases almost linearly during the simulation time.

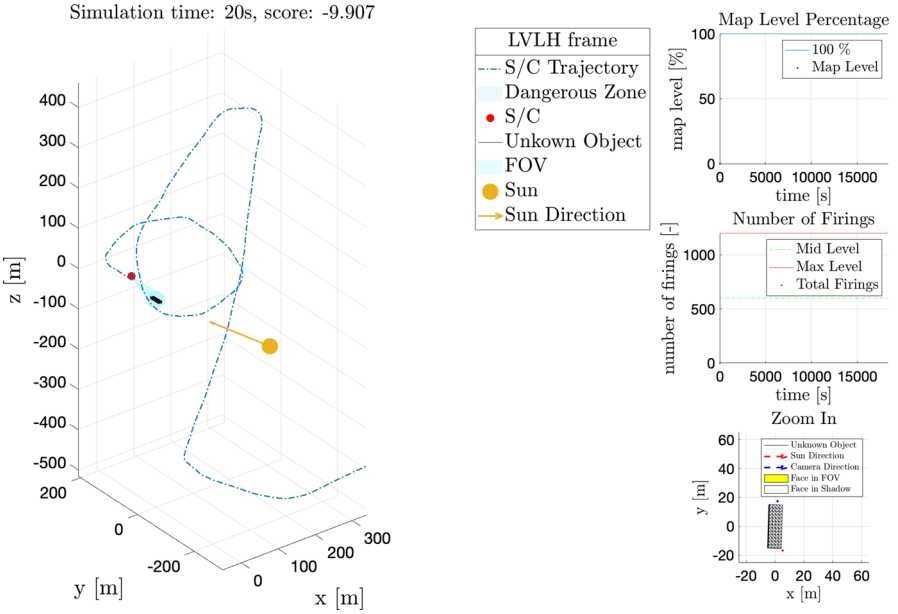


Fig. 11 Example of 100% map level episode: frame 1 (initial epoch)

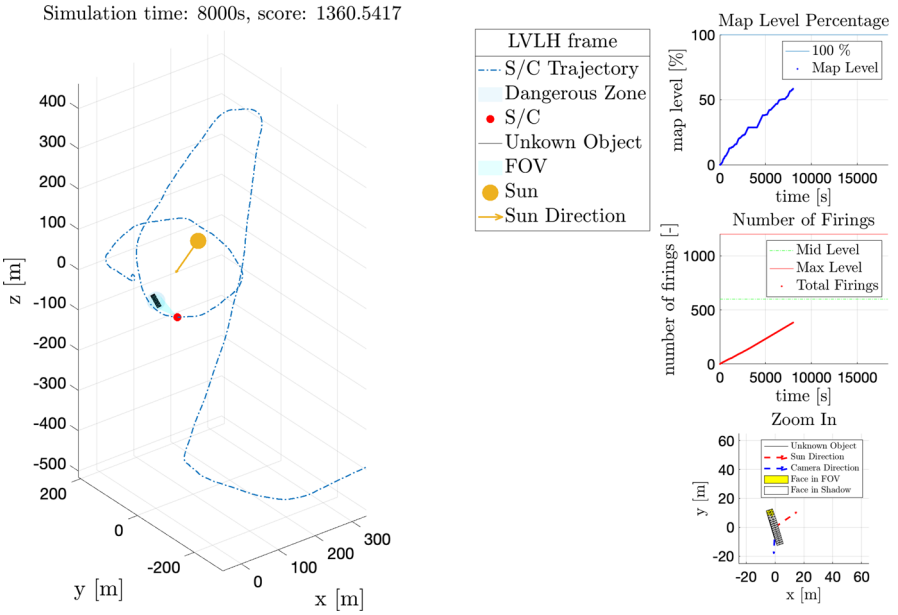


Fig. 12 Example of 100% map level episode: frame 2

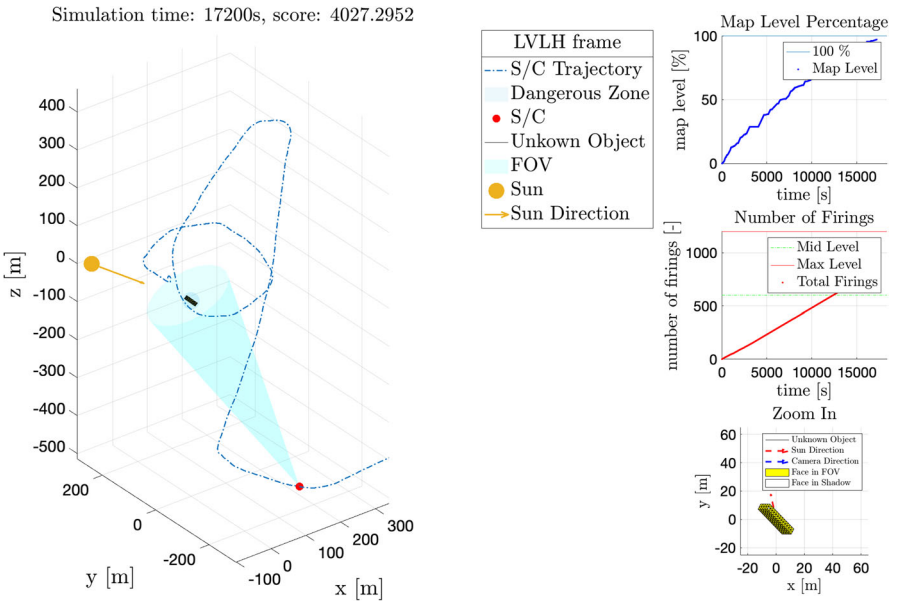


Fig. 13 Example of 100% map level episode: frame 3 (final epoch)

Funding Open access funding provided by Politecnico di Milano within the CRUI-CARE Agreement.

Declarations

Conflict of Interests On behalf of all authors, Andrea Brandonisio states that there is no conflict of interest.

References

1. Davis, J.P., Mayberry, J.P., Penn, J.P.: On-orbit servicing: Inspection repair refuel upgrade and assembly of satellites in space. The Aerospace Corporation report (2019)
2. Gaudet, B., Linares, R., Furfaro, R.: ptive guidance and integrated navigation with reinforcement meta-learning. Acta Astronautica, Vol. 169 (2020)
3. Linares, R., Campbell, T., Furfaro, R., Gaylor, D.: A deep learning approach for optical autonomous planetary relative terrain navigation. AAS/AIAA Spaceflight Mechanics Meeting. San Antonio, Texas (2017)
4. Chan, D.M., Agha-mohammadi, A.A.: Autonomous imaging and mapping of small bodies using deep reinforcement learning. IEEE Aerospace Conference, Big Sky, Montana (2019)
5. Piccinin, M., Lavagna, M.: Deep reinforcement learning approach for small bodies shape reconstruction enhancement. AIAA Scitech 2020 Forum, Orlando, Florida (2020)
6. Oestreich, C.E., Linares, R., Gondhalekar, R.: Autonomous six-degree-of-freedom spacecraft docking with rotating targets via reinforcement learning. J. Aerospace Inf. Sys. **18**(7), 417–428 (2021). https://doi.org/10.2514/1.I010914
7. Broida, J., Linares, R.: Spacecraft rendezvous guidance in cluttered environments via reinforcement learning. Ka' anapali, Maui, HI, USA (2019)
8. Smith, R., Self, M., Cheeseman, P.: Estimating uncertain spatial relationships in robotics. Autonomous Robot Vehicles **1**(1), 435–461 (1986)
9. Feder, H.J.S., Leonard, J.J., Smith, C.M.: Adaptive mobile robot navigation and mapping. Int. J. Robot. Res. **18**(7), 650–668 (1999)
10. Geromichalos, D., Azkarate, M., Tsardoulias, E., Gerdes, L., Petrou, L., Pulgar, C.P.-d.: SLAM for autonomous planetary rovers with global localization. J. Field Robotics **37**(1), 1–18 (2020)
11. Baldini, F., Harvard, A., Chung, S.-J., Nesnas, I., Bhaskaran, S.: Autonomous Small Body Mapping and Spacecraft Navigation. International Astronautical Congress (IAC). Bremen, Germany (2018)
12. Pesce, V., Agha-mohammadi, A.A., Lavagna, M.: Autonomous Navigation and Mapping of Small Bodies. IEEE Aerospace Conference. Big Sky, Montana (2018)
13. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. The MIT Press, Cambridge (2015)
14. Kollar, T., Roy, N.: Trajectory optimization using reinforcement learning for map exploration. Int. J. Robot. Res. **27**(2), 175–196 (2008)
15. Sallab, A., Abdou, M., Perot, E., Yogamani, S.: Deep reinforcement learning framework for autonomous driving. Electronic Imaging **2017**(1), 70–76 (2017)
16. Curtis, H.D.: Orbital Mechanics for Engineering Students. Butterworth-Heinemann, Oxford (2014)
17. Wertz, J.R.: Spacecraft Attitude Determination and Control. Springer Science & Business Media, Torrance (1978)
18. Gaskell, R.: Automated landmark identification for spacecraft navigation. Adv. Astronaut. Sci. **109**(1), 1749–1756 (2002)

19. Gaskell, R.W., Barnhouin-Jha, O.S., Scheeres, D.J., Konopliv, A.S., Mukai, T., Abe, S., Saito, J., Ishiguro, M., Kubota, T., Hashimoto, T., Kawaguchi, J., Yoshikawa, M., Shirakawa, K., Kominato, T., Hirata, N., Demura, H.: Characterizing and navigating small bodies with imaging data. Meteoritics & Planetary Science **43**(6), 1049–1061 (2008)
20. Durrant-Whyte, H., Bailey, T.: Simultaneous localization and mapping (SLAM): part I. IEEE Robot. Autom. Mag. **13**(2), 99–110 (2006)
21. Bailey, T., Durrant-Whyte, H.: Simultaneous localization and mapping (SLAM): part II. IEEE Robot. Autom. Mag. **13**(3), 108–117 (2006)
22. Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., Leonard, J.J.: Past, present, and future of simultaneous localization and mapping: toward the robust-perception age. IEEE Trans. Robot. **32**(6), 1309–1332 (2016)
23. Åström, K.: Optimal control of Markov processes with incomplete state information. J. Math. Anal. Appl. **10**(1), 174–205 (1965)
24. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., Hassabis, D.: Human-level control through deep reinforcement learning. Nature **518**, 529–533 (2015)
25. Brandonisio, A.: Deep Reinforcement Learning to Enhance Fly-around Guidance for Uncooperative Space Objects Smart Imaging. Master's thesis, Politecnico di Milano (2020)

Affiliations

Andrea Brandonisio[1] · Michèle Lavagna[1] · Davide Guzzetti[2]

Michèle Lavagna
michelle.lavagna@polimi.it

Davide Guzzetti
guzzetti@auburn.edu

[1] Department of Aerospace Science and Technologies (DAER), Via La Masa, 34, Milan, Italy

[2] Department of Aerospace Engineering, Auburn University, Auburn, AL 36849, USA