POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Adaptive guidance via Meta-Reinforcement Learning: ARPOD for an under-actuated CubeSat

Laurea Magistrale in Space Engineering - Ingegneria Spaziale

Author: Gaetano Calabrò

Advisor: Prof. Pierluigi Di Lizia

Co-advisor: Michele Maestrini PhD

Academic year: 2022

1. Introduction

The evolution of space industries is, currently, headed towards the application of small satellites for the accomplishment of very complex missions, such as on-orbit servicing, debris removal and the deployment of large constellation for global services. To cope with the complexity of such space missions, autonomous and small spacecrafts solutions are making inroads in almost every space industry.

This Executive Summary aims at briefly describing a solution for an Autonomous Rendezvous, Proximity Operations and Docking (ARPOD) problem for a highly constrained CubeSat, proposed in [1], through the implementation of a Deep Meta-Reinforcement Learning algorithm. Moreover, the scope of this work is to demonstrate that Deep Meta-Reinforcement learning provides a powerful technique for solving complex problems, where uncertainties are considered. In particular, the adaptability and robustness of such algorithm is proved by the results that will be summarized on this document.

2. Problem formulation

Typically, a rendezvous, proximity operations and docking mission ConOps can be divided into three main phases:

- An initial phase, defined as rendezvous phase, during which a spacecraft, called Chaser, approaches another one, named as Target. It ranges from $10\,km$ to $1\,km$ of relative distance between the two.
- A second phase, the proximity operations one, that goes from $1\,km$ to $100\,m$ of separation between the two spacecrafts.
- The docking phase, in which final manoeuvres are performed to dock the two spacecrafts. It can be identified between $100\,m$ and $0\,m$ of separation.

With this in mind, the problem here considered assumes a planar Autonomous Rendezvous, Proximity Operations and Docking between an Earth-orbiting spacecraft, the Target, and an autonomous spacecraft, the Chaser.

The high level objectives of this mission are:

- The chaser must reach less than $10\,m$ of separation from the Target, with a maximum absolute relative velocity of $0.2\,m/s$ and an absolute relative angle lower than $5\,deg$.
- The entire manoeuvre must satisfy some

safety requirements, that are introduced as constraints.

- The docking is considered successful if the Chaser reaches less than 1 m of relative distance with respect to the Target.

2.1. Chaser Model and Dynamics

The Chaser is modelled as a 6U CubeSat, provided with a set of two thrusters, aligned with the body axis $\mathbf{x}_b$ and a reaction wheel, used to control the angle about the $\mathbf{z}_b$ axis. The docking port of the Chaser is assumed to be on the surface with normal aligned to the body axis [1], as illustrated in green in Fig. 1.

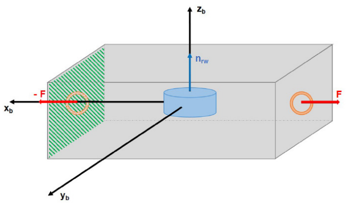

Figure 1: *Model of the Chaser. The thrusters and the reaction wheel are indicated in red and blue, respectively.* Taken from [2].

The Target is on a circular Earth orbit a 500 km of altitude. To simplify the equations of motion that describe the rendezvous, a non-inertial frame is attached to the Target: the Hill's frame. In this frame, the relative position between the Target and the Chaser is denoted as $\mathbf{r} = \{x\ y\ z\}^T$. Hence, the planar equations of motion [3] can be written as:

$$\begin{aligned} \ddot{x} - 2n\dot{y} - 3n^2x &= 0 \\ \ddot{y} + 2n\dot{x} &= 0 \end{aligned} \tag{1}$$

Where n is the mean motion of the Target, that is constant for a circular orbit. As a consequence, since all the coefficients of the differential equations are constant, an analytical solution there exists. The same holds for the rotational equations of motion eq. (2), where I_{zz} and D are the mass moments of inertia of the spacecraft and the reaction wheel, respectively, θ_N is the attitude angle and $\dot{\psi}$ is the angular acceleration provided by the reaction wheel.

$$I_{zz}\ddot{\theta}_N = -D\dot{\psi} \tag{2}$$

For the scopes of this work, the analytical solution of the equations of motion is then discretized in a recursive form.

2.2. Constraints

A set of constraints is implemented to assure safety and feasibility of the trajectory, following the guidelines for an assured satellite proximity operations, given by [1].
For the sake of clarity, the constraints are briefly explained hereafter, where they are mathematically formalized.

Constraint	Expression
1	$F \in [F_{min} : F_{max}]$
2	$\lvert\psi\rvert \leq \psi_{max}$
3	$\lvert\dot{\psi}\rvert \leq \dot{\psi}_{max}$
4	$\lvert\dot{x}\rvert \leq v_{x_{max}} and \lvert\dot{y}\rvert \leq v_{y_{max}}$
5	$\lvert\lvert\mathbf{v}\rvert\rvert \leq v_{dock} + f_s\sqrt{\frac{F_{max}}{2m}\lvert\lvert\mathbf{r}\rvert\rvert}$
6	$\lvert\dot{\theta}_N\rvert \leq \dot{\theta}_{N_{max}}$
7	$\lvert\ddot{\theta}_N\rvert \leq \ddot{\theta}_{N_{max}}$
8	$\alpha_C \leq \alpha_{LoS}$

Table 1: Mathematical expression of the constraints

In particular:

- **Constraint 1** - *Asymmetric bounded thrust*: physical limitation of the thrusters in terms of maximum and minimum available thrust, F_{max} and F_{min}.
- **Constraint 2** - *Maximum reaction wheel velocity*: physical limitation of the reaction wheel in terms of maximum angular velocity ψ_{max}.
- **Constraint 3** - *Maximum reaction wheel acceleration*: physical limitation of the reaction wheel in terms of maximum angular acceleration $\dot{\psi}_{max}$.
- **Constraint 4** - *Recoverable relative velocity limit*: maximum threshold for the relative velocity of the chaser, in order to maintain recoverable relative motion.
- **Constraint 5** - *Bounded relative velocity limit*: links the speed and the relative distance, meaning that the Chaser should not be traveling exceedingly fast when it is getting closer to the Target. The maximum final velocity is thus constrained to fulfill a threshold to ensure a safe docking. In the mathematical expression, f_s is a safety coefficient.
- **Constraint 6** - *Maximum angular velocity*:

limit to the maximum angular velocity of the Chaser.

- **Constraint 7** - *Maximum angular acceleration*: maximum angular acceleration that the structure of the Chaser spacecraft can sustain without serious damages.
- **Constraint 8** - *Docking cone*: imposed to force the Chaser to remain inside the Line-of-Sight (LoS) α_{LoS} of the Target docking port sensors during the docking phase.

3. Deep Meta-Reinforcement Learning

In the Reinforcement learning framework, an Agent learns through repeated interaction with an environment to complete a certain task. A Markov Decision Process (MDP) is an abstraction of this environment, which can be represented, in a continuous form, by a state space $\mathcal{S}$, an action space $\mathcal{A}$, a state transition distribution $\mathcal{P}(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \mathbf{a}_t)$ and a reward function $r = \mathcal{R}(\mathbf{x}_t, \mathbf{a}_t)$, where $\mathbf{x} \in \mathcal{S}$, $\mathbf{a} \in \mathcal{A}$ and r is a scalar reward signal. The Agent interacts with the environment, selecting a certain action $\mathbf{a}_t$ based on the state $\mathbf{s}_t$ it is currently in, receiving then a reward r_{t+1} and ending up in the next state $\mathbf{s}_t$. The optimization of the learning process consists in maximizing the sum of potentially discounted rewards over the trajectories generated from the interaction with the environment.

Deep Meta-Reinforcement learning is a reinforcement learning based algorithm that embeds a particular kind of Artificial Neural Networks, called Recurrent Neural Networks (RNN). The Meta-RL Agent is trained over multiple Markov Decision Processes. At the beginning of a new episode, a new MDP environment is generated and the initial state of the Agent neural network is reset. Afterwards, the Agent performs an action-selection strategy thus exploring the new environment. Finally, the networks are trained to maximize the sum of observed rewards, obtained during the exploration at each episode.

After the training process, the testing phase begins. The Agent's policy is fixed, which means that the parameters of its neural network, the weights and biases, are frozen. Since the policy is fixed, the Agent is now history-dependent. In other words, the adaptability of Meta-RL Agents lies on the use of Recurrent Neural Networks, that are able to store data from earlier events and propagate them to current processing steps, building so a memory of time series events [4] that are recalled when dealing with a new environment. In this work, a Long Short-Term Memory (LSTM) network is adopted as the Recurrent Neural Network that builds up the Agent. For the sake of brevity, its main structure is illustrated in Fig. 2.

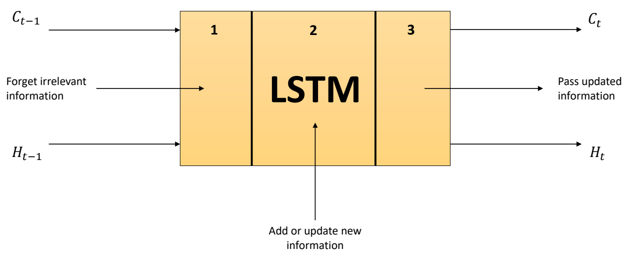

Figure 2: *Common subdivision of an LSTM module into three main parts, respectively the Forget gate, the Input gate and the Output gate. Each of them plays a particular and unique role. C_t is called as cell state and stores long term dependecies, while H_t is the hidden state, that catches short term dependencies.*

Along the Recurrent Network, a standard Feed Forward Neural Network is considered for the implementation of the Agent.

3.1. Actor-Critic and Proximal Policy Optimization

The two kinds of neural networks are used to build up the Agent, that can be divided in two different parts: an Actor and a Critic. The Actor consists in a neural network that tries to learn the adaptive policy π^{θ}, which is a probability distribution of possible actions conditioned on the state the Agent currently is, by optimizing the networks parameters θ. The Critic network, instead, learns the so-called value function, $V_w^{\pi,\gamma}(\mathbf{s}_t)$, parameterized in w, that, dependently on the current policy and the current state, estimates how good is a state assuming to start from that state and reach a future one in a horizon determined by the discounting factor γ.

The learning process of both Actor and Critic consists in the simultaneous optimization of their respective network parameters θ and w through a gradient descent applied to some user-defined cost functions. In this work, the Proximal Policy Optimization (PPO) algorithm is adopted for the policy optimization process [5]. It defines a clipped objective function, based on the advantage function $A^{\pi,\gamma}(\mathbf{s}_t, \mathbf{a}_t)$, which is a measure of how much better an action is compared to the expected outcome that can be obtained by following the current policy π_θ. In this work, as done in [6], the advantage function is calculated as the difference between the discounted sum of rewards, known as empirical return, and the predicted state value function, learned by the Critic. The expression is shown in eq. (3).

$$A^{\pi}_{w}(\mathbf{s}_t, \mathbf{a}_t) = \left[\sum_{\tau=0}^{\infty} \gamma^{\tau} r_{t+\tau}\right] - V^{\pi}_{w}(\mathbf{s}_t) \qquad (3)$$

Finally the two objective functions $J^{PPO}_t(\theta)$ and $J^{VF}_t(w)$ can be written for both the Actor and the Critic, respectively. The parameters are then updated through gradient ascent on θ and gradient descent on w.

Furthermore in the implementation of the PPO algorithm, the clipping parameter used for the objective function $J^{PPO}_t(\theta)$ and the learning rates of the neural networks are dynamically adjusted to target a Kullback-Leibler divergence between the old and the new policies. For the sake of clarity, the Kullback-Leilber divergence is a measure of the difference between two probability distributions [6].

3.2. Algorithm

To sum up, the Actor-Critic based PPO algorithm is reported hereafter.

Algorithm 1 PPO algorithm in Meta-RL

```
1: Initial instructions
2: for episode = $1, 2, ..., E$ do
3:     Reset the environment
4:     Generate new environment
5:     while not done do
6:         Sample $\mathbf{a}_t$ from $\pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)$
7:         $s_{t+1}; r_{t+1}; done \leftarrow Env(\mathbf{s}_t; \mathbf{a}_t)$
8:         Store the roll-outs
9:         for epoch = 1,2, ..., K do
10:            Unroll the recurrent layer of each network
11:            Compute $V^{\pi}_{w}$ and $A^{\pi}_{w}$
12:            Compute Actor objective function $J^{PPO}_t$
13:            Compute Critic objective function $J^{VF}_t$
14:            Perform a gradient ascent on the Actor parameters $\theta$
15:            Perform a gradient descent on the Critic parameters $w$
16:            Adjust clipping parameter and learning rates to target a KL divergence
17:        end for
18:    end while
19: end for
20: Final instructions
```

4. Implementation

The implementation of the ARPOD problem in the Meta-RL framework begins with the discretization of the solution of the equations of motion and the definition of the action and state spaces, respectful of the constraints. Afterwards, a proper reward logic is introduced as the combination of two terms: a shaping reward function and a sparse reward logic, that assigns bonuses and penalties when some user-defined conditions are met. The shaping reward, instead, suggests the Agent what actions must be preferred in order to reach the Target. For this reason, this term is expressed as a function of the distance, according to the definition of a fictitious potential field in the location of the target, and as a function of the attitude angle. Hence:

$$R^{shape}_t(r_t, \theta_{N_t}) = R^{shape,r}_t + R^{shape,\theta}_t \qquad (4)$$

The distance dependent term is obtained by the definition of an attractive Artificial Potential

Field (APF) [7] in the location of the Target, as expressed in eq. (5), where k_{att} is the attractive coefficient. The Potential is defined as a positive function that increases as the relative distance with respect to the Target increases.

$$U_t = \frac{1}{2} k_{att}(||r_t||^2) \qquad (5)$$

After the definition of the reward function, the Agent's networks structure is created and reported in Table 2 and Table 3.

Layer	Neurons	Activation
Hidden 1	130	tanh
Hidden 2	90	tanh
Hidden 3	60	tanh
Output	2	Linear

Table 2: Actor Neural Network architecture

Layer	Neurons	Activation
Hidden 1	130	tanh
Hidden 2	90	tanh
Hidden 3	60	tanh
Output	1	Linear

Table 3: Critic Neural Network architecture

Finally the parameters of the Networks and of the algorithm, usually referred to as hyperparameters, are selected by the user.

5. Results

Two cases are analyzed in this work:

1. An autonomous Proximity Operations and Docking manoeuvre from 1km of relative distance with respect to the Target along the tangential direction of the Hill'frame (V-bar approach), with uncertainties on the initial position of $\pm 50\,m$ and on the initial velocity and attidue.
2. A full ARPOD manoeuvre from 5km of relative distance along the tangential direction of the Hill's frame (V-bar approach), with uncertainties on the initial position of $\pm 10\,m$ and on the initial velocity and attitude.

The first step consists in the training of the neural networks, that aims at optimizing of the adaptive policy. For both cases, the learning curves expressed in terms of rewards are reported in Fig. 3 and Fig. 4.

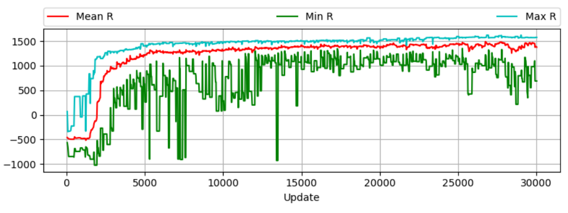

Figure 3: *Optimization of the learning curve for the first case.*

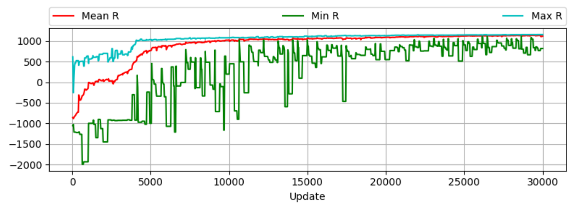

Figure 4: *Optimization of the learning curve for the second case.*

It can be seen from the curves that the Agent's policy is indeed optimized as the reward collected increases with the episodes.
Finally, the Agent is tested for both the two cases. The two policies, one for the first case and the other for the second case, are now fixed and a simulation of 10000 trajectories is executed.

For the Proximity Operations case, 16.58% of the trajectories are acceptable, see Fig. 5, and are all illustrated in Fig. 6. The minimum distance reached by the Agent is $2.74\,m$.

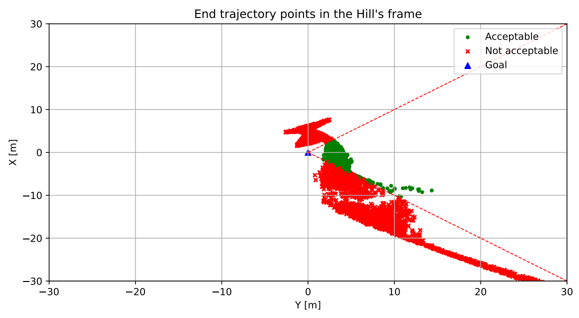

Figure 5: *Terminal points of the 10000 trajectories. Case 1.*

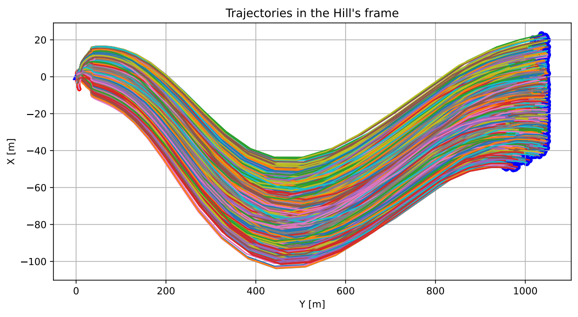

Figure 6: *1658 trajectories for the first case.*

For what concerns the second case, all of the 10000 trajectories are acceptable, yet they stops at 15.8 m. The terminal points and the trajectories are illustrated in Fig. 7 and Fig. 8, respectively.

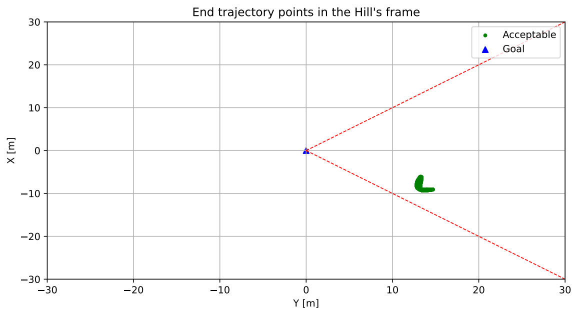

Figure 7: *Terminal points of the 10000 trajectories. Case 2.*

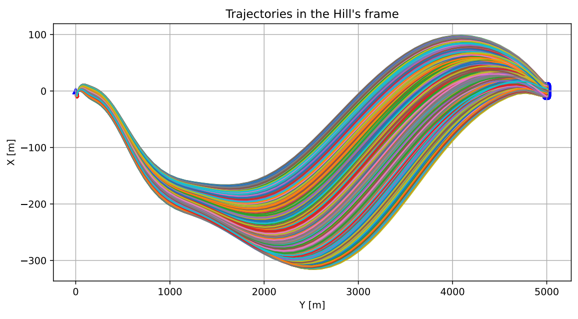

Figure 8: *10000 trajectories for the second case.*

6. Conclusions

From the results, it can be seen that the Agent is correctly trained and that the learned behavior brings it closer to the Target, fulfilling the constraints on the velocities and on the actuators' physical limitations. The robustness and adaptability demonstrated in this work justifies the strength of Meta-RL in the solution of different tasks, without the need of a specific training for each of the problems. However, even if the training can be considered successful, the mission should not be considered as achieved. In particular, it was shown how the Agent was able to get very close to the Target spacecraft yet without reaching the final docking phase.
For future works, a solution to the missing final phase must be found, eventually improving the Meta-RL Agent or implementing new techniques. In addition, actuator failures or more uncertainties on the dynamics can bring to more complex and realistic problems that will be faced using the promising technique of Meta-Reinforcement learning.

7. Acknowledgements

I would like to thank my Advisor Professor Pierluigi Di Lizia and my Co-Advisor Michele Maestrini PhD. They guided me through all the problems I encountered in this work, helped me with their knowledge and experience. I would like also to acknowledge the creators of the codes that inspired me for this work, Professor Roberto Furfaro, Professor Richard Linares and Professor Brian Gaudet. Alongside them, also the people who shared their knowledge through internet free-courses: Professor Sergey Levine, with his CS285 online course on Reinforcement Learning, and Professor David Silver, from Google DeepMind.

References

[1] Christopher D. Petersen, Sean Phillips, Kerianne L. Hobbs, and Kendra Lang. Challenge problem: Assured satellite proximity operations. 31st AAS/AIAA Space Flight Mechanics Meeting, online, 2021. American Astronautical Society (AAS) and American Institute of Aeronautics and Astronautics (AIAA).

[2] Matthieu Paris. Safe arpod for under-actuated cubesat via reinforcement learning. Master's thesis, Politecnico di Milano, 2021.

[3] Gene W. Sparrow and Douglas B. Price. Derivation of approximate equations for solving the planar rendezvous problem. *NASA Technical Note D-4670*, 1968.

[4] Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi

Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. 2016. doi: 10.48550/ARXIV.1611.05763.

[5] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithm, 2017.

[6] Brian Gaudet, Richard Linares, and Roberto Furfaro. Adaptive guidance and integrated navigation with reinforcement meta-learning. *Acta Astronauta*, 2020.

[7] Qingfeng Yao et. al. Path planning method with improved artificial potential field - a reinforcement learning perspective. *IEEE Access*, 8, 2020. doi: http://dx.doi.org/10.1109/ACCESS.2020.3011211.