Point Cloud-Based Reinforcement Learning for Autonomous Navigation of a Robotic Rover on Planetary Surfaces

IAC-23-D1.2.9x80146

Point Cloud-Based Reinforcement Learning for Autonomous Navigation of a Robotic Rover on Planetary Surfaces

Federico Mustich
Politecnico di Torino - DIANA, Italy, federico.mustich@teamdiana.it

Leonardo Maria Festa
Politecnico di Torino - DIANA, Italy, leonardo.festa@teamdiana.it

Fabrizio Stesina
Politecnico di Torino, Italy, fabrizio.stesina@polito.it

Gabriele Tiboni
Politecnico di Torino, Italy, gabriele.tiboni@polito.it

Raffello Camoriano
Politecnico di Torino, Italy, raffello.camoriano@polito.it

Abstract

This work proposes an autonomous navigation software stack to allow robotic rovers in open-field environments to reach given target coordinates in a safe, reliable, and efficient manner. The primary objective of this project is to train a Reinforcement Learning (RL) agent to acquire effective strategies for safely traversing its sorroundings until the desired destination is reached. The rover receives target coordinates from the user and point cloud data from dedicated sensors such as stereocameras or LIDARs, providing information about the surrounding environment morphology. Assuming the robotic rover has a reliable method to localize its position and orientation in the field and track its movements during operations, as well as a robust controller to reach a target pose in its immediate proximity within a few meters from its current position, we introduce *DIANA-Gym*, a simulation environment to train the agent on a virtual rover. Next, a state-of-the-art algorithm is selected to train and test an RL agent on the virtual environment. Although designed for planetary navigation and exploration purposes, we believe that the proposed framework could be adapted with minimal modifications to other similar open-field navigation tasks. By combining Reinforcement Learning with point cloud data, our proposed autonomous navigation software stack provides an efficient, reliable, and safe solution for autonomous exploration and navigation in challenging environments. The entire project is being developed within DIANA from Politecnico di Torino, a student team competing in the Rover Challenge Series, which challenges students from all the engineering areas to design and develop a prototype for an astronaut assistance rover platform.

Keywords: Autonomous Navigation, Reinforcement Learning, Machine Learning, Point Cloud, Rover.

1. Introduction

Autonomous navigation is a challenging task, especially in open-field environments characterized by intricate and rugged morphologies, oftentimes lacking structured visual references and exhibiting dynamical lighting conditions. This challenge is further accentuated in the context of robotic rovers operating on planetary surfaces, where environmental unpredictability and hazards are heightened. The imperative for such rovers is twofold: they must execute cautious maneuvers to avert potentially mission-ending hazards, while simultaneously achieving efficient traversal speeds to optimize their operational and scientific yield over their operational lifespans.

While numerous methodologies for autonomous navigation in open-field scenarios have been proposed or integrated into past and present missions [1], many of these approaches hinge upon heuristic algorithms and conventional pathfinding techniques, including A*, Monte Carlo and RRT [2]. Instead, this study aims to assess the efficacy of a machine learning-driven strategy in tackling this task. Specifically, we propose a Reinforcement Learning (RL) agent that perceives its surroundings through a point cloud representation, as opposed to a 2D representation obtained via common RGB cameras. This is due to the assumption that, since a point cloud natively encodes depth information, it should be more efficient for a RL agent to learn an effective navigation policy using such representation.

The main contribution of this work consists in the development of the required software infrastructure to allow users to train the RL agent with a virtual rover in a simulated environment, namely *DIANA-Gym*.

All of the software is available, free and open-source, at `https://github.com/Anatr1/DIANA-Gym`.

2. Related Works

Recent advancements in the field of machine learning, particularly in the area of Deep Reinforcement Learning (DRL), have had a significant impact on robotics and, in particular, on the approach to autonomous navigation tasks. DRL has mitigated the limitations of static heuristic algorithms by enabling agents to learn optimal actions through interaction with their environments. This dynamic learning paradigm is particularly well-suited for open-field navigation, where environments are subject to change and adaptability is paramount. Through continuous interaction with the environment, DRL agents can develop contextually adaptive navigational strategies, enhancing their performance in complex landscapes.

An important milestone for the DRL community has been the introduction of Gym by OpenAI [3], which has rapidly become the standard framework for building and evaluating DRL models. The introduction of Gym led to the development of Baselines [4] and, subsequently, Stable Baselines by Raffin et al [5], an open-source implementation of many DRL algorithms that follow a consistent, gym-compatible interface to facilitate the training and comparison of different models.

DRL has been applied to robot navigation tasksin a number of settings with different aims and analyzing different scenarios and aspects of the problem, ranging from search and rescue [6] to agriculture automation [7]. Koutras et al. [8] proposed a gym-compatible framework for exploiting DRL models to learn exploration policies in unknown 2D environments and performed preliminary studies on various state-of-the-art DRL algorithms such as SAC [9] and PPO [10].

Martini et al. [11] presented *PIC4rl-gym*, a novel framework based on ROS2 and Gazebo [12], for training and testing DRL agents on several outdoor and indoor navigation scenarios. *Pic4RL-gym* is intended as an alternative to OpenAI Gym and is not gym-compatible. Liang et al., leveraging ROS, Gazebo, and gym-gazebo [13], proposed Parallel Gym Gazebo [14], a gym-compatible training platform for DRL applied to robotic tasks that addresses the computational bottleneck resulting from the simulation of complex and realistic dynamics, which often significantly impedes progress in the field by imposing unsustainable training times and costs. Although interesting, an open-source implementation of this work is currently not available.

Ferigo et al [15] developed *Gym-Ignition,* a framework to create reproducible robot environment for RL research introducing Gym-compatible environment support to *Gazebo-Ignition* [16], a simulator derived from Gazebo with focus on advanced rendering, physics and sensors simulation.

Finally, *ROS2Learn* [17], together with gym-gazebo2 [18], constitutes yet another gym-compatible framework for training and developing RL-based robotic applications using, once again, Gazebo and ROS2.

3. Requirements

The project has been developed within DIANAs, a student team from Politecnico di Torino active in the Rover Challenge Series. As such, its requirements are shaped according to those imposed by similar

competitions, in particular the *European Rover Challenge 2023 (ERC23)* [19], and to the means and necessities of a university student team. For this reason, the listed requirements may differ, sometimes substantially, to those applying to actual spacecraft and space-rated hardware. In particular, the trained networks resulting from this project are designed to be run on an embedded ARM computer such as an NVIDIA Jetson Xavier or NVIDIA Jetson Orin. Similar computers, which usually are supported by a strong GPU, are designed for earth-bounded robots and AI applications, and differ significantly from the industry standard computing hardware for space applications, both in terms of computational power and energy usage.

The requirements desired for the trained agent to fulfill are the following:

- **Safe travel.** The system should keep the rover on a path safe from terrain hazards and avoid collisions while keeping the pitch and roll angles as close as possible to 0°. The system should not guide the rover between passages too narrow for its size or in regions in which it would be hard to maneuver and escape.
- **Ground speed.** For small-sized rovers (~10 kg), at the bare minimum the system should be able to travel at least 100 meters in 20 minutes, which corresponds to an average ground speed of 0.083 m/s. Speeds as high as 0.33 m/s should be achievable, always depending on the considered rover.
- **Light and weather robustness.** The system should be able to operate with comparable performances under any possible light condition.
- **Field-agnostic.** The system should be able to navigate in any open field resembling a mars/lunar landscape not featuring any other moving object/entity except from the rover itself.
- **Platform flexibility.** The system should be flexible enough to be integrated, after a proper retraining, into every ground-based robotic platform compliant with the guidance and control stack described next.

4. Infrastructure

4.1 Perception and Control

This project assumes three core components to be available and properly working on the target rover platform:

- A reliable pose estimation service. The robot needs to be aware of its position and orientation on the field at any moment. The position is calculated with respect to the origin of the reference frame placed on the rover starting point with x-axis facing front and y-axis growing to the left of it, starting with a yaw value of 0 degrees. This value will be used as input by the RL agent and will constitute part of its internal state.
- A sensor returning a 3D representation of the surrounding environment. For the implementation, a depth-camera has been considered. 3D LIDARs providing 360° degrees point-clouds could also be applied. The resulting data will also be utilized by the RL agent as an input.
- A service enabling to control the rover *in position*, meaning that, upon receiving a target point to reach, the rover must be able to compute a trajectory and to follow it until the given point is reached, assuming an obstacle-free terrain and without performing any kind of obstacle detection or avoidance. The output of the RL agent will consist in a series of 2D points to be reached by the position controller.

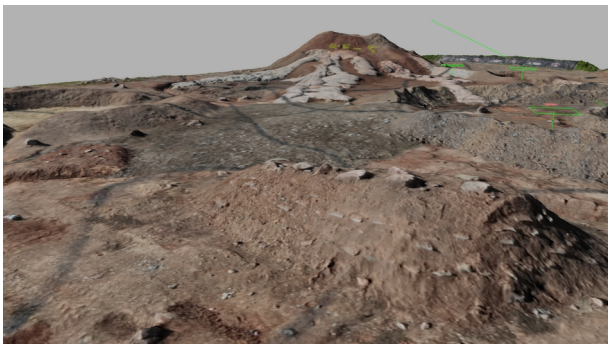
4.2 Simulation Environment: DIANA-Gym

To facilitate the training of RL agents in a simulated environment for autonomous rover navigation across challenging terrains, the selection of an appropriate simulation framework is paramount. In this project, Gazebo was chosen as the foundation due to its inherent support and seamless integration with ROS platforms and for the large support provided by its active community [12], ensuring compatibility with a wide range of robotic hardware and software components.

We introduce *DIANA-Gym*, a novel simulation environment tailored for RL training of robotic rovers. *DIANA-Gym* builds upon the framework proposed by Martini et al. in *PIC4rl-gym*, extending it to offer enhanced compatibility with OpenAI's Gym environments, mirroring the approach taken by Nuin et al. in their *ROS2Learn/Gym-Gazebo2* framework. Beyond these foundations, *DIANA-Gym* incorporates support for *Stable Baselines 3*, easing its adaptation to various reinforcement learning algorithms. This not only simplifies the implementation process but also grants users access to a rich repository of pre-built features for training monitoring and parallelization, making the framework versatile and user-friendly.

DIANA-Gym is equipped with the ERC23 Mars Yard as its default simulation environment, which constitutes of an artificial terrain designed to mimic Martian landscapes. This simulation world spans an expansive 53x41 meter area and has been captured in an high-fidelity 3D model by means of photogrammetry

techniques. The Mars Yard provides a challenging terrain in which to train RL models, featuring uneven terrain, rocky outcrops, sand dunes, and other geological elements that typify Martian landscapes. For testing purposes, or for applications which do not require to navigate similarly harsh terrains, a number of more forgiving simulated worlds are also provided to the user.



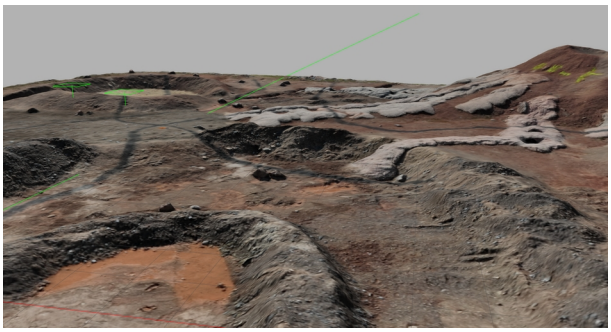


Figure 1-2: Samples from the ERC23 Mars Yard

For simplicity, the position controller that has been implemented follows a basic routine to guide the rover to the target destination, by first spinning in place on its center until it reduces its yaw angle with respect to the target to near 0°, and subsequently moving forward until its tracked position is within a tolerance margin from the target. In future extensions, more sophisticated controllers could be explored and implemented.

Due to the lack of an .urdf file representative of DIANA's rover, the simulation environment currently utilizes as default a Jackal robot, by Clearpath Robotics [20]. Authors plan to add as soon as possible a model of DIANA's most recent rover platform, ARDITO, which features a six-wheels rocker-bogie configuration much like those of NASA's Mars exploration rovers.

5. RL agent

5.1 Action Space

Considering the aforementioned constraints and the requirement for the agent's output to consist of the next-step 2D coordinates, used as input for the underlying Position Controller, this project defines the action space as the collection of 2D points situated in proximity of the rover's current position.

Specifically, when a user opts for a square action space, the available coordinates encompass the points within a square, with sides measuring seven meters, centered on the rover's current position. Alternatively, when a circular action space is chosen, the available coordinates are confined to a circle with a radius of four meters, centered on the rover's current location. In both instances, the resulting output is expressed in Cartesian coordinates within the absolute reference frame. This reference frame is established with its origin coinciding with the rover's pose at the commencement of the navigation task.

5.2 Observation Space

In this project, the observation space is fundamentally comprised of two distinct data structures, fed to the learning model to access to the status of the robot:

- **Agent Position Relative to Navigation Goal:** This component encodes critical information about the agent's position concerning the designated navigation goal. It includes the agent's Euclidean distance from the goal, expressed in meters, and the yaw angle, presented in degrees.

- **Surrounding Environment's Morphology:** The second component encompasses information related to the three-dimensional morphology of the environment surrounding the rover. To incorporate this information into the observation space, data obtained from a dedicated sensor, in the form of a point cloud or depth image, necessitates preprocessing. In the implemented configuration, a simulated depth camera is employed, yielding a two-dimensional matrix that conveys spatial distance information pertaining to objects and terrain in front of the rover.

Figure 3: Sample from the unprocessed depth-image captured from the simulated sensor

5.3 Reward Function

The reward function implemented takes into consideration several metrics to assess the performance of the agent during training and guide the algorithm learning process:

- **The distance travelled towards the goal.** If during a training step the agent moves closer to the target goal, it receives a reward bonus proportional to the distance travelled. Equivalently, a penalty is assigned to the agent should it travel further from the goal during a training step.

- **The yaw angle to the goal.** The reward is maximum for yaw angles equal to 0°, progressively decreasing until reaching a maximum penalty for values equal to +-180°, when the rover is traveling in the opposite direction with respect to the target it should approach.

- **Roll and pitch angles.** The safest traverse is the one which avoids dangerous slopes whenever possible. For this reason, small reward bonuses are granted for maintaining roll and pitch angles, separately, close to 0°. Since this project is developed for planetary exploration rovers, which usually grant notable mobility capacities on uneven terrains, the penalty for pitch and roll angles is kept negligible for absolute values up until 5°, and then progressively incremented. The intended objective is to let the agent choose to traverse over moderately steep routes if considered necessary or particularly convenient to reach the target.

- **Success or Failure conditions.** A large reward is granted whenever the rover reaches the target within a given tolerance. Similarly, a major penalty is assigned should the rover collide with environmental features. Finally, a penalty is also assigned depending on the number of steps it took the agent to reach the target, or otherwise to end the run either by collision or timeout.

Each of these metrics can be tweaked and fine-tuned by modifying existing hyperparameters while performing application specific reward shaping.

5.4 Algorithms

DIANA-Gym seamlessly integrates with the entire suite of reinforcement learning algorithms provided by *Stable Baselines 3*, catering to the diverse needs and means of users. The selection of algorithms at the disposal of the user spans over many state-of-the-art options supporting environments featuring continuous action spaces. This includes algorithms of considerable repute, such as the Soft Actor-Critic (SAC) [9], Proximal Policy Optimization (PPO) [10], and the Advantage Actor-Critic (A2C) [21]. These algorithms are known for their efficacy in solving complex robotic control tasks, offering varying trade-offs in terms of exploration, sample efficiency, and stability. Researchers and engineers can effortlessly switch between these algorithms, experimenting and fine-tuning their choice to optimize training outcomes and enhance the performance of autonomous agents operating in challenging terrains.

6. Conclusions and future works

With this work we presented *DIANA-Gym*, a novel software stack to enable training of Reinforcement Learning models on autonomous navigation tasks for planetary exploration rovers, particularly focused on robots employing depth sensors able to return a point-cloud/depth representation of the surrounding environment. By merging *Gazebo, ROS2, Stable Baselines 3,* and the ERC23 Mars Yard 3D model, *DIANA-Gym* offers a versatile, open-source, gym-compatible and easily expandible environment for robotics researchers and space or AI enthusiasts to familiarize with RL for robotics or advance knowledge in this field.

DIANA-Gym will continue to be updated from DIANA Computer Science department and new features are likely to be added in next future. New robot models and simulated environments are planned to be added soon as well as support to other kinds of sensors, such as RGB cameras, to enable multimodality support.

Acknowledgements

The authors are pleased to express their gratitude to:

- All DIANA members, for their indomitable passion and work they put in everyday to develop and build planetary rover technology demonstrators.
- Politecnico di Torino and DIMEAS, which fund and sponsor DIANA from more than fifteen years now.
- VANDAL research laboratory for their technical support and mentoring.

References

[1] M. Winter et al. Detailed Description of the High Level Autonomy Functionalities Devoloped for the ExoMars Rover, 14th Symposium on Advanced Space Technologies in Robotics and Automation, 2017.

[2] Tahirovic, A., & Magnani, G. (2011). A roughness-based rrt for mobile robot navigation planning. *IFAC Proceedings Volumes*, *44*(1), 5944–5949. https://doi.org/10.3182/20110828-6-IT-1002.03351

[3] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). Openai gym. arXiv. https://doi.org/10.48550/arXiv.1606.01540

[4] Open AI Baselines public github repository https://github.com/openai/baselines (accessed 30.08.23).

[5] A Raffin et al. Stable-Baselines3: Reliable Reinforcement Learning Implementations, Journal of Machine Learning Research, 2021.

[6] Niroui, F., Zhang, K., Kashino, Z., & Nejat, G. (2019). Deep reinforcement learning robot for search and rescue applications: Exploration in unknown cluttered environments. IEEE Robotics and Automation Letters, 4(2), 610–617. https://doi.org/10.1109/LRA.2019.2891991

[7] Martini, M., Cerrato, S., Salvetti, F., Angarano, S., & Chiaberge, M. (2022). Position-agnostic autonomous navigation in vineyards with deep reinforcement learning. 2022 IEEE 18th International Conference on Automation Science and Engineering (CASE), 477–484. https://doi.org/10.1109/CASE49997.2022.9926582

[8] Koutras, D. I., Kapoutsis, A. C., Amanatiadis, A. A., & Kosmatopoulos, E. B. (2021). Marsexplorer: Exploration of unknown terrains via deep reinforcement learning and procedurally generated environments. *Electronics*, *10*(22), 2751. https://doi.org/10.3390/electronics10222751

[9] Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). *Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor*. arXiv. https://doi.org/10.48550/arXiv.1801.01290

[10] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). *Proximal policy optimization algorithms*. arXiv. https://doi.org/10.48550/arXiv.1707.06347

[11] Martini, M., Eirale, A., Cerrato, S., & Chiaberge, M. (2022). *Pic4rl-gym: A ros2 modular framework for robots autonomous navigation with deep reinforcement learning*. arXiv. http://arxiv.org/abs/2211.10714

[12] Gazebo Simulator, Open Source Robotics Foundation, https://gazebosim.org/home

[13] Zamora, I., Lopez, N. G., Vilches, V. M., & Cordero, A. H. (2017). *Extending the OpenAI Gym for robotics: A toolkit for reinforcement learning using ROS and Gazebo*. arXiv. http://arxiv.org/abs/1608.05742

[14] Liang, Z., Cai, Z., Li, M., & Yang, W. (2019). Parallel gym gazebo: A scalable parallel robot deep reinforcement learning platform. *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, 206–213. https://doi.org/10.1109/ICTAI.2019.00037

[15] Ferigo, Diego et al. "Gym-ignition: reproducible robotic simulations for reinforcement learning" *2020 IEEE/SICE International Symposium on System Integration (SII)*. IEEE, 2020

[16] Gazebo Ignition, Ignition Robotics, https://gazebosim.org/api/gazebo/2.10/index.html

[17] Nuin, Y. L. E., Lopez, N. G., Moral, E. B., Juan, L. U. S., Rueda, A. S., Vilches, V. M., & Kojcev, R. (2019). *ROS2Learn: A reinforcement learning framework for ROS 2*. arXiv. http://arxiv.org/abs/1903.06282

[18] *Papers with Code—Gym-gazebo2, a toolkit for reinforcement learning using ROS 2 and Gazebo*. (n.d.). Retrieved August 31, 2023, from https://paperswithcode.com/paper/gym-gazebo2-a-toolkit-for-reinforcement

[19] European Rover Challenge, https://roverchallenge.eu/en/main-page/

[20] Jackal, Unmanned Ground Vehicle, Clearpath Robotics, https://clearpathrobotics.com/jackal-small-unmanned-ground-vehicle/

[21] Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., Silver, D., & Kavukcuoglu, K. (2016, February 4). *Asynchronous methods for deep reinforcement learning*. arXiv.Org. https://arxiv.org/abs/1602.01783v2