PIC4rl-gym: a ROS2 Modular Framework for Robots Autonomous Navigation with Deep Reinforcement Learning

PIC4rl-gym: a ROS2 modular framework for Robots Autonomous Navigation with Deep Reinforcement Learning

1st Mauro Martini
Department of Electronics and Telecommunications
Politecnico di Torino
Torino, Italy
mauro.martini@polito.it

2nd Andrea Eirale
Department of Electronics and Telecommunications
Politecnico di Torino
Torino, Italy
andrea.eirale@polito.it

3rd Simone Cerrato
Department of Electronics and Telecommunications
Politecnico di Torino
Torino, Italy
simone.cerrato@polito.it

4th Marcello Chiaberge
Department of Electronics and Telecommunications
Politecnico di Torino
Torino, Italy
marcello.chiaberge@polito.it

***Abstract*—Learning agents can optimize standard autonomous navigation improving flexibility, efficiency, and computational cost of the system by adopting a wide variety of approaches. This work introduces the *PIC4rl-gym*, a fundamental modular framework to enhance navigation and learning research by mixing ROS2 and Gazebo, the standard tools of the robotics community, with Deep Reinforcement Learning (DRL). The paper describes the whole structure of the PIC4rl-gym, which fully integrates DRL agent's training and testing in several indoor and outdoor navigation scenarios and tasks. A modular approach is adopted to easily customize the simulation by selecting new platforms, sensors, or models. We demonstrate the potential of our novel gym by benchmarking the resulting policies, trained for different navigation tasks, with a complete set of metrics.**

***Index Terms*—Mobile Robots, Autonomous Navigation, Deep Reinforcement Learning, ROS2, Gazebo, Simulation, Gym**

I. Introduction

Autonomous navigation algorithms aim at providing mobile robots with efficient planning and control policies to go through cluttered and dynamic environments. Advanced autonomous navigation systems have been explored to improve planners' and controllers' robustness, reliability, and computational efficiency in real-world applications. In the last decade, learning methods have seen a tremendous success among robotics researchers, motivating an increasing collection of innovative works which adopt Deep Reinforcement Learning (DRL) for general autonomous navigation [1], socially aware path planning [2], and agile aerial vehicles autopilot [3]. Besides the most common paradigm of sensorimotor agents or local planners, learning agents can be successfully mixed up in alternative ways with the navigation system. Recent works proposed hybrid solutions to optimize classic planners like the Dynamic Window Approach (DWA) [4]. Moreover, [5], [6] recently showed the effectiveness of the planner's parameters learning approach compared to end-to-end policy learning, resulting in an adaptive optimized planner. Despite the impressive research output, many works never reach real robotic applications, and a great amount of them present hard reproducibility limitations. These problems are often tight to the lack of a common tool for robotics to easily develop and compare solutions in the same conditions. Robot Operating System (ROS), recently updated to ROS2 [7], is the standard software exoskeleton for any robotics project. Learning in simulation is certainly the most convenient and safe procedure to train DRL agents for robotics, and Gazebo[1] is the common choice among simulators. For this reason, we propose the PIC4rl-gym[2], an open-source framework in ROS2/Gazebo realized to enhance, simplify and uniform the research on learning-based autonomous navigation, bridging the gap between DRL research and mobile robotics applications. The modular structure of our new gym allows the user to easily adjust the simulation training settings, selecting the desired robotic platform, sensors, neural network architecture, and DRL policy. The importance of this flexibility resides in the fact that autonomous navigation may present infinite different contexts compared to game-like benchmarks where RL algorithms are usually proposed. Moreover, a testing package is also embedded in the gym to encourage the comparison of resulting policies and their transition to real-world applications. It allows to automatically load trained agents and compute navigation metrics for each task of interest in different testing scenarios. We demonstrate the wide potential of the PIC4rl-gym by presenting diverse ablation studies that can be conducted within the framework. These include training and testing navigation policies from scratch and reference to already published works built upon the gym. The resulting benchmarks are reported for each of the considered navigation categories. We want to point out that the PIC4rl-gym does not focus on a specific DRL methodology. It paves the way for developing novel, generic DRL-based navigation solutions, from end-to-end to hybrid approaches.

Therefore, the contributions of this work are:

We thank PIC4SeR (PoliTO interdepartmental center for service robotics) for supporting this project.

[1] https://gazebosim.org/home

[2] Full code accesible at https://github.com/PIC4SeR/PIC4rl_gym

- a modular ROS2/Gazebo framework to develop learning-based navigation solutions for mobile robotic platforms;
- a starting collection of training and testing environments, sensors, and neural networks models;
- a testing package for trained agents to easily build common benchmarks for each navigation task considered with an established and expandable set of metrics.

II. Related Works

Previous works exist with similar scopes and objectives. Multiple versions of a simulation gym for DRL specifically applied to manipulator controls are proposed by [8], [9]. [10] is a recent similar work that proposes a gym in ROS, handled with behaviour trees. Regarding this work, we are proposing several advantages with PIC4rl-gym: first, ROS2 presents several benefits and updates and is the actual standard for robotics developers. Second, our parameter-based approach can be better understood by new users for customization, while different behaviour trees may result difficult to be modified. Differently, [11] adheres to our idea of a common benchmark for advanced autonomous navigation, proposing interesting metrics to evaluate a difficulty score for each generated environment in Gazebo. Despite the rich collection of Gazebo worlds proposed, this work proposes a dataset of challenging, although not realistic, scenarios. The training framework used to train the adaptive planner with reinforcement and parameter learning approach in [12] has not been released. Moreover, it was based on the previous ROS navigation stack and a single platform.

Nonetheless, disparate research works have already been developed within the PIC4rl-gym framework: end-to-end local planners [13], position-agnostic vineyard navigation [14], and RL-DWA person following [15].

III. Approach

In this section, we describe the structure of the PIC4rl-gym and briefly formalize the theoretical framework of Reinforcement Learning for autonomous navigation. The ROS2 nodes architecture to perform parametric training sessions in Gazebo is discussed. Then, the organization of the testing package included in the gym is also presented, introducing the metrics used to evaluate the different robot navigation tasks considered so far.

A. Deep Reinforcement Learning formulation for navigation

A typical Reinforcement Learning (RL) framework can be formulated as a Markov Decision Process (MDP) described by the tuple $(\mathscr{S}, \mathscr{A}, \mathscr{P}, R, \gamma)$. An agent starts its interaction with the environment in an initial state s_0, drawn from a prefixed distribution $p(s_0)$ and then cyclically selects an action $\mathbf{a_t} \in \mathscr{A}$ from a generic state $\mathbf{s_t} \in \mathscr{S}$ to move into a new state $\mathbf{s_{t+1}}$ with the transition probability $\mathscr{P}(\mathbf{s_{t+1}}|\mathbf{s_t}, \mathbf{a_t})$, receiving a reward $r_t = R(\mathbf{s_t}, \mathbf{a_t})$.

In reinforcement learning, a parametric policy π_θ describes the agent behavior. In the context of autonomous navigation, we usually model the MDP with an episodic structure with maximum time steps T. Hence, the agent's policy is trained to maximize the cumulative expected reward $\mathbb{E}_{\tau\sim\pi}\sum_{t=0}^{T}\gamma^t r_t$ over each episode, where $\gamma \in [0,1)$ is the discount factor. More in detail, we aim at obtaining the optimal policy π_θ^* with parameters θ through the maximization of the discounted term:

$$\pi_\theta^* = \arg\max_{\pi} \mathbb{E}_{\tau\sim\pi} \sum_{t=0}^{T} \gamma^t r_t \tag{1}$$

which can present alternative expressions according to the specific deterministic or stochastic policy adopted. As mentioned in Section I, the agent's policy can play a wide variety of roles within a navigation framework, depending on the task and the methodological approach.

B. PIC4rl-gym training framework

The PIC4rl-gym is designed to provide robotics developers with an easy tool to start custom DRL training sessions in simulation with minimum action on the code. The gym focuses on autonomous navigation tasks, which can be approached with novel solutions based on learning agents or hybrid classic and learning-based navigation components. To this end, we design an extremely modular framework, leveraging ROS parameters for fast simulation tuning. Indeed, two sets of parameters are used to manipulate both the simulation settings and the training details. The overall scheme of the PIC4rl-gym framework is shown in Figure 1, depicting the organization of ROS nodes and other elements composing the complete system. Arrows indicate how they communicate with the Gazebo simulation environment and the policy Trainer class.

PIC4rl training and environment The core section of the gym consists of two elements: a training interface *pic4rl_training* which bridges the ROS system with the Trainer, and the environment *pic4rl_environment*. The overall system has been optimized such that training and environment modules are condensed in a single ROS2 node execution, avoiding delays derived from massive data exchange between different nodes. Hence, a class inheritance-based structure is chosen: *training(env(Node))*. The environment reflects the typical design of a DRL gym, presenting two main methods to be called by the Trainer loop: *step()* to get observation and reward data, and *reset()* to restart a new episode, as shown in the schematic in Figure 1. Each of the fundamental sub-methods in *step()* defines the specific navigation task. Indeed, different navigation tasks can be tackled by defining a clean environment with appropriate methods to send the predicted action, calling the correct sensor data to process, defining the end-of-episode state condition, and designing the agent's reward function and observation. The associated *pic4rl_training* will therefore instantiate the environment and the desired training policy, starting the Trainer loop. ROS2 parameters allow the user to easily customize the training in simulation as requested by the navigation task. For example, it is possible to set episode duration and the number of initial episodes, regulate the exploration, and change the robot's starting pose. This ensemble of practical settings strongly

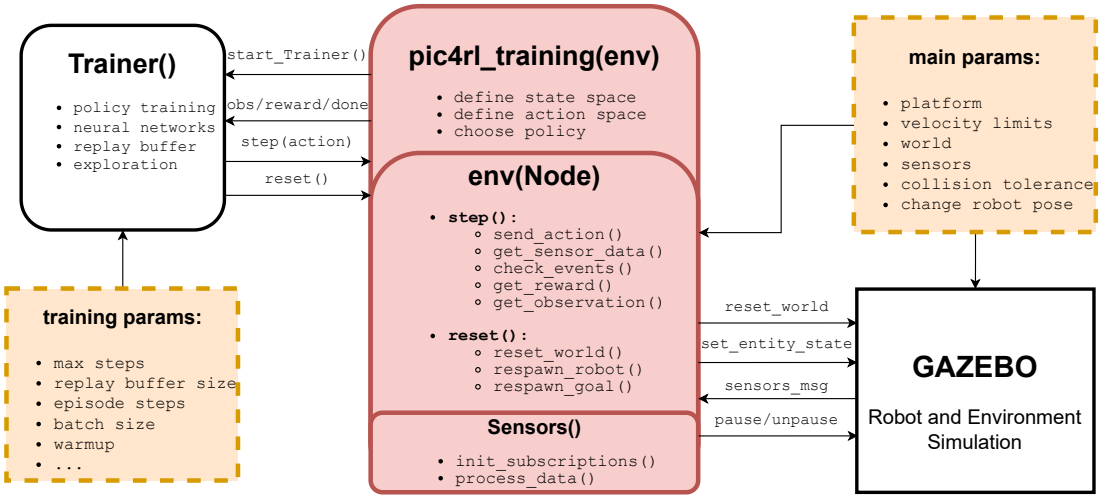

Fig. 1: Schematic of ROS nodes and processes composing the PIC4rl gym framework.

affects the success of the DRL training, being autonomous navigation a much more complex task to learn compared to standard RL benchmarks. Changing the robot pose along the simulation is a fundamental feature for the generalization properties of the agent. In such a way, it is possible to augment the variety of states experienced by the agents without drastically increasing the computational cost of the simulation training.

Gazebo simulation It is possible to launch a simulation in Gazebo, choosing the desired world, robotic platform, sensor data, and velocity limits in the main parameters file. In the actual version of the gym, we include standards packages to spawn differential drive platforms such as Jackal[3] and Husky[4] UGV from ClearPath Robotics or TurtleBot[5], together with omnidirectional custom platforms [16]. The structure of the simulation package, which handles the Gazebo virtual environment, grants to easily add new platforms and sensor plugins. Moreover, the gym automatically manages a precise detection of collisions with obstacles by specifying the shape and desired tolerance for collisions in the parameters. Whenever a *reset()* is performed to start a new episode, the environment calls the *reset_world* service of Gazebo to reset the simulation to the original state. Then, the gazebo-ros *set_entity_state* service is called to re-spawn the robot and the visual model of the goal in a new starting pose. When requested by the task, *pause/unpause* services can be used at each training step to stop the simulation while computing a new action and guarantee the MDP property of the agent-environment interaction loop.

Sensors A particular focus is devoted to developing a modular framework for sensors. No available tools offer the possibility to choose or combine data deriving from different sensors for navigation policies with DRL. In PIC4rl-gym, it is possible to specify the desired sensors and the associated properties in the parameters: the number of range points and max distance for 2D LiDAR, resolution and max depth for RGB-D images. The *Sensors()* class will take care of subscribing to the desired topics and process sensor data at each environment step call. Nonetheless, according to the sensor data shape, a set of suitable neural network architectures is already available to be selected in the pic4rl-training node. Simple dense networks are used for 2D LiDAR, while Convolutional neural networks with multiple inputs are implemented to organically deal with feature extraction from images (single-channel or 3-channels) and task-dependent state information such as goal location or robot pose.

Trainer The gym's DRL training section is developed by customizing and integrating the TF2RL library [6] in a ROS2 system. The *Trainer()* class takes care of interacting with the ROS environment at each step and training the RL policy. Real-world autonomous navigation usually requires agents with continuous action space. According to this, the Trainer focuses on state-of-the-art policy training algorithms which respect this condition. Among them, we have Deep Deterministic Policy Gradients (DDPG) [17], Twin Delayed DDPG (TD3) [18], and Soft Actor-Critic (SAC) [19]. The TF2RL library also includes diverse replay buffer options, from basic to prioritized and N-step experience replay. The original implementation of the Trainer has been adapted to our gym case integrating it into the modular framework where different network architectures can be selected according to state shape and task and automatically tune the action output shape. Nonetheless, the library only provided an initial warm-up phase for experience collection with a random policy. An ε-greedy exploration policy with exponential decay γ_ε has also been included to guarantee a continuous exploration rate during the entire training.

[3]https://clearpathrobotics.com/jackal-small-unmanned-ground-vehicle/
[4]https://clearpathrobotics.com/husky-unmanned-ground-vehicle-robot/
[5]https://www.turtlebot.com/

[6]https://github.com/keiohta/tf2rl

C. Testing package

A testing package has also been developed to easily evaluate trained agents and compute navigation metrics in different testing scenarios. The structure of the *Tester()* reflects the one used for training, only performing policy evaluation. Necessary data are collected from the simulation thanks to ros topics and the Gazebo service *get_entity_state*, which allows the robot to store the full traveled path, which can be used to compute metrics for trajectory comparison. The scheme of the PIC4rl-gym testing package is shown in Figure 2. New metrics can be easily added to the testing framework and selected from the parameters configuration interface.

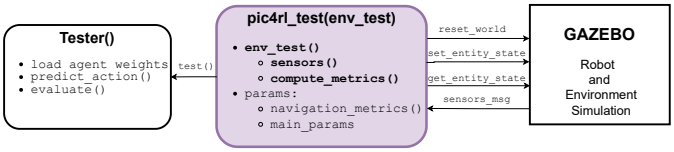

Fig. 2: Schematic description of the testing package.

IV. Experiments and Benchmarks

In this section, some representative experimental sessions are described to demonstrate the effectiveness of the PIC4rl-gym on a wide variety of potential applications. Nonetheless, this work aims to introduce the framework without focusing on a specific approach. We report three different case studies, a representative one conducted from scratch, the others referring to already published works realized with the gym.

A. End-to-end policy comparison

The classic point-to-point navigation is tackled here with a basic end-to-end policy learning approach, training sensorimotor agents that directly map different sensor data to robot velocity commands $[v, \omega]$ for mapless navigation. We conducted an ablation study by training and testing four different policies, changing sensors observation and DRL algorithms. A TD3 and a SAC dense agents are trained with 2D LiDAR ranges as observations, and a TD3 and a SAC convolutional agents are trained with depth images as input. Figure 3 shows the indoor world used to train the agents. In this case, a Jackal UGV is used. Alternative ablation studies can be conducted by varying the velocity ranges, the platform, the reward function, and all the training settings. Figure 4 shows the reward signal trends obtained during the training. The agents receive a reward $r_g = 1000$ if the goal is reached, $r_c = -150$ for collisions, and a dense reward proportional to distance reduction from goal $r_d = (d_{t-1} - d_t)$. The first 300 episodes are conducted in the central area of the world with random goals. All the agents are perfectly able to learn this behaviour, as confirmed by the growing reward trend. Then, the robot is positioned among realistic obstacles, and the agents should adapt to this complex condition. The metrics used to evaluate the point-to-point navigation are:

- number of successes;
- clearance time, cumulative heading average, total path distance, distance/path ratio;
- mean and standard deviation of linear and angular velocities, max and mean linear/yaw acceleration;
- min and mean distance from obstacles;

Table I reports the main results obtained from the testing stage of the study, averaging metrics over five different testing episodes with five different *start-goal* couples assigned around the indoor world.

Fig. 3: Example of indoor environment in Gazebo where navigation policies can be trained or tested, accessing different areas with realistic obstacles and narrow passages. A Jackal UGV has been spawned to reach the goal (red circular target).

B. Row-based crops navigation

A position-agnostic navigation agent for vineyards row-based crops has been trained in the PIC4rl-gym and already presented in detail in the paper [14]. In this case, a SAC convolutional agent learns how to safely guide a robot through vineyard rows without localization data. Besides velocity comparison, the robot's trajectory is evaluated in terms of difference from the center of the row. Different Gazebo vineyard worlds are available in the gym for testing. Row-based crops navigation is a fundamental and novel benchmark in the precision agriculture domain that the PIC4rl-gym can significantly boost and uniform.

C. Person monitoring

The PIC4rl-gym has also been used to develop an RL-DWA peculiar person monitoring algorithm presented in [15]. In this alternative assistive navigation task, a DRL agent is trained to keep the orientation of an omnidirectional robot towards the person to be monitored, while the robot is moving avoiding obstacles.

TABLE I: End-to-end policy comparison: average navigation metrics obtained testing the agents on 5 different episodes.

	success	time[s]	cum. heading[rad]	path[m]	dist/path	v_{mean}[m/s]	ω[rad/s] std. dev.	max linear acc.$[m/s^2]$	max yaw acc.$[rad/s^2]$	min obstacle dist[m]	mean obstacle dist[m]
TD3 LiDAR	5/5	24.11	0.10	5.84	0.96	0.28	0.54	3.41	18.73	0.37	3.08
TD3 Depth	5/5	19.29	0.11	5.77	0.94	0.31	0.43	2.60	17.23	0.53	3.65
SAC LiDAR	5/5	29.36	0.28	6.06	0.94	0.25	0.44	2.98	16.98	0.31	2.66
SAC Depth	5/5	16.65	0.13	5.74	0.98	0.36	0.41	2.51	16.58	0.42	2.78

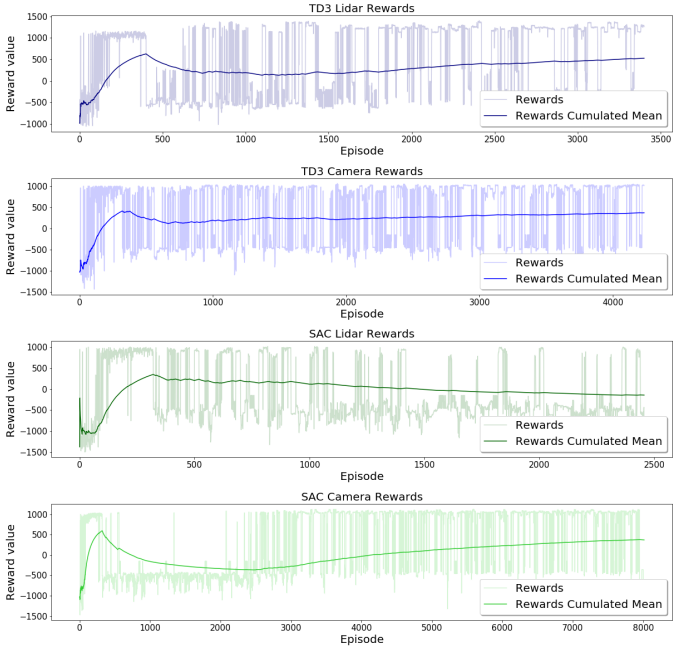

Fig. 4: End-to-end policy comparison: reward signals obtained during training the agents. Average reward trend in bold.

V. Conclusion and Future Works

In this paper, we presented *PIC4rl-gym*, a novel modular framework in ROS2/Gazebo to train and test Deep Reinforcement Learning agents specifically for mobile robot navigation tasks. The highly modular and adaptable structure of PIC4rl-gym provides robotics researchers with a complete and fast tool for developing cutting-edge DRL-based navigation solutions for various tasks. Our experiments highlight the large set of possible configurations one can explore by combining different platforms, sensors, policies, and neural network models, leading to performance optimization and flexibility. Nonetheless, we also release a testing package and a standard set of testing environments to enhance the practice of comparing proposed solutions on a common research benchmark.

- Increase the variety of training and testing scenarios
- Expand available sensors (stereo depth camera and Ultrawide-band anchors)
- Include other navigation tasks and benchmarks like exploration and social navigation.

References

[1] K. Zhu and T. Zhang, "Deep reinforcement learning based mobile robot navigation: A review," *Tsinghua Science and Technology*, vol. 26, no. 5, pp. 674–691, 2021.

[2] Y. F. Chen, M. Everett, M. Liu, and J. P. How, "Socially aware motion planning with deep reinforcement learning," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 1343–1350.

[3] Y. Song, M. Steinweg, E. Kaufmann, and D. Scaramuzza, "Autonomous drone racing with deep reinforcement learning," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 1205–1212.

[4] U. Patel, N. K. S. Kumar, A. J. Sathyamoorthy, and D. Manocha, "Dwa-rl: Dynamically feasible deep reinforcement learning policy for robot navigation among mobile obstacles," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 6057–6063.

[5] Z. Xu, X. Xiao, G. Warnell, A. Nair, and P. Stone, "Machine learning methods for local motion planning: A study of end-to-end vs. parameter learning," in *2021 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2021, pp. 217–222.

[6] X. Xiao, Z. Wang, Z. Xu, B. Liu, G. Warnell, G. Dhamankar, A. Nair, and P. Stone, "Appl: Adaptive planner parameter learning," *Robotics and Autonomous Systems*, vol. 154, p. 104132, 2022.

[7] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science Robotics*, vol. 7, no. 66, p. eabm6074, 2022. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.abm6074

[8] N. G. Lopez, Y. L. E. Nuin, E. B. Moral, L. U. S. Juan, A. S. Rueda, V. M. Vilches, and R. Kojcev, "gym-gazebo2, a toolkit for reinforcement learning using ros 2 and gazebo," *arXiv preprint arXiv:1903.06278*, 2019.

[9] Y. L. E. Nuin, N. G. Lopez, E. B. Moral, L. U. S. Juan, A. S. Rueda, V. M. Vilches, and R. Kojcev, "Ros2learn: a reinforcement learning framework for ros 2," *arXiv preprint arXiv:1903.06282*, 2019.

[10] W. G. La, L. Kong, S. Muralidhara, and P. Nichat, "Deepsim: A reinforcement learning environment build toolkit for ros and gazebo," *arXiv preprint arXiv:2205.08034*, 2022.

[11] D. Perille, A. Truong, X. Xiao, and P. Stone, "Benchmarking metric ground navigation," in *2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2020, pp. 116–121.

[12] Z. Xu, G. Dhamankar, A. Nair, X. Xiao, G. Warnell, B. Liu, Z. Wang, and P. Stone, "Applr: Adaptive planner parameter learning from reinforcement," in *2021 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2021, pp. 6086–6092.

[13] *Local Planners with Deep Reinforcement Learning for Indoor Autonomous Navigation*. Zenodo, Oct. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.6367976

[14] M. Martini, S. Cerrato, F. Salvetti, S. Angarano, and M. Chiaberge, "Position-agnostic autonomous navigation in vineyards with deep reinforcement learning," in *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2022, pp. 477–484.

[15] A. Eirale, M. Martini, and M. Chiaberge, "Rl-dwa omnidirectional motion planning for person following in domestic assistance and monitoring." [Online]. Available: https://arxiv.org/abs/2211.04993

[16] A. Eirale, M. Martini, L. Tagliavini, D. Gandini, M. Chiaberge, and G. Quaglia, "Marvin: An innovative omni-directional robotic assistant for domestic environments," *Sensors*, vol. 22, no. 14, p. 5261, 2022.

[17] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[18] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*. PMLR, 2018, pp. 1587–1596.

[19] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.