Platform-portable reinforcement learning method to localize underwater targets

Ivan Masmitja
Institut de Ciències del Mar (ICM), CSIC
Barcelona, Spain
masmitja@icm.csic.es

Mario Martin
KEMLG Research Group
Universitat Politècnica de Catalunya, Barcelona Tech. Barcelona, Spain
mmartin@cs.upc.edu

Tom O'Reilly
Monterey Bay Aquarium Research Institute
Moss Landing, USA
oreilly@mbari.org

Narcis Palomeras
Computer Vision and Robotics Institute, Universitat de Girona
Girona, Spain
narcis.palomeras@udg.edu

Kakani Katija
Bioinspiration Lab, Monterey Bay Aquarium Research Institute
Moss Landing, USA
kakani@mbari.org

Joan Navarro
Institut de Ciències del Mar (ICM) CSIC
Barcelona, Spain
joan@icm.csic.es

I. Introduction

One of the main concerns about reinforcement learning (RL) methods is how to transfer the policies learned in simulated environments to reality while obtaining similar behaviors and performance (i.e., sim-to-real transferability), which is of special importance in robot controllers [1]. Multiple research directions have been followed during the last few years to reduce the gap between the simulated and real worlds to accomplish more efficient policy transfer. One of the most widely used methods for learning transfer is domain randomization, which exposes the model to a variety of conditions, to make the model robust to modeling inaccuracies in these aspects. Randomizations are considered pivotal to achieving sim-to-real transfer and robust polices in general [2]. Another common method is system identification, which uses high-fidelity environments with precise mathematical models of physical and dynamic systems. However, system identification has the drawback of being computationally demanding, thus requiring more time for training. Other relevant methods are zero-shot transfer and domain adaptation [3].

Most of the studies on RL have been focused on low-level controllers that use the end-to-end approach, where the RL network uses as input the raw information provided by onboard sensors and gives as output the continuous control actions to apply to the actuators [4]. However, this approach has two main limitations: (i) it has a strong dependency on the platform's configuration, e.g., related to the information available from the sensors and their quality, or the number of actuators, such as thrusters, and their configuration; and (ii) the sim-to-real transfer gap is even harder to reduce, as the trained policy is strongly affected by the dynamics of the robotic platform. For example, in [5] the authors used a second training process in the real vehicle, where the learning procedure continued online. In [6] the controller needed additional adjustments to compensate for the difference between the simulations and the real world, but even with that, the field results showed lower performance.

This work received financial support from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie, grant agreement No 893089. The Spanish Ministerio de Economía y Competitividad, grant agreement No PID2020-114732RB-C31, project BITER-ECO.

In this study, we present a platform-portable deep reinforcement learning method that has been used as a path-planning system to localize underwater objects with autonomous vehicles, Fig. 1. We have designed a high-level control system to reduce the above-mentioned problems and have great sim-to-real transfer capabilities. Moreover, our method is easily configurable to be deployed on different platforms and under different conditions. For example, the trained agent has been successfully deployed in two different vehicles: (i) a Wave Glider, an autonomous surface vehicle (ASV) from Liquid Robotics (USA); and (ii) a Sparus II, an autonomous underwater vehicle (AUV) from IQUA Robotics (Spain). The tests were conducted at Monterey Bay in California, and the Sant Feliu de Guíxols harbor in Catalonia (Spain). In both cases, the vehicles used range-only target tracking methods to localize an anchored transponder [7].

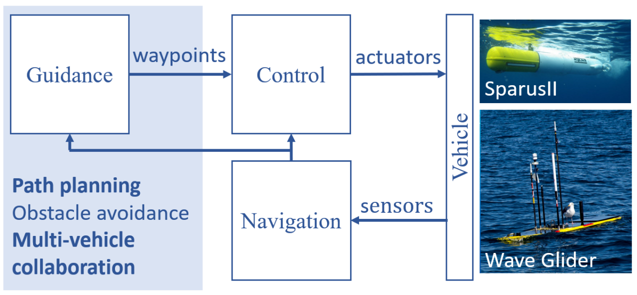

Fig. 1: Guidance, navigation and control system, and some of the main associated research lines in Guidance. In bold, those aspects described in detail are reported.

II. Reinforcement learning path planning

The autonomous navigation system on marine vehicles is typically divided into three main layers, which are known as Guidance, Navigation, and Control systems (GNC) [8], Fig. 1. The Guidance system is the highest control level of the vehicle during a mission. Usually, it has a waypoint generator, which establishes the points to cross to accomplish the goal of the mission. In addition, it can incorporate several algorithms such as path planning, obstacle avoidance, or multi-vehicle collaboration, which are essential to improving the autonomy of the vehicle.

Many algorithms have been designed for AUV path planning (and obstacle avoidance) during the last decades [9]. Most of these works have focused on path planning under the influence of marine currents, obstacle avoidance, or seabed coverage. The optimization objectives for AUV path planning are usually the path length, time, and power consumption. However, path planning can also be used to optimize for underwater target localization [10].

A. Range-only target localization

The relationship between the acoustic sensor location and the accuracy that can be achieved in parameter estimation under different measurement typologies has been widely studied [11]. In general, the computation of the optimal sensor configuration can be carried out analytically by examining the corresponding Cramer-Rao Bound (CRB) or its Fisher Information Matrix (FIM) [12]. If a set of noisy observations is used to estimate a certain parameter of interest, the CRB sets the lowest bound on the covariance matrix that can asymptotically be achievable using any unbiased estimation algorithm. For example, the CRB method was used to find the optimal sensors' locations of an underwater sensor network to find a target using their ranges [13]. This approach was adapted to derive the optimal path shape that an ASV should take to compute the position of an underwater target using range-only and single-beacon techniques [10].

Within this framework, the use of deep RL techniques has been proposed to find the optimal trajectory for an autonomous vehicle to track underwater targets. Deep RL uses the formal framework of the Markov Decision Process (MDP) to define the interaction between a learning agent and its environment in terms of states, actions, and rewards. Whereas most of the attention in deep RL has focused on game theory (e.g., to solve Atari games [14]), the same principles can be used to solve path planning and trajectory optimization. In our previous study [15], we showed that the RL agent[1] can learn a policy with a performance comparable to the analytically derived optimal trajectory. This approach highlighted the potential for tracking marine animals by autonomous underwater vehicles and could enable coordinated fleets of vehicles to localize and track a set of underwater assets via multi-agent, multi-target approaches that are currently intractable with existing methodologies.

In this study, the development of a complete Guidance system for an adaptive AUV or ASV has been proposed, which is based on a deep RL algorithm. The system will be designed to be detached from the lower Control and Navigation layers to make it platform-portable and easily deployable in real environments. It is worth noticing that this is envisioned as a first necessary step to validate the use of RL to tackle such problems, which could be used later on in more complex scenarios.

The case of a single tracker (an AUV/ASV) and a single object to localize have been used, hereinafter the agent and the target, respectively. The final goal of the agent is to localize and track the target. Two key algorithms run simultaneously to achieve this goal: (i) agent path planning, which is based on the policy learned using the RL; and (ii) the target position estimation based on range data acquired online, where we used a Least Square (LS) and a Particle Filter (PF) approaches [16]. In this work, we focus on solving the agent path planning problem by employing the typical scenario where the agent moves in a 2D environment (e.g., an ASV or an AUV at constant depth) and the target's depth is known by the agent. Both the agent and the target have an acoustic modem, which can be used to measure the distance between them. Finally, we also assume that the agent knows its position by using its navigation methods (e.g., GPS or dead reckoning).

B. Deep reinforcement learning architecture

A Soft Actor-Critic (SAC) [17] algorithm has been implemented and tested for solving this problem. SAC is a state-of-the-art model-free actor-critic deep RL algorithm in continuous action domains. The main characteristic is that it maximizes reward while also maximizing the entropy of the policy as a regularizer to obtain robust policies. We conducted many tests with other architectures, such as Deep Deterministic Policy Gradient (DDPG). However, SAC outperformed them in all the studied scenarios.

In addition, the actor-critic algorithm implemented can enable/disable a Long-Short-Term-Memory (LSTM) network, following the work conducted by [18]. The LSTM is a type of RNN that has an outer recurrence from the outputs to the inputs of the hidden layer and also an internal recurrence between LSTM-Cells. A part of the information is transmitted to the next moment in the form of memory and participates in the training of input-output data pairs. Therefore, the training results at the current moment are determined by both the current training data and the historical training data.

A top-level representation of the structure of this recurrent actor-critic framework is illustrated in Fig. 2.

C. Training environment

The training environment is based on the OpenAI particle [19], which is a multi-agent particle world with a continuous observation and action space. This environment has been

[1] **Data and materials availability:** The range-only target localization algorithms with deep RL are available on GitHub: github.com/imasmitja/RLforUTracking

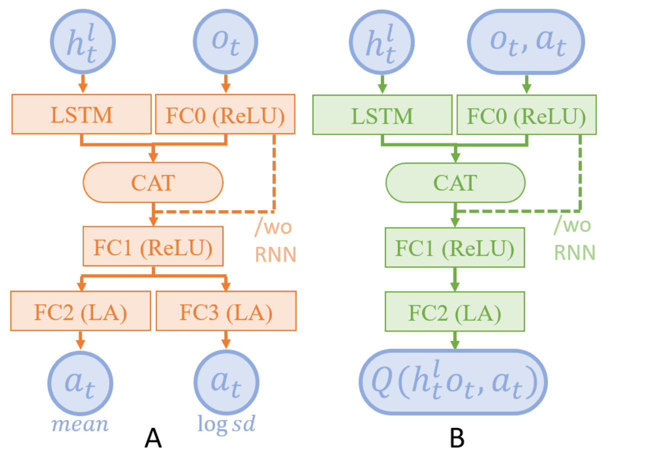

Fig. 2: Overview of the recurrent actor-critic network, (A) Actor and (B) Critic, both with the possibility of enabling/disabling the RNN. Notation: LA is a Linear Activation function, ReLU is a Rectified Linear Unit activation function, FC is a Fully Connected layer, and CAT is a concatenate function.

modified to incorporate the target estimation algorithm (based on an LS or a PF range-only triangulation technique) and its visualization. The OpenAI particle action space has been modified to fit the constraints of our scenario, which is explained below.

In this experiment, we have considered an agent with a constant velocity v, and a single action space referred to the variation of the yaw angle ψ. This is a common operational mode when it is applied to torpedo-shape AUVs (e.g., the Sparus II AUV (Iqua Robotics, Spain)) or vehicles that do not use thrusters (e.g., the Wave Glider (Liquid Robotics, USA)).

The agent is equipped with a sensor that measures distances to the targets at specified discrete intervals of time defined by $\bar{d}_t = ||\mathbf{d}_t|| + w_t, t \in \{1, 2, \ldots, m\}$. Where $\mathbf{d}_t = \mathbf{p}_t - \mathbf{q}_t$ is the relative position vector of the target with respect to the agent, m indicates the number of measurements carried out, and $w_t \sim \mathcal{N}(\epsilon, \sigma^2)$ is a non-zero mean Gaussian measurement error where σ^2 is the variance and ϵ is the systematic error, mostly due to the sound speed uncertainty under water. These measurements can be used to obtain an estimation of the target's position $\hat{\mathbf{q}}$ using range-only and single-beacon techniques [16]. In this study, a simple unconstrained LS algorithm has been chosen during the training for its run-time performance (orders of magnitudes below its competitors). It is worth mentioning that a PF can also be used during the test for its great capability to track a moving target.

The observations at each time-step t that we can get from the environment include the position $\mathbf{p}$ and velocity $\mathbf{v}$ vectors of the agent, the relative position vector of the estimated target position ($\hat{\mathbf{d}}_t = \mathbf{p}_t - \hat{\mathbf{q}}_t$), and the projected distance measured by the sensor $\bar{d}_{pt}$: $\mathbf{o}_t = [\mathbf{p}_t, \mathbf{v}_t, \hat{\mathbf{d}}_t, \bar{d}_{pt}]$. The action space is determined by the force applied to the yaw (ψ) angle of the agent, as $a_t \triangleq u_\psi$.

Additionally, a mini-batch of N experiences $\{(\mathbf{h}_t^l, \mathbf{o}_t, a_t, r_t, \mathbf{o}_{t+1}, d_t)_i\}_{i=1}^N$ is sampled from the replay buffer D of experiences at each iteration. The past history $\mathbf{h}_t^l$ is only taken into consideration if the LSTM is enabled inside the actor-critic algorithm.

Finally, in RL, the agent obtains rewards as a function of the state and the agent's actions. The agent aims to maximize the total expected return $R = \sum_{t=0}^{T} \gamma^t r^t$, where γ is a discount factor and T is the time horizon. A combination of dense and sparse reward methods has been proposed, where we have defined two different goals to optimize the agent's trajectory: (i) a reward function based on the distance between the agent and the target, and (ii) a reward function based on the estimated target position error. In addition, a terminal reward was implemented. The overall goal is to optimize for error reduction but also to maintain the agent close to the target to increase the acoustic link performance while reducing collisions. Consequently, the final reward is given by $r = r_d + r_e + r_{terminal}$, defined as

$$r_d = \begin{cases} \lambda(0.5 - \hat{d}) & if \quad \hat{d} > d_{th} \\ 1 & \text{else} \end{cases}, \tag{1}$$

where λ is a positive constant, $\hat{d}$ is the distance between the agent and the estimated target position, and d_{th} is the predefined distance threshold to be reached by the agent.

$$r_e = \begin{cases} \lambda(0.5 - e_q) & if \quad e_q > e_{th} \\ 1 & \text{else} \end{cases}, \tag{2}$$

where $e_q = ||\hat{\mathbf{q}}_t - \mathbf{q}_t||$ is the error between the predicted target position and the real target position at time-step t, and e_{th} is the predefined error threshold to be reached by the agent.

$$r_{terminal} = \begin{cases} -100 & if \quad \hat{d} > \hat{d}_{max} \\ -1 & if \quad \hat{d} < \hat{d}_{min} \\ 0 & \text{else} \end{cases}, \tag{3}$$

where $\hat{d}_{max}$ is the maximum distance where the agent can go related to the target, and $\hat{d}_{min}$ is a threshold set to avoid collisions between the target and the agent.

III. Vehicle's Guidance System

The deep RL algorithm generates an angle action based on the observation state space (i.e., the position of the tracker, the estimated position of the target, and the range measured). In general, the guidance system does not have access to the closed-loop Control algorithms in most autonomous vehicles. The vehicle can only be controlled by specifying the waypoints that it has to reach. Consequently, the angle actions must be translated into waypoints by placing them at the specified RL angle action related to the current vehicle's yaw. The distance of these generated waypoints has been chosen to be long enough to maintain a desired vehicle's speed (some vehicles will reduce their velocity if they are close to the final position) and also to maintain the safety of the vehicle, as too far away points could cause the vehicle to reach dangerous places such as coastal zones. This step can be observed in the block diagram represented in Fig. 3 referred to as "*Compute next WP with RL*" (in red).

The remaining components of the vehicle's guidance system consist of the following steps: (i) If the system is not initialized, the first position of the vehicle is saved as the origin and the initial target prediction. (ii) A new waypoint will be computed only if the vehicle has traveled beyond a threshold. This is conducted to ensure that the measurements and associated updates are equally distributed across the vehicle's trajectory without influence from its velocity. (iii) If a new range is measured, the estimated target position is updated using either PF or LS. Also, an additional weighted filter can be applied to the estimations. If the range is not measured (e.g., due to poor acoustic communications), the system computes a planar range based on the last predicted target position, as it is required as input to the RL algorithm to compute the new action. Additionally, if the communication between the target and tracker fails for too long, the system will stop updating the vehicle's waypoints as a safety measure (i.e., to allow an external user to stop the vehicle or to stop sending the vehicle to an unwanted place). Finally, in step (iv), the system computes the next waypoint to reach based on the RL action and sends the specific command to the vehicle's main control unit.

IV. Field Tests

To validate the platform-portable approach presented in this study, two different tests were designed. First, a Wave Glider was used to localize an acoustic transponder attached to an AUV docking station located at 70 m depth in Monterey Bay (California), see Fig. 4. And second, a Sparus II AUV to localize a transponder attached to a boat inside the harbor of Sant Feliu de Guíxols (Spain), see Fig. 5.

Besides the differences between both vehicles, it is worth noting that the environment was also significantly different. The Wave Glider was moving in an open area without the possibility of colliding with any obstacles, whereas the Sparus II test was conducted inside a harbor where the vehicle's movement was more constrained. To overcome this issue, a scaling factor was applied in all the dimensions of the state vector used by the RL algorithm deployed in Sparus II. During the training, the reward function had a parameter that gave a maximum reward to the agent (i.e., the tracker) when the distance concerning the target was below 0.3 (which represented 300 m in the field). With this reward, the agent learned an optimal trajectory that consisted of conducting loops with a radius of ∼200 m around the target. To reduce this radius, we applied a scaling factor of 20 to all the distances (e.g., the distance measured between the tracker and the target, or the relative x and y positions of the target). With this, the same policy learned by the agent could be used in the harbor experiment. The Sparus II with this tuned RL policy conducted loops of ∼10 m instead of ∼200 m. Because the scaling factor was applied to all the state space, the policy was not affected, and the target could be localized.

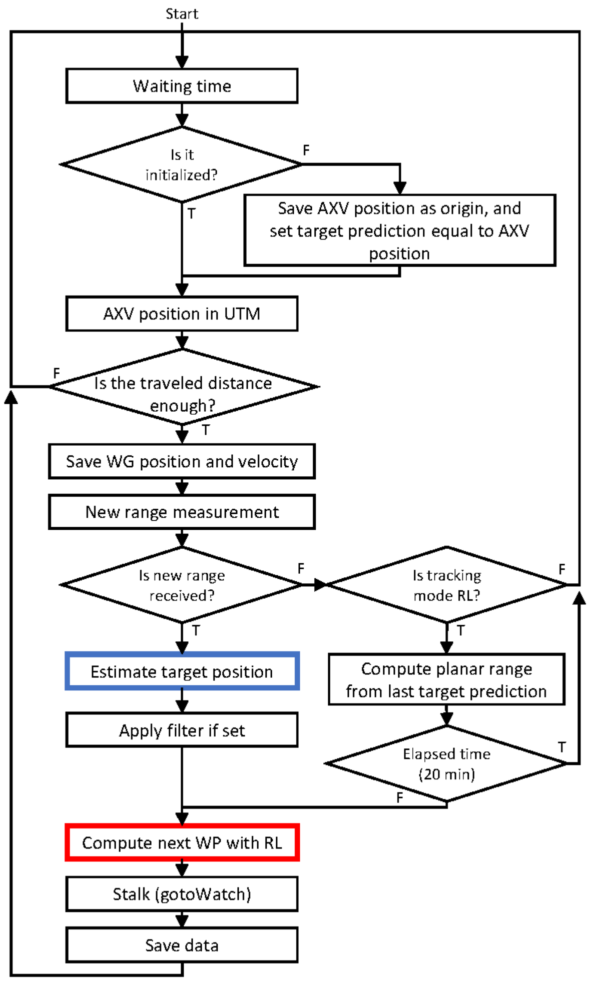

Fig. 3: Block diagram of the vehicle's guidance system. The target position is estimated in the blue block using a Particle Filter or a Least Square method. The next waypoint is computed based on the Reinforcement Learning algorithm in the red block and sent to the main vehicle's control unit using the desired command (e.g., a *GoToWatch*).

The vehicle's guidance system was installed inside the Wave Glider Hotspot payload. The Hotspot is an embedded computer with a set of communication interfaces (acoustic and aerial) that has access to the main vehicle control unit. This makes the vehicle a communication hub between the underwater assets and onshore servers. This payload was in charge of executing the RL path planning and target estimation algorithms, as well as other safety measures, such as avoiding navigation close to the shore. Because the vehicle was on the surface and had cell phone communications, the evolution of the experiment could be monitored continuously, which allowed the debugging of the system.

On the contrary, the Sparus II architecture is based on the Robot Operating System (ROS). This allowed to the simulation of the RL path planning and the overall guidance system in a realistic simulated environment from Iqua Robotics

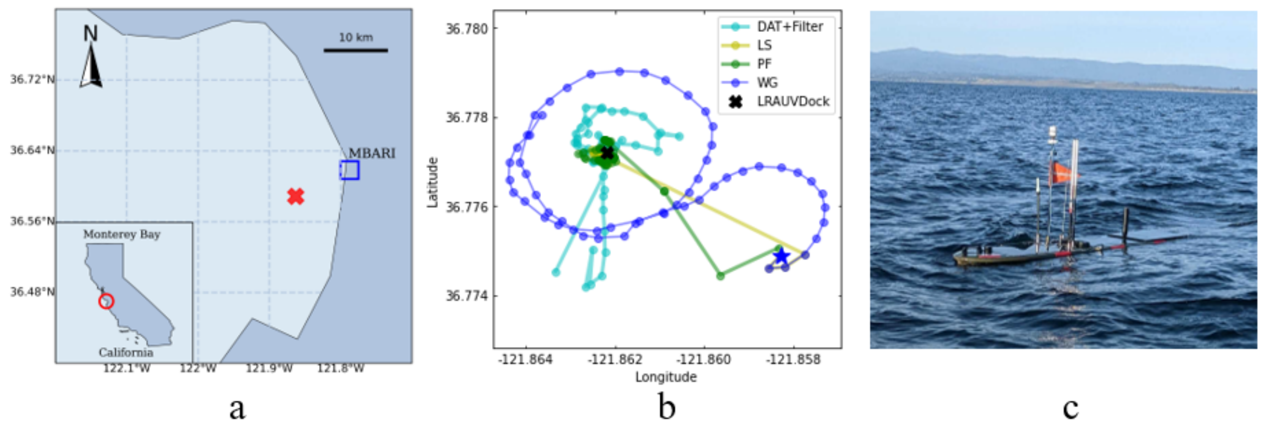

Fig. 4: Test conducted at Monterey Bay (a) with the Wave Glider ASV (c). Results are shown in (b), where the vehicle trajectory is represented (blue) and the target position is estimated with ultra-short baseline (light-green), least squares (yellow), and particle filters (green) methods.

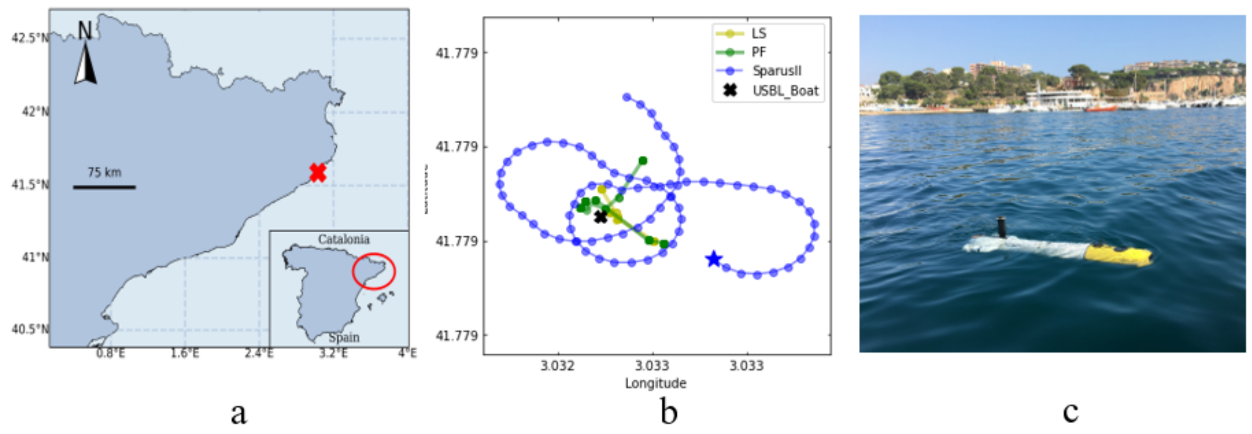

Fig. 5: Test conducted at Sant Feliu de Guíxols harbor (a) with the Sparus II AUV (c). Results are shown in (b), where the vehicle trajectory is represented (blue) and the target position is estimated with least squares (yellow), and particle filters (green) methods.

(iquarobotics.com), using its control architecture COLA2 [20]. Different simulations were conducted before the field tests to ensure that the implementation was ready to be used and the implemented ROS Guidance system package was interacting correctly with the vehicle's architecture. For example, in Fig. 6 the vehicle was able to localize a static target.

V. Conclusions

This study demonstrates a path planning system based on deep RL that has been deployed on two different robotic platforms and conducted field tests. The strategy proposed here shows how the sim-to-real transferability of this kind of algorithm can be addressed by using a high-level control system. The main guidance system architecture is presented, and two field experiments are conducted in an open-sea area and inside a harbor, using as agents a Wave Glide ASV and a Sparus II AUV. The RL algorithm was trained in a simulated environment to optimize the agent's trajectory for range-only underwater target localization. The architecture developed here could also be used to train multi-agent and multi-target scenarios where a group of coordinated agents can navigate to find and track a series of underwater assets at previously unknown positions.

Acknowledgment

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 893089. This work also received financial support from the Spanish Ministerio de Economía y Competitividad (BITER-ECO: PID2020-114732RBC31). This work acknowledges the 'Severo Ochoa Centre of Excellence' accreditation (CEX2019-000928-S). We gratefully acknowledge the David and Lucile Packard Foundation.

References

[1] E. Salvato, G. Fenu, E. Medvet, and F. A. Pellegrino, "Crossing the reality gap: A survey on sim-to-real transferability of robot controllers in reinforcement learning," *IEEE Access*, vol. 9, pp. 153 171–153 187, 2021.

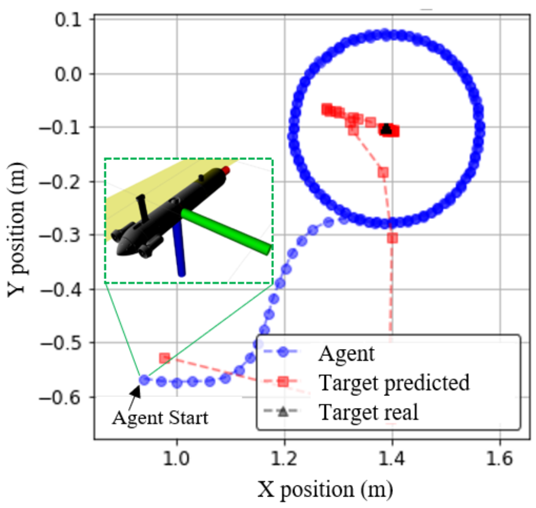

Fig. 6: Vehicle's guidance system simulated using a ROS environment with the Sparus II AUV architecture

[2] B. Osiński, A. Jakubowski, P. Ziecina, P. Miloś, C. Galias, S. Homoceanu, and H. Michalewski, "Simulation-based reinforcement learning for real-world autonomous driving," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 6411–6418.

[3] W. Zhao, J. P. Queralta, and T. Westerlund, "Sim-to-real transfer in deep reinforcement learning for robotics: a survey," in *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2020, pp. 737–744.

[4] W. Luo, P. Sun, F. Zhong, W. Liu, T. Zhang, and Y. Wang, "End-to-end active object tracking via reinforcement learning," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. PMLR, 10–15 Jul 2018, pp. 3286–3295. [Online]. Available: https://proceedings.mlr.press/v80/luo18a.html

[5] I. Carlucho, M. De Paula, S. Wang, Y. Petillot, and G. G. Acosta, "Adaptive low-level control of autonomous underwater vehicles using deep reinforcement learning," *Robotics and Autonomous Systems*, vol. 107, pp. 71–86, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889018301519

[6] C.-H. Pi, K.-C. Hu, S. Cheng, and I.-C. Wu, "Low-level autonomous control and tracking of quadrotor using reinforcement learning," *Control Engineering Practice*, vol. 95, p. 104222, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0967066119301923

[7] I. Masmitja, S. Gomariz, J. Del-Rio, B. Kieft, T. O'Reilly, P.-J. Bouvet, and J. Aguzzi, "Optimal path shape for range-only underwater target localization using a wave glider," *The International Journal of Robotics Research*, vol. 37, no. 12, pp. 1447–1462, 2018.

[8] T. Fossen, "Marine control systems: guidance, navigation and control of ships, rigs and underwater vehicles," *Marine Cybernetics*, 2002.

[9] C. Cheng, Q. Sha, B. He, and G. Li, "Path planning and obstacle avoidance for auv: A review," *Ocean Engineering*, vol. 235, p. 109355, 2021.

[10] I. Masmitja, S. Gomariz, J. Del-Rio, B. Kieft, T. O'Reilly, P.-J. Bouvet, and J. Aguzzi, "Optimal path shape for range-only underwater target localization using a wave glider," *The International Journal of Robotics Research*, vol. 37, no. 12, pp. 1447–1462, 2018.

[11] D. Ucinski, *Optimal Measurement Methods for Distributed Parameter System Identification*. Boca Raton: CRC Press, 2004.

[12] H. L. Van Trees, K. L. Bell, and Z. Tian, *Detection, estimation, and modulation theory, part I: detection, estimation, and linear modulation theory*. John Wiley & Sons, 2004.

[13] D. Moreno-Salinas, A. Pascoal, and J. Aranda, "Optimal sensor placement for acoustic underwater target positioning with range-only measurements," *IEEE Journal of Oceanic Engineering*, vol. 41, no. 3, pp. 620–643, 2016.

[14] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[15] I. Masmitja, M. Martin, K. Katija, S. Gomariz, and J. Navarro, "A reinforcement learning path planning approach for range-only underwater target localization with autonomous vehicles," in *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*, 2022, pp. 675–682.

[16] I. Masmitja, S. Gomariz, J. Del-Rio, B. Kieft, T. O'Reilly, P.-J. Bouvet, and J. Aguzzi, "Range-only single-beacon tracking of underwater targets from an autonomous vehicle: From theory to practice," *IEEE Access*, vol. 7, pp. 86 946–86 963, 2019.

[17] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," *arXiv preprint arXiv:1801.01290*, 2018.

[18] B. Li and Y. Wu, "Path planning for uav ground target tracking via deep reinforcement learning," *IEEE Access*, vol. 8, pp. 29 064–29 074, 2020.

[19] J. K. Terry, B. Black, N. Grammel, M. Jayakumar, A. Hari, R. Sulivan, L. Santos, R. Perez, C. Horsch, C. Dieffendahl, N. L. Williams, Y. Lokesh, R. Sullivan, and P. Ravi, "Pettingzoo: Gym for multi-agent reinforcement learning," *arXiv preprint arXiv:2009.14471*, 2020.

[20] N. Palomeras, A. El-Fakdi, M. Carreras, and P. Ridao, "Cola2: A control architecture for auvs," *IEEE Journal of Oceanic Engineering*, vol. 37, no. 4, pp. 695–716, 2012.