MARINE ROBOTS

Dynamic robotic tracking of underwater targets using reinforcement learning

I. Masmitja[1,2]*, M. Martin[3,4], T. O'Reilly[2], B. Kieft[2], N. Palomeras[5], J. Navarro[1], K. Katija[2]

To realize the potential of autonomous underwater robots that scale up our observational capacity in the ocean, new techniques are needed. Fleets of autonomous robots could be used to study complex marine systems and animals with either new imaging configurations or by tracking tagged animals to study their behavior. These activities can then inform and create new policies for community conservation. The role of animal connectivity via active movement of animals represents a major knowledge gap related to the distribution of deep ocean populations. Tracking underwater targets represents a major challenge for observing biological processes in situ, and methods to robustly respond to a changing environment during monitoring missions are needed. Analytical techniques for optimal sensor placement and path planning to locate underwater targets are not straightforward in such cases. The aim of this study was to investigate the use of reinforcement learning as a tool for range-only underwater target-tracking optimization, whose promising capabilities have been demonstrated in terrestrial scenarios. To evaluate its usefulness, a reinforcement learning method was implemented as a path planning system for an autonomous surface vehicle while tracking an underwater mobile target. A complete description of an open-source model, performance metrics in simulated environments, and evaluated algorithms based on more than 15 hours of at-sea field experiments are presented. These efforts demonstrate that deep reinforcement learning is a powerful approach that enhances the abilities of autonomous robots in the ocean and encourages the deployment of algorithms like these for monitoring marine biological systems in the future.

INTRODUCTION

Exploration of Earth's final frontier, the ocean, has become increasingly important due to the potential consequences that human activities could have on ecosystem functioning and marine biodiversity. The ocean provides approximately 15% of the animal protein consumed worldwide by humans (*1*) and influences important biological processes such as the carbon cycle and its contribution to climate change (*2*, *3*). All these factors highlight the importance of engaging stakeholders, scientists, resource managers, and members of the public to monitor these vulnerable marine ecosystems [for example, through marine protected areas (*4*–*6*)] and develop new technologies capable of characterizing their change (*7*).

Knowing the position of underwater targets (for example, tagged species or underwater vehicles that monitor the marine environment) and being able to follow them over time and space is critical to many applications. Quantifying animal movements informs spatial ecology and enables the effective application of conservation and management strategies, such as in marine protected areas (*8*). Autonomous vehicles have been used in conjunction with fixed stations to monitor and localize tagged species (*7*, *9*). Multiple robotic platforms, in coordination with each other, have been used to characterize different oceanographic phenomena, such as fronts and eddies (*10*, *11*). To study these currents and the biological communities drifting within them, each robot must cooperate and exchange information to coordinate their activities, and vehicle localization and tracking are essential for this to succeed (*12*, *13*). Periodically localizing seabed platforms ensures their operation and recovery at the end of sampling missions (*3*, *14*). These studies used underwater vehicles with preprogrammed trajectories and had a limited degree of flexibility. As a result, new methods and tools are still needed to improve the localization and tracking of moving underwater targets in a more adaptable manner.

Oceanographic monitoring is challenged by the low reliability and bandwidth of most underwater communication systems, where the GPS does not function (*15*). Robotic platforms and autonomous underwater vehicles (AUVs) have become useful tools for ocean observation by reducing cost and increasing mission duration and efficiency (*3*, *7*, *16*, *17*). However, the increasing complexity of missions carried out by such platforms have pushed them to the limits of their reliability and persistence. Because energy is limited in such platforms, vehicle control optimization is needed to maximize their utility. The ability to study animals and phenomena on vast scales with high degrees of uncertainty could be enabled by developing adaptive vehicles that respond to conditions in the environment. New vehicles driven by machine learning (ML) algorithms to increase autonomy are increasingly being developed and implemented in the hopes of filling this capability gap (*18*, *19*).

ML is the area of artificial intelligence concerned with computer algorithms that can solve problems by learning from data using three main approaches: supervised, unsupervised, and reinforcement learning (RL) (*20*). Whereas supervised learning is learning from a training set of labeled examples [for example, object detection (*21*)], unsupervised learning is about finding hidden structures in collections of unlabeled data [for example, personalized digital marketing recommendations (*22*)]. RL contrasts with these

[1]Institut de Ciències del Mar (ICM), CSIC, Barcelona 95062, Spain. [2]Research and Development, Bioinspiration Lab, Monterey Bay Aquarium Research Institute MBARI, Moss Landing, CA 95062, USA. [3]Knowledge Engineering and Machine Learning Group, Universitat Politècnica de Catalunya, Barcelona Tech., Barcelona 08034, Spain. [4]Barcelona Supercomputing Center (BSC), Barcelona 08034, Spain. [5]Computer vision and Robotics Institute, Universitat de Girona, Girona 17003, Spain.
*Corresponding author. Email: masmitja@icm.csic.es

approaches in that learning takes place through trial and error, where examples can be illustrative of reconstructing a field, pattern, or phenomenon. RL is a suitable solution for designing robots that can learn optimal policies by interacting directly with the environment.

To enhance the target-tracking capabilities of marine robots and our understanding of the ecosystem, a guidance system based on soft actor-critic (SAC) (*23*) deep RL algorithms has been developed. Whereas most of the attention in deep RL has focused on game theory [for example, to solve Atari games (*24*) or to master the game of Go (*25*)], the same principles can be used to solve path planning and trajectory optimization problems (Table 1). Previous studies have shown that aerial gliders can navigate atmospheric thermals autonomously (*26*), stratospheric Loon superpressure balloons have learned optimal control to maintain their position at multiple locations (*27*), and an RL agent has been trained to efficiently navigate in simulated vortical flow fields (*28*). Actor-critic architecture has also been used to track a ground target using an uncrewed autonomous vehicle (*29*), where recurrent neural networks (RNNs) were able to control an agent to avoid collisions with different obstacles and to reach the target using range and angle information. However, these existing efforts have been largely carried out in simulated environments or under controlled conditions (*30*, *31*).

In this study, a path planning system using deep RL for an adaptive autonomous surface vehicle (ASV), which can locate and track submerged targets autonomously, has been demonstrated. The overall goal is to optimize the tracking trajectory to reduce the target prediction error and ensure a good acoustic communication link. This path planning system is the core component of the ASV's guidance system, which establishes the mission waypoints. The algorithm was designed independently of the control and navigation layers to make it platform independent and is easily deployable in real environments, in contrast to other studies focused on low-level control (*31*–*33*). Last, we have implemented the algorithms in Python, which can be used to benchmark future developments in the field.

RESULTS

Learning the path for underwater tracking

Problem formulation

We consider the case of a single tracker (an ASV) and a single moving target (an AUV or an animal instrumented with an electronic device or acoustic tag), referred to as the agent and the target, respectively. Two key algorithms run simultaneously onboard the ASV to achieve the localization and tracking goal: (i) the target position estimation and (ii) the agent path planning. The

Table 1. RL application comparison. Different studies have been conducted related to the use of RL methods to control an autonomous vehicle in different environments (air/land/sea). Here, some of the most relevant in the sea environment are presented and compared with our method (in chronological order). In addition, the newest publications in other journals in air and land environments are also presented. –, information not applicable; ●, yes; ○, no.

Year	Environment	Mission	Algorithm	Sim tests	Lab tests	Field tests	Comparison	Open source	Reference
2020	Air	Navigation	QR-DQN	●	–	●	●	○	(*27*)
2018	Air	Navigation	Q-learning	○	–	●	○	○	(*26*)
2020	Land	Locomotion	TNC	●	–	●	●	○	(*52*)
2022	Sea	Target tracking*	H-LSTM-SAC	●	–	●	●	●	Ours
2022	Sea	Path following/collision	DDPG	●	–	○	●	○	(*53*)
2022	Sea	Target tracking†	MAG	●	–	○	●	○	(*54*)
2022	Sea	Dynamic positioning	NMPC	●	–	●‡	●	○	(*55*)
2022	Sea	Interception	SLDDPG	●	–	○	○	○	(*56*)
2022	Sea	Path following/collision	Actor-critic	●	●	○	●	○	(*57*)
2021	Sea	Navigation	V-RACER	●	–	○	●	●	(*28*)
2021	Sea	Dynamic positioning	PPO	●	–	○	●	●	(*58*)
2021	Sea	Path following/collision	PPO	●	–	○	○	●	(*59*)
2020	Sea	Collision	DDQN	●	–	●§	●	○	(*60*)
2020	Sea	Path following	SPMPC	●	–	●‖	●	○	(*61*)
2019	Sea	Docking control	DQN	●	–	○	●	○	(*32*)
2019	Sea	Path following	DQN	●	–	○	○	○	(*62*)
2019	Sea	Path following	DDPG	●	–	●¶	●	○	(*63*)
2019	Sea	Target search	A3C	●	●	○	●	○	(*30*)
2015	Sea	Path optimization	Q-learning	●	–	○	●	○	(*64*)
2005	Sea	Target tracking*	Q-learning	○	●	○	○	○	(*65*)

*Target tracking using range-only methods. †Target tracking with video processing. ‡Field tests conducted only for approximately 5 min and 10 m. §Field tests conducted in a lake where the obstacle to avoid was simulated, not real. ‖Field tests conducted in a harbor with limited boat displacement (approximately 10 m) and experiment duration (approximately 5 min). ¶Field tests conducted in a lake.

target's position is estimated using a single-beacon and range-only technique (*34*), which outperforms other methods because of its scalability, flexibility, and accuracy (*35*). The distance between the agent and the target can be measured using the same acoustic modems used for intervehicle communications. However, one of the main drawbacks of using range to localize underwater targets is the inherent ambiguity presented during triangulation. Having accurate target predictions is key to this method, because the agent must stay close to the target to increase the acoustic link performance. To tackle this issue, a deep RL algorithm is used for agent path planning that generates the next agent direction (Fig. 1).

Simulated environment

The simulation environment is based on the OpenAI particle (*36*), which is a multiagent particle world with a continuous observation and action space. This environment was modified to incorporate the range-only target estimation algorithm and its visualization using least square (LS) and particle filter (PF) methods. Whereas LS has excellent performance for static target estimation scenarios, PF outperforms it when the target is moving. LS was used during the training process because its computational runtime is orders of magnitude below that of PF.

The distance measured between the agent and the target was modeled using a nonzero mean Gaussian measurement error $w_t \sim \mathcal{N}(\varepsilon, \sigma^2)$, where σ^2 is the variance and ε is the systematic error, mostly due to the sound velocity uncertainty in underwater environments (*37*). During training, these values were set to $\sigma^2 = 1$ m and $\varepsilon = 1\%$ of the distance measured. Furthermore, if the distance between the agent and the target is greater than a threshold, the agent does not receive a range measurement and, therefore, is encouraged to search for the target.

The agent was trained with random target position, velocity, and direction for each simulation run, which was the beginning of a new tracking mission. The simulation environment approximates the effect of ocean currents by modifying the position of the agent at each time step. These currents were also randomly initialized at the beginning of each episode. Last, a dropping factor was implemented to simulate the lack of communication, and therefore range measurements, between the agent and the target.

Deep RL algorithms

The use of deep Q-learning and off-policy algorithms in continuous state and action space presents a major challenge for stability and convergence (*38*). To overcome such limitations, a separate actor network is often used to perform the maximization over actions in the optimal action-value function, such as in deep deterministic policy gradient (DDPG) (*39*). However, its applicability is not straightforward because of hyperparameter sensitivity. In (*23*), the authors drew on the maximum entropy framework with SAC algorithms, which aim to maximize the expected reward while also maximizing entropy (to succeed at the task while acting as randomly as possible), showing great potential for both sample-efficient learning and stability. Maximizing the entropy of the behavior while optimizing the reward obtained allows for the finding of alternative options to solve the problem, which is useful in robotic systems with noise in sensor readings or when conditions of the environment change. These two actor-critic algorithms were implemented and tested to compare their performance as a range-only target-tracking and path planning system.

A long-short-term-memory (LSTM) network was integrated to study its effects on algorithm performance. Specifically, two different structures were implemented: (i) an RNN that takes as input the last n states (LSTM-SAC) following (*29*) and (*40*) and (ii) a simpler

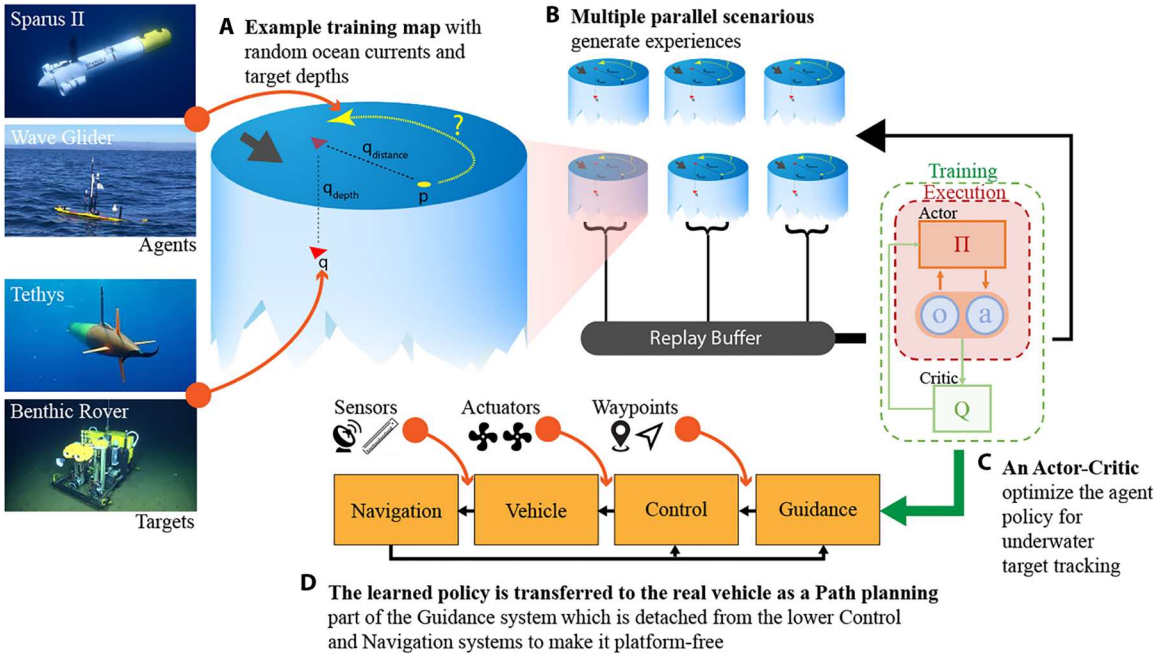

Fig. 1. Deep RL concept as range-only path planning. An agent was trained in a virtual environment that uses real conditions, such as ocean currents and distance measurement noise (**A**). During the training, multiple parallel scenarios were used to boost the process (**B**), and different actor-critic algorithms were studied (**C**). Last, the policy learned was transferred to the real vehicle as a path planning method as part of its guidance system (**D**).

implementation where one of the hidden layers was replaced by an RNN (H-LSTM-SAC) with a history length of 1 and the internal hidden state was used as input on the next step as a memory unit (Fig. 2, A and B). 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. Part of the state information is transmitted to the next moment in the form of memory and participates in the training of input-output data pairs. Hence, the training results at the current moment are determined by both the current training data and the historical training data. A high-level representation of the two recurrent actor-critic architectures is illustrated in Materials and Methods.

The agent obtains rewards as a function of the state and the agent's actions. The agent's goal is 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. In this study, a combination of dense and sparse reward methods was used, where two different goals to optimize the agent's trajectory were defined as (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. Last, a terminal reward was also implemented. The overall goals are to optimize error reduction, reduce collisions between the agent and the target (for example, if both the target and the agent are at the same depth or on the surface), and minimize the distance between the agent and target so as to increase the acoustic link performance. A detailed explanation with mathematical notations describing these algorithms can be found in Materials and Methods.

Experimental outcomes

A set of trials were conducted in the simulated environment to evaluate deep RL algorithms as a guidance system for an ASV to track underwater targets using range-only triangulation techniques. The agent's constant velocity was set to $v = 1$ ms^{-1}, and the sampling time interval was set to $\Delta t = 30$ s. For each simulation run, all the distances between the agent and target had a maximum value of 1, which represented 1 km, and the agent's and target's velocities were scaled accordingly. The measurement noise w_t was set with a σ of 1 m and an ε of 1% of the distance, which are values close to real conditions (*41*). Ocean currents up to 1/2 of the agent's velocity were introduced, which had a random velocity and direction at each episode. A dropping factor of 10% was also used to simulate the probability of dropped communications between the agent and the target. Last, the number of steps per episode was set to 200, which represents more than 1.5 hours of tracking in real conditions. The remaining simulation settings and hyperparameters are presented in table S1.

Training results

One key concern was whether an agent can find the optimal policy to localize and track an underwater target using range-only methods. The agent was trained using a moving target with a constant direction of $\psi_q \in [0,2\pi)$ degrees and a velocity equal to $v_q \in [0,1/3]$ ms^{-1} of the agent's velocity, both of which were initialized randomly at the beginning of each episode. The training was conducted using the DDPG and SAC algorithms (with and without the LSTM network) and the proposed H-LSTM-SAC architecture. We observed the best results using the SAC algorithms (fig. S1), which are the focus in the following sections.

At the beginning (with a random agent), the agent conducted a nearly linear trajectory, which yields a poor predicted position (see training evolution and policy learned in Fig. 3). After a few hundred thousand episodes, the agent learned to reach the target, which produced an increase in the number of collisions between them (Fig. 3). In parallel, the agent also started to learn to circle around the target to improve the estimated target position, thereby reducing its prediction error. After 0.6 million episodes, the agent learned to circle closer to the predicted target position, but the loops were not conducted on top of the predicted target position, which yields a suboptimal solution. After a few more iterations, the agent finally improved its policy, learning to circle on top of the predicted target position, which increased the overall performance. Last, differences in performance among the SAC, LSTM-SAC, and H-LSTM-SAC algorithms were observed after 1 million episodes (Fig. 3). Whereas the SAC stopped improving, the H-LSTM-SAC learned a better strategy to avoid collisions with the target, and

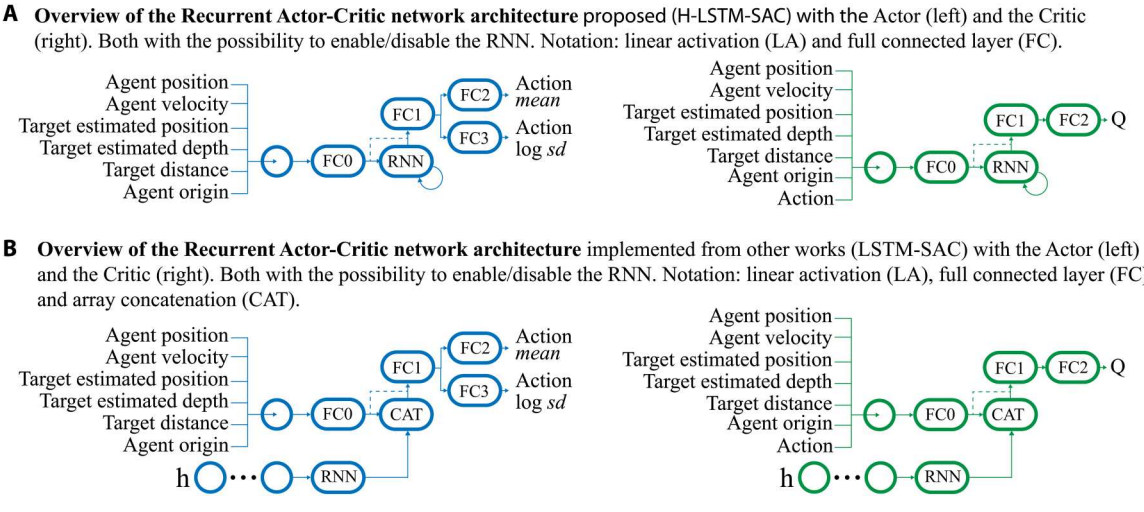

Fig. 2. A high-level representation of the implemented deep recurrent RL algorithms. The proposed H-LSTM-SAC algorithms in which a single-cell RNN was used and the hidden state was passed to the next step (**A**) and the version implemented from previous works, where an RNN with a history *h* of the last *n* observations was used (**B**). Both architectures can enable or disable the RNN part (dotted line). In addition, besides the SAC architecture, a DDPG and a TD3 were also implemented.

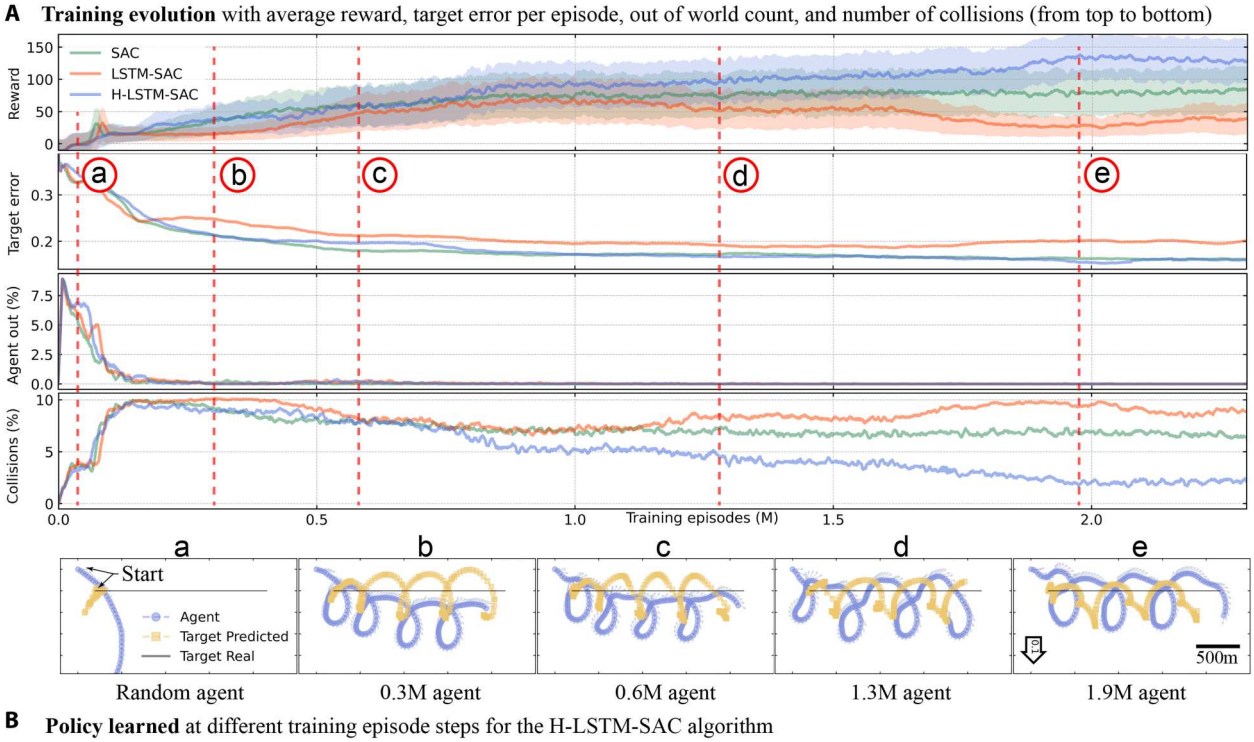

Fig. 3. Overview of the training performance. The reward error, the target prediction error, the agent out of world count, and the number of collisions evolution during the training (**A**); some examples of the policy learned at different training phases (**B**).

the reward increased until 2 million steps (Fig. 3). This contrasts with the LSTM-SAC implementation, which was not able to maintain its performance beyond 1 million steps and started to decrease, thereby resulting in poor target prediction and more collisions (Fig. 3).

Test results using the simulated environment

The trained agent was compared against the optimal trajectory derived analytically (*41*), which is a set of measurements equally distributed on a circumference centered on top of the target (or predefined path). The circumference's radius must be at least equal to $\sqrt{2}$ times the target's depth, and here, a circumference with a radius of 180 m was used. The performance of the trained agent was compared with the reliable evaluation procedures reported in (*42*). The trials were conducted 100 times in the simulated environment, which used a random seed for each execution, and median values were computed for different parameters to compare their performance. The target prediction error (fig. S2), the probability of improvement of the deep RL algorithms versus the predefined path (fig. S3), and the distance between the agent and the target at each time step (fig. S4) were used as performance indicators that are summarized in Table 2.

The predefined path showed the greatest values in the steady state error (0.2 m) for a static target scenario and without ocean currents (an ideal environment). This was expected because the predefined path follows the optimal trajectory that can be conducted to obtain the greatest accuracy of the target's position. RL path planning was also capable of obtaining great accuracy, especially with the SAC algorithm. The SAC and H-LSTM-SAC algorithms learned a better approximation to the target, which resulted in increased accuracy at the start of tracking (4.6 m of error) and provided faster target localization (greater initial time values). When the effect of ocean currents was added, the predefined path could not maintain its performance, and both SAC and H-LSTM-SAC algorithms showed greater values. This poor performance is due to the limitations of the environment in implementing a proper closed-loop controller. Despite these limitations, the simulation environment can still be used to evaluate the performance of the deep RL methods.

When the different algorithms were tested in a moving target scenario, the performance of LS to localize the target decreased quickly with its velocity. In such scenarios, the PF algorithm outperforms the LS algorithm. The RL algorithms can generate a path that is more capable than the predefined path of accurately localizing and tracking the target, especially at high target velocities, for both steady and transient states. The agents were tested using a target with a random walk with Lévy flight distribution (*43*). This test shows that the agent, which did not see this kind of behavior during the training sessions, was still capable of tracking the target. Some of the trajectories conducted by the predefined path and the H-LSTM-SAC algorithm can be observed in fig. S5.

The probability of improvement of the SAC and SAC-LSTM algorithms versus the predefined path (fig. S2) was also computed using Mann-Whitney *U* statistic methodology, where a probability of improvement greater than 0.5 means that the algorithm has greater performance (*42*). The RL path planning methods have, in general, an equal or greater probability of improvement when tracking targets, which means that they have learned a policy that is at least as good as more traditional methods. The real distance between the agent and the target was measured, and stability in distance between the agent and target was compared between

Table 2. Summary of the test results conducted to evaluate and compare the performance of SAC and H-LSTM-LSTM algorithms against the predefined path (def. Path) for static, low-speed (1/10 of the agent's velocity), high-speed (1/3 of the agent's velocity), and random Levy movement targets. The minimum error achievable with an ideal solution obtained analytically is indicated by a double dagger. ●, value out of range; □, value not valid: probability of improvement versus the predefined path.

Scenario	Parameter	def. Path LS*	SAC LS*	/w LSTM LS*	def. Path LS†	SAC LS†	SAC + LSTM LS†	def. Path PF†	SAC PF†	/w LSTM PF†
Static target	Initial time (steps)	69	50	51	72	45	45	●	●	●
	Median transient error (m)	6.1	4.6	4.7	6.2	4.4	4.6	25.2	19.8	19.0
	Median steady error (m)	0.2‡	0.4	0.6	1.4	0.5	0.6	21.8	15.3	14.7
	Median transient distance (m)	356	330	303	371	332	342	357	351	358
	Median steady distance (m)	181	251	202	195	251	260	209	257	267
	Probability of improvement	□	0.4	0.3	□	0.8	0.7	□	0.7	0.7
Low-speed target	Initial time (steps)	60	46	48	64	51	76	10	8	9
	Median transient error (m)	61.4	59.2	59.2	63.9	60.3	60.9	32.8	29.1	28.5
	Median steady error (m)	52.8	56.7	56.6	54.7	57.8	57.8	26.1	22.6	23.2
	Median transient distance (m)	363	329	337	379	330	337	357	351	353
	Median steady distance (m)	211	251	257	233	250	254	267	267	267
	Probability of improvement	□	0.5	0.5	□	0.5	0.5	□	0.6	0.6
High-speed target	Initial time (steps)	●	●	●	●	●	●	5	5	5
	Median transient error (m)	225.7	217.7	219.0	235.2	223.7	224.6	114.1	83.8	82.2
	Median steady error (m)	310.5	205.4	203.8	343.1	222.8	224.2	149.8	67.9	69.9
	Median transient distance (m)	466	344	359	490	354	371	419	357	361
	Median steady distance (m)	514	249	244	615	287	298	477	259	267
	Probability of improvement	□	0.6	0.7	□	0.6	0.6	□	0.6	0.6
Levy movement target	Initial time (steps)	53	38	41	55	39	40	9	6	7
	Median transient error (m)	143.8	134.2	137.1	149.4	137.0	138.4	68.5	65.2	61.5
	Median steady error (m)	78.7	82.4	79.5	79.5	84.9	81.5	43.3	41.0	42.7
	Median transient distance (m)	392	333	347	408	340	353	375	356	358
	Median steady distance (m)	226	248	250	246	248	248	210	257	267
	Probability of improvement	□	0.5	0.5	□	0.5	0.5	□	0.5	0.5

*Without ocean currents and 10% measurement dropping. †Considering ocean currents with a velocity equal to 1/3 of the agent's velocity and 10% measurement dropping.

algorithms. The RL path planning algorithm provided greater performance (the distance between the agent and the target decreases faster) at the beginning of the mission, and the distance was more constantly maintained at higher target velocities.

A performance evaluation was conducted with a set of trial tests using the high-fidelity environment developed by Iqua Robotics (iquarobotics.com). This approach was used to ensure the RL path planning applicability to open-ocean scenarios. This environment uses a realistic dynamic model of a commercial vehicle and implements the appropriate communication and command architecture control (see Supplementary Methods and fig. S11).

Field test demonstration

The validation of the whole system was conducted in Monterey Bay, between 5 and 10 km off the coast of California (Fig. 4 and figs. S6 to S10). The RL agent was a Wave Glider (Liquid Robotics, USA) that was equipped with Monterey Bay Aquarium Research Institute (MBARI)'s Hotspot payload, which provides a relay between acoustics and cellular/satellite communications. The tracked target was an MBARI long-range AUV (LRAUV), which was equipped with an imaging payload.

The LRAUV was programmed with a "drift mission" (the vehicle drifted with ocean currents at depths between 30 and 50 m below the surface). In addition, a 60-min communication threshold was established, and the vehicle surfaced periodically to establish radio communication and fix a GPS position. Fig. 4C shows how the Wave Glider was able to track the LRAUV for more than 2.5 km and approximately 5 hours.

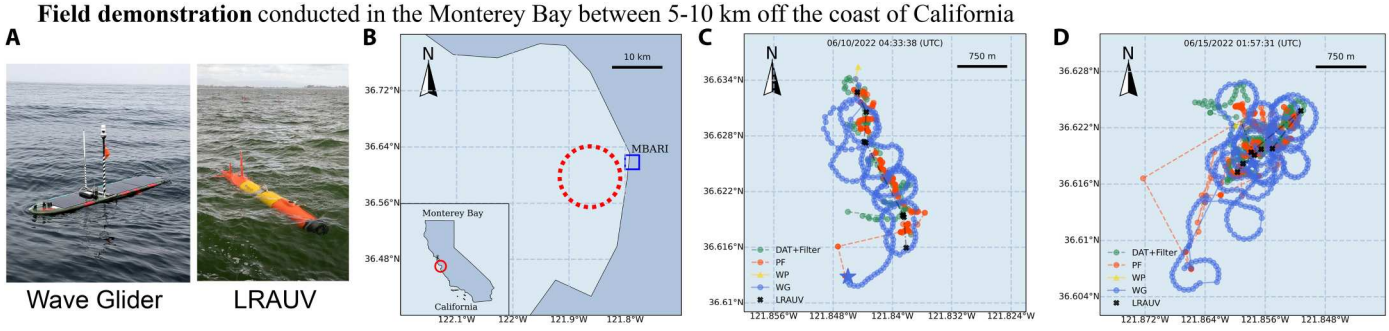

Fig. 4. Field test results. Several real-world tests were conducted in Monterey Bay. These tests were carried out with a Wave Glider that used the H-LSTM-SAC algorithm to track a LRAUV (**A**) between 5 and 10 km off the coast of California (**B**). Two missions are represented in (**C**) and (**D**) where the blue points are the Wave Glider trajectory, the gray cross is the LRAUV real position, and the red and green points are the LRAUV estimated positions using the PF and DAT.

Other mission profiles were also tested. In Fig. 4D, the tracking performance can be observed during a "keep-station mission": The LRAUV started with a drifting mission, but if it reached the maximum distance from a specified point (the station), it returned to its original location at high speed, doing a straight-line transect. These kinds of trajectories are the most challenging ones for range-only target-tracking approaches, especially for PF algorithms, because the target does not move at a constant velocity, and, even worse, its direction changes suddenly and drastically. During one of these transects, the PF algorithm temporarily lost the position of the LRAUV. Even in this worst-case scenario, the Wave Glider was able to maintain tracking for approximately 9.5 hours of the mission. Consequently, the Wave Glider was close enough for excellent acoustic communications but far enough away to avoid any collisions with the LRAUV while on the surface.

The real-world intricacies of underwater target tracking impose severe constraints on the performance of the underlying methods, reflecting a fundamental trade-off between complexity and accuracy. The selection of an appropriate abstract control as part of the guidance system was critical in enabling deep RL to be deployed on multiple platforms with minimal effort. The navigational strategy we used here constitutes a set of general waypoints to be reached by the actions of the RL model, with no fine-tuning occurring during a particular field setup.

DISCUSSION

A platform-independent method was successfully trained to achieve optimal performance in tracking underwater moving targets and validated in at-sea experiments. The fieldwork conducted in Monterey Bay, using a Wave Glider as an agent and an underwater vehicle as a target, highlights the potential of RL as a path-planning algorithm by steering the autonomous vehicle toward a submerged target using acoustic communication links. These field trials represent more than 4 km of tracking over 15 hours of vehicle deployments, undertaking the important gap of demonstrating RL algorithms in the ocean environment. The policy learned in the simulated environment can be used with real vehicles, laying the groundwork for an approach to controlling more adaptive and cooperative marine robots for target tracking.

In GPS-denied environments such as underwater marine ecosystems (*37*), other approaches to localize and track underwater mobile targets are necessary. If we are able to acoustically track targets sufficiently well from the surface, we will be able to provide relatively accurate GPS-aided positioning using different triangulation methods (*34*), such as range-only target localization techniques (*41*). Unfortunately, AUV path optimization to increase system observability and therefore target estimation accuracy is not straightforward, which is especially true in scenarios where the dynamics of ocean currents or acoustic communication loss impose important operability constraints. Other works have focused on field applications where acoustic-instrumented marine organisms were detected by marine robots (*7*, *9*). Often, with these acoustic devices, the distance between the robot and the instrumented organism cannot be measured; only the presence of the transmitted signal is used to infer their position, which results in poor localization. Autonomous vehicles have typically been used to track targets that are capable of holding large payloads that have substantial energy consumption (*12*). To overcome this issue, researchers have studied passive acoustic monitoring techniques, which use the sound produced by underwater targets to detect and track their sources (*44*). Although it can be used to monitor targets that emit characteristic sound profiles, it is difficult to differentiate between different objects with similar sound profiles. Conversely, other studies have focused on detecting and tracking underwater features using optical-based devices such as video cameras or lasers (*45*). However, in such scenarios, the vehicle must be close enough to the target (within a few meters or even less) for identification, which is often very challenging.

Deep RL can be used as an optimization tool for underwater target tracking using range-only methods for autonomous vehicles, which can be extended to other applications that include the ecological movements of deep-water marine species (*8*), the coordination of a fleet of autonomous vehicles (*12*), the localization and recovery of deployed platforms (*3*), and autonomous docking (*46*). In simulations, the RL agent was able to learn the optimal policy by achieving similar performance of an analytical formulation (*41*). Moreover, the trained agent was tested in different simulated environmental conditions (for example, tracking a random walk target) to estimate the target's position. This indicates that the RL agent may have a relatively high zero-shot transfer performance, because the agent was evaluated in scenarios that were not present during training.

This deep RL approach opens the possibility of learning strategies to track underwater targets and find solutions for complex scenarios. In the study of the movement ecology of marine species,

adaptive AUVs and ASVs could be used in two ways: (i) as a tool to study the movement of electronically tagged individuals to inform management strategies of commercially exploited species in marine protected areas (*7*, *9*) and (ii) for tracking individuals for long periods of time to improve the understanding of their complex spatiotemporal behaviors (*18*). This capability is especially valuable for species that never reach the sea surface and a GPS fix on their location cannot be obtained. ASVs equipped with deep RL algorithms could instead patrol specific areas to provide temporally varying location information of tagged species carrying acoustic transponder tags for valuable and often challenging in situ behavioral studies.

Last, the tracking surface vehicle could also be used as a communication hub to enable human-in-the-loop operations and to relay information (for example, imagery) from the underwater asset during prescribed missions. This supervised autonomy approach could be used to extend the duration of missions, enable cooperation between shoreside experts, and allow for mission changes and corrections based on human input, thereby offering other ways to study the ocean (*12*, *47*, *48*). In such scenarios, seamless multivehicle coordination and collaboration with human operators become crucial. Although promising, additional research in multivehicle cooperation and collaboration, especially in the ocean environment, still needs to be conducted.

MATERIALS AND METHODS

System overview

Autonomous navigation systems are typically divided into three main layers, which are known as guidance, navigation, and control systems (*49*). The navigation and control systems strongly depend on a platform's configuration and the instruments or sensors implemented (*50*) and use low-level control techniques to control the vehicle (for example, to maintain a specific velocity or altitude from the seabed). In contrast, the guidance system is responsible for a higher-level of control by defining the path to follow or implementing obstacle avoidance methods. Here, the RL algorithm will be the core component of the guidance system as path planning for an adaptive ASV to track underwater moving targets. Specifically, the case of a single tracker (an ASV) and a single target (an AUV) is considered. 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 deep RL, and (ii) the target position estimation based on range data acquired online, where a LS approach was used for its simplicity and low computational runtime (*34*). In this work, the agent path planning problem was constrained using the typical scenario where the agent moves in a two-dimensional (2D) environment 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. Last, we also assume that the agent knows its position using its own navigation methods (for example, GPS).

Agent model

In the absence of ocean currents, the kinematics model of an autonomous vehicle is given by

$$\begin{cases} \dot{\mathbf{p}}(t) = \boldsymbol{v}(t) \\ \dot{\mathbf{v}}(t) = \mathbf{F}/m \end{cases} \tag{1}$$

where $t \in [0, t_f]$, $t_f > 0$, and $\mathbf{p} \in \mathbb{R}^2$ is the position vector of the agent in a 2D plane, $\mathbf{v} \in \mathbb{R}^2$ is the velocity vector, $\mathbf{F} \in \mathbb{R}^2$ is a force vector, and m is the mass of the agent. In this experiment, an agent with a constant velocity v and a single action space referring to the variation of the yaw angle ψ was considered. This is a common mode of operation when it is applied to torpedo-shaped AUVs (for example, MBARI's LRAUV) or vehicles that do not use thrusters (for example, the Wave Glider from Liquid Robotics). Consequently, using a state space formulation and defining the input action vector $u = F + \omega \in \mathbb{R}^1$ as the force applied to the ψ, with zero-mean additive Gaussian noise $\omega \sim \mathcal{N}(0, \sigma^2)$, the simplified dynamic discrete model at time-step t can be defined as

$$\begin{bmatrix} \mathbf{p}_{t+1} \\ \mathbf{v}_{t+1} \\ \psi_{t+1} \end{bmatrix} = \begin{bmatrix} \mathbf{p}_t \\ 0 \\ \psi_t \end{bmatrix} + \begin{bmatrix} v\mathbf{g}(\psi_{t+1})\Delta t \\ v\mathbf{g}(\psi_{t+1}) \\ u_t/m \end{bmatrix} \tag{2}$$

where $\mathbf{g}(\cdot) \triangleq [\cos(\cdot), \sin(\cdot)]$ and Δt is the sampling time interval. This equation will set the following waypoint to be reached by the agent, given a defined time step and the agent's velocity.

Target model

In this study, two scenarios were used: one with a fixed target position and one with a random walk with Lévy flight distribution (*43*). The simplified dynamic discrete model at time-step t can be obtained as follows:

$$\begin{bmatrix} \mathbf{q}_{t+1} \\ \mathbf{v}_t \end{bmatrix} = \begin{bmatrix} \mathbf{q}_t \\ 0 \end{bmatrix} + \begin{bmatrix} \mathbf{v}_t \\ \mathbf{u}_t/m \end{bmatrix} \Delta t \tag{3}$$

where the input vector $\boldsymbol{u}_t$ is equal to

$$\boldsymbol{u}_t = f_t \mathbf{g}(\chi_t) \tag{4}$$

where f_t is the force applied to the target at time step t, generated following the Lévy distribution, using the so-called Mantegna algorithm for a symmetric Lévy stable distribution and $\chi_t \in [0, 2\pi)$ is the angle of the applied force, which is sampled using a random uniform distribution.

Measurement model

The agent is equipped with a sensor that measures distances to the targets at specified, discrete-time intervals. Therefore, the range measurement is naturally modeled in a discrete-time setting as

$$\overline{d}_t = \|\boldsymbol{d}_t\| + \omega_t, \;\; t \in \{1, 2, \ldots, m\} \tag{5}$$

where $\boldsymbol{d}_t = \boldsymbol{p}_t - \boldsymbol{q}_t$ is the relative position vector of the target with respect to the agent, m indicates the number of measurements carried out, and $\omega_t \sim \mathcal{N}(\varepsilon, \sigma^2)$ is a nonzero 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 (*37*). Last, the projected planar range measurement $\overline{d}_p$ can be derived from target depth d_q as $\overline{d}_p = \sqrt{(\overline{d}_t^2 - d_q^2)}$.

Predicted target position model

Different methods can be used to obtain an estimation of the target's position $\hat{\boldsymbol{q}}$ using range-only and single-beacon techniques (*34*), for example, the simple unconstrained LS algorithm. The main idea behind LS algorithms lies in the linearization of the system using the squared range measurements to obtain a linear equation as a function of the unknown target's position.

Although this technique is suitable for static target localization, its capability to track a moving target can be compromised, and thus it is inferior to other algorithms (for example, PF). However, the run-time performance of LS is orders of magnitude faster than its competitors, which is key in RL techniques to accelerate the training phase. Consequently, we used LS during the training phase and both LS and PF during the test phase.

Observation and action space

The observations at each time-step t that we can obtain from the environment include the position $\boldsymbol{p}$ and velocity $\boldsymbol{v}$ vectors of the agent, the relative position vector of the estimated target position ($\hat{\boldsymbol{d}}_t = \boldsymbol{p}_t - \hat{\boldsymbol{q}}_t$), and the projected distance measured by the sensor $\overline{d}_{pt}$. In addition, target depth d_p and agent origin point $\boldsymbol{p}_o$, which is the agent's last position where a range measurement could be made, were included. This last vector is to help the agent when several consecutive range measurements fail:

$$\boldsymbol{o}_t = [\boldsymbol{p}_t, \boldsymbol{v}_t, \hat{\boldsymbol{d}}_t, \overline{d}_{pt}, d_p, \boldsymbol{p}_o] \tag{6}$$

The action space is determined by the force applied to the yaw ψ angle of the agent, because $a_t \triangleq u_\psi$.

Reward function

In RL, the agent obtains rewards as a function of the state s and the agent's actions a. 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.

The design of a good reward function is a key aspect in RL. In dense reward settings, the agent receives diverse rewards in most states (for example, a reward proportional to distance to the goal), which allows the agent to quickly differentiate good states from bad ones. However, such an approach can easily exploit poorly designed rewards, get stuck in local optima, and induce behavior that the designer did not intend. In contrast, goal-based sparse rewards are appealing because they do not suffer from the reward exploration problem (*24*). In addition, this small, simple set of rules has similarities with biological behaviors and is therefore applicable to animals with a very limited level of information processing (*51*).

A combination of both reward methods was used in this study: (i) a nonsparse reward to guide the agent toward the goal when its performance was poor and (ii) a sparse reward when the agent's performance reached a predefined threshold. In addition, two different goals were defined to optimize the agent's trajectory, which influence the reward obtained by the agent: (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.

The reward as a function of the distance between the agent and the target is defined as

$$r_d = \begin{cases} \lambda(0.5 - \hat{d}) & \text{if } \hat{d} > d_{\text{th}} \\ 1 & \text{else} \end{cases} \tag{7}$$

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. The smaller the distance $\hat{d}$ is, the closer the agent is to the estimated target, and therefore, this reward is the most important reward to guide the agent to navigate toward the target.

The reward as a function of the predicted target error is defined as

$$r_e = \begin{cases} \lambda(0.5 - e_q) & \text{if } e_q > e_{\text{th}} \\ 1 & \text{else} \end{cases} \tag{8}$$

where $e_q = \|\hat{\boldsymbol{q}}_t - \boldsymbol{q}_t\|$ 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. This is key to optimizing the agent's trajectory toward the goal of finding the optimal path that leads to the greatest accuracy in the estimated target position.

Last, a terminal reward related to the success of the mission is defined as

$$r_{\text{terminal}} = \begin{cases} -100 & \text{if } \hat{d} > d_{\max} \\ -1 & \text{if } \hat{d} < d_{\min} \\ 0 & \text{else} \end{cases} \tag{9}$$

where $d_{\max}$ is the maximum distance that the agent can go with respect to the target, and $d_{\min}$ is a threshold set to avoid collisions between the target and the agent. Consequently, this sparse reward gives a higher penalty if the distance between the target and the agent is greater than a maximum or less than a minimum threshold. Then, the final reward is given by the equation $r = r_d + r_e + r_{\text{terminal}}$.

State space

In the proposed approach, a mini-batch of N experiences $\{(\boldsymbol{o}_t, a_t, r_t, \boldsymbol{o}_{t+1}, d_t)_i\}_{i=1}^{N}$ is sampled from the replay buffer $\mathbf{D}$ of experiences at each iteration. In the LSTM-SAC and LSTM-DDPG algorithms, the past history $\boldsymbol{h}_t^l$ is also taken into consideration, where l is the number of past observations used. In this case, the mini-batch of experiences is $\{(\boldsymbol{h}_t^l, \boldsymbol{o}_t, a_t, r_t, \boldsymbol{o}_{t+1}, d_t)_i\}_{i=1}^{N}$.

Algorithms

As mentioned in the previous section, two different actor-critic algorithms have been implemented: a DDPG, which is an actor-critic model-free (enables the reuse of previously collected data for efficiency) deep Q-learning algorithm (*39*), and a SAC, where the actor aims to maximize the expected reward while also maximizing entropy (*23*). The actor-critic architecture incorporates separate policy and value function networks. The critic network estimates the value function on the basis of the optimal action-value Q-function, and the actor network updates the policy distribution in the direction suggested by the critic network. Therefore, they interleave learning an approximator to the optimal Q-function $Q^*(s, a)$ with learning an approximator to the optimal action $a^*(s)$, which can be calculated for any given state by the equation: $a^*(s) = \arg\max Q^*(s, a)$.

In addition, in these algorithms, an LSTM network was added at the beginning of the structure (LSTM-DDPG and LSTM-SAC), with the ability to enable and disable it. In addition, a single-cell

LSTM architecture was also designed and added in one of the hidden layers of the SAC algorithm (H-LSTM-SAC).

Supplementary Materials

This PDF file includes:
Methods
Figs. S1 to S11
Table S1

Other Supplementary Material for this manuscript includes the following:
Movies S1 to S5

REFERENCES AND NOTES

1. R. Guenard, *The State of World Fisheries and Aquaculture 2020* (FAO, 2020); www.fao.org/documents/card/en/c/ca9229en, vol. 32.
2. K. L. Smith, H. A. Ruhl, C. L. Huffard, M. Messié, M. Kahru, Episodic organic carbon fluxes from surface ocean to abyssal depths during long-term monitoring in NE Pacific. *Proc. Natl. Acad. Sci. U.S.A.* **115**, 12235–12240 (2018).
3. K. L. Smith, A. D. Sherman, P. R. McGill, R. G. Henthorn, J. Ferreira, T. P. Connolly, C. L. Huffard, Abyssal Benthic Rover, an autonomous vehicle for long-term monitoring of deep-ocean processes. *Sci. Robot.* **6**, eabl4925 (2021).
4. L. W. Botsford, J. W. White, M. H. Carr, J. E. Caselle, Marine protected area networks in California, USA, in *Marine Managed Areas and Fisheries*, M. L. Johnson, J. Sandell, Eds., vol. 69 of *Advances in Marine Biology* (Academic Press, 2014); https://linkinghub.elsevier.com/retrieve/pii/B9780128002148000062, pp. 205–251.
5. M. Gleason, S. McCreary, M. Miller-Henson, J. Ugoretz, E. Fox, M. Merrifield, W. McClintock, P. Serpa, K. Hoffman, Science-based and stakeholder-driven marine protected area network planning: A successful case study from north central California. *Ocean Coast. Manag.* **53**, 52–68 (2010).
6. N. Perera, A. De Vos, Marine protected areas in Sri Lanka: A review. *Environ. Manag.* **40**, 727–738 (2007).
7. I. Masmitja, J. Navarro, S. Gomariz, J. Aguzzi, B. Kieft, T. O'Reilly, K. Katija, P.-J. Bouvet, C. Fannjiang, M. Vigo, P. Puig, A. Alcocer, G. Vallicrosa, N. Palomeras, M. Carreras, J. del Rio, J. B. Company, Mobile robotic platforms for the acoustic tracking of deep-sea demersal fishery resources. *Sci. Robot.* **5**, eabc3701 (2020).
8. M. Vigo, J. Navarro, I. Masmitja, J. Aguzzi, J. García, G. Rotllant, N. Bahamón, J. Company, Spatial ecology of Norway lobster *Nephrops norvegicus* in Mediterranean deep-water environments: Implications for designing no-take marine reserves. *Mar. Ecol. Prog. Ser.* **674**, 173–188 (2021).
9. D. Cote, J. M. Nicolas, F. Whoriskey, A. M. Cook, J. Broome, P. M. Regular, D. Baker, Characterizing snow crab (*Chionoecetes opilio*) movements in the sydney bight (Nova Scotia, Canada): A collaborative approach using multiscale acoustic telemetry. *Can. J. Fish. Aquat. Sci.* **76**, 334–346 (2019).
10. E. R. Abraham, The generation of plankton patchiness by turbulent stirring. *Nature* **391**, 577–580 (1998).
11. A. Pascual, D. L. Rudnick, S. Ruiz, J. Tintoré, E. D'Asaro, Coherent pathways for vertical transport from the surface ocean to interior. *Bull. Am. Meteorol. Soc.* **101**, E1996–E2004 (2020).
12. Y. Zhang, J. P. Ryan, B. W. Hobson, B. Kieft, A. Romano, B. Barone, C. M. Preston, B. Roman, B.-Y. Raanan, D. Pargett, M. Dugenne, A. E. White, F. H. Freitas, S. Poulos, S. T. Wilson, E. F. DeLong, D. M. Karl, J. M. Birch, J. G. Bellingham, C. A. Scholin, A system of coordinated autonomous robots for Lagrangian studies of microbes in the oceanic deep chlorophyll maximum. *Sci. Robot.* **6**, eabb9138 (2021).
13. J. Kalwa, D. Tietjen, M. Carreiro-Silva, J. Fontes, L. Brignone, N. Gracias, P. Ridao, M. Pfingsthorn, A. Birk, T. Glotzbach, S. Eckstein, M. Caccia, J. Alves, T. Furfaro, J. Ribeiro, A. Pascoal, The European Project MORPH: Distributed UUV systems for multimodal, 3D underwater surveys. *Mar. Technol. Soc. J.* **50**, 26–41 (2016).
14. R. Gwiazda, C. K. Paull, B. Kieft, D. Klimov, R. Herlien, E. Lundsten, M. McCann, M. J. Cartigny, A. Hamilton, J. Xu, K. L. Maier, D. R. Parsons, P. J. Talling, Near-bed structure of sediment gravity flows measured by motion-sensing "boulder-like" Benthic Event Detectors (BEDs) in Monterey Canyon. *Case Rep. Med.* **127**, e2021JF00643 (2022).
15. J. Heidemann, M. Stojanovic, M. Zorzi, Underwater sensor networks: Applications, advances and challenges. *Philos. Trans. A Math. Phys. Eng. Sci.* **370**, 158–175 (2012).
16. E. Zereik, M. Bibuli, N. Mišković, P. Ridao, A. Pascoal, Challenges and future trends in marine robotics. *Annu. Rev. Control.* **46**, 350–368 (2018).
17. S. Ravindran, Underwater robot can follow marine organisms over record distances. *Nature* 10.1038/news.2010.573 (2010).
18. D. R. Yoerger, A. F. Govindarajan, J. C. Howland, J. K. Llopiz, P. H. Wiebe, M. Curran, J. Fujii, D. Gomez-Ibanez, K. Katija, B. H. Robison, B. W. Hobson, M. Risi, S. M. Rock, A hybrid underwater robot for multidisciplinary investigation of the ocean twilight zone. *Sci. Robot.* **6**, eabe1901 (2021).
19. A. Saad, A. Stahl, A. Våge, E. Davies, T. Nordam, N. Aberle, M. Ludvigsen, G. Johnsen, J. Sousa, K. Rajan, Advancing ocean observation with an AI-driven mobile robotic explorer. *Oceanography* **33**, 50–59 (2020).
20. R. S. Sutton, A. G. Barto, *Reinforcement Learning: An Introduction* (MIT Press, ed. 2, 2015).
21. Z. Q. Zhao, P. Zheng, S. T. Xu, X. Wu, Object detection with deep learning: A review. *IEEE Trans. Neural Netw. Learn. Syst.* **30**, 3212–3232 (2019).
22. R. K. Behera, A. Gunasekaran, S. Gupta, S. Kamboj, P. K. Bala, Personalized digital marketing recommender engine. *J. Retail. Consum. Serv.* **53**, 101799 (2020).
23. T. Haarnoja, A. Zhou, P. Abbeel, S. Levine, Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, in *Proceedings of the 35th International Conference on (ICML 2018)* (ICML, 2018), vol. 5, pp. 2976–2989.
24. F. Memarian, W. Goo, R. Lioutikov, S. Niekum, U. Topcu, Self-Supervised Online Reward Shaping in Sparse-Reward Environments, in *Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robot and Systems (IROS)*, Prague, Czech Republic, 2021 September 27–October 1, pp. 2369–2375.
25. D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, D. Hassabis, Mastering the game of Go with deep neural networks and tree search. *Nature* **529**, 484–489 (2016).
26. G. Reddy, J. Wong-Ng, A. Celani, T. J. Sejnowski, M. Vergassola, Glider soaring via reinforcement learning in the field. *Nature* **562**, 236–239 (2018).
27. M. G. Bellemare, S. Candido, P. S. Castro, J. Gong, M. C. Machado, S. Moitra, S. S. Ponda, Z. Wang, Autonomous navigation of stratospheric balloons using reinforcement learning. *Nature* **588**, 77–82 (2020).
28. P. Gunnarson, I. Mandralis, G. Novati, P. Koumoutsakos, J. O. Dabiri, Learning efficient navigation in vortical flow fields. *Nat. Commun.* **12**, 7143 (2021).
29. B. Li, Y. Wu, Path planning for UAV ground target tracking via deep reinforcement learning. *IEEE Access* **8**, 29064–29074 (2020).
30. X. Cao, C. Sun, M. Yan, Target search control of AUV in underwater environment with deep reinforcement learning. *IEEE Access* **7**, 96549–96559 (2019).
31. I. Carlucho, M. De Paula, S. Wang, Y. Petillot, G. G. Acosta, Adaptive low-level control of autonomous underwater vehicles using deep reinforcement learning. *Rob. Auton. Syst.* **107**, 71–86 (2018).
32. E. Anderlini, G. G. Parker, G. Thomas, Docking control of an autonomous underwater vehicle using reinforcement learning. *Appl. Sci.* **9**, 3456 (2019).
33. H. Wu, S. Song, K. You, C. Wu, Depth control of model-free AUVs via reinforcement learning. *IEEE Trans. Syst. Man Cybern. Syst.* **49**, 2499–2510 (2019).
34. I. Masmitja, S. Gomariz, J. Del-Rio, B. Kieft, T. O'Reilly, P. J. Bouvet, J. Aguzzi, Range-only single-beacon tracking of underwater targets from an autonomous vehicle: From theory to practice. *IEEE Access* **7**, 86946–86963 (2019).
35. I. Ullah, J. Chen, X. Su, C. Esposito, C. Choi, Localization and detection of targets in underwater wireless sensor using distance and angle based algorithms. *IEEE Access* **7**, 45693–45704 (2019).
36. J. K. Terry, B. Black, N. Grammel, M. Jayakumar, A. Hari, R. Sullivan, L. Santos, R. Perez, C. Horsch, C. Dieffendahl, N. L. Williams, Y. Lokesh, P. Ravi, PettingZoo: A standard API for multi-agent reinforcement learning. *Adv. Neural Inf. Process. Syst.* **18**, 15032–15043 (2021).
37. M. Stojanovic, J. Preisig, Underwater acoustic communication channels: Propagation models and statistical characterization. *IEEE Commun. Mag.* **47**, 84–89 (2009).
38. V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, D. Hassabis, Human-level control through deep reinforcement learning. *Nature* **518**, 529–533 (2015).
39. T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, Continuous control with deep reinforcement learning, in *4th International Conference on Learning Representations (ICLR 2016)–Conference Track Proceedings* (ICLR, 2016).
40. L. Meng, R. Gorbet, D. Kulic, Memory-based Deep Reinforcement Learning for POMDPs, in *Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Prague, Czech Republic, 2021 September 27–October 1, pp. 5619–5626.
41. I. Masmitja, S. Gomariz, J. Del-Rio, B. Kieft, T. O'Reilly, P. J. Bouvet, J. Aguzzi, Optimal path shape for range-only underwater target localization using a Wave Glider. *Int. J. Rob. Res.* **37**, 1447–1462 (2018).

42. R. Agarwal, M. Schwarzer, P. S. Castro, A. Courville, M. G. Bellemare, Deep reinforcement learning at the edge of the statistical precipice. *Adv. Neural Inf. Process. Syst.* **35**, 29304–29320 (2021).
43. X.-S. Yang, in *Nature-Inspired Optimization Algorithms* (Elsevier, 2014), pp. 45–65.
44. M. Poupard, M. Ferrari, P. Best, H. Glotin, Passive acoustic monitoring of sperm whales and anthropogenic noise using stereophonic recordings in the Mediterranean Sea, North West Pelagos Sanctuary. *Sci. Rep.* **12**, 2007 (2022).
45. K. Katija, P. L. D. Roberts, J. Daniels, A. Lapides, K. Barnard, M. Risi, B. Y. Ranaan, B. G. Woodward, J. Takahashi, in *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)* (IEEE, 2021), pp. 859–868.
46. P. W. Kimball, E. B. Clark, M. Scully, K. Richmond, C. Flesher, L. E. Lindzey, J. Harman, K. Huffstutler, J. Lawrence, S. Lelievre, J. Moor, B. Pease, V. Siegel, L. Winslow, D. D. Blankenship, P. Doran, S. Kim, B. E. Schmidt, W. C. Stone, The ARTEMIS under-ice AUV docking system. *J. Field Robot.* **35**, 299–308 (2018).
47. A. S. Ferreira, M. Costa, F. Py, J. Pinto, M. A. Silva, A. Nimmo-Smith, T. A. Johansen, J. B. de Sousa, K. Rajan, Advancing multi-vehicle deployments in oceanographic field experiments. *Auton. Robots.* **43**, 1555–1574 (2019).
48. A. Jarrot, A. Gelman, G. Choi, A. Speck, G. Strunk, A. Croux, T. P. Osedach, S. Vannuffelen, S. Ossia, J. Vincent, S. Grall, G. Eudeline, High-speed underwater acoustic communication for multi-agent supervised autonomy, in *Proceedings of the 2021 Fifth Underwater Communications and Networking Conference (UComms)*, Lerici, Italy, 2021 August 31–September 2, pp. 5–8.
49. T. I. Fossen, *Marine Control Systems Guidance, Navigation, and Control of Ships, Rigs and Underwater Vehicles* (Marine Cybernetics, Trondheim, Norway, 2002).
50. I. Masmitja, J. Gonzalez, C. Galarza, S. Gomariz, J. Aguzzi, J. del Rio, New vectorial propulsion system and trajectory control designs for improved AUV mission autonomy. *Sensors* **18**, 1241 (2018).
51. E. E. Nuzhin, M. E. Panov, N. V. Brilliantov, Why animals swirl and how they group. *Sci. Rep.* **11**, 20843 (2021).
52. J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, M. Hutter, Learning quadrupedal locomotion over challenging terrain. *Sci. Robot.* **5**, eabc5986 (2020).
53. C. Zhang, P. Cheng, B. Du, B. Dong, W. Zhang, AUV path tracking with real-time obstacle avoidance via reinforcement learning under adaptive constraints. *Ocean Eng.* **256**, 111453 (2022).
54. Y. Mao, F. Gao, Q. Zhang, Z. Yang, An AUV target-tracking method combining imitation learning and deep reinforcement learning. *J. Mar. Sci. Eng.* **10**, 383 (2022).
55. A. B. Martinsen, A. M. Lekkas, S. Gros, Reinforcement learning-based NMPC for tracking control of ASVs: Theory and experiments. *Control Eng. Pract.* **120**, 105024 (2022).
56. B. Du, B. Lin, C. Zhang, B. Dong, W. Zhang, Safe deep reinforcement learning-based adaptive control for USV interception mission. *Ocean Eng.* **246**, 110477 (2022).
57. N. Wang, Y. Wang, Y. Zhao, Y. Wang, Z. Li, Sim-to-Real: Mapless navigation for USVs using deep reinforcement learning. *J. Mar. Sci. Eng.* **10**, 895 (2022).
58. S. S. Øvereng, D. T. Nguyen, G. Hamre, Dynamic positioning using deep reinforcement learning. *Ocean Eng.* **235**, 109433 (2021).
59. S. T. Havenstrm, A. Rasheed, O. San, Deep reinforcement learning controller for 3D path-following and collision avoidance by autonomous underwater vehicles. arXiv:2006.09792 (2020).
60. J. Woo, N. Kim, Collision avoidance for an unmanned surface vehicle using deep reinforcement learning. *Ocean Eng.* **199**, 107001 (2020).
61. Y. Cui, S. Osaki, T. Matsubara, Autonomous boat driving system using sample-efficient model predictive control-based reinforcement learning approach. *J. Field Robot.* **38**, 331–354 (2021).
62. Q. Zhang, J. Lin, Q. Sha, B. He, G. Li, Deep interactive reinforcement learning for path following of autonomous underwater vehicle. *IEEE Access* **8**, 24258–24268 (2020).
63. J. Woo, C. Yu, N. Kim, Deep reinforcement learning-based controller for path following of an unmanned surface vehicle. *Ocean Eng.* **183**, 155–166 (2019).
64. B. Yoo, J. Kim, Path optimization for marine vehicles in ocean currents using reinforcement learning. *J. Mar. Sci. Technol.* **21**, 334–343 (2016).
65. M. Carreras, J. Yuh, J. Batlle, P. Ridao, A behavior-based scheme using reinforcement learning for autonomous underwater vehicles. *IEEE J. Ocean. Eng.* **30**, 416–427 (2005).

Acknowledgments: We thank C. Wahl and J. Daniels (MBARI) for the assistance with the sea trials, P. Roberts and E. Orenstein (MBARI) for developing the software and configuring the mission of the underwater vehicle, and S. Gomariz (UPC) for ongoing technical advice on vehicle control. This work acknowledges the "Severo Ochoa Centre of Excellence" accreditation (CEX2019-000928-S). The work's experiments were run at the Barcelona Supercomputing Center in collaboration with the HPAI group. We gratefully acknowledge the David and Lucile Packard Foundation. **Funding:** This work was supported by the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie, grant agreement no. 893089 (I.M.); the Spanish Ministerio de Economia y Competitividad under the project SASES, grant agreement no. RTI2018-095112-B-I00 (I.M.); the Spanish Ministerio de Economia y Competitividad under the project BITER-ECO, grant agreement no. PID2020-114732RB-C31 (I.M. and J.N.); and the Spanish Ministerio de Economia y Competitividad under the project BITER-AUV, grant agreement no. PID2020-114732RB-C33 (N.P.). **Author contributions:** Conceptualization: I.M., M.M., K.K., N.P., and J.N. Methodology: I.M., M.M., K.K., N.P., T.O., and B.K. Investigation: I.M. and M.M. Visualization: I.M., M.M., N.P., and J.N. Funding acquisition: I.M., K.K., and J.N. Project administration: I.M., K.K., and J.N. Supervision: M.M., K.K., and J.N.. Writing–original draft: I.M. and K.K. Writing–review and editing: I.M., M.M., K.K., N.P., J.N., T.O., and B.K. **Competing interests:** The authors declare that they have no competing interests. **Data and materials availability:** Deep RL algorithms used in this project and the developed environment are available in the open-source repository: https://doi.org/10.5281/zenodo.8063918. The ROS implementation of the proposed RL algorithm is available in the open-source repository: https://doi.org/10.5281/zenodo.8063968.

Submitted 16 September 2022
Accepted 26 June 2023
Published 26 July 2023
10.1126/scirobotics.ade7811

Science Robotics

Dynamic robotic tracking of underwater targets using reinforcement learning

I. Masmitja, M. Martin, T. O'Reilly, B. Kieft, N. Palomeras, J. Navarro, and K. Katija

Sci. Robot. **8** (80), eade7811. DOI: 10.1126/scirobotics.ade7811

Science Robotics (ISSN 2470-9476) is published by the American Association for the Advancement of Science. 1200 New York Avenue NW, Washington, DC 20005. The title *Science Robotics* is a registered trademark of AAAS.