Expert Systems With Applications 270 (2025) 126483

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Variational model-based Deep Reinforcement Learning for Non-Homogeneous Patrolling aquatic environments with multiple unmanned surface vehicles

Samuel Yanes Luis [a,*], Nicola Basilico [b], Michele Antonazzi [b], Daniel Gutiérrez-Reina [a], Sergio Toral Marín [a]

[a] Department of Electronics Engineering, University of Sevilla, Camino Avenida de Los Descubrimientos s/n, Sevilla, 41005, Spain
[b] Department of Computer Science, University of Milan, Via Celoria 18, Milano, 20133, Italy

ARTICLE INFO

Keywords:
Deep Reinforcement Learning
Environmental patrolling
Multi-agent path planning
Model-based decision making

ABSTRACT

This paper addresses the challenge of Non-Homogeneous Patrolling for Autonomous Surface Vehicles in non-homogeneous importance water environments with a dissimilar biological monitorization criterion. Traditional monitoring methods fail, especially in expansive areas such as Lake Ypacaraíin Paraguay. The proposed solution employs a cooperative Deep Reinforcement Learning framework, specifically a multi-agent version of the Double Deep Q-Learning algorithm based on safe-consensus decision making. This framework optimizes adaptive policies for such vehicles by simultaneously modeling the environment and patrolling high-importance zones. The incorporation of a Variational Auto-Encoder based on the U-Network architecture directly addresses the non-observability of the environment by predicting biological importance from partial observations. The methodology is validated in a realistic algae bloom contamination scenario, demonstrating superior performance and computational efficiency compared to traditional approaches like Gaussian Processes and K-Nearest-Neighbors. The Deep Reinforcement Learning framework, coupled with the Variational Auto-Encoder model, showcases flexibility and efficiency in addressing multi-agent cooperation and long-term objective optimization for water quality monitoring. The results reveal significant improvements, with the proposed model exceeding well-founded approaches with a 30% faster minimization of the patrolling score compared to these methods.

1. Introduction

Lakes, rivers, and shores play a vital role in the ecosystems of the Earth and are critical components of our planet's water cycle. The biological state of such resources is constantly changing due to various natural and human-caused factors: rainfall patterns, temperature fluctuations, and human activities such as agriculture and urbanization (Baron, Poff, Angermeier, et al., 2002). Monitoring water for pollutants and disease-causing organisms is essential to understand and manage the impact on health. However, the size of these resources makes it a challenging task. This has been a usual problem in places such as Mar Menor (Spain) or Lake Ypacaraí(Paraguay). Traditional manual monitoring methods are limited in their ability to cover large areas and collect comprehensive data on changing conditions of water resources (Arzamendia, Gutierrez et al., 2019). The use of Autonomous Surface Vehicles (ASVs) for monitoring offers a unique opportunity to model water resources in real time with high resolution (Sánchez-García et al., 2018). With the use of advanced water quality sensors and an appropriate coordinated monitoring policy, ASVs can continuously collect data on various physical and chemical properties of water, providing a more comprehensive understanding of the risks these environments suffer.

However, effective monitoring of water resources using ASVs requires the optimization of these adaptive policies that take into account the unique challenges of real biophysical environments. These policies must decide sequentially where to take samples of the water quality parameters (WQP), at the same time considering different objectives: (i) obstacle avoidance, (ii) efficient coordination between vehicles, and (iii) integration of data gathered by multiple vehicles into a comprehensive picture of the water resource conditions. The use of multiple agents also has the problem of scalability, since its complexity increases with the number of independent mobile measurement stations, represented by the ASVs. By far, the solution to cooperative information gathering with multiple agents remains an open challenge that must be addressed

* Corresponding author.
E-mail address: syanes@us.es (S. Yanes Luis).

https://doi.org/10.1016/j.eswa.2025.126483
Received 29 March 2024; Received in revised form 24 October 2024; Accepted 7 January 2025
Available online 13 January 2025
0957-4174/© 2025 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

from multiple perspectives.

In the particular case of water quality monitoring, this task required to be continuous. Persistently monitoring an environment involves revisiting zones in which the information could become obsolete or there is a higher risk of biological danger. This problem has been commonly formulated as the Patrolling problem (Chevaleyre, 2004; Yanes, Reina, & Marín, 2020), which is a particular case of Path Planning problems. The Patrolling problem consists of continuously sampling the zones of higher interest with a cyclic temporal criterion (Chevaleyre, 2004). In general, those zones that have been unvisited the longest are the most important zones to cover.

Among environmental applications, such as the problem of monitoring water quality, seen in Yanes et al. (2020) or in the surveillance of wildfires (Julian & Kochenderfer, 2018), it is convenient to define a dissimilar importance between zones for monitoring idleness. In these scenarios, there are zones of higher biological importance that need to be covered more persistently, such as zones of high-risk biological activity in algae blooms or clusters of high turbidity in waters. This problem can be formulated as the Non-Homogeneous Patrolling Problem (NHPP). NHPP is defined as the sequential optimization problem of finding the best route $\psi := [p_0, \dots, p_T]$ that covers the maximum information possible ω at the same time it maximizes the coverage of zones that have remained unvisited for longer w.

An important aspect addressed in this article is the imposition that the importance criterion ω is not known a priori and is discovered as the fleet explores the navigation space. A poor model of importance distribution implies inefficient patrolling as the important zones remain unknown and therefore unvisited for longer. This aspect configures the problem into a Partially Observable Markov Decision Process (POMDP), where the fleet of agents has limited vision of the surroundings. This indicates that agents face the dual challenge of continuously modeling a changing environment while patrolling critical zones. In the context of a non-observable NHPP, when information importance is unknown in advance, it is proven that optimal solutions via off-line path planners are infeasible. Therefore, two modules are required: one to address the environment's non-stationarity and another to develop an adaptive policy that can respond to these changes.

Patrolling also involves cooperation between agents to avoid measurement redundancies and to share the navigation space. This paper proposes to use Deep Reinforcement Learning (DRL) as a methodology to obtain dynamic policies for the ASVs to infer the importance model ω and to patrol efficiently. Previous approaches that used DRL (Yanes et al., 2020; Yanes, Reina, Marín & Toral, 2021; Yanes Luis, Gutiérrez-Reina, & Toral Marín, 2023) for WQP patrolling have shown promising results in generating optimized tailored patrolling policies, allowing ASVs to make decisions on the fly based on real-time interactions with the environment. Nevertheless, there is a clear uncovered aspect of these methods when modeling and patrolling must be performed at the same time, moreover, when it involves multiple vehicles.

In this sense, this paper proposes a multi-agent version of the Double Deep Q-Learning algorithm (DDQL) from the DRL methodology (Mnih, Kavukcuoglu, Silver, et al., 2015), to deal with different water quality monitoring tasks with this particular partial observability condition. In DDQL, a deep neural network $Q(s, a)$ is defined to represent the future discounted reward given an observation s and a possible action $a \in A$ for the agent to choose. This network will be trained in a model-free manner, using only simulated interactions of the agents with the environment. The original approach from Yanes, Gutiérrez-Reina, and Toral Marín (2021) is modified to train multiple agents using one single Q function. For multiple cooperative agents, a unique function Q is defined. This function receives an egocentric observation o^j, where every agent j differentiates itself from other agents visually. With regard to the training scenario, a realistic water contamination simulator is proposed, based on the dynamics of the sparse algae blooms (Qiu, Ren, Li, Tao, & Zhou, 2021). This scenario generator introduces stochasticity to obtain infinite possible contamination cases subjected to the natural limits of the algae phenomenon.

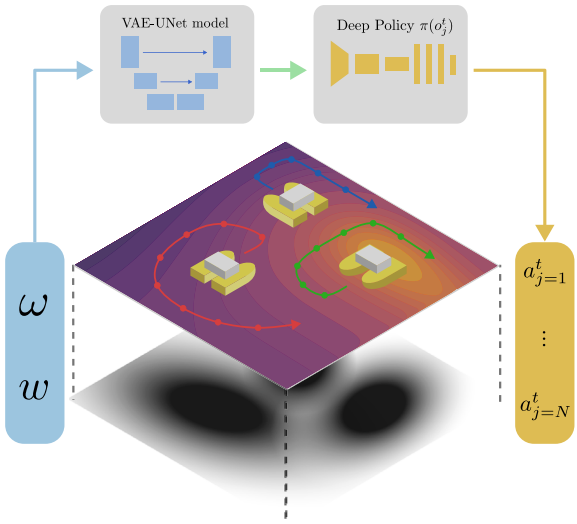

Fig. 1. Diagram of the proposed framework.

Deep Q-Learning and other reinforcement learning methods base their estimation on observations of the environment. As mentioned above, without proper observation, it is harder to infer the optimal actions for agents to move (Yanes et al., 2020). To address importance modeling, an enhanced model is proposed online based on a Deep Variational Auto-encoder, to infer the biologic importance map ω given partial observations like in an Inverse Problem Estimation (Yi, Guo, Fan, Hamann, & Wang, 2020). The architecture of this module is made up of a variant encoder–decoder based on the popular UNet network (Ronneberger, Fischer, & Brox, 2015). Our architecture implements a variational version of this UNet similar to Wang et al. (2020) with the novelty of a Gaussian operation in the channel for stochasticity generation. The input of the model is visually constructed by all the measurements taken, which results in a constant size input, constant size inference time, much more efficient than other classical approaches like the Gaussian process (GPs). See Fig. 1 for a complete overview of the proposed framework.

In summary, this paper proposes the following:

- A realistic scenario of algae bloom contamination for water resources based on a stochastic physical model.
- A Deep Reinforcement Learning framework to obtain Deep patrolling policies by means of an appropriate NHPP formulation, reward, and state.
- A Variational Auto-Encoder model to enhance the observability of the agents and to predict the model of the environment with partial observations.

This paper is organized as follows. In Section 2 previous approaches are discussed. In Section 3, the NHPP is presented and the movement and the final optimization objective is explained. In Section 4, the DRL methodology is presented within the network architectures. The importance model based on the VAE-UNet is also presented in Section 4. In Section 5, we present and discuss the simulations and the results. Finally, in Section 6, the conclusions and future work are explained.

2. Related works

The use of autonomous surface vehicles has gained relevance in recent years due to advances in battery autonomy and, above all, to the capacity for remote computing and sensing (Sánchez-García et al., 2018). These vehicles are particularly convenient because they can be used to obtain a status of water resource quality combined with good

GNSS localization capability, maneuverability, and autonomy. Those vehicles dedicated to biological conservation are usually equipped with modules such as water quality sensors, bathymetry, and spectral cameras, for environmental data acquisition. These sensors will define the observation capability in monitoring tasks and observability within the patrol optimization problem.

Multiple previous works have addressed the issue of acquiring environmental water quality information. Depending on the final objective and conditions of the patrol, we can highlight different types of path planning: (I) When the objective is to find an accurate model of one or several parameters of water quality, we usually refer to Informative Path Planning (IPP) (Popovic, Vidal-Calleja, Hitz, et al., 2020). (II) When planning requires continuous monitoring due to the need for temporal monitoring, we refer to Informative Patrolling (Yanes et al., 2020). In Peralta, Reina, Marín, Gregor, and Arzamendia (2021) the use of Bayesian optimization and Gaussian processes (GP) is proposed for optimal sampling with different agents. The search space for each vehicle is equally distributed using a Voronoi tessellation. For the policy, the Expected Improvement acquisition function with regularization based on the sample distance is used. This limits the vehicles in the travel distance before sampling. With this formulation, the aim is to obtain the highest precision with a minimum number of samples. However, no cyclic criterion could be easily incorporated into this framework, which is absolutely necessary for the patrolling task.

It can be seen how the Particle Swarm Optimization (PSO) method is used to find the maximum contamination levels (Jara Ten Kathen, Peralta, Johnson, Jurado Flores, & Gutiérrez Reina, 2024). In this work, each vehicle behaves as a particle in a continuous search space. In Kathen, Flores, and Reina (2021), a modification of the classic PSO algorithm is proposed that encourages exploration using a predictive uncertainty of GP as a surrogate model. The main problem with applying these methods for patrolling is that they are not intended for continuous monitoring. The optimization of Gaussian processes has $\mathcal{O}(N^3)$ complexity with the number of samples, which is not ideal for large amounts of information from patrolling paths. Moreover, GPs involve imposing how information correlates by choosing a kernel and hyperparameter boundaries (Kathen, Peralta, Johnson, Jurado Flores, & Gutiérrez Reina, 2023). In our proposal, the use of Deep policies trained by a DRL kernel allows the behavior to be optimized without any previous heuristic in a model-free scheme. In addition, the VAE mode will exhibit a much more efficient prediction with less inference time, as it incorporates qualified information from the environment during offline training.

Regarding the use of DRL with patrolling missions, some previous works have explored this possibility (Piciarelli & Foresti, 2019; Yanes et al., 2020; Yanes, Reina et al., 2021; Yanes Luis et al., 2023). In Piciarelli and Foresti (2019), the use of flying autonomous vehicles is proposed for the first time for the resolution of nonhomogeneous patrolling. This work posits the use of spectral cameras as a form of information acquisition with a single agent. However, the adopted scenario does not appear to be based on any real phenomena, is tailored for only one agent, and the navigation space is assumed to be convex. This condition allows traveling from one point to another in space without obstacles, and all actions are valid. In this paper, a mechanism is implemented to avoid risky or directly forbidden actions, as was done in Yanes Luis et al. (2023). This mechanism has been shown to allow for more homogeneous and direct learning, since the avoidance of known obstacles is no longer part of learning.

The use of DDQL for NHPP patrolling is also discussed in Yanes, Gutiérrez-Reina et al. (2021). The conclusions of this work allow us to determine that DDQL is capable of optimizing large state–action policies for this task. In addition, a detailed study of the sensitivity of DDQL hyperparameters is carried out in Yanes et al. (2020). Finally, in Yanes, Reina et al. (2021), a methodology for the multi-agent case is proposed. Nevertheless, the agents have all the information of the problem a priori, in a fully-observable NHPP, which could be seen as unrealistic. There, it is proposed to use a neural network whose last layer has an output for each agent. In contrast, a shared weight policy is proposed as in Yanes, Reina et al. (2021), however, with this architecture, it is not necessary to modify its size depending on the agents. Instead, it incorporates an *egocentric observation* for each agent, in which we separate the observation of the agent that observes from the other agents that are observed. Other works related to patrolling put the focus on pure fleet-movement coordination, such as Sun, Li, Chen, Dong, and Wang (2024). In this work, a related subtask of patrolling is addressed for cooperative defense using ASVs. Behavioral cloning is proposed to enhance the performance of a DRL-based policy. This idea explores the fact that there are very competitive algorithms that DRL can learn from.

The DRL has been used in other tasks beyond patrolling. It is common to find that reinforcement learning algorithms are used to design motion controllers for obstacle avoidance or tracking. An example of DRL application in surface vehicles is in Song, Gan, Yao, Zang, and Qu (2023). In this work, the use of an Actor Critic algorithm is proposed to solve the tracking problem. Here, the use of a realistic model of the environment is proposed: dynamic model of waves, inertias and associated hydrodynamic phenomena. However, the work limits its scope to low-level target tracking, without any model of environment perception or exogenous variables (biological or human). Other works, such as Sawada, Sato, and Majima (2021), propose a similar task but incorporating LSTM recurrent networks in the algorithm. This is a particularly interesting mechanism when, in the absence of an explicit dynamic model, past state information needs to be incorporated in decision making. This work realistically models the obstacles in the environment using proximity sensors and uses this LSTM mechanism to implicitly estimate how the obstacles and the rest of the fleet will move. In the latter two works, continuous actions have been used because the control task imposes continuous control signals. To reduce the dimension of the decision space, in Zhang, Wang, Bi, and Huang (2024), it is proposed to use a Soft-Actor Critic algorithm with 11 discrete motion actions for the control of a water vehicle. In this work the model of the environment is static, and obstacle avoidance is again delegated to the policy through implicit modeling of the environment. However, the work proposes a hierarchical training mechanism to separate the tasks of obstacle detection, use of tides to promote motion, and tracking.

With respect to the environment information model, we explore the use of reconstruction techniques based on variational encoders (Kohl et al., 2019; Yi et al., 2020). In Kohl et al. (2019), a VAE network is proposed for image segmentation tasks, which translates a visual input into semantic maps using a stochastic kernel operator. This work also proposes processing the prior and posterior distributions of visual input using independent CNNs. In Yi et al. (2020), a VAE framework is proposed for the reconstruction of partially occluded solar radiation images. We make use of the proposed loss functions proposed in Yi et al. (2020) and the prior/posterior architecture in Kohl et al. (2019) but with variations to adapt it to our particular application process. These kind of techniques enable the possibility to obtain a deep regressor able to invert the dynamics of the hidden information. This is different from approaches such as Kathen et al. (2021) and Peralta et al. (2021) where the model has no other knowledge of the environment rather than the samples taken on every mission.

3. Problem statement

In this section, we discuss the Non-Homogeneous Patrolling Problem (NHPP). We also enumerate the assumptions taken in the particular case of the biological monitorization of Lake Ypacaraí, which will serve as our reference real-world scenario.

In the Static Non-Homogeneous Patrolling Problem (NHPP), the environment is modeled as a grid map, represented by a connected graph $G = (V, E, \omega)$. Each node $v \in V$ corresponds to a location a

robot can reach, while each edge $(u, v) \in E$ represents an unobstructed and shortest path between adjacent locations. Node adjacency is defined under the assumption that the grid is 8-connected. Thus, G is formally depicted as an 8-connected grid graph, with holes indicating the presence of obstacles. In our reference scenario, the nodes represent areas within the navigable waters where robots can travel to collect measurements. Time is treated as discrete in this model, and moving along any edge of the graph – whether horizontal, vertical, or diagonal – requires 1 temporal unit.

The function $\omega : V \to \mathbb{R}_{\geq 0}$ assigns to each node its *information gain*, namely a value that, in the scope of the global monitoring task, quantifies the importance of a measurement taken in the area associated with the node. The higher $\omega(v)$ the more valuable a measurement gathered at v. Clearly, ω captures a key aspect for the efficiency of the monitoring task, contingent upon the specific environment in which such a task is carried out. This function is commonly employed to characterize the spatial biological interest across the area to be surveyed. For instance, in Kathen et al. (2021) and Yanes et al. (2020), this relevance is expressed through a contamination index within the range of $[0, 1]$, which denotes the level of anomaly in water quality. In this work, we define the information gain for a node v as $\omega(v) = \omega_0(v) + \omega_1(v)$. The component $\omega_0(v)$ is independent of the measurement taken at node v. Essentially, ω_0 represents the intrinsic significance of v, stemming solely from its presence in the environment. The other component, $\omega_1(v)$, is directly proportional to the informational value at v and thus varies based on the measurement a robot acquires there. Notably, ω_1 is initially unknown and can only be determined by having a robot performing a measurement at the related location. Being related to real measurements taken in the environment, this value typically exhibits some degree of spatial correlation.

Given a set of agents $J = \{1, 2, \dots, m\}, m \geq 1$, tasked with deployment in the environment for monitoring purposes. Addressing the NHPP involves determining a collection of paths $\psi = \{\psi_1, \psi_2, \dots, \psi_m\}$, assigning one distinct path to each ASV. In particular, the path for robot j is a sequence of nodes $\psi_j = (v_1, v_2, \dots, v_T)$ that the agent is scheduled to visit, where each consecutive pair of nodes (v_i, v_{i+1}) is an element of the edge set E. This implies that an agent j conducts a measurement at every node it visits along its path, specifically at time step t it will perform a measurement at node $\psi_j(t)$. Given the practical constraints of real-world monitoring applications, it is imperative that the entire monitoring mission concludes within a predefined time limit T. When a measurement is taken at node v, a robot uncovers $\omega_1(v)$ and designates a group of nodes around v, within a certain maximum distance, as monitored at the current time. This set of nodes is referred to as the neighborhood of v, defined as $\mathcal{N}(v) = \{q \in V \mid d(v, q) \leq \kappa\}$. Here, $d(v, q)$ represents the shortest time required to traverse from v to q across the graph G, and κ is a parameter specifying the range.

In this context, a candidate solution ψ yields the function $M_\psi(t) = \{v \in \cup_{j \in J} \mathcal{N}(\psi_j(t))\}$, indicating the set of nodes monitored by the team of robots at a given time t. Through this function, a solution ψ generates an idleness profile for each node in the environment. The idleness at a node, usually defined as the number of time steps since its last visit, can be interpreted as a constant cost τ incurred for every time step the node remains unvisited. Formally, we denote the idleness of node v at time step t as $w(v, t)$, where $w(v, 0) = 0$ and

$$w(v, t+1) = \begin{cases} \min\{W_{MAX}, w(v, t) + \tau\} & \text{if } v \notin M_\psi(t) \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

where W_{MAX} denotes the maximum idleness expected in a node, usually 1 if the idleness is min–max normalized as in Yanes et al. (2020).

An optimal solution to NHPP can be defined as a set of paths π^* that minimizes an environment-cumulative weighted idleness over the mission horizon T. That is,

$$\pi^* = \arg\min_{\pi} \sum_{v \in V} w(v, T)\omega(v) \tag{2}$$

4. Methodology

In this Section, we will explain the multi-agent Reinforcement Learning framework and the proposed algorithm to optimize the fleet policy. We will also address the Variational Deep scheme and how it is trained to invert the observations into the complete state of the environment.

4.1. DDQL framework

Deep Reinforcement Learning algorithms use a neural network to represent a deep policy $\pi(s \mid \theta)$. The deep policy is trained by trial and error through the agent interaction with the environment. In DRL algorithms, the cornerstone of optimization is the generation of experiences $< s_t, a_t, s_{t+1}, r_t >$ composed by the state s, actions a (which represents any feasible movements in the aforementioned graph G for an agent, and the generated rewards r that result from this movement. DRL algorithms seek to adjust the policy parameters to maximize the cumulative future reward $R = \sum_{t=0}^{T} \gamma^t r_t$ over a control horizon T and given a discount factor γ that balances the importance of short-term rewards. One of the most widely used algorithms in scenarios with discrete actions $a \in A \subseteq \mathbb{R}^{|A|}$, is Double Deep Q-Learning (DDQL). This algorithm, first proposed in Hasselt, Guez, and Silver (2016), attempts to estimate the state–action function $Q(s, a)$, which represents the cumulative and discounted reward, given a state s and each action a:

$$Q(s_t, a) = \sum_{k=t}^{T} \gamma r(s_k, a) \tag{3}$$

The core of DQL lies in minimizing the Bellman error, which quantifies the discrepancy between the current estimation of the action-value function and a target value based on the Bellman equation. Given a transition (s_t, a_t, r_t, s_{t+1}), the Bellman error is defined as follows:

$$\delta = r_t + \gamma \cdot Q(s_{t+1}, \arg\max_{a'} Q(s_{t+1}, a'; \theta); \theta^-) - Q(s_t, a_t; \theta) \tag{4}$$

where γ is the discount factor that weights the importance of future rewards, $Q(s_t, a_t; \theta)$ represents the current estimation of the Q-value for state s_t and action a_t using the online network with parameters θ, $\max_{a'} Q(s_{t+1}, a'; \theta)$ represents the maximum Q-value over all possible actions in the next state s_{t+1}, and where $Q(s_{t+1}, \arg\max_{a'} Q(s_{t+1}, a'; \theta); \theta^-)$ represents the Q-value of the action selected by the online network but evaluated using the target network parameters θ^-. Note that DDQL employs two separate neural networks: the online network and the target network. The target network is updated every N_{update} training step, which updates the targets to the already seen data. Once the optimal Q-value function is reached, the optimal action is given by:

$$a^* = \arg\max_{a'} Q(s, a') \tag{5}$$

4.2. The ϵ-greedy exploratory policy

In order to explore the state–action space, it is important to take actions different from those indicated by Q. An ϵ-greedy policy is employed to achieve this balance between exploration and exploitation of Q. At each time step, the agent selects the action with the highest Q-value with a probability of $(1-\epsilon)$ (exploitation), and with a probability of ϵ, selects a random action (exploration). The value of ϵ determines the degree of randomness in the policy and is annealed over the training time to gradually shift the agents' behavior from exploration to exploitation as learning progresses.

4.3. Prioritized experience replay

For this algorithm to converge, it is necessary to employ a replay memory buffer that stores the agent's experiences during interactions

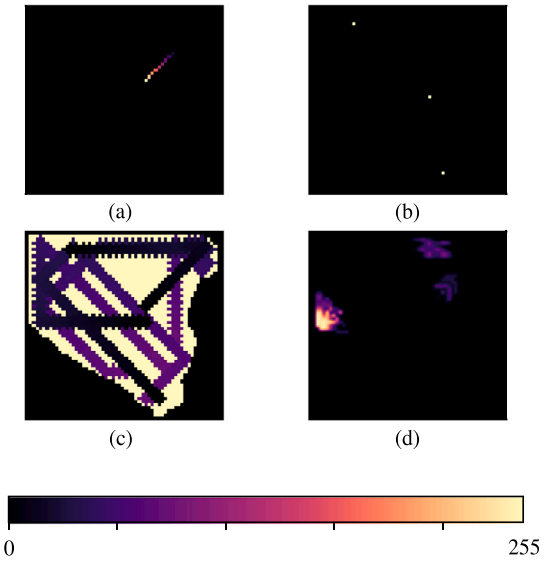

Fig. 2. Example of the observation of an agent j.

with the environment. By randomly sampling mini-batches of experiences from the replay memory, we decorrelate the training data and mitigate potential issues caused by temporal correlations in the sequences of experiences. To improve the better learning of experiences with higher temporal errors δ, we implement a Prioritized Experience Replay (PER) like in Schaul, Quan, Antonoglou, and Silver (2015). PER assigns priorities, denoted by p_i, to experiences based on their absolute Bellman errors δ_i, which measure the significance of a transition. The priority p_i of experience i is computed as:

$$p_i = |\delta_i| \tag{6}$$

The higher the absolute Bellman error, the higher the priority, indicating that the experience is more informative for learning.

4.4. Safe DDQL for multiple agents

Due to the priorities of Eq. (6), observations become interchangeable between agents and can be stored in the same experience buffer to indistinctly train the shared policy. The reward function, which is the same for all agents, also allows multiple agents to be trained with a single neural network for the same cooperative objective. The policy simply maps each agent's egocentric observation into its Q values, and each agent takes the highest valued action independently. This approach is halfway between Independent Q-Learning where each agent has its own neural network and centralized Q-learning (Yanes et al., 2020), where agents' actions are decided jointly with a single global observation. A direct benefit of this strategy is that the fleet size can change, and the policy can continue to be used. In addition, this architecture allows for the setting up of a serverless scheme, where each agent can carry the policy on its local hardware. Another aspect related to the problem scaling is that the neural network architecture does not change when the number of agents changes. Algorithm 1 reports the proposed Multi-agent DDQL Algorithm.

With respect to obstacle avoidance, the DRL is able to effectively learn the boundaries of the map (Yanes et al., 2020), the actions that generate an agent-scenario collision can be deterministically calculated (Yanes Luis et al., 2023). It is different with the collision between agents since, by deciding the actions simultaneously, two agents may decide to move to the same place. This is a particular NP-hard subproblem of Multi-agent Path Finding (Stern et al., 2019). In

Algorithm 1 Safe Multi-agent Double Deep Q-Learning Algorithm

Initialize replay memory D to capacity $|D|$
Initialize target Q-network Q' with weights $\theta' = \theta$
Initialize policy network Q with weights θ
for episode = 1 to E_{max} **do**
  Reset environment
  Get initial observation $o_0 = \mathcal{O}(s_0)$
  for timestep = 1 to T **do**
    $p \sim U(0,1)$
    if $p < \epsilon$ **then**
      $\boldsymbol{a}_j \leftarrow SafeConsensus(U(0,1), \dots, U(0,1))$
    else
      $\boldsymbol{a}_j \leftarrow SafeConsensus(Q(o_0, a), \dots, Q(o_{|A|}, a))$
    end if
    Execute action $\boldsymbol{a}_j$
    Observe rewards r_j and new observations o_j^{t+1}
    Store every transition $(o_j^t, \boldsymbol{a}_j^t, r_j^t, o_j^{t+1})$ in D
    Sample random batch B of $(o_j, \boldsymbol{a}_j, r_j, o_{j+1})$ from D
    Set $y_j = r_j + \gamma Q'(s_{j+1}, \arg\max_{\boldsymbol{a}} Q(s_{j+1}, \boldsymbol{a}; \theta); \theta')$
    Update weights by minimizing the loss:

$$\mathcal{L}(\theta) = \frac{1}{B}\sum_{j=1}^{B}(y_j - Q(s_j, \boldsymbol{a}_j; \theta))^2$$

    if $mod(t, M) == 0$ **then**
      $\theta' \leftarrow \theta$ ▷ Target weights update.
    end if
  end for
  $\epsilon \leftarrow \min(\epsilon_{min}, \epsilon - d\epsilon)$.
end for

this paper, it is proposed to overcome these situations by means of a heuristic based on the conditional decision $a^j = \pi(o^j | a^{j^-})$ of an action taken by an agent a^j, depending on the actions of the other agents a^{j^-}. First, we order the agents according to the largest estimated value of Q. The agent with the highest value in Q, which is under the assumption of good estimation the most promising action, follows the ϵ-greedy policy free of fixed obstacles without taking into account the movements of the rest of the agents. The other agents also take an action considering the new position of the previous agent. Those values of Q that place the agent in collision are deterministically censored with a value of $-\infty$. A new action, already safe, is taken by that agent and the next agent does the same. When all actions have been decided, the movements are processed, and measurements are taken. In this way, this heuristic relies on the optimism of the agents to take precedence. In the event that a random action is taken, following the ϵ-greedy policy, again only safe actions are considered and no collision is produced. We provide the pseudo-code of the whole Safe Dueling DQL algorithm in Algorithm 1 and the particular subroutine for consensus in Algorithm 2.

4.5. Neural network

The shared policy is implemented by a neural network composed of an initial Convolutional Neural Network (CNN), a fully connected 3-Layer Perceptron, and a final two-headed Dueling scheme (see Fig. 3). The CNN, formed by 3 convolutional layers with ReLU activations, extracts spatial features from the state o_j^t, which is modeled as a single-channel image (see Fig. 2). The observation channels correspond to: (a) the navigation map, the information model $\hat{\omega}$, (c) the idleness map $w(v,t)$, (d) the position of the agent j, and (e) the position of the other agents j^-.

The resulting embedding is processed by 3 linear layers with ReLU and then fed into a Dueling Deep Q-Network (DQN) like in Wang, Freitas, and Lanctot (2015). A Dueling network augments the traditional DQN architecture with two streams for value estimation. The

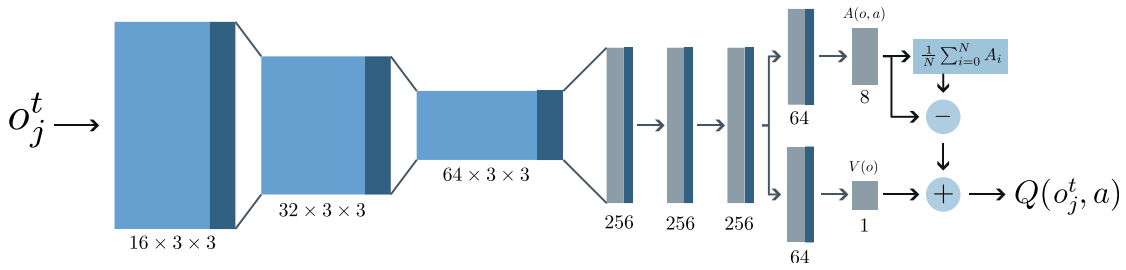

Fig. 3. Dueling Deep Q-Network architecture with a convolutional encoder.

Algorithm 2 Safe Consensus Algorithm

Require: Positions $P^t = \boldsymbol{p}_1^t, \boldsymbol{p}_2^t, \dots, \boldsymbol{p}_N^t$ of N agents at time t
Require: Values $Q = \{Q_1, Q_2, \dots, Q_{|A|}\}$ that weight each agent's action.
1: Initialize empty set of future positions $P^{t+1} := \emptyset$
2: Obtain order of agents' actions in decreasing order of their Q values: $j_1, j_2, \dots, j_N$, such that $\max Q_{j_1} \geq \max Q_{j_2} \geq \dots \geq \max Q_{j_N}$.
3: **for** each agent j in order of actions **do**
4: Select greedy safe action

$$\boldsymbol{a}_j = \arg\max_{\boldsymbol{a} \in A} Q_j(a)$$

subjected to:

$$\|(\boldsymbol{p}_j^t + \boldsymbol{a}_j) - \boldsymbol{p}'\|_2 \leq d_{safe} \quad \forall \boldsymbol{p}' \in P^{t+1}$$

5: $P^{t+1} \leftarrow P^{t+1} \cup (\boldsymbol{p}_j^t + \boldsymbol{a}_j)$ ▷ Update next fleet positions.
6: $A_{selected} \leftarrow A_{selected} \cup \boldsymbol{a}_j$ ▷ Update consensus actions.
7: **end for**
8: **return** $A_{selected}$

first stream computes the state value function $V(o_j)$, representing the expected cumulative reward given an observation in state o_j and following the current policy. The second stream calculates the action advantage function $A(o_j, a)$, which measures how much better taking action a given an observation o_j is compared to the average action-value in that state. The two steams are combined using the following equation:

$$Q(o_j^t, a) = V(o_j^t) + A(o_j^t, a) - \frac{1}{|A|}\sum_{a'} A(o_j^t, a') \tag{7}$$

This separation improves the representation and approximation of the Q function and is particularly useful for tasks with large action spaces or complex environments, leading to more stable and accurate policy learning in DRL (Wang et al., 2015). The output of the layer will be, as stated in Section 5.2, the estimation of the Q-values for each possible Cartesian movement in the graph $G(V, E)$ for an ASV j in the current observed state O_j^t, with respect to the future discounted reward.

4.6. Reward function

The reward function is fundamental in all DRL algorithms (Sutton & Barto, 2018). The reward must consider in a quantitative manner the optimality of an action in a particular state. In this paper, the reward will be individually evaluated for the exact action. The reward function will be designed using the definitions of optimality of the NHPP in Eq. (2). With each new sample, every agent will receive the sum of idleness ω collected in its new position multiplied by the term of importance of the information given the importance model $\hat{\omega}$. To address redundancy in the measurement, we define a redundancy factor $\rho(v)$ as the number of ASVs that cover a particular node v_j in a particular instant. A node is considered covered if any agent is in such position v_i that, v_j is a neighbor $v_j \in \mathcal{N}(v_i)$. Thus, the reward associated with overlapping areas of different agents will be weighted depending on how many agents cover this particular position $\rho(v)$. Consequently, this will decrease the individual reward for taking measurements with multiple agents in the same place. The final reward for an action a_j^t will be:

$$r(s^t, a_j^t) = \sum_{v \in \mathcal{N}(v)} \frac{w(v,t) \times \omega(v,t)}{\rho(v)} \tag{8}$$

4.7. Information importance model

The information model is composed of a convolutional-deconvolutional neural network based on the UNet architecture (Ronneberger et al., 2015). This architecture is composed of a fully convolutional neural network (CNN) that combines both contracting and expanding paths. The contracting path consists of 4 convolutional and max-pooling layers. This process enables the network to learn high-level features from the entire input image. Following this, there is the deconvolutional path, which consists of 4 convolutional and up-sampling layers, which increases the spatial resolution of the feature maps. In the central part of the U, the contraction and expansion paths are connected through residual connections. The input of the network consists of two channels: i) a pre-processed model of importance Y^t constructed by a composition of the samples taken by the agents so far in the locations in which they were taken, and (ii) a binary visit mask M^t, with those places that have been sampled represented by a 1, and 0 when this place has not been sampled yet. This input will be queried from the UNet architecture to produce an estimate of the importance model at that time $\hat{\omega}^t$.

In this paper, this original architecture is expanded to implement a variational version of the UNet, as it was used in Yi et al. (2020) for image reconstruction. Two distinct convolutional networks are implemented, one to estimate the prior probabilistic Gaussian distribution $\mathcal{N}(\mu, \sigma)$, and the other to estimate the posterior distribution $\mathcal{N}(\hat{\mu}, \hat{\sigma})$, with $\mu, \sigma \in \mathbb{R}^N$. The prior network (used in inference) receives partial observations Y^t and is trained to minimize the Kullback–Leibler (KL) distance with respect to the output of the posterior network that also receives the ground truth of the information importance ω. With this probabilistic distribution, we generate a set of convolutional filters, each sampled from every distribution of $\mathcal{N}(\mu, \sigma)$. The resulting stochastic filters are added to the final output layer of the UNet.

The entire architecture is trained by minimizing the loss composed of a weighted sum of the following terms:

- **Reconstruction loss**: the Mean Squared Error (MSE) between the generated model and the real ground truth. This term is related to the capacity of the network to invert the observation and provide an estimate of the complete map:

$$\mathbb{L}_{recons} = MSE\left(\hat{\omega}, \omega\right) \tag{9}$$

- **KL loss**: The Kullback–Leibler divergence between the prior and posterior distributions. By minimizing this term, the network learns the inner distribution of the data.

$$\mathbb{L}_{KL} = KL\left(\mathcal{N}(\mu, \sigma), \mathcal{N}(\hat{\mu}, \hat{\sigma})\right) \tag{10}$$

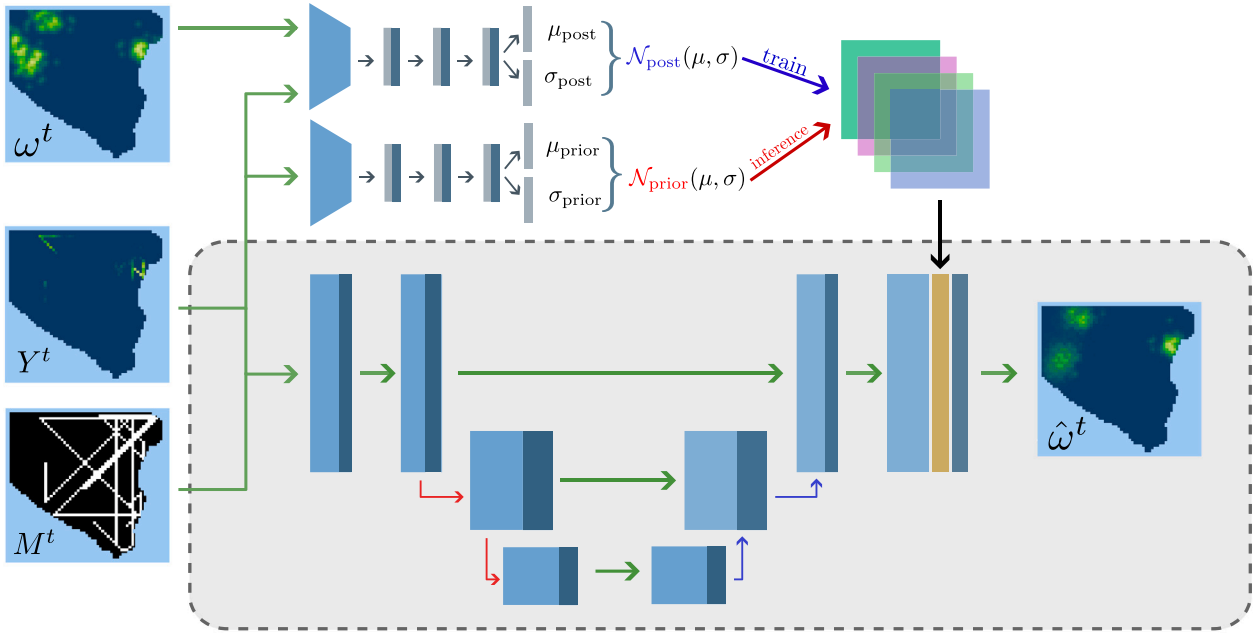

Fig. 4. VAE-UNet training architecture for training and inference.

- **Perceptual loss**: The difference between the high-level feature map ξ of the output and the ground truth from a pre-trained deep convolutional model like $VGG16$ (Gatys, Ecker, & Bethge, 2015).

$$\mathbb{L}_{perceptual} = MSE\left(\xi(\hat{\omega}), \xi(\omega)\right) \tag{11}$$

The final loss will be:

$$\mathbb{L} = \beta_0 \mathbb{L}_{recons} + \beta_1 \mathbb{L}_{KL} + \beta_2 \mathbb{L}_{perceptual} \tag{12}$$

with β_1 and β_2 being parameters to weigh the importance of each term in the minimization of total loss. These weights must be chosen to minimize the final prediction error between the final prediction of importance and the ground truth. A complete diagram of the model and its submodules is shown in Fig. 4.

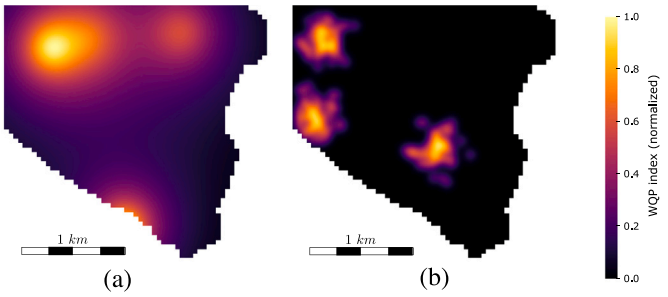

Fig. 5. Two examples of WQP distributions with both benchmarks in the southern part of Lake Ypacaraí: (a) Shekel benchmark, and (b) Algae Bloom Benchmark.

5. Training and results

5.1. Environment simulation

In this section, we first present the training results of both the Deep Model and the Deep Policy. Secondly, we present the results of the training of the Deep Model and we analyze its performance compared to other regression methods. Third, we show the results of the DRL training and discuss different swarm behaviors. Finally, we compare our approach with other state-of-the-art approaches in the literature. All simulations and training have been carried out on a server running Ubuntu 20.04, equipped with an Intel Dual Xeon Gold 5220R CPU2.20 GHz, 192Gb of RAM and two GPUs: Nvidia Quadro A4000 48 GB and Nvidia RTX 3090 25 GB.

All simulations were conducted with Lake Ypacaraías the baseline scenario with 4 vehicles, as it happened in previous approaches to monitoring contamination in large water resources (Yanes et al., 2020; Yanes, Reina et al., 2021; Yanes Luis et al., 2023). The movement-related parameters (d_{max}, d_{move} and N) has been selected based on Yanes et al. (2020). Their values has been selected by considering realistic values in terms of battery life of real ASV prototypes. The value of the forgetting factor τ is chosen as a user parameter, which imposes the effective time in which the information is valid for the user. This parameter is extrinsic to the problem and can be modified according to the desire for more or less monitoring rate. Finally, the simulation constant parameter Δt is simply a value for the integration of the laws of motion of each moving element. It must be chosen small enough to avoid integration errors.

Two different benchmarks are used to represent the information importance ω. On the one hand, a Shekel benchmark function[1] is used, similarly to many other previous works on monitoring water quality (Kathen et al., 2021; Peralta et al., 2021). This standard WQP-model function treats the contamination in a similar way to the data collected in the Mar Menor Data Server,[2] which is a particular case of contaminated water quality resource. The WQPs have a smooth distribution over the waters with peaks and valleys.

On the other hand, we present an Algae Bloom simulator based on a simplified diffusion model of blue–green algae bacteria. Up to 6 random algae blooms can appear in any part of the water. We treat these algae bloom bacteria as particles with random speeds v_r for each one, to model the diffusion effect of contaminants on the surface of the waters. In addition, two speeds related to wind speed v_w and water currents v_c are introduced. These three components are weighted to compose the final speed. The position of every particle is updated depending on these speeds by computing the discrete integral with a fixed time step Δt. Finally, to model the effects of the shores, the physical boundaries

[1] https://www.sfu.ca/~ssurjano/shekel.html
[2] https://marmenor.upct.es/maps/Transparency

Table 1
Summary of the simulation and benchmark parameters.

Parameter	Value
Number of agents (N)	4
Number of actions $\|A\|$	8
Max. distance (d_{max})	32 km
Movement distance (d_{move})	160 m
Forgetting factor (τ)	0.01
Simulator time constant (Δt)	0.05

Table 2
Summary of the training parameters of the Deep VAE.

Parameter	Value
Learning rate (lr)	$1 \cdot 10^{-3}$
Batch Size	64
Max. Epochs	30
Discount factor (γ)	0.99
Training dataset size (Shekel)	3000 ground truths
Training dataset size (Algae Bloom)	3000 ground truths
Validation dataset size (Algae Bloom)	100 ground truths
Validation dataset size (Algae Bloom)	100 ground truths

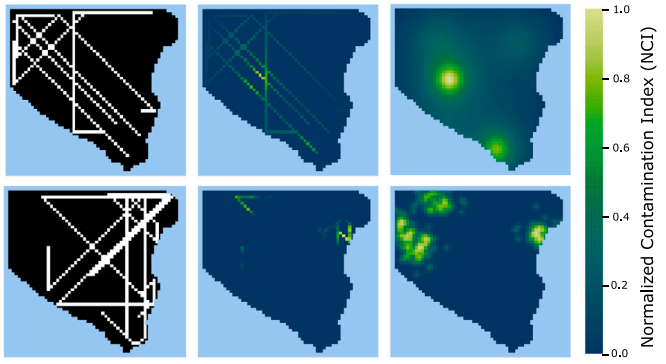

Fig. 6. Example of two inputs for the VAE-UNet taken in a random timestep for both benchmarks. In the left, the first channel with the measurement mask. In the middle, the measured values in their positions. In the right, the respective ground truths.

of the navigable zones exert a pushing-back force to the particles. The Algae Bloom simulations will be termed static when the blooms are simulated for a random period and then considered, in general terms, stationary (see Tables 1 and 2).

5.2. Importance model training

In order to train the VAE Importance Model, a training set of 3000 functions from both the WQP simulator and from the static Algae Bloom simulator are generated. In addition, a validation set and a test set of 100 ground truths of each are also simulated for performance comparison (see Fig. 5 for an example of both benchmarks). This data is generated with different random seeds and different starting conditions such as initial Algae Blooms, maxima-minima positions, currents, and wind conditions. Consequently, no single ground truth sampled will be found simultaneously in the training and validation set. An example of the input and ground truth for both benchmarks can be seen in Fig. 6.

Thus, each ground truth is paired with the random path generated for each vehicle using a Non-Redundant Path Planner (NRPP), as explained in Yanes Luis et al. (2023). This path planner consists of a random coverage heuristic used in exploratory scenarios. Each vehicle selects an obstacle-free trajectory, and when the shore is reached, another random direction is selected. This random path selection is used to reduce the bias of the Importance Model with respect to the type of path planner used in the application. As path planning also plays a role in the quality of the estimation, it is important to decouple the importance inference task from the patrolling task to avoid cross bias.

Table 3
Best obtained loss weights and its ANOVA significance for the hyperparametrization of both ground truths.

Parameter	Best	ANOVA signf.
Algae Bloom benchmark		
Learning Rate	2.8×10^{-3}	0.71
Recons. Loss (β_0)	6.402	0.08
KL Loss (β_1)	4.561	0.16
Percep. Loss (β_2)	4.18	0.03
WQP benchmark		
Learning Rate	1.4×10^{-3}	0.54
Reconstruction Loss (β_0)	3.42	0.29
KL Loss (β_1)	1.84	0.08
Perceptual Loss (β_2)	4.23	0.08

The VAE is trained for 50 epochs on each benchmark. The weight loss ($\beta_0, \beta_1, \beta_2$) from Eq. (12) are optimized using a Tree-structured Parzen Estimator (TPE) sampler (Bergstra, Bardenet, Bengio, & Kégl, 2011), by minimizing the Mean Squared Error (MSE) of the predictions over the validation datasets. The most significant term for both benchmarks is the reconstruction loss. In Table 3 we present all the hyperparameters related to the training of the model and the estimated importance given its Analysis of Variance (ANOVA) test.

For performance evaluation, we will analyze 3 metrics: (I) the Root Mean Squared Error (RMSE) that will measure the average residuals between the estimated model and the ground truth at each timestep. (II) A weighted RMSE that will ponder the error of every zone with its importance, so the error in highly important will weight more. (III) The cost of computing every method in seconds to evaluate the aptitude of the model to work in a time-constrained application. We compare our method with other approaches for 2D regression:

1. **Myopic naive model:** This is a naive model used as a baseline. This model is updated only in the vicinity κ of every measurement with the most updated information.
2. **Radial k-Nearest Neighbors (R-kNN):** A 2D k-Nearest Neighbors algorithm. Every position in $\hat{\omega}$ is updated with the nearest sampled value within a radius r_{KNN}.
3. **Gaussian Process (GP):** Taken directly from Peralta et al. (2021). This model uses a Radial Basis Kernel to model the correlation between samples. The model output will be directly the mean of the GP.

In Fig. 7a-c the RMSE and the weighted RMSE are represented with respect to the number of measurement steps. Each step involves one measurement per vehicle. It can be seen that the proposed model is able to predict the real importance map with higher accuracy in all cases. In the WQP benchmark, with enough distance budget, both the GPs and VAE-UNet algorithms are able to find low errors. However, the proposed methodology can reduce the error earlier in this first scenario by a 28% and a 30% in the first third and second third, respectively, with respect to the second-best method (GP) (see Table 5 and Fig. 7a). In the Algae Bloom scenario, the improvement is higher, with a 35% and 29% in the first and two thirds (see Table 4 and Fig. 7b). This improved performance has been observed to be related to the lack of smoothness in algae blooms. The GPs seem to overfit the hyperparameters of the RBF kernel to the low-bandwidth features of the blooms. The VAE algorithm is able to deal, at the same time and more efficiently, with smoothness predictions in valley zones and noisy high-importance areas. The main benefit of the proposed algorithm with respect to the patrolling task, inferred from the results in Fig. 7, is the faster convergence of the proposed algorithm. With battery restrictions, a faster convergence in the model provides the path planner with a better understanding of hidden ω to deal with intensification much more efficiently. To ensure that these results are statistically significant, a Wilcoxon Ranked test (Wilcoxon, 1945) has been used to validate the

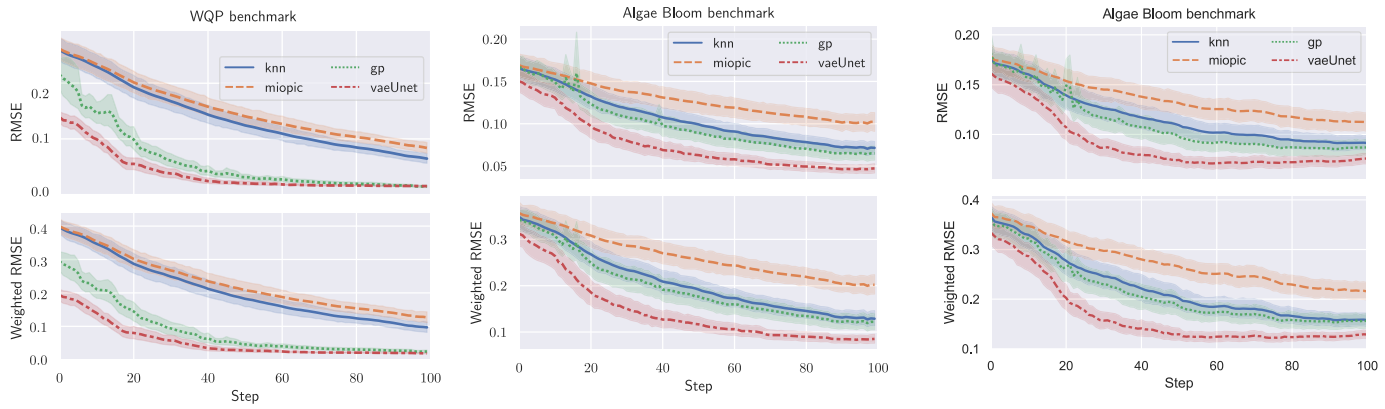

Fig. 7. Comparison of the average RMSE and Weighted RMSE between different regression models with 100 sampled ground truths from (a) WQP function, (b) static Algae Bloom benchmark, and (c) dynamic Algae Bloom benchmark simulator.

Table 4
Average RMSE and its Confidence Interval values for each model in the Algae Bloom benchmark.

	Error 33%		Error 66%		Error 100%	
	μ	CI (95%)	μ	CI (95%)	μ	CI (95%)
rKNN	0.117	±0.012	0.086	±0.009	0.071	±0.008
Myopic	0.137	±0.012	0.115	±0.011	0.102	±0.011
GP	0.107	±0.013	0.078	±0.010	0.065	±0.008
VAE-UNet	0.077	±0.011	0.055	±0.007	0.047	±0.005

Table 5
Average RMSE and its Confidence Interval values for each model in the WQP benchmark.

	Error 33%		Error 66%		Error 100%	
	μ	CI (95%)	μ	CI (95%)	μ	CI (95%)
rKNN	0.163	±0.017	0.102	±0.012	0.065	±0.010
Myopic	0.175	±0.017	0.120	±0.012	0.084	±0.010
GP	0.056	±0.010	0.024	±0.003	0.015	±0.002
VAE-UNet	0.036	±0.007	0.017	±0.001	0.013	±0.001

Table 6
Average RMSE and its Confidence Interval values for each model in the dynamic Algae Bloom benchmark.

	Error 33%		Error 66%		Error 100%	
	μ	CI (95%)	μ	CI (95%)	μ	CI (95%)
rKNN	0.125	±0.012	0.100	±0.009	0.092	±0.007
Myopic	0.144	±0.012	0.124	±0.011	0.113	±0.009
GP	0.115	±0.012	0.091	±0.009	0.087	±0.008
VAE-UNet	0.086	±0.009	0.072	±0.006	0.076	±0.005

comparisons between the proposed method and others with $p < 0.05$.

When introducing dynamics in the Algae Bloom simulation, the proposed VAE architecture is still able to give better predictions even when the training dataset is static. Since in this scenario, new measurement substitute old ones in the visual input, the VAE network can still process estimations without much loss of generalization. It is observed a 20% improvement (with a $p < 0.05$ according to the Wilcoxon Ranked test), over 100 episodes across the measurement campaign (see Fig. 7(b)) with respect to the GPs. It is seen that GPs suffer from loss of generalization when two samples too close are acquired in different times. The other algorithms, although they can come up with a comprehensive image of the contamination at the end of a exploratory mission, cannot capture the realistic image of this benchmark (see Table 6).

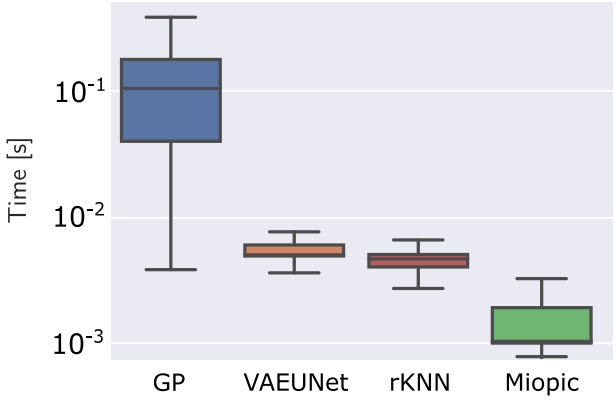

Fig. 8. Inference times for each model under test.

In Fig. 8 it can be observed the probabilistic distribution of the different computation times for every model under test. The computation times are represented there for 100 different multi-agent paths and for the three benchmarks. It can be observed that the second-best algorithm on average in terms of the importance model error needs one order of magnitude more time per computation than the VAE-UNet proposed algorithm. The computation time is similar for the rKNN algorithm, with a 10% slower time. The faster approach is obviously the Myopic approach, which obtains the worst performance in the model accuracy. This result can be interpreted as the metric robustness of the proposed information model to be used in real-time scenarios with more strict time-critic requirements. It is important to note that the computation time of the VAE-UNet is constant regardless of the number of samples M. This is not true for the rKNN benchmark and (especially) for the GPs, which suffer from a complexity of $\mathcal{O}(M)$. Then, the computation times are expected to increase significantly with larger paths in a continuous monitorization task. This fact supports the decision to use alternative methods such as the one proposed here with VAE-UNet for the continuous patrolling task, provided that there is some kind of information or prior on how the information to be monitored behaves.

5.3. Deep policy training

Once the VAE-UNet model is trained, each information model is embedded in the DRL framework for training. Two different trainings are conducted for the two benchmarks, with a total training budget of 20.000 episodes each. During training, every mission is conducted in a completely new sample of the benchmark, not used in the model

Table 7
Summary of the training parameters of the DRL algorithm.

Parameter	Value
Learning rate (lr)	$4 \cdot 10^{-4}$
Batch Size	64
Episodes	20.000
Target update freq. (M)	500 episodes
Number of agents (N)	4

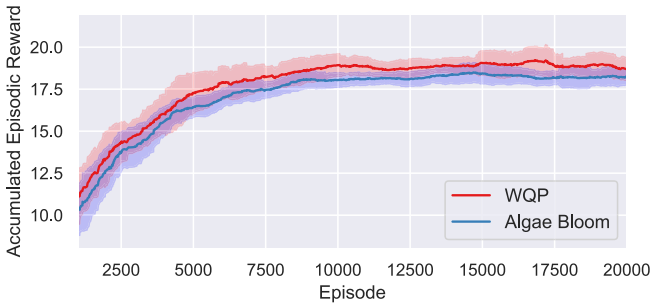

Fig. 9. Average rewards during training for both benchmarks.

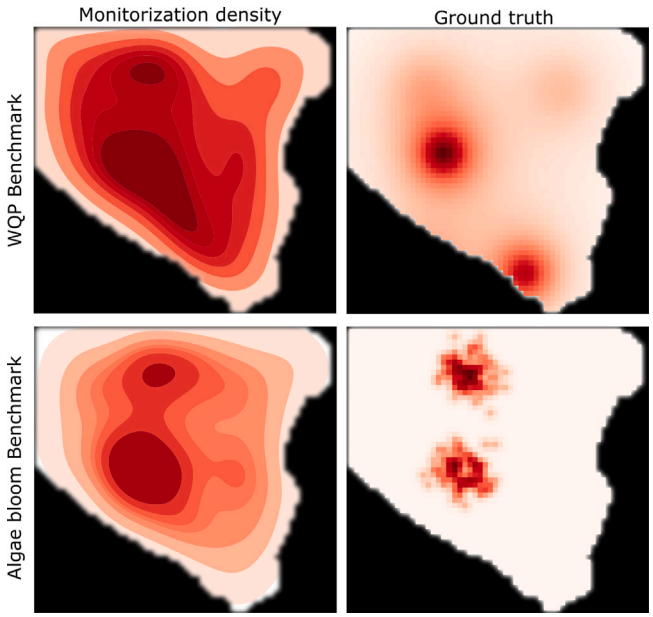

Fig. 10. Heat map of measurements taken by the DRL agents trained.

training, nor in the validation, to avoid estimation bias. Each policy is trained with the very same simulation parameters reported in Table 7. During training, every agent will act according to the aforementioned ϵ-greedy explorative policy. After training, every policy is evaluated on 100 different benchmarks that are not used during the training. For evaluation, every agent will act completely greedy ($\epsilon = 0$) only considering the predicted Q values of the network and the safety mechanism to avoid overlapping measurements until the distance budget is reached (d_{max}).

In Fig. 9, the collective accumulated rewards are represented for both benchmarks. It can be observed that, as training progresses, the performance of the agents improves. The improvement stalls after 10.000 episodes, as ϵ reaches a minimum (exploitative learning). This indicates that training can be shortened if needed for these two benchmarks. However, in this work, the training budget is imposed to be high enough to observe the full potential of the DRL capabilities. The total duration of a training is approximately 16.5 h and 18.5, respectively, for the WQP and Algae Bloom benchmark (using the hardware setup described at the beginning of Section 5.1).

Fig. 10 shows the intensity of the monitoring when deploying the trained policy in two random ground truths with the proposed benchmarks. This image conforms to a heat map of the non-homogeneous coverage that constitutes the patrolling paths. The intensity has been computed using a kernel density estimator (KDE) with Scott's rule for bandwidth selection (Scott, 2012). It can be observed that deep policies intensify the coverage especially in the zones of high informative importance, which is a visual confirmation of the expected behavior of the deep agents. In the WQP benchmark (top of Fig. 10) the monitoring is more homogeneous throughout the environment, since the function is smoother and there is a smaller spatial variation of importance. In the Algae Bloom benchmark (bottom of Fig. 10), the resulting behavior is more focused on contamination peaks, as covering the nonimportant areas (white in the bottom right of Fig. 10) only provides a reward of $w \times \omega_0$.

5.4. Validation of the DRL planner

To address the efficiency of the proposed method, the DRL is compared to the path planning methods from previous work (Arzamendia, Gregor, Gutierrez-Reina, Toral, 2019; Krause & Guestrin, 2007). For a fair comparison, all algorithms will use the VAE-UNet model if needed. The following methods are used for comparison:

- **Vehicle Routing Problem solver**: The Vehicle Routing Problem (VRP) is a combinatorial optimization challenge in logistics. It involves determining the most efficient routes for a fleet of vehicles to visit a set of nodes considering constraints such as vehicle distance budget, time windows, while balancing the vehicles charge. To solve the VRP, a lattice of visitable points from the navigation space, equally distanced by d_{move}, is created. Then, a VRP solver such as OR-Tools[3] optimizes the fleet paths. The resulting paths constitute a cyclic solution that covers the entire space once per cycle. To comply with the patrolling task, the agents repeat the cycle when finished until the end of the mission.
- **Greedy algorithm**: This Patrolling problem is a candidate to be a submodular function. A submodular function is a set function that exhibits a diminishing returns property: the marginal gain of adding measurement element to a smaller set of measurements is greater than adding it to a larger set, as proposed in Krause and Guestrin (2007) to address the Informative Path Planning with marine vehicles. Greedy algorithms are known to provide near-optimal solutions for maximizing submodular functions. The greedy approach iteratively selects the next point that maximizes the reward given the current observation of the idleness and importance model. Due to the diminishing returns property, this strategy ensures that each chosen element contributes significantly to the overall objective. Although greedy algorithms may not guarantee the absolute optimal solution, they often achieve approximation guarantees, making them computationally efficient for large-scale problems like this (Krause & Golovin, 2014).

The results in Fig. 11 show that the DRL approach is able to obtain paths with a lower weighted idleness $w \times (\omega_0 + \omega_1)$ on average for steady-state monitoring in less time. The VRP solution, although it can perform well in the homogeneous coverage task, cannot address the criterion of different importance imposed by ω. This approach is more suitable for pure homogeneous scenarios or when no assumptions can be made about the importance. However, it serves as a strong baseline for comparing more sophisticated algorithms with robust offline path

[3] https://developers.google.com/optimization

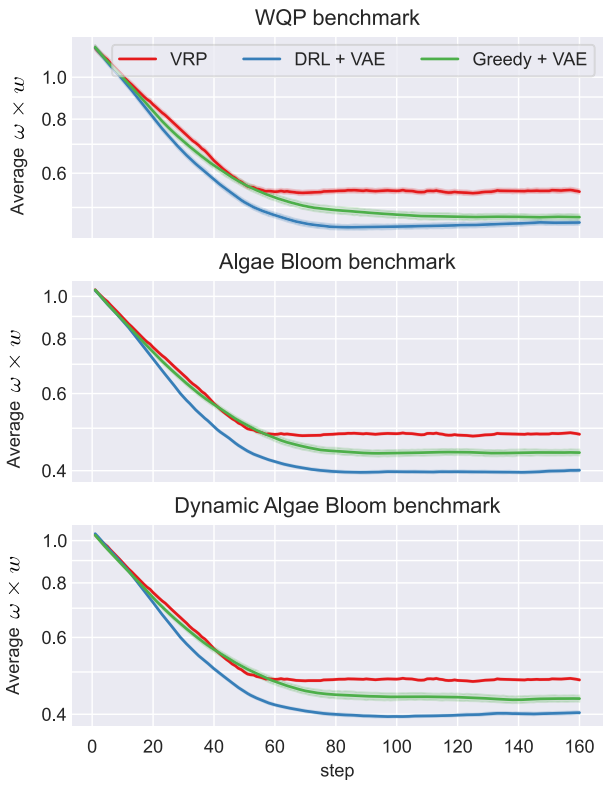

Fig. 11. Non-homogeneous Patrolling score $w \times (\omega_0 + \omega_1)$ from VAE-UNet evaluation experiments.

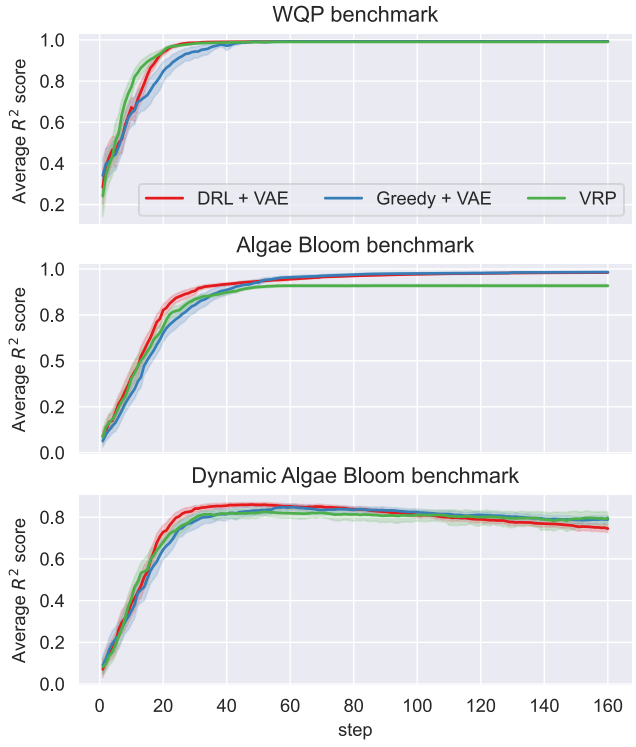

Fig. 12. R^2 score resulting from VAE-UNet evaluation experiments.

Table 8
Final metrics for the ablation study between algorithms with and without VAE in the WQP benchmark.

	$w \times (\omega_0 + \omega_1)$		R^2	
	μ	CI (95%)	μ	CI (95%)
DRL + Myopic	0.484	±0.0062	0.965	±0.0031
DRL + VAE	0.462	±0.0062	0.992	±0.0008
Greedy + VAE	0.475	±0.0078	0.992	±0.0008
Greedy + Myopic	0.490	±0.0096	0.964	±0.0032

planning. In this sense, in Fig. 12, it can be seen that the accuracy of the model (represented by the score R^2) is still high (> 0.9).

With regard to the second-best algorithm, it has been observed that, despite the simplicity, the greedy approach obtains well-founded results. This result indicates that the patrolling problem's objective function in this paper shows high levels of submodularity. The greedy algorithm is able to obtain close results to the DRL resulting algorithm in the WQP benchmark. There, the importance of information is smoother and greedy agents act as gradient-ascent particles with respect to weighted idleness $w \times (\omega_0 + \omega_1)$. In the Algae bloom benchmark (both static and dynamic), the patrolling is less efficient than the DRL policies since the DRL is able to consider long-term spatial dependencies over the actions. For example, with a better partition of the monitorization space between agents and avoiding the monitorization redundancy.

In the end, the three algorithms are able to obtain a good accuracy model of the environment. Nevertheless, the dissimilar performance in the patrolling indicates that a good model is only complementary to a good patrolling, especially when different agents need to cooperate in the same scenario. One important aspect of the DRL usefulness for real applications is that the DRL is able to explicitly incorporate mechanisms of cooperation by means of the reward function, which can be beneficial on the long term behavior for the fleet objective.

In order to compare the model importance in the patrolling task, the DRL policies and the Greedy agents have been evaluated with and without the VAE-UNet to help with importance prediction. Two new DRL policies have been trained without the VAE-UNet mechanism and only using the Myopic model with the same training conditions explained in the previous case (see Table 7). This experiment is an ablation study to understand how a model can improve non-homogeneous patrolling and to what extent.

In Fig. 13 the performance is depicted for both benchmarks of the different algorithms. If the marginal improvement of the algorithms with respect to their Myopic baseline is analyzed, it can be observed that, for the Greedy algorithm, the average improvement is below 4% at the end of the episode (see Tables 8 and 9). For the DRL, the use of the VAE model implies, at the end of the episode, a 6% lower $w \times (\omega_0 + \omega_1)$. This is explained because, as observed in Fig. 14, at the end of the episode, all algorithms obtain a comprehensive image of importance, which in the end collapses in a good patrolling. A major improvement is seen in the convergence speed of the VAE enhanced algorithms. With VAE estimates on information importance, the DRL agent is able to perform slightly better approximately a 30% faster than the DRL + Myopic fleet.

When comparing the performance of the myopic fleet between DRL and the Greedy algorithm, it can be observed that the DRL is more robust to an inefficient model than the Greedy counterpart. The DRL algorithm is able to learn to some extent the patrolling task in spite of the model. There are also other aspects related to performance in the DRL that surpass the greedy algorithm as the long-term cooperation that effectively enhances the performance in the first case.

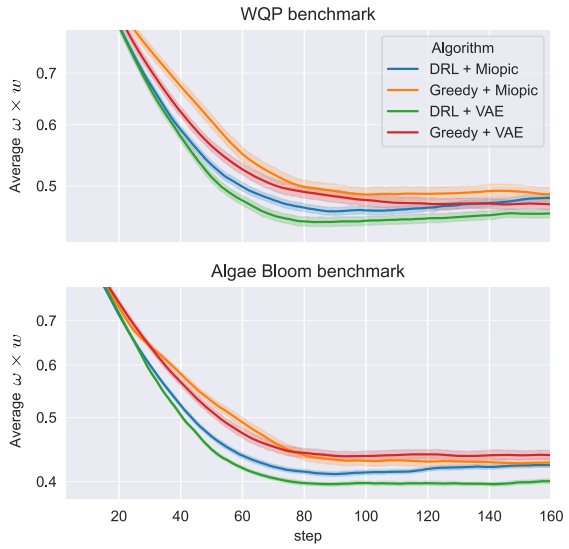

Fig. 13. Average patrolling score $w \times (\omega_0 + \omega_1)$ for the ablation study with the two benchmarks.

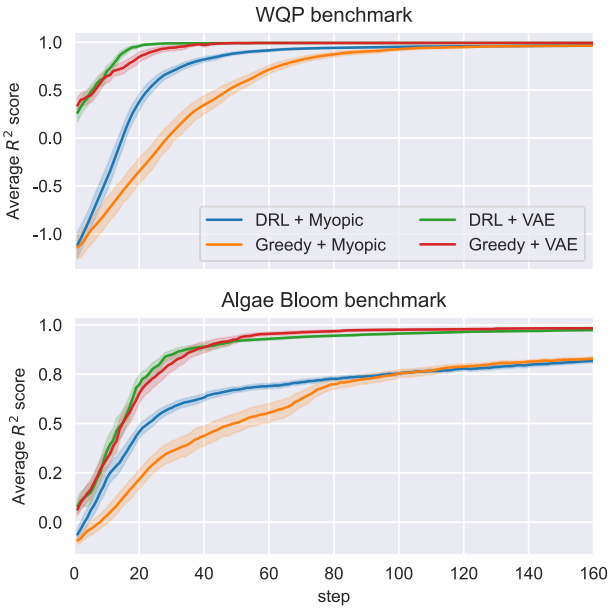

Fig. 14. R^2 score of the model for the ablation study with the two benchmarks.

Table 9
Final metrics for the ablation study between algorithms with and without VAE in the Algae Bloom benchmark.

	$w \times (\omega_0 + \omega_1)$		R^2	
	μ	CI (95%)	μ	CI (95%)
DRL + Myopic	0.424	±0.0036	0.818	±0.0052
DRL + VAE	0.401	±0.003	0.974	±0.0002
Greedy + VAE	0.429	±0.0074	0.983	±0.0011
Greedy + Myopic	0.429	±0.0074	0.820	±0.0066

6. Conclusions

In this paper, a novel approach that addresses the Non-homogeneous patrolling for multiple agents has been presented. This approach proposes the use of a Deep Reinforcement Learning algorithm to train multiple agents to continuously patrol a water scenario in two different contamination scenarios. The framework incorporates a Variational Auto Encoder that uses the famous UNet architecture to predict the information importance of the scenario beforehand to speed up and enhance the patrolling coverage. This VAE-UNet model takes advantage of realistic simulators and known data from water resources to conform to a prior-fed information model.

From the results, it is possible to conclude that the performance of the VAE-UNet is capable of surpassing well-established approaches such as rKNN and GPs in accuracy. In addition to that, an exceptional speedup of 10 less computational time has been observed with respect to GPs, the second-best approach. This property proves that this approach can be used in even more critical time-sensitive path planning than the marine patrolling problem or in larger measurement campaigns. With respect to the patrolling performance, the DRL has been seen as a flexible and efficient approach that is capable of addressing the cooperation of multiple agents in a model-free manner and the long-term objective optimization. The DRL also benefits from the VAE-UNet model that allows a 30% faster minimization of the patrolling score $w \times (\omega_0 + \omega_1)$.

Future lines of work should address two main aspects of model estimation. On the one hand, it is necessary to further study the generalization ability of the VAE-UNet model when facing information that is much different from the original simulation. While the simulations can be realistic to some extent and the VAE-UNet has been proven to estimate even when some dynamics are incorporated to the scenario, it is still open how robust the approach is to failures and information noise. However, it can be interesting to incorporate recurrent mechanisms into the model to predict the importance in the future, not only in the current model. Long-Short-Term Memory Cells (LSTMs) inside of the latent space of the architecture are expected to be useful in future lines of work.

With respect to future lines of improvement in patrolling policies, it is still an open challenge to address two well-established paradigms in the informative path planning literature that have not addressed the Patrolling task: (I) The Multi-Objective patrolling and (II) The heterogeneous-fleet patrolling. In the first case, more than one importance criterion must be optimized, in which the optimal solution is not a single policy but a Pareto front of policies. In the latter case, the agents pursue a cooperative goal, but with different measurement abilities and with different types of sensors like sonars, WQP sensors, or multispectral cameras.

CRediT authorship contribution statement

Samuel Yanes Luis: Conceptualization, Methodology, Validation, Writing – original draft. **Nicola Basilico:** Supervision, Formal Analysis, Methodology, Validation, Writing – review & editing. **Michele Antonazzi:** Formal Analysis, Methodology, Validation, Writing – review & editing. **Daniel Gutiérrez-Reina:** Supervision, Conceptualization, Methodology, Funding acquisition, Project administration, Writing – review & editing. **Sergio Toral Marín:** Supervision, Conceptualization, Methodology, Funding acquisition, Project administration, Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Samuel Yanes Luis reports article publishing charges was provided by University of Seville. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Arzamendia, M., Gregor, D., Gutierrez-Reina, D., & Toral, S. (2019). An evolutionary approach to constrained path planning of an autonomous surface vehicle for maximizing the covered area of Ypacarai Lake. *Soft Computing*, *23*(5), 1723–1734.

Arzamendia, M., Gutierrez, D., Toral, S., Gregor, D., Asimakopoulou, E., & Bessis, N. (2019). Intelligent online learning strategy for an autonomous surface vehicle in lake environments using evolutionary computation. *IEEE Intelligent Transportation Systems Magazine*, *11*(4), 110–125.

Baron, J. S., Poff, N. L., Angermeier, P. L., et al. (2002). Meeting ecological and societal needs for freshwater. *Ecological Applications*, *12*(5), 1247–1260. http://dx.doi.org/10.1890/1051-0761(2002)012[1247:MEASNF]2.0.CO;2.

Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, & K. Weinberger (Eds.), *Advances in neural information processing systems, Vol. 24*. Curran Associates, Inc.

Chevaleyre, Y. (2004). Theoretical analysis of the multi-agent patrolling problem. In *Proceedings. IEEE/WIC/aCM international conference on intelligent agent technology, 2004.(IAT 2004)*. (pp. 302–308). IEEE.

Gatys, L. A., Ecker, A. S., & Bethge, M. (2015). A neural algorithm of artistic style. arXiv:1508.06576,

Hasselt, H. v., Guez, A., & Silver, D. (2016). Deep reinforcement learning with double Q-learning. In *Proceedings of the thirtieth AAAI conference on artificial intelligence* (pp. 2094–2100). AAAI Press.

Jara Ten Kathen, M., Peralta, F., Johnson, P., Jurado Flores, I., & Gutiérrez Reina, D. (2024). AquaFeL-PSO: An informative path planning for water resources monitoring using autonomous surface vehicles based on multi-modal PSO and federated learning. *Ocean Engineering*, *311*, Article 118787. http://dx.doi.org/10.1016/j.oceaneng.2024.118787.

Julian, K. D., & Kochenderfer, M. J. (2018). Distributed wildfire surveillance with autonomous aircraft using deep reinforcement learning. ArXiv arXiv:1810.04244.

Kathen, M. J. T., Flores, I. J., & Reina, D. G. (2021). An informative path planner for a swarm of ASVs based on an enhanced PSO with Gaussian surrogate model components intended for water monitoring applications. *Electronics*, *10*(13), 1605.

Kathen, M. J. T., Peralta, F., Johnson, P., Jurado Flores, I., & Gutiérrez Reina, D. (2023). Performance evaluation of AquaFeL-PSO informative path planner under different contamination profiles. In G. Rivera, L. Cruz-Reyes, B. Dorronsoro, & A. Rosete (Eds.), *Data analytics and computational intelligence: novel models, algorithms and applications* (pp. 405–431). Cham: Springer Nature Switzerland, http://dx.doi.org/10.1007/978-3-031-38325-017.

Kohl, S. A. A., Romera-Paredes, B., Meyer, C., De Fauw, J., Ledsam, J. R., Maier-Hein, K. H., et al. (2019). A probabilistic U-net for segmentation of ambiguous images. arXiv:1806.05034 [cs, stat].

Krause, A., & Golovin, D. (2014). Submodular function maximization. In L. Bordeaux, Y. Hamadi, & P. Kohli (Eds.), *Tractability: practical approaches to hard problems* (pp. 71–104). Cambridge University Press.

Krause, A., & Guestrin, C. (2007). Near-optimal observation selection using submodular functions. In *Proceedings of the 22nd national conference on artificial intelligence - volume 2* (pp. 1650–1654). AAAI Press.

Mnih, V., Kavukcuoglu, K., Silver, D., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533. http://dx.doi.org/10.1038/nature14236.

Peralta, F., Reina, D. G., Marín, S. L. T., Gregor, D. O., & Arzamendia, M. (2021). A Bayesian optimization approach for water resources monitoring through an autonomous surface vehicle: The Ypacarai lake case study. *IEEE Access*, *9*(1), 9163–9179. http://dx.doi.org/10.1109/ACCESS.2021.3050934.

Piciarelli, C., & Foresti, G. L. (2019). Drone patrolling with reinforcement learning. *ACM International Conference Proceeding Series*, *1*(1), 1–6. http://dx.doi.org/10.1145/3349801.3349805, ISBN: 9781450371896.

Popovic, M., Vidal-Calleja, T., Hitz, G., et al. (2020). An informative path planning framework for UAV-based terrain monitoring. *Autonomous Robots*, *44*, 889–911. http://dx.doi.org/10.1007/s10514-020-09903-2.

Qiu, S., Ren, H., Li, H., Tao, R., & Zhou, Y. (2021). An improved particle number-based oil spill model using implicit viscosity in marine simulator. In Y. Dimakopoulos (Ed.), *Mathematical Problems in Engineering*, *2021*, Article 5545051. http://dx.doi.org/10.1155/2021/5545051.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. CoRR arXiv:1505.04597.

Sánchez-García, J., García-Campos, J., Arzamendia, M., Reina, D., Toral, S., & Gregor, D. (2018). A survey on unmanned aerial and aquatic vehicle multi-hop networks: Wireless communications, evaluation tools and applications. *Computer Communications*, *119*, 43–65. http://dx.doi.org/10.1016/j.comcom.2018.02.002.

Sawada, R., Sato, K., & Majima, T. (2021). Automatic ship collision avoidance using deep reinforcement learning with LSTM in continuous action spaces. *Journal of Marine Science and Technology*, *26*(2), 509–524. http://dx.doi.org/10.1007/s00773-020-00755-0.

Schaul, T., Quan, J., Antonoglou, I., & Silver, D. (2015). Prioritized Experience Replay. http://arxiv.org/abs/1511.05952.

Scott, D. W. (2012). Multivariate density estimation and visualization. In *Handbook of computational statistics: concepts and methods* (pp. 549–569). Berlin, Heidelberg: Springer Berlin Heidelberg, http://dx.doi.org/10.1007/978-3-642-21551-3_19.

Song, D., Gan, W., Yao, P., Zang, W., & Qu, X. (2023). Surface path tracking method of autonomous surface underwater vehicle based on deep reinforcement learning. *Neural Computing and Applications*, *35*(8), 6225–6245. http://dx.doi.org/10.1007/s00521-022-08009-3.

Stern, R., Sturtevant, N. R., Felner, A., Koenig, S., Ma, H., Walker, T. T., et al. (2019). Multi-agent pathfinding: Definitions, variants, and benchmarks. In *Symposium on combinatorial search*.

Sun, S., Li, T., Chen, X., Dong, H., & Wang, X. (2024). Cooperative defense of autonomous surface vessels with quantity disadvantage using behavior cloning and deep reinforcement learning. *Applied Soft Computing*, *164*, Article 111968. http://dx.doi.org/10.1016/j.asoc.2024.111968.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (2nd ed.). The MIT Press.

Wang, F., Eljarrat, A., Müller, J., Henninen, T. R., Erni, R., & Koch, C. T. (2020). Multi-resolution convolutional neural networks for inverse problems. *Scientific Reports*, *10*(1), 5730. http://dx.doi.org/10.1038/s41598-020-62484-z.

Wang, Z., Freitas, N. d., & Lanctot, M. (2015). Dueling network architectures for deep reinforcement learning. CoRR, arXiv:1511.06581.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics*, *1*, http://dx.doi.org/10.2307/3001968.

Yanes, S., Gutiérrez-Reina, D., & Toral Marín, S. (2021). A dimensional comparison between evolutionary algorithm and deep reinforcement learning methodologies for autonomous surface vehicles with water quality sensors. *Sensors*, *21*(8), http://dx.doi.org/10.3390/s21082862.

Yanes, S., Reina, D. G., & Marín, S. L. T. (2020). A deep reinforcement learning approach for the patrolling problem of water resources through autonomous surface vehicles: The Ypacarai lake case. *IEEE Access*, *6*(1), 63. http://dx.doi.org/10.1109/ACCESS.2020.3036938.

Yanes, S., Reina, D. G., & Marín, S. L. T. (2021). A multiagent deep reinforcement learning approach for path planning in autonomous surface vehicles: The Ypacaraí lake patrolling case. *IEEE Access*, *9*, 17084–17099.

Yanes Luis, S., Gutiérrez-Reina, D., & Toral Marín, S. (2023). Censored deep reinforcement patrolling with information criterion for monitoring large water resources using Autonomous Surface Vehicles. *Applied Soft Computing*, *132*, Article 109874. http://dx.doi.org/10.1016/j.asoc.2022.109874.

Yi, K., Guo, Y., Fan, Y., Hamann, J., & Wang, Y. G. (2020). CosmoVAE: Variationa autoencoder for CMB image inpainting. arXiv:2001.11651 [astro-ph, stat].

Zhang, A., Wang, W., Bi, W., & Huang, Z. (2024). A path planning method based on deep reinforcement learning for AUV in complex marine environment. *Ocean Engineering*, *313*, Article 119354. http://dx.doi.org/10.1016/j.oceaneng.2024.119354.