IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 8 April 2024, accepted 15 May 2024, date of publication 20 May 2024, date of current version 29 May 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3402980

RESEARCH ARTICLE

Informative Deep Reinforcement Path Planning for Heterogeneous Autonomous Surface Vehicles in Large Water Resources

ALEJANDRO MENDOZA BARRIONUEVO, SAMUEL YANES LUIS, DANIEL GUTIÉRREZ REINA, AND SERGIO L. TORAL MARÍN
Department of Electronic Engineering, University of Seville, 41003 Seville, Spain

Corresponding author: Alejandro Mendoza Barrionuevo (amendoza1@us.es)

This work was supported in part by the Spanish Ministry of Science Ministerio de Ciencia, Innovación y Universidades (MCIN)/Agencia Estatal de Investigación (AEI)/10.13039/501100011033 under Grant TED2021-131326B-C21; in part by the European Union with ''NextGenerationEU/Plan de Recuperación, Transformación y Resiliencia (PRTR)'' funds, the Spanish Ministry of Science MCIN/AEI/10.13039/501100011033, under Grant TED2021-131326A-C22; in part by the European Union with ''Next Generation EU/PRTR'' funds, the Spanish Ministry of Science PID2021-126921OB-C21, under Grant TED2021-131326A-C22; in part by the European Union with ''Next Generation EU/PRTR'' funds; in part by the Spanish Ministry of Science PID2021-126921OA-C22 under Grant TED2021-131326A-C22; in part by European Union with ''Next Generation EU/PRTR'' funds; in part by the ''Junta de Andalucía: Consejería de Universidad, Investigación e Innovación'' through the project ''Monitorization of Environmental Dangers with Unmanned Surface Agents: (MEDUSA)'' under Grant PCM_00019; and in part by the research network REINFORCED http://grupos.us.es/reinforced/, funded by the University of Seville, through the ''Plan Propio de Investigación.''

ABSTRACT Water contamination in extensive aquatic resources is a pressing issue, especially during current drought conditions across the world. To adress this, a novel approach involving a heterogeneous sensing capabilities fleet of four autonomous surface vehicles is introduced for efficient contamination mapping. To reduce costs, vehicles may be equipped with low quality sensors meaning measurements reliability differs between vehicles and affects model accuracy. The diverse sensing capabilities are characterized by a wide range of sensor standard deviations, addressing the applicability of the framework in real-world scenarios with commercial sensors. This research leverages Gaussian Processes to accurately model spatial distribution of contamination, integrating measurements from the vehicles to understand contamination patterns comprehensively. Additionally, an informative path planning strategy is introduced based on a centralized neural network which implements a Double Deep Q-Learning algorithm, driving the decision-making process of all agents. Effective learning hinges on accurately defining the observation and reward functions, for which several proposals will be compared. These tailored definitions are essential for guiding the learning process, and minimizing the error towards the main goal: to obtain the best possible contamination model. Remarkably, the proposed system demonstrates superior performance in Ypacaraí Lake scenario, surpassing traditional heuristics like lawn mower or particle swarm optimization by up to 82% in reducing mean squared error in highly contaminated regions for several combinations of agents.

INDEX TERMS Autonomous vehicles, deep reinforcement learning, environmental monitoring, heterogeneous multirobot systems, informative path planning.

I. INTRODUCTION

Water resources, such as rivers, seas, or lakes, play a vital role in preserving life and sustaining the diverse ecosystems and societies of the planet. Unfortunately, these invaluable resources are increasingly threatened by pollution, drought, climate change, and uncontrolled overexploitation [1]. These factors result in a serious deterioration of water quality, as in the case of Ypacaraí Lake, the largest body of water in Paraguay. It covers an area of roughly 60 km^2, and is surrounded by important cities such as San Bernardino and Areguá. Despite its importance, the lack of connections

The associate editor coordinating the review of this manuscript and approving it for publication was Wai-Keung Fung.

© 2024 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

to external water flows makes it vulnerable to human-induced problems. Untreated wastewater, agricultural pollution, or unregulated development cause excessive nutrient enrichment in the lake, known as eutrophication. This produces uncontrolled growth of harmful organisms, depletion of oxygen levels in deep water, and mass deaths of aquatic species, which endanges its ecosystem's sustainability [2]. Urgent intervention is required to prevent further damage.

Addressing this problem requires a reliable model based on water quality parameters. However, due to the lake's size, gathering accurate data is challenging, especially in the inner areas. The manual sampling method is resource intensive and inefficient. To overcome this challenge, more advanced and efficient monitoring mechanisms need to be implemented. The use of Autonomous Surface Vehicles (ASVs) is gaining importance in aquatic environments [3]. Equipped with water quality sensors, as shown in Fig. 1, they offer an affordable and versatile solution to collect samples from any part of the lake. The challenge here relies on an efficient Informative Path Planning (IPP) for ASVs, where they use sensor data to plan routes, considering factors like data importance, obstacle avoidance, and battery efficiency. As will be detailed in Section III, IPP is a strategy to optimize the paths of autonomous agents to efficiently collect data in environments, focusing on areas of highest informational value.

When dealing with large bodies of water, a fleet of ASVs is essential. Although each ASV presents a cost-effective alternative individually, the collective expense grows with the size of the fleet. A significant part of this cost comes from onboard measurement sensors, whose prices and quality vary, often dictated by budgetary constraints. This budgetary variability results in a heterogeneous sensing capabilities fleet where some ASVs are equipped with high-quality and expensive sensors, offering precise measurements with a low standard deviation, indicating high reliability in the data. On the contrary, other ASVs are outfitted with more economical sensors that, while cost-effective, offer data with a higher standard deviation, denoting less reliability and precision. This heterogeneity in sensor quality plays a crucial role in IPP, since it directly affects each vehicle's ability to tune and update the contamination model. The challenge of this work is to synergize these various inputs to obtain the best possible contamination model, especially in heavily contaminated locations. To do this, it must be ensured that the data collection path for each ASV is efficiently planned to compensate for the shortcomings of the sensors and take advantage of their strengths. To maximize the effectiveness of the ASV fleet in characterizing contaminated areas, a targeted approach is employed: ASVs with best quality sensors are designated to the most contaminated zones, ensuring precise data collection where it is crucial, and the others should cover less critical areas. This strategy ensures detailed and accurate mapping of the most affected areas, while efficiently utilizing the fleet's diverse sensor capabilities. This requires intricate and active coordination among vehicles and a continuous exchange of information. Due to the large amount of information to be processed, IPP can be considered NP-hard [4], so their scalability and dimensionality make an approach based on Machine Learning one of the best alternatives, proven in cases such as the salesman problem [5].

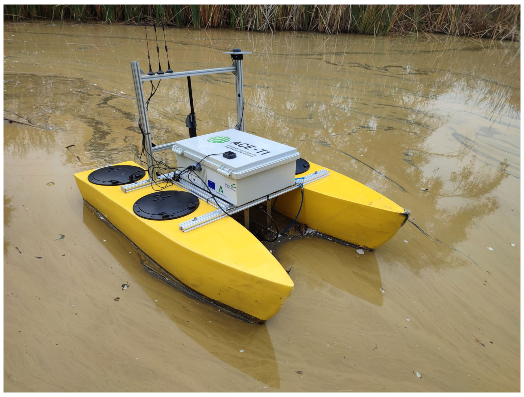
FIGURE 1. Example of an ASV prototype for the measurement of water quality parameters, equipped with GPS for path planning.

This work will address the problem of finding an efficient Ypacaraí Lake monitoring algorithm for heterogeneous ASVs using Deep Reinforcement Leaning (DRL) techniques, more specifically the Deep Q-Learning algorithm [6]. In pursuit of an efficient and scalable solution, the proposed strategy involves employing a single centralized neural network that coordinates a diverse fleet of vehicles. However, given the egocentric nature of the agents, if desired each could run its own on-board neural network simply by requesting the information of current state. The crux of the proposed approach lies in the design of a sophisticated scenario observation function. This function is responsible for feeding the neural network with comprehensive structured input. These inputs take the form of multiple matrices, representing the collected data, information to differentiate agents from each other and their positions, as well as the discretized lake map. As a result, the framework will estimate the most appropriate movement based on a tailored reward function. The design of the reward function is critical as it aims to encourage actions that are predicted to reduce the model's error relative to the ground truth, even though the actual error is not directly accessible. It must be based on other indirect accessible indicators. Water quality parameter (WQP) samples will be taken at each position and treated as Gaussian random variables. A spatial correlation matrix, using the Gaussian Process (GP) with RBF kernel [7], captures the statistical relationship between these samples. This method is especially useful when working with noisy sensors, since depending on the standard deviation of the measurement, the model will take it into account to a greater or lesser degree. Consequently, the IPP task should involve a sequential determination of the next sampling location that maximizes information gain.

In summary, the main contributions of this work are:

i) The development of a model-free DRL framework to solve the IPP problem for a heterogeneous multi-agent fleet of ASVs according to the quality of their sensors.
ii) The definition of two tailored observation functions and two reward functions to follow a policy which minimizes the error of the model.
iii) A comprehensive evaluation and comparison with other path planners for water quality monitoring approaches, addressing the scalability problem in the context of the Ypacaraí Lake.

The structure of the article is as follows: it starts with a review of related research in Section II. Section III formally outlines the Informative Path Planning problem and explains how a simulated space with Ground Truth data is generated for sampling. Section IV details the methodology, introducing the Gaussian Process and describing the Deep Reinforcement Learning algorithm. Section V presents, analyzes, and compares diverse results and simulations, including comparisons with other path planning techniques. The article concludes in Section VI with some conclusions and future research directions.

II. RELATED WORK

The application of autonomous vehicles in the field of scientific applications is increasingly emerging [8], [9], thanks to technological advances such as machine and deep learning. These unmanned vehicles find applications in scientific fields such as subsea pipeline inspections [10], water quality monitoring [11], bathymetry [12], and agriculture [13], among others. They enable more efficient, accurate, and sustainable environmental monitoring and data collection. However, optimizing their trajectories is essential due to battery limitations.

Planning these paths is not a trivial task, so the optimization algorithms to solve this problem may be diverse, including bio-inspired algorithms [3], [14], Bayesian Optimization [15], or DRL algorithms [16]. DRL has proven to be particularly effective in the context of path planning for multi-agent agent case, such in [17], where two approaches are suggested: a Double Deep Q-Network and a Dueling Architecture for Q-values optimization. The suggested methods are relying on a centralized Deep Q-Network, employing convolutional neural networks (CNN), with an individual fully-connected layer for up to three agents, following the assumption that agents possess identical properties and abilities. The objective is to maximize area coverage before the vehicles' battery runs out. Contrary to the architecture mentioned above, in this work the layers of the proposed network are shared by all the agents, so that the network does not grow even if the number of agents increases.

In [18], a completely different method is proposed, introducing a Bayesian Optimization method for predicting multiple water quality parameters in Ypacaraí Lake using a single ASV. The approach employs Gaussian Processes (GPs) and combines different acquisition functions to balance objectives and improve overall model performance. However, minimizing all functions simultaneously is challenging, as they often are counterbalanced. Simulations demonstrate that more collected samples lead to reduced Mean Squared Error, supporting the suitability of the GP regression model to effectively model water quality variables. Inspired by these compelling findings, the GPs will be adopted in the current article to harness their proven capabilities.

In [19], a monitoring system for Ypacaraí Lake is developed that uses a fleet of ASVs controlled by an improved Particle Swarm Optimization (PSO) algorithm based on a GP as surrogate model. With PSO, a population of potential solutions, represented as particles, moves through the search space to find the best solution. Each particle adjusts its position and velocity based on its own experience and the experiences of its neighbors. The algorithm aims to find the global optimum in the search space by iteratively updating the particles positions and velocities. In [20], a heterogeneous PSO-based swarm is proposed which uses a topological k-nearest neighbor, capable of switching between exploration or exploitation dynamics with an adaptive repulsion parameter. Despite the fact that PSO is an heuristic that excels in simpler, static optimization tasks where the primary goal is to find a good-enough solution efficiently, the methodology employed in this paper choose DRL over PSO for autonomous path planning due to the DRL's ability to learn and adapt to the environment through its interactions, which makes it better suited for problems involving complex sequential decision-making, like autonomous navigation.

The work [13] introduces an adaptive path planning method for precise agricultural monitoring with Unmanned Aerial Vehicles (UAVs). It used a deep learning model to optimize flight paths, detecting for necessary high-resolution semantic segmentation. This allows depth inspections at low altitude to be performed only when necessary, conserving battery power. In [21], a cooperative IPP strategy is presented that uses UAV squads for terrain monitoring, employing DRL to achieve a collective mission purpose in a efficient manner. In [22], DRL works in conjunction with local Gaussian processes to obtain a contamination map, achieving great results. Despite the similarities, unlike this previous work, the one proposed in this paper takes noise into account in the agent measurements, making the optimization of the routes and the achievement of a good contamination map a more complex task. The demonstrated efficiency of DRL in IPP and its successful application in conjunction with Gaussian Processes, as evidenced in the referenced studies, provide a solid foundation for its utilization in the methodologies employed in this research.

In [23], innovative algorithms are proposed for heterogeneous multirobot systems (HMRS) to enhance their performance in adversarial catching tasks, such as security and rescue missions. These algorithms employ actor-critic DRL approaches, coupled with asymmetric self-play and curriculum learning strategies, to promote effective cooperation among agents within and across groups, facilitating

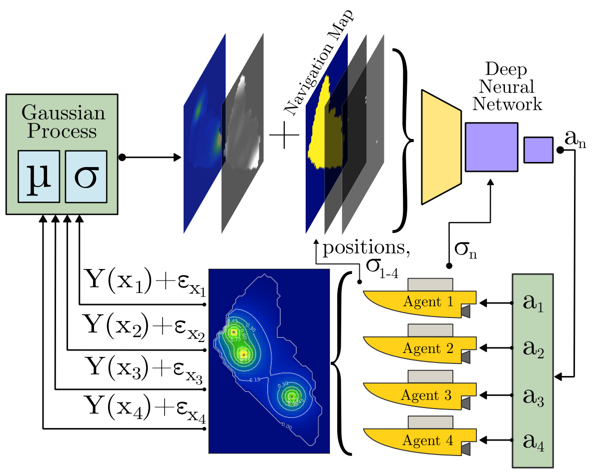
Gaussian Process
μ
σ
Navigation Map
Deep Neural Network
a_n
positions, $\sigma_{1\text{-}4}$
σ_n
$Y(x_1)+\varepsilon_{x_1}$
$Y(x_2)+\varepsilon_{x_2}$
$Y(x_3)+\varepsilon_{x_3}$
$Y(x_4)+\varepsilon_{x_4}$
Agent 1
Agent 2
Agent 3
Agent 4
a_1
a_2
a_3
a_4

FIGURE 2. **Conceptual image of the framework proposed in this work. The deep neural network is in charge of the decision making process of the actions performed by the agents. In each movement, each agent takes a noisy sample of the ground truth, which is used to update the contamination model estimated by a Gaussian Process, which knows the standard deviation associated to the sensor of each agent. The neural network input is formed by the contamination map estimation and its uncertainty, the navigation map and the positions of the agents, whose values also represent the sensor quality by its standard deviation. The standard deviation of the observing agent is also an additional input that is introduced to the network directly in the dense layer.**

effective teamwork in complex scenarios with real-world constraints. In contrast to that previous work, in the present paper agents can take the same actions and share the joint information, but having different measurement capabilities they cannot contribute equally to the final goal: to obtain the best model. These differences in capabilities are introduced through the observation function, which indicates the quality of the observing agent and that of its partners.

Given the complexity of the stated problem, the basis of the framework proposed (see Fig. 2) combines DRL with a GP as surrogate model, as seen in [18] and [19], for a multi-agent fleet of four ASVs. The use of representative matrices as states to guide the actions of agents is inspired by [22]. Two observation functions are proposed that offer deterministic knowledge of the environment and self-perception of the quality and position of the agent and of the rest. In addition, two reward functions based on these observations are proposed. Although previous works has shown potential, they has primarily focused on homogeneous multirobot systems, leaving a research gap for the task of monitoring water resources with heterogeneous ASVs. They are cost-effective, but sensors are one of the most expensive components. As a result, the fleet may have sensors of differing standard deviation, resulting in a HMRS. Current literature overlooks sensor quality and its impact on modeling. This approach introduces a new consideration by incorporating the inherent noise associated with sensor measurements into the decision-making process of the agents. In this sense, the associated sampling error directly influences the accuracy of the contamination model estimation obtained and the agent's ability to influence the surrogate model, with lower standard deviation measurements being more significant. In the proposed framework, agents adopt different policies to optimize global information collection based on the quality of their respective sensors, a facet that has not yet been addressed in previous work. Moreover, this is a highly generalizable perspective, as it allows the incorporation of sensors in a wide range of standard deviation. This implies that agents will have to follow different policies depending on their measurement capability to obtain the highest possible collective reward. Using DRL in HMRS applications presents challenges outlined in [8]: training complexity due to collaboration patterns between teams and within a team, and uncertainty in real-world scenarios making it difficult to flexibilize training to cope with unexpected events. To address this, a noise-adaptive centralized network is proposed that adapts agent policies based on sensor quality to simplify training. This approach enables future real-world implementation for monitoring with ASVs.

III. STATEMENT OF THE PROBLEM

IPP can be described as a computational strategy or algorithm designed to plan routes in a way that maximizes the acquisition of valuable information about the environment. In this application, the IPP focuses on optimizing data collection by guiding vehicles to locations where the model error $E(t)$ decreases with respect to the real information. In the multi-agent assumption, the objective is to find a set of trajectories $\Psi := [\psi_1, \psi_2, \ldots, \psi_N]$ optimized to collectively minimize the model error (1), while respecting the constraint that they do not cause collisions between agents or between agents and obstacles.

$$\Psi^* = \arg\min_{\Psi} \sum_{\psi \in \Psi} E(t)_\psi \tag{1}$$

Under this strategy, a trajectory can be defined by a sequence of visited waypoints at which to take measurements X^n for each agent n in a fleet of N agents.

A. SCENARIO AND OTHER ASSUMPTIONS

With the purpose of training agents for real-world situations, it is necessary to develop a simulator that accurately reflects the real constraints. Throughout this research, several assumptions are considered. At each time step t, every vehicle takes one action from the set of possible actions A, which corresponds to the eight motion directions $[S, SE, E, NE, N, NW, W, SW]$. Only actions that go to a safe place are allowed, by means of a low level control that ensures it without any error. With each action taken, the agents move a fixed distance of $d_{straight}$ or $d_{diagonal}$ in the selected direction, depending on whether the action follows a straight or diagonal direction, and collect a water measure at their new location, repeating iteratively. All movements are assumed to take the same time to be performed, so all ASVs move synchronously, although the Euclidean distance between two consecutive waypoints is greater for diagonal movements. Trips will be limited to a maximum distance of

d_{max}, beyond which the battery energy reaches a minimum safety reserve for the vehicle to return to its base, so it will be eliminated from the map and it will not be able to take any action. This research assumes that vehicles can be heterogeneous from the sensors quality point of view. They will have the same motion capabilities, but the noise of the measurement sensors may be different. The measures collected are taken from a ground truth, denoted as Y, which is generated from a static scalar field fixed before launching the mission. However, the field is unknown to the agents, except at the points where they collect samples. The samples model for each agent can be represented as:

$$y_x = Y(x) + \epsilon_x, \tag{2}$$

where $x = \left[x_{lat}, x_{long}\right]$ is the position of the vehicle taking the sample and ϵ_x is a noise depending on the standard deviation associated with the vehicle sensor. In this way, ϵ_x behaves as a uniform probability distribution for each agent over the course of an episode. The function $Y : \mathbb{R}^2 \rightarrow \mathbb{R}$ returns the ground truth value in that position. To facilitate training and comparisons across different scenarios, the values of Y are normalized within the range of 0 to 1, referred to as the Normalized WQP. The measurement values do not represent any particular WQP, but can describe any of them, as will be seen in Section III-B. Under these circumstances, this will mean that a value of 0 indicates a reasonable value for the WQP, while a value of 1 indicates an out-of-normal parameter value, and is associated with high contamination. In all simulations, it will be assumed that the contamination is static throughout the episode, since the dynamics of movement of the parameters to be measured evolve much slower than the measurement process.

To make the simulation more manageable and efficient, the real navigation map is discretized into a homogeneous grid through which the agents will move, turning into a cell map, such that $\mathcal{M} : \mathbb{R}^2 \rightarrow \mathbb{R}$. Non-navigable areas, such as obstacles or shorelines, are indicated with $\mathcal{M}(x) = 0$, and it will be assumed that vehicles cannot sample in these locations. There will be no moving obstacles in the scenario other than the possible interaction of the vehicles with each other, thus, agents cannot occupy the same cell. To ensure the safety of the four vessels, the deployment positions will always correspond to four fixed locations in the south of the map (near the San Blas Pier), from which the vehicles can depart in no particular order, as seen in Fig. 3. Additionally, vehicles are not required to finish their trip at the same location where they departed. It is assumed that they still have sufficient remaining autonomy to return to shore from any location at the conclusion of their measurement period.

B. GROUND TRUTHS MODELS

In the context of contamination modeling, ground truth models play a crucial role in representing the true underlying state of contamination in natural water resources. It has been previously demonstrated in research such as [24] that some physical-chemical water parameters such as pH,

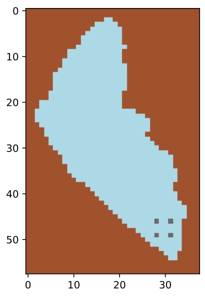

FIGURE 3. Discretized Ypacaraí Lake map with the four deployment positions marked.

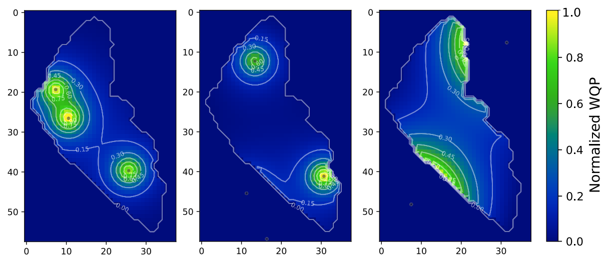

FIGURE 4. Example of several contamination maps of Ypacaraí Lake obtained through a Shekel function.

electrical conductivity, dissolved oxygen levels, ORP or particle concentrations (nutrients, heavy metals, bacteria) follow a smooth spatial distribution. This research will focus on contamination models that follow this pattern. To create a simulation model that matches the behavior of these parameters, a Shekel function will be used [25]. It is one of such mathematical models commonly employed for simulation purposes in various fields, including contamination analysis. This is a mathematical benchmark function that allows researchers and practitioners to generate known and controlled environments with specific characteristics. The function has two main parameters that determine the positions of the peaks in the input space and the prominence of the peaks, through the c elements. In this work, this ground truth generator will provide randomly generated Y scalar functions to the learning algorithm preprocessed to fit the Ypacaraí Lake map, as seen in Fig. 4. The highest values of each peak will be homogeneously distributed over the map and may be found outside the navigable areas.

IV. METHODOLOGY

In view of the statements and constraints in Section III, the problem can be formulated as a Partially Observable Markov Decision Process (POMDP). These decision processes can be scaled up to the multi-agent case by establishing a partially observable state s^t only accessible by an observer n, whereby the observation function can be set as $o_n^t = \mathcal{O}(s^t)$. The primary objective in a POMDP is to find the optimal policy π^* that maximizes the expected accumulated reward over time, that is, a mapping from an observation o_n^t of the state space to an action such that $\boldsymbol{a}_n^{t+1} = \pi^*(o_n^t)$. The accumulated reward can be denoted as $\sum_{t=0}^{T} R(s^t, \boldsymbol{a}_n^t)$, limited by a determined optimization time T for each agent $n \in [1, N]$ belonging to the fleet. The complete state s_t contains comprehensive information about the environment. However, the observation of agents does not have direct access to all information, such as the ground truth scalar field. Instead, agents only observe a subset of this information, such as the navigation map $\mathcal{M}$, the position of the agents (observer and non-observers), or the model generated from the measurements obtained up to a given time, among others.

This section will explain in detail the methodology used to i) obtain a contamination model from the samples taken by using GPs, and ii) train the movement policy of the ASVs using DRL.

A. GAUSSIAN PROCESS AS SURROGATE MODEL

A Gaussian Process is a probabilistic model that represents a distribution over functions, treating functions as joined random variables [7]. Unlike parametric models that have a fixed set of parameters, a GP is a non-parametric modeling approach characterized by a mean function $\mu(x)$ and a covariance (or kernel) function $k(x, x')$, allowing it to capture complex patterns and uncertainties in data. Therefore, they provide a powerful framework for modeling spatial dependencies and uncertainties, such in contamination maps, as they can be employed to predict values of an unknown function $Y(x)$. Thus, the prediction using a GP can be expressed as:

$$Y(x) \sim \mathcal{GP}(\mu(x), k(x, x')). \tag{3}$$

The covariance function, on the one hand, captures the underlying patterns and relationships between the data points, i.e., is a way of expressing how similar the function values are at different inputs x and x', as seen in [7]. On the other hand, the mean function ensures that the model is consistent with the observed data distribution and is a way of expressing the prior knowledge about the function.

In the same way as [26], in this paper GPs are employed as online prediction methods to generate contamination models. These models have a dual purpose: firstly, the main reason for the trips, which is to capture an estimate of the spatial structure of lake contamination levels, and secondly, it serves as an observer of the real information not accessible in the POMDP, the ground truth scalar field. To predict the values at each location on the map, the function must be pre-fitted with sets of input and output values $\{X, \boldsymbol{y}\}$. In the context of measuring WQPs, X is the set of sampling coordinates, and $\boldsymbol{y}$ is the set of measurements of WQPs obtained at each of these locations.

Given that the selection of the kernel function plays a crucial role in the formulation of hypotheses on spatial correlations, it is imperative to choose a kernel that aligns well with the characteristics of the WQPs under consideration. As presented in [26], a strong initial option is to employ a Radial Basis Function (RBF) type kernel, also known as the Gaussian kernel, as it is particularly suitable for scenarios where smooth and continuous spatial variations in contamination levels are anticipated. This kernel decreases the correlation exponentially as the distance between the samples increases, adjusting the degree of influence through a hyperparameter ℓ called length-scale. In the context of pollution modeling, this can be interpreted as the distance over which WQPs levels show a significant correlation, i.e., the spatial spread. For its part, the scale kernel provides additional flexibility through the s parameter, as it allows the amplitude (or variance) of the kernel to be better modeled based on the variability of the data. This influences the smoothness of the covariance function. The optimization of these hyperparameters is performed by maximizing the log marginal likelihood through Adam optimization method, given a set of collected data from the training data $\{X, \boldsymbol{y}\}$ and a prior value of 0 in this case, assuming that the measurements are normalized [7]. During optimization, it is crucial to set reasonable bounds for the hyperparameters (minimum and maximum values) to ensure that they remain within a plausible range and to promote numerical stability and faster convergence.

The evaluation of the covariance function for each pair of points (x, x') can be grouped into a square matrix called covariance matrix $\mathbf{K}$, where each element $\mathbf{K}_{i,j} = k(x_i, x_j)$. In realistic modeling situations, as in this scenario, it is necessary to deal with noisy observations, since there is no access to the values of the functions themselves (2). Assuming that the noise is Gaussian, additive, independent, and identically distributed, it can be modeled by adding a diagonal variance term $\sigma_n^2 I$ to the part of the covariance matrix associated with the observed data points. Thus, the covariance matrix for all possible locations X_* is defined as follows:

$$\mathbf{K} = \begin{bmatrix} K + \sigma_n^2 I & K_* \\ K_*^T & K_{**} \end{bmatrix} = \begin{bmatrix} K(X, X) + \sigma_n^2 I & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix} \tag{4}$$

Usually X_* represents the locations of unknown values, i.e., the locations not visited and to be estimated, but since the measurements taken by the agents in this work have noise it is also necessary to make predictions at the locations where samples have already been taken, so X_* denotes all the locations that can be visited. Hence, all map data can be estimated with the GP by its mean and uncertainty as:

$$\begin{aligned} \mu(X_*) &= K_*^T [K + \sigma_n^2 I]^{-1} \boldsymbol{y} \\ \sigma(X_*) &= K_{**} - K_*^T [K + \sigma_n^2 I]^{-1} K_* \end{aligned} \tag{5}$$

In this research, the standard deviation of the measurement sensors is known. Thus, the diagonal variance terms are not parameters to be adjusted, they are assumed constant and can be adequately modeled such that $\sigma_n^2 = \sigma_{sensors}^2$.

B. DEEP REINFORCEMENT LEARNING

DRL is a branch of machine learning that combines deep learning with reinforcement learning. Deep learning uses neural networks to learn from data, while reinforcement learning guides the learning of agents who make sequential decisions in an environment, as in this proposal, for which they receive feedback in the form of positive or negative rewards [27]. The agent learns a policy π, which maps the best action given a state of the environment. The optimal policy π^* should give actions that result in the highest expected cumulative reward Q(s,a), considering the uncertain nature of the environment [28].

$$\pi^*(s) = \underset{a}{\operatorname{argmax}}\, Q(s, a) \tag{6}$$

In this paper, the Double Deep Q-Learning algorithm (D-DQL) is suggested as a universal and effective framework for optimizing discrete action policies, as seen in [17], [23], and [21]. In D-DQL the idea is to decouple the selection of the best action from the evaluation of that action to mitigate the overestimation bias present in standard DQL, as seen in [29]. D-DQL also demonstrates greater stability during training, and often converges to policies closer to optimality compared to DQL, making it more effective in complex environments, as in this case. To do this, two functions are introduced, and therefore two deep neural networks are used to estimate each of them. The $Q(s, a)$ function is used to select the best action using an epsilon-greedy policy applied to the estimated rewards. To stabilize the training, a duplicate Q function called target $Q^{target}(s, a)$ is employed, which estimates the reward obtained by performing the best action selected by the other network [29]. During training, the Q learning algorithm employs an iterative process to update the Q values based on observed rewards and estimated discounted future rewards, with the goal of minimizing the difference between the network prediction and the actual reward obtained, to converge to an optimal policy. These updates are performed by taking steps to reduce the Bellman error [30], such that:

$$\begin{aligned} Q(s, a) &\leftarrow Q(s, a) \\ &\quad + \alpha \left[R + \gamma \cdot Q^{target}(s', \underset{a'}{\operatorname{argmax}}(Q(s'))) - Q(s, a) \right] \end{aligned} \tag{7}$$

where Q(s,a) is the current Q-value for state s of the environment and the selected action $a = \operatorname{argmax}(Q(s))$, i.e., the expected cumulative reward, α is the learning rate, R is the immediate reward obtained for taking action a in state s, γ is the discount factor that weights future rewards, and $Q^{target}(s', \operatorname{argmax}_{a'}(Q(s')))$ is the expected cumulative reward computed using the target Q-network to evaluate in the next state s' the best action selected for the function Q, as the $\operatorname{argmax}_{a'}$ operation selects the action with the highest Q-value.

As with DRL the Q-function is approximated by a Deep Neural Network (DNN), this can be expressed as $Q(s, a, \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ are the trainable weights of the DNN. The Q-function is estimated by collecting experiences, which are sequences of states (s), actions (a), next states (s'), and rewards (R). These experiences are stored in a prioritized experience-replay buffer to train the parameters $\boldsymbol{\theta}$ of the network with batches where the most informative experiences are the most likely to be selected, as developed in [31]. The parameters are tuned by error backpropagation and taking a stochastic gradient descent step in the direction that reduces the loss $\mathcal{L}$, given the learning rate. The loss function is constructed from the quadratic difference between the current Q estimate and the Q-target value:

$$\begin{aligned} &\mathcal{L}(\boldsymbol{\theta}) \\ &= \left[R + \gamma \cdot Q^{target}(s', \underset{a'}{\operatorname{argmax}}(Q(s'; \boldsymbol{\theta})); \boldsymbol{\theta}^-) - Q(s, a; \boldsymbol{\theta}) \right]^2 \end{aligned} \tag{8}$$

where $\boldsymbol{\theta}^-$ are the frozen parameters of the target network. The special feature of the target network is that its parameters are updated more gradually than the main Q-network. This update is performed using Polyak averaging [32], where its parameters are replaced with those of the main network, depending on a smoothing factor τ that controls the update rate. If this factor is one, the update is known as hard update, directly copying the main network's parameters, while a smaller factor ensures gradual updates, enhancing learning stability.

To achieve an effective learning, an early stage of exploration of the state-action space is necessary, where both good and bad experiences must be stored to avoid bias. In order to balance this exploration-exploitation of the network, a ϵ-greedy policy is employed. Each agent has a probability of ϵ to perform a random action, inducing exploration movements. With a probability of 1-ϵ, the agent tends to select the action that the Q-function suggests as the optimal one, prioritizing exploitation of learned knowledge. This strategy allows a balance between trying out new actions and sticking to what the agent has found to be effective, by trial and error. During the initial stages of training, ϵ is set high to gather a diverse set of experiences, and as training progresses, it is gradually reduced to promote exploitation of knowledge gained from past experiences.

C. OBSERVATION FUNCTION

The observation function, $\mathcal{O}(s)$, is a mapping mechanism that transforms the complete state of the environment into a partial observable state. This enables an agent to interpret and respond effectively, learning to interact effectively with the environment as it extracts the crucial parts of its state. The representativeness of the observation is crucial, as it determines the information available to the agent. The observation process is defined as a function $o_n = \mathcal{O}(s)$, where the state s of an agent n is observed. This work will use a visual representation of the state, since this method has been previously shown to significantly improve performance in similar situations [17], [22]. To do this, as mentioned in Section III, the navigation map and all other information to which the agent has access (it does not have access to the complete state) are discretized and represented in matrices.

This representation proves to be efficient, as CNN excel at extracting features from such visual states, as will be outlined below. Therefore, the observing representation will consist of 5 channel-states, such that:

1) Discretized navigation map $\mathcal{M}$, with 1 value wherever it corresponds to visitable areas, and 0 where it does not.
2) The mean map expected by the Gaussian Process $\mu(X)$, normalized in the range of [0, 1], where X are the visitable locations.
3) Two alternatives are proposed to provide information about the level of environment exploration performed by the agents:
   A) The min-max normalized standard deviation (SD) predicted by the GP $\sigma(X)$, denoted as uncertainty map, previously used in other works [22].
   B) A matrix symbolizing a knowledge map, $K_{map}(X)$, where each pixel has a knowledge value between 0 and 1, representing the understanding of that area. Initially, all pixels are set to 0, indicating unexplored areas. When an agent explores a pixel, the value of the pixel is updated, storing the reliability of the n agent's sensor k^n_{agent}, measured by the inverse of the sensor's standard deviation, scaled between 0.25 and 1. Thus, a higher pixel value means a better knowledge of the area, since higher quality sensor samples have been taken. The value 0.25 ensures differentiation from unvisited pixels (value 0). If a pixel is revisited by a more reliable agent (higher k_{agent}), the value of the pixel is updated to reflect the more accurate knowledge. This setup creates a dynamic map where each pixel's value represents the best-available knowledge of that area, continually improving as more reliable agents visit, as represented in Fig. 5. This alternative to the uncertainty map is proposed due to the fact that constant changes in uncertainty estimation in relation to the measurements made can make it a hesitant reference and affect the exploration of the agents.
4) A matrix with zeroes except in the position of the observing agent n on the map, $\mathcal{P}(n)$. The value stored in the position of the agent is the k^n_{agent}, which allows to differentiate the quality of each agent's sensor by the observation of the status, determining the best action to be taken accordingly.
5) A matrix with zeroes except in the positions of the other n^- active agents, $\mathcal{P}(n^-)$. The values are defined in the same way as for channel 4, storing $k^{n^-}_{agent}$.

In this way, the two suggested approaches to be assessed as observation functions can be described as follows:

$$o_n = \begin{cases} \mathcal{O}_\sigma(s) \text{ where: } s = \langle \mathcal{M}, \mu(X), \sigma(X), \mathcal{P}(n), \mathcal{P}(n^-) \rangle \\ \quad \text{or} \\ \mathcal{O}_K(s) \text{ where: } s = \langle \mathcal{M}, \mu(X), K_{map}(X), \mathcal{P}(n), \mathcal{P}(n^-) \rangle \end{cases} \tag{9}$$

The difference between the two proposed observation functions is, therefore, that one ($\mathcal{O}_\sigma$) introduces in the observation the uncertainty map of the GP, and the other ($\mathcal{O}_K$) an innovation of this paper that has been named knowledge map.

D. REWARD FUNCTION

The reward function evaluates the optimality of actions taken by any agent given a mission objective in a reinforcement learning environment. It assigns a numerical value to each state-action, indicating how beneficial or detrimental the choice of that action was in that specific state. The formulation of the reward function is of utmost importance, since it guides the agent toward the goal that the agent is trying to achieve through action making [28]. This paper proposes the use of three weighting parameters, obtaining two reward functions by combining the first one with each of the other two.

1) GP-MEAN WEIGHTING

As seen in [22], changes in the mean $\mu(X)$ of the GP model serve as a reliable metric for reward adjustments. Thus, this first reward function $r_\mu(s, a)$ will consider as a positive reward the disparity between the mean of the model at two consecutive time steps, such that $\Delta\mu(X) = \sum |\mu'(X) - \mu(X)|$. This formulation is based under the reasonable assumption that obtaining data that change the mean with respect to the previous one implies an improvement in the estimation due to a gain of information. When dealing with heterogeneous agents with different measurement capabilities, the use of GPs will also adjust the model according to the measurement confidence apart from the likelihood of the data. That is, a good agent will cause a greater change in the model than one with a higher SD if they obtain the same measurement. This will encourage the better agents to go to areas where a high change is likely to occur, such as the most polluted areas, as they will get a larger reward than the worst ones.

To reward the change in the model it is necessary to create the concept of influence area $\mathcal{I}$ of the measure taken by the agent. Thus, this reward function will calculate the discrepancy between the models of two consecutive steps only in the radius $R_{influence}$ that delimits the area of influence of the agent. The length of this area is a parameter that is chosen based on evidence of correlation between nearby measurements and the size of contamination peaks. When areas of influence overlap, there can be instances where an agent, despite not taking an optimal action, benefits from the positive action of another, leading to an overvaluation of its contribution. To prevent such non-optimal policies from occurring, it becomes essential to weight rewards according to how much each agent has contributed to the improvement of the model in that area. For this purpose, a weighting w_{SD} is made according to how good each of the overlapping agents is. That is, the lower the standard deviation of its sensor, the higher the percentage of that zone's total reward it will take.

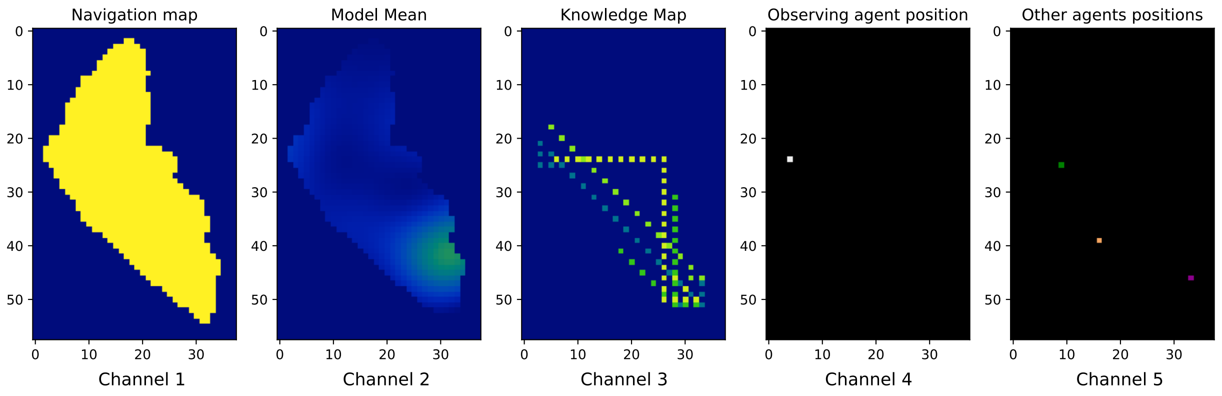

FIGURE 5. Representative example of a state seen through the observation function $\mathcal{O}_K$.

The weighting for an agent n can be denoted as:

$$w_{SD}^n(x) = 1 - \frac{SD_n}{\sum_j^N SD_j} \tag{10}$$

where N are all the active agents that satisfy $x \in \mathcal{I}_N$. Thus, the reward associated with this weighting parameter can be defined as:

$$r_\mu(s, a_n) = \sum_{\boldsymbol{x} \in \mathcal{I}_N} \left[|\Delta\mu(\boldsymbol{x})| \cdot w_{SD}^n(x) \right] \tag{11}$$

2) GP-UNCERTAINTY WEIGHTING

In addition to the use of the mean, an additional reward term is also proposed to benefit those actions that produce a high decrease in the expected uncertainty of the GP, with the aim of promoting exploration. This reward function will be associated with the observation function $\mathcal{O}_\sigma$, which captures the uncertainty map. In the same way as for the mean, the disparity in model uncertainty between two consecutive steps can be defined such that $\Delta\sigma(X) = \sum |\sigma'(X) - \sigma(X)|$. It is important to emphasize that the absolute value is used because, with the adjustment of the internal parameters of the model when taking new samples, the uncertainty could increase or decrease, and both cases can be beneficial for improvement, as seen in [22]. That is, in absolute terms, there may be an increase in uncertainty when new hyperparameters of the GP are found that overcome the previous ones and provide a better model from a likelihood point of view. Similarly to the mean, agents with better measurement sensors will achieve a larger change in uncertainty, as their measurement will be more reliable. Furthermore, the reward received by the agent will also be that which belongs to its area of influence and is weighted by w_{SD}, as seen in (10), when there is overlap between several agents.

$$r_\sigma(s, a_n) = \sum_{\boldsymbol{x} \in \mathcal{I}_N} \left[|\Delta\sigma(\boldsymbol{x})| \cdot w_{SD}^n(x) \right] \tag{12}$$

3) KNOWLEDGE AND MEASURE WEIGHTING

As the uncertainty of the GP during the adjustment process can be very volatile and overly optimistic, another approach is introduced to encourage exploration which used the knowledge map, associated with the observation function $\mathcal{O}_K$. For this reward function there will be a first weighting parameter w_{know} determined from the K_{map} information, and a second weighting w_{meas} depending on the measure obtained by the agent. In this way, the first weighting will be calculated as a function of a value denoted as k_{local}, meaning the local knowledge of the visited location. To obtain it, there can be 2 casuistries:

i) The current cell x has already been visited in the episode: the k_{local} of the cell will be directly the value stored in $K_{map}(x)$.

ii) The current cell x has not yet been visited: in this case, the 3 best neighbors in the adjacent cells of x are searched in K_{map}, i.e., the 3 with the highest values. The average of them is calculated and taken as the k_{local} of the current cell. Therefore, if $K_{map}(x) = 0$ and all its neighbors are zero, $k_{local}(x)$ will be zero as well.

Once the k_{local} of the current cell is obtained, it is compared to whether it is better than the knowledge provided by the agent, k_{agent}. Recall that the knowledge provided by the agent is inversely proportional to its standard deviation scaled in the range $[1, 0.25]$. Hence, in the case of being better, the weighting will be calculated by the difference between the knowledge contributed by the agent and the k_{local}, with an offset of 0.25 due to the scaling range, limiting to 1 the maximum possible. If it is not better, the weighting value will be the complement of the k_{local}, i.e., one minus k_{local}. Thus, if the knowledge is already high in the area, the reward will be low, while if the cell has not been visited it will be the maximum: 1, encouraging the visit of areas with low knowledge.

$$w_{know}^n(x) = \begin{cases} 0.25 + \left(k_{agent}^n - k_{local}(x_n)\right) & \text{if } k_{agent}^n < k_{local}(x_n) \\ 1 - k_{local}(x_n) & \text{if } k_{agent}^n > k_{local}(x_n) \end{cases} \tag{13}$$

On the other hand, the weighting parameter w_{meas} will be directly the measurement taken by the sensor, since it is normalized between 0 and 1 and can be used directly as a

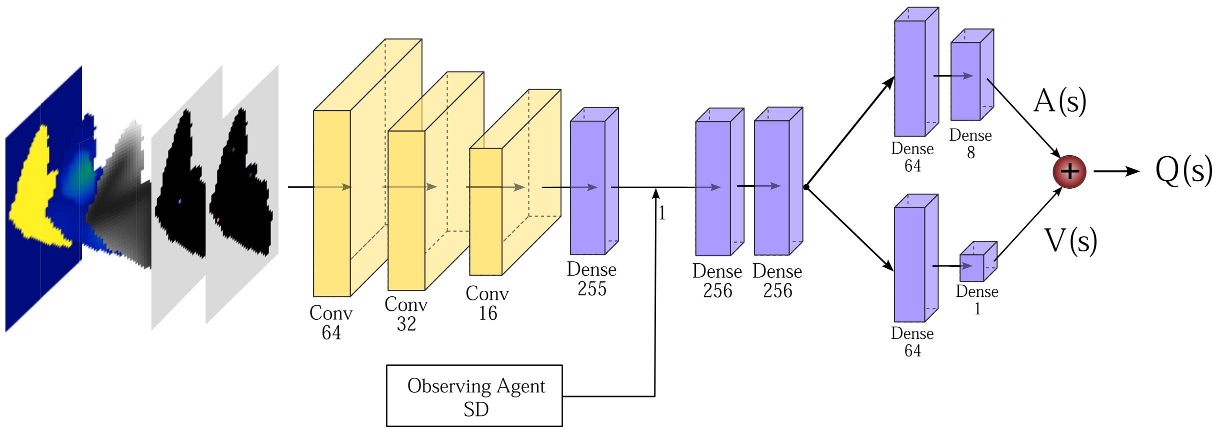

FIGURE 6. Conceptual image of DNN architecture used to estimate *Q*-function. It consists of a first convolutional stage as a feature extractor, followed by a second stage of fully connected layers. It has an additional input to recognize the quality of the observer agent by its SD. The last stage is split in two, which is known as Dueling-DQN. One part estimates the Advantage Function $A(s, a)$ and the other the Value Function $V(s)$. All activation functions between layers are performed by the ReLU function.

multiplication factor. This factor is intended to maximize the zones with high contamination sites, since it is where the change in the model is more abrupt, and more measurements need to be taken to obtain a good model. The combination of the two weightings parameters results in:

$$r_{k,m}(s, a_n) = w_{know} \cdot w_{meas} \tag{14}$$

4) RESULTING REWARD FUNCTIONS

From these reward sub-functions (11)(12)(14), the two final reward functions proposed in this work for every n agent can be defined as:

$$\begin{aligned} R_1(s, a_n) &= C_1 \cdot r_\mu(s, a_n) + C_2 \cdot r_\sigma(s, a_n) \\ R_2(s, a_n) &= C_1 \cdot r_\mu(s, a_n) + C_3 \cdot r_{k,m}(s, a_n) \end{aligned} \tag{15}$$

where C_1, C_2 and C_3 are parameters to be assessed in order to obtain an effective learning during training. In this way, R_1 will be the reward function associated with the observation function $\mathcal{O}_A(s)$, the one with the uncertainty map of the model, and R_2 will be associated with observation function $\mathcal{O}_B(s)$, the one with the knowledge map.

E. ACTION COLLISION MASKING

Although DRL proves to be effective in learning to avoid obstacles in the environment [17], the nature of individual action computation prevents each agent from having knowledge of the actions planned by others. In order to facilitate training and improve efficiency, it will adopt the coordination technique proposed in [22], where actions that lead to collision are prohibited. There, agents are ordered according to the joint highest value of Q, and the agent with the highest value of Q decides an action without considering the other agents. Sequentially, the following agents obtain the new future position of the previous one, considering it as an obstacle, thus blocking actions leading to the same cell. Therefore, although the agents decide their actions sequentially, the actual movement of the vehicles occurs simultaneously. This greedy heuristic, denoted as `AMask` in Algorithm 1, allows the most optimistic agent to perform the action first, giving it priority since it is the one with the highest expected reward.

F. DUELING NEURAL NETWORK ARCHITECTURE

Once the Double Deep Q-Learning, the observation and reward functions and action masking for collisions have been defined, the neural network proposed to be employed can be explained in more detail. In this multi-agent problem, a single neural network will be used for all agents. This approach, known as Centralized Training and Exploration with Decentralized execution via policy Distillation [33], has achieved good results in similar work [22]. The main advantage of this method is its scalability and low computational cost compared to other proposals where there is a network for each agent. With this method, due to the egocentric observation of the state, the same neural network can determine the best actions depending on the agent it observes. This method becomes especially challenging when facing with a fleet of heterogeneous agents, where the same action can give very disparate rewards depending on the quality of the agent's measurement, but since the actions they can take and the perception of the environment are the same, these are considered interchangeable experiences, and all the experiences generated are stored indistinctly in the experience buffer so that they can be used later during training.

The implementation of the neural network depicted in Fig. 6 is proposed, similar to the one used in [22]. This consists of a first stage of three convolutional layers, such as a convolutional encoder, originally proposed in [6]. It serves as a feature extractor of the scenario observation, such as the navigation map, the model, and the positions of the agents with the value k_{agent}. The output of the three convolutional layers is followed by a fully connected layer, and its output is merged with an additional external input where the standard deviation value of the observing agent is introduced. This additional input proposed is intended to help the network in the process of identifying the observing agent, and to serve as a reinforcement to know that depending on this identifier, it can get a different reward for the same pair of environment state and action. This value fusion is followed by two more fully connected layers, and after them, given the complexity of the Q-function to be estimated, a division known as Dueling Q-Network [34] is introduced, where two estimators are used for the Q-function: the Value Function $V(s)$ and the Advantage Function $A(s, a)$, allowing a better representation of the cumulative reward. The former returns a value based on the estimated reward according to the state s in which it is, regardless of the action to be taken. That is, it is a measure of the intrinsic quality of the current state, e.g., if the value is high, it means that it is an area where a high reward is expected regardless of the action to be taken. The latter returns the advantage of taking a specific action a in a state s versus the other possible actions, i.e., it measures how each action changes the intrinsic quality of the current state. Thus, as denoted in [34], the $Q(s, a)$ value is computed as follows, being $|\mathcal{A}|$ the dimension of the action space.

$$Q(s,a) = V(s) + \left(A(s,a) - \frac{1}{|\mathcal{A}|} \sum_{\boldsymbol{a}} A(s, \boldsymbol{a}) \right) \tag{16}$$

Finally, the pseudo-code of Dueling DQL with other implementations is described in Algorithm 1.

V. SIMULATION RESULTS

This section presents the settings and parameters selected for the training and simulations performed. First, a comparison will be made between the proposed observation functions and the reward functions, with a discussion of the advantages and scalability of the framework. Subsequently, the best approach obtained will be compared with other previous algorithms and heuristics from 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 48GB and Nvidia RTX 3090 25GB. Each training process has lasted around 30 hours of computation, due to the high number of episodes. GPyTorch[1] libraries are used to define the Gaussian processes and PyTorch[2] for the neural networks. The code is available in the Github repository [https://github.com/amendb/] for reproduction of the results.

A. PERFORMANCE METRICS

In the field of model and algorithm evaluation, the proper selection of performance metrics is essential to understand the effectiveness and efficiency of the proposed solutions. In this case, the metrics will provide a quantitative measure of performance towards the ultimate goal of the study: to obtain the best possible contamination model. Thus, three metrics will be analyzed:

[1]https://gpytorch.ai/

[2]https://pytorch.org/

Algorithm 1 Double Deep Q-Learning Algorithm With Action Masking (`AMask`) to Avoid Collisions

```
1:  Initialize buffer replay memory $B$ to capacity $|B|$
2:  Initialize $Q$-network with random weights $\boldsymbol{\theta}$
3:  Initialize target $Q$-network $Q'$ cloning weights $\boldsymbol{\theta}' = \boldsymbol{\theta}$
4:  for episode = 1 to $N_{episodes}$ do
5:      $active_i \leftarrow$ True
6:      Reset environment
7:      while any($active_i$) do
8:          Take new measures and update GP
9:          for every agent i active do
10:             Get observation of the state $o_i = \mathcal{O}(s)$
11:             if $U(0,1) < \epsilon$ then
12:                 $a_i \leftarrow$ AMask $(U(0,1), \ldots, U(0,1))$
13:             else
14:                 $a_i \leftarrow$ AMask $(Q(o_i,a), \ldots, Q(o_i,a))$
15:             end if
16:         end for
17:         Move every agent $i$ following action $a_i$
18:         Get every reward $r_i$ using (15)
19:         Get new observations $o'_i$
20:         Store in B every transition $< o_i, a_i, r_i, o'_i >$
21:         if $B > b$ then
22:             Sample random batch $b$ of $(o, a, r, o')$ from $B$
23:             Loss backpropagation
24:             Update weights $\theta$ by gradient descent step
25:             Soft update $Q$-target parameters $\theta'$ from $\theta$
26:         end if
27:         if $d_i > d_{max}$ then $active_i \leftarrow$ False
28:         end if
29:     end while
30:     if $\epsilon > \epsilon_{min}$ then $\epsilon \leftarrow \epsilon - \epsilon_{reduction}$
31:     end if
32: end for
```

- **MSE**: Mean Squared Error (MSE) between the contamination Ground Truth and the model obtained through Gaussian Process regression. It will be calculated at three stages: 33%, 66% and 100% of the episode in order to observe the evolution. The lower the MSE, the better is the model.

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (Y(x_i) - \mu(x_i))^2$$

- **MSE^{peaks}**: The same metric, but calculated only in locations where $Y(x_i) > 0.9$, i.e., contamination peaks, at three stages: 33%, 66% and 100% of the episode.
- **R2**: Quantifies the proportion of variance in the dependent variable that can be predicted by the independent variable. The closer the value is to 1, the better is the

TABLE 1. Environment and gaussian process parameters.

Environment parameter	Value
Map size (H, W)	$(58,\ 38)$ pixels
Estimated cell length	250 m
Estimated cell area	0.06 km^2
Movement distance straights $(d_{straights})$	500 m
Movement distance diagonals $(d_{diagonals})$	725 m
Max. distance traveled (d_{max})	25 km
Agent measure SD range (σ_n)	$[0.005,\ 0.5]$
Influence radius $R_{influence}$	1590 m
Shekel Function (Ground Truth)	**Value**
Maximum peaks range each episode	$[1, 4]$
Elements c range each episode	$[0.5, 1.5]$
Gaussian Process parameter	**Value**
Initial length-scale ℓ	1325 m
Length-scale range (ℓ_{min}, ℓ_{max})	$[132,\ 2650]$ m
Prior mean (μ_0)	0

model.

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (Y(x_i) - \mu(x))^2}{\sum_{i=1}^{n} \left(Y(x_i) - \overline{Y(X)}\right)^2}$$

- **CMSE**: The Cumulative MSE (CMSE) is the aggregated MSE across all time step points t. It allows assessing the overall trend of the model performance over multiple episodes, giving a global view. The lower the CMSE, the better is the model.

$$CMSE = \sum_{t=1}^{T} MSE_t$$

B. BASE SETTINGS

Although the chosen scenario is the Ypacaraí Lake, this algorithm can be implemented in any other environment. The movements are performed synchronously, although the diagonal movements travel a larger space than the straight ones. At each movement, a sample of the location is taken and the GP-model is updated. The model parameters have been defined on the basis of the known information of the chosen scenario, but could be adapted if another scenario is selected. This information is compiled in Table 1. The range of standard deviations of the sensor coupled to the agents has been obtained through a market study, with the purpose of comparing the possibilities of modeling water contamination with more expensive and with cheaper sensors.

C. REWARD AND OBSERVATION STUDY

In order to consider all the proposals, both the two observation functions $\mathcal{O}_\sigma(s)$ and $\mathcal{O}_K(s)$ (Section IV-C), as well as the two reward functions R_1 and R_2 (Section IV-D), several trainings have been performed. Each training performs 100.000 missions of the same duration (25km), being 51 movements the maximum possible per episode. In each mission, the four agents appear randomly in four fixed positions, as described in Section III-A. The learning parameters used during training are detailed in Table 2, adjusted with reference to those used in similar studies [22]. To balance exploration and exploitation and allow the network to learn very diverse experiences stored in the prioritized experience replay, the ϵ value of the ϵ-greedy policy during the first 33% of episodes will decrease linearly from 1 to 0.05. After 33% of episodes, ϵ will remain constant at the minimum value. To adjust the network weights, batches of 128 in size will be taken by applying a learning rate of 1×10^{-4}. Similarly, the selection of the parameters of the GP kernel is fundamental to guarantee the convergence of the estimation, so the parameters shown in Table 2 were set to achieve a good fit to the expected contamination maps.

TABLE 2. List of hyperparameters for training DQL algorithm.

Learning hyperparameter	Value
Training episodes	1×10^5
Learning Rate α	1×10^{-4}
Prioritized buffer replay size $\|B\|$	1×10^6
Randomness of the sample in buffer	0.2
Bias compensating constant in buffer	0.6
Batch size b	128
Target network update smoothing factor τ	1×10^{-3}
ϵ interval	$[1, 0.05]$
ϵ threshold	33%
Discount factor γ	0.99

During the training process, several configurations of observation functions and parameters C_1, C_2 y C_3 have been tested, being those shown in Table 3 the ones that have obtained the best results. When dealing with heterogeneous drones that may be modifiable, which would totally change the behavior of them, in order to compare between proposals it is necessary to set conditions. Therefore, as the combinations of agents could be infinite within the standard deviation range of 0.005 to 0.5, three combinations of agents will be chosen for validation:

i) *Combination 1*: All agents have sensors that obtain relatively good measurements, so their standard deviations values are set to mean/low: [0.007, 0.020, 0.056, 0.091].
ii) *Combination 2*: All agents have sensors that obtain relatively bad measurements. Their standard deviations values are set to mean/high: [0.213, 0.381, 0.130, 0.197].
iii) *Combination 3*: Two agents have sensors that obtain relatively good measurements, and two others have sensors that obtain relatively bad measurements. The standard deviations values of the ensemble are set to: [0.007, 0.020, 0.213, 0.130].

Under the selected configurations for the agents, the three fundamental cases are comprehensively addressed, thus providing a robust estimate of the overall performance. Several combinations of observation-reward functions have been evaluated. However, the comparison will focus on the five that have shown the most outstanding results. Two of these approaches are associated with the observation function $\mathcal{O}_\sigma$, and R_1 is set as the reward function, with parameters $[C_1, C_2] = [10, 0]$ and $[10, 5]$. The remaining

TABLE 3. Average metrics comparison between best combinations of observation and reward function parameters for D-DQL during the same 100 episodes. Three combinations of agents are evaluated. Highlighted values correspond to the best performance value.

Agents	Obs. Fn.	Reward			$\text{MSE}_{33\%}(X)$		$\text{MSE}_{66\%}(X)$		$\text{MSE}_{100\%}(X)$		$\text{MSE}^{peaks}_{33\%}(X)$		$\text{MSE}^{peaks}_{66\%}(X)$		$\text{MSE}^{peaks}_{100\%}(X)$		$\text{R}^2(X)$	$\text{CMSE}(X)$
		C_1	C_2	C_3	Mean	CI 95%	Mean	CI 95%	Mean	CI 95%	Mean	CI 95%	Mean	CI 95%	Mean	CI 95%		
Comb. 1	$\mathcal{O}_\sigma$	10	0	–	0.1031	0.0096	0.0461	0.0036	0.0389	0.0034	0.5046	0.0654	0.2523	0.0411	0.2191	0.0389	0.9131	4.2019
	$\mathcal{O}_\sigma$	10	5	–	0.1019	0.0095	0.0446	0.0034	0.0359	0.0024	0.4880	0.0666	0.2559	0.0426	0.2246	0.0374	0.9249	4.1599
	$\mathcal{O}_K$	10	–	25	**0.1002**	0.0103	0.0392	0.0031	0.0316	0.0025	0.4308	0.0714	0.1901	0.0373	0.1631	0.0345	0.9417	4.0103
	$\mathcal{O}_K$	10	–	50	0.1028	0.0112	**0.0344**	0.0024	**0.0263**	0.0015	0.4337	0.0730	0.1315	0.0281	0.0896	0.0158	**0.9626**	**3.8749**
	$\mathcal{O}_K$	10	–	100	0.1030	0.0107	0.0376	0.0037	0.0267	0.0023	**0.4294**	0.0764	**0.1014**	0.0298	**0.0623**	0.0109	0.9589	3.9932
Comb. 2	$\mathcal{O}_\sigma$	10	0	–	0.1302	0.0073	0.1000	0.0039	0.0910	0.0040	0.7078	0.0453	0.6052	0.0388	0.5529	0.0384	0.5549	6.1046
	$\mathcal{O}_\sigma$	10	5	–	0.1353	0.0073	0.1060	0.0039	0.0957	0.0042	0.7367	0.0448	0.6259	0.0393	0.5667	0.0423	0.5074	6.3342
	$\mathcal{O}_K$	10	–	25	0.1317	0.0078	0.0958	0.0040	0.0863	0.0038	0.6909	0.0481	0.5513	0.0400	0.4853	0.0389	0.6010	6.0614
	$\mathcal{O}_K$	10	–	50	**0.1281**	0.0073	**0.0956**	0.0046	0.0828	0.0039	0.6727	0.0494	0.5344	0.0396	0.4592	0.0374	0.6228	**5.9487**
	$\mathcal{O}_K$	10	–	100	0.1288	0.0078	0.0959	0.0043	**0.0808**	0.0034	**0.6718**	0.0517	**0.5283**	0.0406	**0.4396**	0.0351	**0.6426**	5.9869
Comb. 3	$\mathcal{O}_\sigma$	10	0	–	0.1071	0.0101	0.0487	0.0035	0.0411	0.0034	0.4969	0.0657	0.2774	0.0420	0.2396	0.0404	0.9008	4.32314
	$\mathcal{O}_\sigma$	10	5	–	0.1006	0.0088	0.0534	0.0041	0.0420	0.0032	0.5148	0.0646	0.3084	0.0484	0.2455	0.0430	0.8922	4.41198
	$\mathcal{O}_K$	10	–	25	0.1076	0.0106	0.0480	0.0034	0.0372	0.0026	0.4869	0.0700	0.2217	0.0422	0.1574	0.0343	0.9220	4.2790
	$\mathcal{O}_K$	10	–	50	**0.0977**	0.0088	**0.0435**	0.0041	**0.0330**	0.0025	0.4532	0.0723	0.1422	0.0361	0.1045	0.0262	**0.9400**	**4.1384**
	$\mathcal{O}_K$	10	–	100	0.1102	0.0104	0.0457	0.0035	0.0338	0.0024	**0.4426**	0.0769	**0.1069**	0.0320	**0.0766**	0.0196	0.9374	4.2796

three approaches correspond to the observation function $\mathcal{O}_K$, and R_2 is configured as the reward function, with parameters $[C_1, C_3] = [10, 25]$, $[10, 50]$, and $[10, 100]$. These particular configurations have been selected due to their superior performance and will undergo further evaluation. For this purpose, 100 episodes with different ground truths generated by Shekel functions will be conducted for each of the five best combinations of observation-reward functions. In this comparison, all environment conditions are identical between methods to avoid bias in the interpretation of the results. The episodes of the evaluation process were performed on a laptop equipped with an 8 GB Nvidia RTX 3070 Laptop graphics card, where each motion step composed of the four agents was processed in an average of approximately 24 milliseconds. It should be noted that the computational load is not significant in the context of this application since vehicles travel long distances for long periods of time, making the computational time meaningless relative to the task at hand.

Table 3 shows the evaluation results for the selected performance metrics. The approaches using the observation function $\mathcal{O}_K$ with its corresponding reward function R_2 are clearly the ones with the best overall metrics at the end of the episode. Of their three combinations of reward weights, the last two are the ones that obtain the best contamination models according to the metrics, being the reward function with $C_1 = 10$ and $C_3 = 100$ awarded with the lowest MSE^{peaks} at the end of the episode. Comparing it to the worst case of this metric, which is obtained for all combinations of agents with the set $< \mathcal{O}_\sigma, C_1 = 10, C_2 = 5 >$, the improvement is substantial: over 72% for the first combination of agents, over 20% for the second, and about 68% for the third.

Compared to the set $< \mathcal{O}_K, C_1 = 10, C_3 = 50 >$, the second set with the lowest $\text{MSE}^{peaks}_{100\%}$, the first improves it by more than 30% for the first combination of agents, more than 4% in the second combination, and close to 27% for the third combination. In contrast, these two best performing cases obtain very similar MSE, R^2 and CMSE values for the complete map at 100% of the episode for all agent configurations. In fact, in $\text{MSE}_{100\%}$ metrics, the differences are close to 2% between these two best cases, where depending on the combination of agents one or the other leads, as happens with the R^2 metric. It is also observed that the convergence of both is faster than the rest of the proposals, generally achieving the best MSE metrics for both 33% of the episode and 66%, as well as lower CMSE values throughout the entire episode. In addition, with the confidence intervals (CI) of the MSE something similar happens: these two cases obtain a smaller value than the other ones, but among them, depending on the combination of agents, sometimes one obtains better values than the other. However, either the CI of the MSE^{peaks} or its mean are always better for the combination $< \mathcal{O}_K, C_1 = 10, C_3 = 100 >$, which is logical, since it is the one that more rewards finding high measurements during the training process, and therefore visiting high contamination areas.

When all sensors have poor quality in the measurement (high standard deviation), as in agent combination 2, the differences between algorithms are not so remarkable, which shows that it is very complicated even to get an acceptable model due to the high noise in the measurement and, therefore, also to take good actions for the agents. However, in agent combination 3, where two high-quality agents are mixed with two low-quality ones, the differences in the metrics with respect to combination 1 are not so pronounced.

TABLE 4. Average metrics comparison between D-DQL, RWPP, LMPP, and PSO [19] algorithms during the same 100 episodes. Three combinations of agents are evaluated. Highlighted values correspond to the best performance value.

Agents	Algorithm	$\text{MSE}_{33\%}(X)$		$\text{MSE}_{66\%}(X)$		$\text{MSE}_{100\%}(X)$		$\text{MSE}^{peaks}_{33\%}(X)$		$\text{MSE}^{peaks}_{66\%}(X)$		$\text{MSE}^{peaks}_{100\%}(X)$		$\text{R}^2(X)$	$\text{CMSE}(X)$
		Mean	CI 95%	Mean	CI 95%	Mean	CI 95%	Mean	CI 95%	Mean	CI 95%	Mean	CI 95%		
Comb.1	D-DQL	0.1030	0.0107	**0.0376**	0.0037	**0.0267**	0.0023	**0.4294**	0.0764	**0.1014**	0.0298	**0.0623**	**0.0109**	**0.9589**	**3.9932**
	RWPP	0.1045	0.0106	0.0519	0.0065	0.0393	0.0044	0.4946	0.0656	0.2589	0.0476	0.2051	0.0381	0.8986	4.1586
	LMPP	0.1067	0.0103	0.0767	0.0089	0.0637	0.0078	0.5098	0.0656	0.4003	0.0614	0.3404	0.0570	0.7295	5.0494
	PSO	**0.0961**	0.0075	0.0716	0.0061	0.0707	0.0070	0.5247	0.0611	0.2933	0.0631	0.2241	0.0657	0.7043	4.6069
Comb.2	D-DQL	**0.1288**	0.0078	**0.0959**	0.0043	**0.0808**	0.0034	**0.6718**	0.0517	**0.5282**	0.0406	**0.4396**	0.0351	**0.6426**	**5.9869**
	RWPP	0.1318	0.0072	0.1071	0.0042	0.0982	0.0041	0.7184	0.0418	0.6186	0.0368	0.5551	0.0358	0.5004	6.2371
	LMPP	0.1366	0.0080	0.1183	0.0071	0.1072	0.0061	0.7123	0.0469	0.6483	0.0460	0.6070	0.0448	0.3975	6.6837
	PSO	0.1374	0.0072	0.1205	0.0056	0.1084	0.0062	0.7233	0.0442	0.6488	0.0463	0.5559	0.0550	0.3763	6.4262
Comb.3	D-DQL	0.1102	0.0104	**0.0457**	0.0035	**0.0338**	0.0024	**0.4426**	0.0769	**0.1069**	0.0320	**0.0766**	0.0196	**0.9374**	**4.2796**
	RWPP	0.1118	0.0109	0.0607	0.0066	0.0493	0.0058	0.5352	0.0656	0.2947	0.0532	0.2431	0.0464	0.8383	4.5191
	LMPP	0.1116	0.0095	0.0825	0.0084	0.0691	0.0076	0.5629	0.0644	0.4446	0.0636	0.3833	0.0611	0.6939	5.3014
	PSO	**0.0993**	0.0076	0.07810	0.0068	0.0749	0.0073	0.5407	0.0600	0.3195	0.0626	0.2051	0.0628	0.6842	4.8401

This indicates two important things: i) When there are agents with such a difference in measurement quality, the model is able to adjust for variations in data quality, indicating the robustness of the GP. In other words, even when some agents provide less accurate or less reliable measurements, the model manages to mitigate their impact and maintain an acceptable performance. ii) The "bad" agents take advantage of the reliable measurements taken by the "good" ones, who generate a greater change in the model, and consequently they also receive a state in the observation that is more faithful to reality, allowing them to make better decisions.

Having reached this point, the crucial question arises: What should be prioritized in the model, a slightly better overall MSE but worse performance in contamination peaks, or vice versa? As stated in the introduction Section I, the objective was to develop a system for detecting high contaminated areas in order to treat them, thus prioritizing them in the IPP. Furthermore, since the differences in the MSE for the entire map are minor between the two best performing cases, it is decided to prioritize the model that excels in predicting contamination peaks, and the combination $< \mathcal{O}_K, C_1 = 10, C_3 = 100 >$ should be chosen as the best candidate.

D. COMPARISON WITH OTHER METHODS

In this section a comparison of the selected D-DQL policy with other algorithms will be performed. The same comparison metrics, GPs and ground truth maps will be used, and the algorithms will also use action collision masking to choose only safe actions. Three algorithms will be used for the comparison:

- **Random Walker Path Planner (RWPP)**: In this approach, each agent moves following a fixed random exploration direction in space, until it encounters an obstacle. At that point it randomly re-sets a new exploration direction, but avoids the direction from which it came so as not to retrace its steps.
- **Lawn Mower Path Planner (LMPP)**: This algorithm uses a strategy to explore that ensures complete and efficient coverage of an area. Its name comes from the way a lawn mower cuts the grass. Each agent moves in a random initial direction until it encounters an obstacle, then takes a perpendicular direction, and continues along a path parallel to the previous one.
- **GP-based Particle Swarm Optmization (PSO)**: This approach is based on the research [19], from which the acceleration coefficients have been extracted. There, the fleet is a group of particles which are used to explore the search space. Each particle has a position and velocity, and these values are updated over time influenced by the distance between the current position and other four components: the personal best position, the global best position including all particles, the position of maximum uncertainty of the GP, and the position of maximum mean of the GP. To balance exploration and exploitation, each of these distances is weighted by a fixed value and a random component.

Another set of 100 scenarios is used to compare the D-DQL with these other approaches. The combination of agents properties will also remain the same. The results are presented in Table 4. Here it can be seen how D-DQL demonstrates its higher performance, being the algorithm that obtains the best results in almost all metrics. In general, the improvement is remarkable, especially in the $\text{MSE}^{peaks}_{100\%}$, where it improves the RWPP by about 70%, 20% and 69% for the three agent combinations respectively, the LMPP by 82%,28% and 80% and the PSO by 72%, 21% and 63%. This improvement, although less noticeably, is also reflected in other metrics such as the MSE of the total map (see also Fig. 8), the CMSE, and the R^2 score. In combination 2 of agents (agents with high standard deviation), as was the case in the comparison of DRL approaches, the differences are not so pronounced, since the impossibility of taking reliable samples means that the virtues

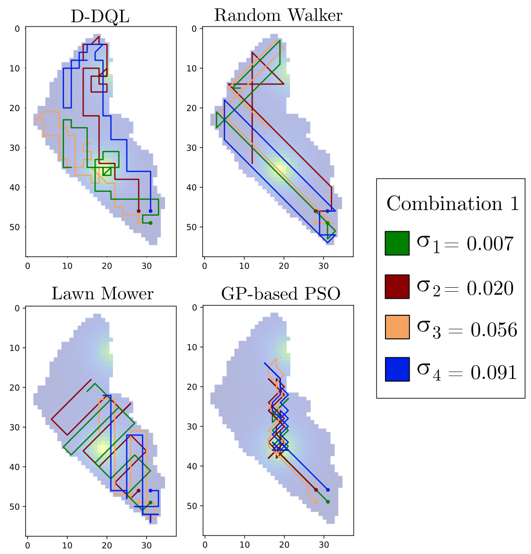

FIGURE 7. Example of paths followed by ***Combination 1*** of agents for the proposed D-DQL algorithm compared to other heuristic algorithms (RWPP, LMPP, GP-based PSO [19]) in one random episode.

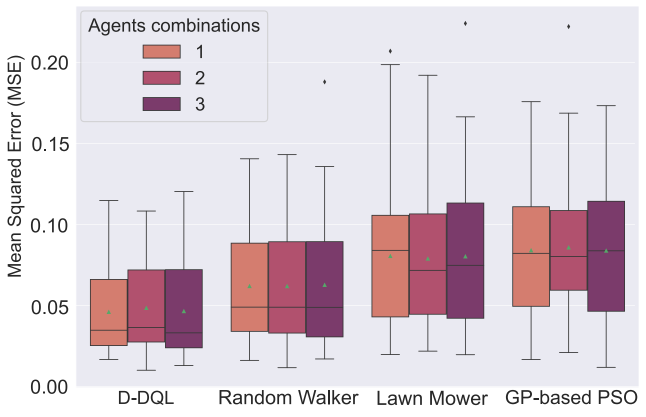

FIGURE 8. Box plot of the MSE$_{100\%}$ for the 3 combinations of agents according to the four algorithms compared during 100 episodes.

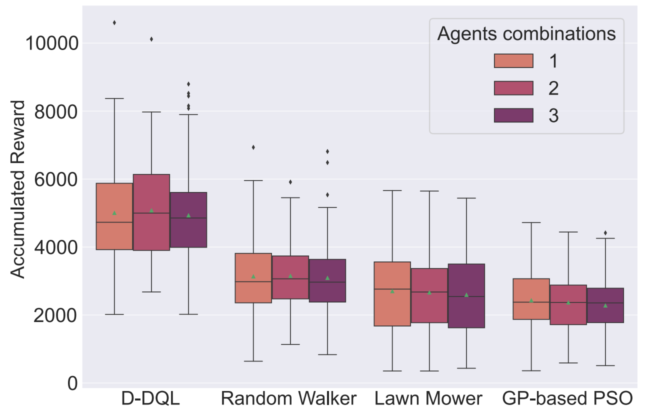

FIGURE 9. Box plot of the accumulated reward for the 3 combinations of agents according to the four algorithms compared during 100 episodes. The reward that would have been obtained by the algorithms that are not reward-driven is also computed, that is, heuristics other than DRL.

of the algorithms are minimized, and the differences with the ground truth are high despite performing good actions. Fig. 9 shows the average cumulative reward over 100 episodes of the proposed D-DQL algorithm for the three combinations of agents, as well as the reward that the other heuristics presented for comparison would obtain in the case that they were rewarded with the same reward function. It can be seen, as expected, that the D-DQL algorithm has the highest reward, but it is interesting to note that the order of the other algorithms sorted from highest to lowest reward matches the order obtained in Fig. 8, which shows the MSE. This consistency indicates that the proposed reward function is effective in incentivizing the desired behavior of the agents in the system, which is to obtain a model with low error.

Fig. 7 clearly illustrates the strategies of each algorithm. LMPP proves to be robust in intensifying specific areas; however, by performing such redundant and inefficient trajectories in the current problem, with the addition of the constraint on the maximum distance traveled, agents find it difficult to cover large areas of the map. Additionally, lack of consideration of the actions taken in the decision process means that areas with high contamination are addressed with the same priority as any other area. Similarly, the RWPP shows surprisingly highly explorative trajectories, but being also a basic heuristic, it does not take into account contextual information of the environment it visits, resulting in a suboptimal use of available resources to address the current problem. Still, despite following a simple strategy, it obtains the second place in most metrics, showing that a good policy to follow to obtain a good model is to perform a highly explorative patrol. D-DQL shows a wide exploration of the map, as well as an intensification in the contamination peaks, especially with the best agents, achieving an adequate policy for the needs of the problem.

Meanwhile, the PSO algorithm, despite being based on acquiring information from the GP and taking it into account for decision making, does not obtain good overall metrics, being the one that obtains the worst MSE for the complete map in all combinations of agents. As shown in Fig. 7, GP-based PSO intensifies heavily on contamination peaks when it finds them, but even then it does not get good results. The main reason is that, despite being a good heuristic algorithm, for the particles in PSO to acquire information independently and efficiently, it is crucial that their initial starting points are sparse in the search space. If the particles start too close together, they will perceive similar information and tend to converge to similar solutions, behaving almost as a single particle. This causes the trajectories to converge to suboptimal or local solutions, rather than exploring the search space more exhaustively in search of peaks. To overcome this limitation two solutions are possible: i) Perform an initial positioning phase without sampling, which would imply a battery expense, and therefore a reduction in the time available for the subsequent measurement phase. ii) Distribute the starting points of the vessels in different locations of the lake. However, this solution would require the

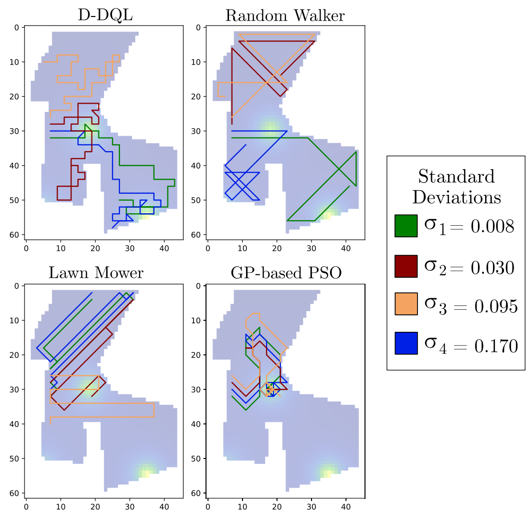

FIGURE 10. Example of paths followed in a new environment (Port of a Coruña, Spain) with a new set of agents for the proposed D-DQL algorithm compared to other heuristic algorithms (RWPP, LMPP, GP-based PSO [19]) in one random episode.

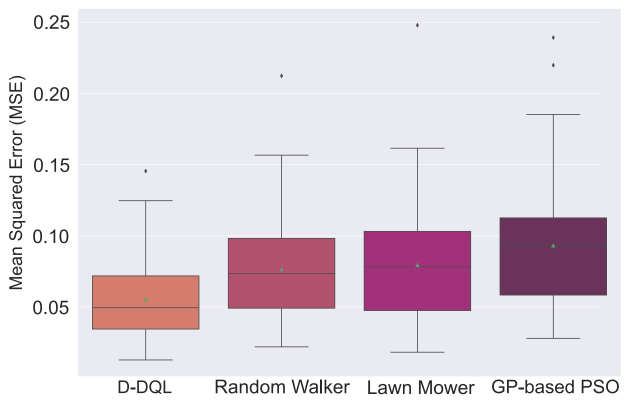

FIGURE 11. Box plot of the $MSE_{100\%}$ for the new set of agents in the new environment (Port of a Coruña, Spain) according to the four algorithms compared during 100 episodes.

transport of the boats to their starting positions, which could involve the use of another type of vehicle, adding logistical complexity to the process. Both solutions are inefficient, and it is mainly at this point where the proposed DRL algorithm stands out, which is capable of performing an adaptive deployment by having the ability to dynamically adapt the behavior of the algorithm regardless of the initial positions.

E. GENERALIZATION TO OTHER SCENARIOS

To ensure the generalizability of the algorithm, a new training is performed with the same parameter configuration but in a new environment: the port of A Coruña (Galicia, Spain). The main novelty compared to Ypacaraí Lake is the greater complexity of its layout, representing a non-convex set. In this map there is much less possibility of going between two points in a straight line than in the previous case, due to the tighter angles of its borders. In addition, new values for the standard deviations of the agent's sensor set will also be determined to prove the correct operation of the algorithm with the above selected parameters. These new standard deviations values of the ensemble are set to: $[0.008, 0.030, 0.095, 0.170]$. An example of traced paths is shown in Fig.7, where it can be seen how the D-DQL algorithm visits a high percentage of the map and focuses on the peaks, as in the case of Lake Ypacaraí. In contrast, for the other heuristics, the task is further complicated due to the higher complexity of the boundaries. In addition, the $MSE_{100\%}$ resulting from the estimation of the contamination map of each algorithm is shown by a box plot in Fig. 11, where again it can be observed how the D-DQL algorithm significantly outperforms the other methods.

VI. CONCLUSION AND FUTURE LINES

In this paper an Informative Path Planning approach has been developed for a fleet of Autonomous Surface Vehicles for contamination monitoring in large water resources. The fleet consists of heterogeneous vehicles with different sensing capabilities, i.e. they all measure the same variable but do so with different qualities depending on the standard deviation of the sensor they incorporate. The objective is for agents to cooperate to achieve the best possible contamination model by optimizing their movements, paying special attention to high-contamination areas. For this purpose, a discretized in cells scenario is created that simulates the acquisition of WQP in the visited locations. With these measurements, by employing a Gaussian process, a surrogated model of the lake is obtained. Ypacaraí Lake is the selected scenario for the simulations, due to its serious real pollution problems, but any other map can be considered instead.

This paper proposes to solve this problem by applying a DRL framework that receives as input a visual representation of the state. For this framework, state-of-the-art methods such as prioritized experience replay, Dueling Deep Q-Network or action collision masking have been applied to obtain a reward-maximizing policy. In order to achieve the objectives, the most outstanding contributions have been: i) to define several modalities of observation of the states, including in them information about the qualities of the sensors of the agents and their movement through the map and ii) to propose reward functions that motivate actions resulting in the improvement of the model, but taking into account the different qualities of the agents. The main advantage of the proposal, which is a contribution to the monitoring processes compared to other approaches, is that the heterogeneous agents can be modified for each scenario, since the network has been trained for a wide range of standard deviation values, so that the combinations are infinite and can cover a large number of real measurement devices only by knowing their standard deviation. This implies a great scalability of the method, since it only uses a single centralized neural network that shares the experiences of all the agents, and decides the best action for each one individually. Although this algorithm has been applied in aquatic monitoring, it can also be extended to other similar applications of pollution detection or other measurements, either aerial or terrestrial.

The proposed method, although promising, has some limitations that need to be considered. Firstly, although it is intrinsic to the problem itself, in order to obtain the best model it is necessary to have sensor data with low standard deviations, which is not always possible. Despite this, in this method agents learn policies that help to mitigate this drawback, being able to adapt to situations in which low quality sensors are presented, and taking advantage of that information for the common goal. In addition, the application of this method requires some level of prior knowledge of the environment, such as contamination behavior, to design an appropriate reward function. D-DQL and other DRL algorithms may face stability issues during training, such as divergence of the Q-function value or instability in policy learning, which may necessitate a long training period with significant computational capacity. Lastly, these algorithms might struggle to generalize the acquired knowledge from one environment to significantly different environments, demanding additional training.

The best DRL algorithm obtained, in the end, is compared with other common heuristics such as Lawn Mower, Random Walker or Particle Swarm Optimization, obtaining in the resulting contamination models a very significant improvement in metrics such as MSE, R^2 and CMSE. Thus, the fusion of Gaussian processes with deep reinforcement learning stands out as an outstanding technique for this type of mission. In the comparisons made, the drones start from the same deployment area for practical and efficiency reasons, reflecting real-world scenarios. However, further research could explore the deployment of teams of agents from different positions, such as a configuration where two agents in each fleet start from separate locations. Following the current line of research, future work may explore the application of the proposed methodology in similar situations with multi-objective maps. Specifically, the measurement of various water quality variables could be considered, thus extending the approach to a more diverse set of targets. The primary goal would be to maximize the acquisition of information from all WQPs to simultaneously improve all models involved in the process. This approach could provide a more complete and holistic view of water quality, addressing multiple relevant aspects by integrating them into a unified process. In addition, it could be explored how the adaptability and efficiency of the approach proposed in this article is maintained or improved in more complex and multifaceted contexts.

REFERENCES

[1] G. Whytock, *Water Resources Across Europe: Confronting Water Stress: An Updated Assessment*. Copenhagen, Denmark: European Environment Agency, 2021.

[2] G. A. Lopez Moreira M, L. Hinegk, A. Salvadore, G. Zolezzi, F. Holker, R. A. Monte Domecq S, M. Bocci, S. Carrer, L. De Nat, and J. Escriba, "Eutrophication, research and management history of the shallow Ypacaraí Lake (Paraguay)," *Sustainability*, vol. 10, no. 7, p. 2426, 2018.

[3] Y. Long, S. Liu, D. Qiu, C. Li, X. Guo, B. Shi, and M. S. AbouOmar, "Local path planning with multiple constraints for USV based on improved bacterial foraging optimization algorithm," *J. Mar. Sci. Eng.*, vol. 11, no. 3, p. 489, Feb. 2023. [Online]. Available: https://www.mdpi.com/2077-1312/11/3/489

[4] Y. Cao, Y. Wang, A. Vashisth, H. Fan, and G. A. Sartoretti, "CAtNIPP: Context-aware attention-based network for informative path planning," in *Proc. Conf. Robot Learn.*, 2023, pp. 1928–1937.

[5] Z. Zhang, H. Liu, M. Zhou, and J. Wang, "Solving dynamic traveling salesman problems with deep reinforcement learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 4, pp. 2119–2132, Apr. 2023.

[6] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, and G. Ostrovski, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[7] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning* (Adaptive Computation and Machine Learning). Cambridge, MA, USA: MIT Press, 2006.

[8] Y. Rizk, M. Awad, and E. W. Tunstel, "Cooperative heterogeneous multi-robot systems: A survey," *ACM Comput. Surveys*, vol. 52, no. 2, pp. 1–31, Mar. 2020.

[9] S. Hu, S. Tian, J. Zhao, and R. Shen, "Path planning of an unmanned surface vessel based on the improved A-star and dynamic window method," *J. Mar. Sci. Eng.*, vol. 11, no. 5, p. 1060, May 2023. [Online]. Available: https://www.mdpi.com/2077-1312/11/5/1060

[10] A. G. Rumson, "The application of fully unmanned robotic systems for inspection of subsea pipelines," *Ocean Eng.*, vol. 235, Sep. 2021, Art. no. 109214. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0029801821006442

[11] W. Chen, X. Hao, K. Yan, J. Lu, J. Liu, C. He, F. Zhou, and X. Xu, "The mobile water quality monitoring system based on low-power wide area network and unmanned surface vehicle," *Wireless Commun. Mobile Comput.*, vol. 2021, pp. 1–16, Oct. 2021.

[12] C. Specht, E. Switalski, and M. Specht, "Application of an autonomous/unmanned survey vessel (ASV/USV) in bathymetric measurements," *Polish Maritime Res.*, vol. 24, no. 3, pp. 36–44, Sep. 2017.

[13] F. Stache, J. Westheider, F. Magistri, C. Stachniss, and M. Popovic, "Adaptive path planning for UAVs for multi-resolution semantic segmentation," *Robot. Auto. Syst.*, vol. 159, Jan. 2023, Art. no. 104288. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889022001774

[14] C. Guo, H. Tang, B. Niu, and C. Boon Patrick Lee, "A survey of bacterial foraging optimization," *Neurocomputing*, vol. 452, pp. 728–746, Sep. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231220319172

[15] R. Marchant and F. Ramos, "Bayesian optimisation for informative continuous path planning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2014, pp. 6136–6143.

[16] M. Grzelczak and P. Duch, "Deep reinforcement learning algorithms for path planning domain in grid-like environment," *Appl. Sci.*, vol. 11, no. 23, p. 11335, Nov. 2021. [Online]. Available: https://www.mdpi.com/2076-3417/11/23/11335

[17] S. Y. Luis, D. G. Reina, and S. L. T. Marín, "A multiagent deep reinforcement learning approach for path planning in autonomous surface vehicles: The Ypacaraí lake patrolling case," *IEEE Access*, vol. 9, pp. 17084–17099, 2021.

[18] F. Peralta, D. G. Reina, S. Toral, M. Arzamendia, and D. Gregor, "A Bayesian optimization approach for multi-function estimation for environmental monitoring using an autonomous surface vehicle: Ypacarai lake case study," *Electronics*, vol. 10, no. 8, p. 963, Apr. 2021. [Online]. Available: https://www.mdpi.com/2079-9292/10/8/963

[19] M. J. T. Kathen, P. Johnson, I. J. Flores, and D. G. E. Reina, "AquaFeL-PSO: A monitoring system for water resources using autonomous surface vehicles based on multimodal PSO and federated learning," 2022, *arXiv:2211.15217*.

[20] H. L. Kwa, G. Tokic, R. Bouffanais, and D. K. P. Yue, "Heterogeneous swarms for maritime dynamic target search and tracking," in *Proc. Global Oceans*, Oct. 2020, pp. 1–8.

[21] J. Westheider, J. Rückin, and M. Popovic, "Multi-UAV adaptive path planning using deep reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2023, pp. 649–656.

[22] S. Y. Luis, D. Shutin, J. M. Gómez, D. G. Reina, and S. T. Marín, "Deep reinforcement multi-agent learning framework for information gathering with local Gaussian processes for water monitoring," 2024, *arXiv:2401.04631*.

[23] Y. Gao, J. Chen, X. Chen, C. Wang, J. Hu, F. Deng, and T. L. Lam, "Asymmetric self-play-enabled intelligent heterogeneous multirobot catching system using deep multiagent reinforcement learning," *IEEE Trans. Robot.*, vol. 39, no. 4, pp. 2603–2622, Aug. 2023.

[24] A. Kale, N. Bandela, J. Kulkarni, and K. Raut, "Factor analysis and spatial distribution of water quality parameters of Aurangabad district, India," *Groundwater Sustain. Develop.*, vol. 10, Apr. 2020, Art. no. 100345. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352801X19303273

[25] J. Shekel, "Test functions for multimodal search techniques," in *Proc. 5th Annu. Princeton Conf. Inf. Sci. Syst.*, 1971, pp. 354–359.

[26] F. Peralta, D. G. Reina, and S. Toral, "Water quality online modeling using multi-objective and multi-agent Bayesian optimization with region partitioning," *Mechatronics*, vol. 91, May 2023, Art. no. 102953. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957415823000090

[27] B. Brown and A. Zai, *Deep Reinforcement Learning in Action*. New York, NY, USA: Simon and Schuster, Mar. 2020.

[28] R. S. Sutton and A. G. Barto, *Reinforcement Learning, Second Edition: An Introduction*. Cambridge, MA, USA: MIT Press, Nov. 2018.

[29] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. AAAI Conf. Artif. Intell.*, 2016, vol. 30, no. 1, pp. 1–7.

[30] R. Bellman, *Dynamic Programming*. New York, NY, USA: Dover, 1957.

[31] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," 2016, *arXiv:1511.05952*.

[32] B. T. Polyak, *Introduction to Optimization*. New York, NY, USA: Optimization Software, 1987.

[33] G. Chen, "A new framework for multi-agent reinforcement learning—Centralized training and exploration with decentralized execution via policy distillation," 2019, *arXiv:1910.09152*.

[34] Z. Wang, T. Schaul, M. Hessel, H. van Hasselt, M. Lanctot, and N. de Freitas, "Dueling network architectures for deep reinforcement learning," 2015, *arXiv:1511.06581*.

ALEJANDRO MENDOZA BARRIONUEVO was born in Sevilla, in 1998. He received the B.Eng. degree in electronics and robotics engineering and the M.S. degree in logic, computing, and artificial intelligence from the University of Seville, Spain, in 2021 and 2022, respectively, where he is currently pursuing the Ph.D. degree with the Department of Electronic Engineering. His research interests include path optimization algorithms with heterogeneous autonomous vehicles using deep learning. This includes scenarios for monitoring large water resources. Looking ahead, he aims to explore new horizons in autonomous systems and smart technologies, driven by a passion for innovation, and the development of practical solutions.

SAMUEL YANES LUIS was born in Tenerife, Spain, in 1997. He is currently pursuing the Ph.D. degree with the University of Seville. He is also a Pre-Doctoral Researcher and a Trainee Professor with the Department of Electronic Engineering, University of Sevilla. He has been a Visiting Researcher with the Institute of Navigation and Communications, German Aerospace Center, and the Department of Computer Science, State University of Milan. His research interests include the development of optimization algorithms for decision problems with autonomous vehicles, the estimation of environmental models using machine learning and deep learning, and the development of aquatic surface robots for hydrological monitoring.

DANIEL GUTIÉRREZ REINA received the B.E. degree (Hons.) in electronic engineering, the M.S. degree in electronics and telecommunications, and the Ph.D. degree (Hons.) in electronic engineering from the University of Seville, Seville, Spain, in 2009, 2011, and 2015, respectively. He was an Assistant Professor with Loyola University, from October 2018 to April 2019. He has been a Visitor Researcher with Liverpool John Moores University, U.K.; the Free University of Berlin, Germany; the Colorado School of Mines, USA; and Leeds Beckett University, U.K. He is currently an Associate Professor with the Departamento de Ingeniería Electrónica, University of Seville. He has published about 60 articles in JCR journals with an impact factor. His current research interests include the application of meta-heuristic, machine learning, and deep algorithms to solve monitoring problems using autonomous systems. He is part of the editorial board of several journals, such as *International Journal of Distributed Sensor Networks* (SAGE), *Electronics* (MDPI), and *Future Internet* (MDPI), organizing numerous SI for these journals.

SERGIO L. TORAL MARÍN was born in Rabat, Morocco, in 1972. He received the M.S. and Ph.D. degrees in electrical and electronic engineering from the University of Seville, Spain, in 1995 and 1999, respectively. He is currently a Full Professor with the Department of Electronic Engineering, University of Seville. He is the author or coauthor of 95 articles in major international peer-reviewed journals (with JCR impact factor) and over 100 articles in well-established international conferences and workshops. His research interests include ad hoc networks and their routing protocols, flying ad hoc networks, deployment of unmanned vehicles, and intelligent transportation systems.