Deep Reinforcement Multiagent Learning Framework for Information Gathering with Local Gaussian Processes for Water Monitoring

Samuel Yanes Luis, Dmitriy Shutin, Juan Marchal Gómez, Daniel Gutiérrez Reina, and Sergio Toral Marín*

The conservation of hydrological resources involves continuously monitoring their contamination. A multiagent system composed of autonomous surface vehicles is proposed herein to efficiently monitor the water quality. To achieve a safe control of the fleet, the fleet policy should be able to act based on measurements and fleet state. It is proposed to use local Gaussian processes and deep reinforcement learning to jointly obtain effective monitoring policies. Local Gaussian processes, unlike classical global Gaussian processes, can accurately model the information in a dissimilar spatial correlation which captures more accurately the water quality information. A deep convolutional policy is proposed, that bases the decisions on the observation on the mean and variance of this model, by means of an information gain reward. Using a double deep Q-learning algorithm, agents are trained to minimize the estimation error in a safe manner thanks to a Consensus-based heuristic. Simulation results indicate an improvement of up to 24% in terms of the mean absolute error with the proposed models. Also, training results with 1–3 agents indicate that our proposed approach returns 20% and 24% smaller average estimation errors for, respectively, monitoring water quality variables and monitoring algae blooms, as compared to state-of-the-art approaches.

1. Introduction

Environmental monitoring is vital for assessing various aspects of the natural world, including air, water, soil quality, biodiversity, and climate. This practice plays a crucial role in addressing environmental issues such as pollution, climate change, and biodiversity loss.[1] By focusing on key indicators like air and water quality, scientists and policymakers can pinpoint pollution sources and implement effective measures to reduce or eliminate them, informing the development of impactful environmental policies and regulations. To efficiently monitor water resources, autonomous surface vehicles (ASVs) emerge as a promising solution, particularly for vast geographic areas challenging to survey manually.[2] ASVs offer a cost-effective means of data collection using diverse sensors, including physicochemical sensors and cameras.[3,4] Leveraging these autonomous agents allows for the swift acquisition of comprehensive environmental data, enhancing our understanding of environmental conditions accurately and efficiently.

This article focuses on a specific issue in natural conservation, building upon prior research.[5,6] It addresses the monitoring of Ypacaraí Lake, Paraguay's largest drinking water source, using ASVs. Ypacaraí's water quality, crucial for the population, is characterized by variables like pH, turbidity, dissolved oxygen, and chlorophyll concentration. Contamination sources vary due to geographical, human, and biological factors, influencing the optimal information acquisition process. Monitoring the contamination is challenging due to their localized and temporal occurrence, influenced by wind and tides within the lake. This article proposes the adaptive deployment of a multiagent vehicle fleet capable of efficiently monitoring common water quality phenomena such as algae blooms.

The approach proposed in this article for monitorization of biological phenomena will fall into the category of the so-called informative path planning (IPP).[7] The ultimate goal of this family of problems is to sequentially decide on an optimal path for every vehicle that maximizes the information I collected, while minimizing the cost C associated with data acquisition. However, when agents have little or no information about the environment, it is difficult to make effective decisions in the presence of such high uncertainty. This problem has been previously addressed in works such as ref. [6], where by means of Bayesian optimization (BO) and an acquisition function, the next

S. Yanes Luis, D. Gutiérrez Reina, S. Toral Marín
Department of Electronic Engineering
University of Sevilla
41005 Sevilla, Spain
E-mail: syanes@us.es

D. Shutin, J. Marchal Gómez
Institute of Navigation and Communications
German Aerospace Center
82234 Wessling, Germany

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/aisy.202300850.

DOI: 10.1002/aisy.202300850

position for data acquisition is chosen considering the overall uncertainty and the current information model.

IPP strategies are intrinsically adaptive as, depending on the successive observations of the scenario, the agents must change their monitoring waypoints. When different cooperating agents are considered for monitoring, other characteristics come into play that increase the complexity of the IPP. This is the case of redundancy in measurement or collision avoidance between agents. In the first case, it is obvious that when paths are constrained to a distance and a time budget, taking redundant measurements is against efficiency. Agents must coordinate by means of some methods to avoid oversampling areas of which they already have sufficient knowledge or to simultaneously monitor the same area. To address these multiple aspects of IPP, this article proposes to embed this IPP into a reinforcement learning framework,[8] that explicitly considers the information gain with the redundancy measure of every vehicle. The other important aspect in multiagent IPP is agent–agent and agent–environment collision avoidance. which is an aspect usually neglected in previous works.[5]

Second, the Gaussian Processes (GPs) provide a probabilistic description of the process in terms of its mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\sigma}$, which naturally provide a measure of uncertainty about the learnt process.

However, GPs have certain issues that make them difficult to work with. First, they have a complexity of $O(N^3)$, with N being the number of measurement samples.[9] Moreover, the use of GPs in environmental monitoring typically assumes constant hyperparameters across the entire data range. In the case of most kernels used in the literature,[6] the length scale hyperparameter defines the size of spatial correlation between samples of the process. Having the same length scale implies constant spatial correlation properties across the whole area of interest. However, practically, this is often not the case, as, for example, concentrations of the algae can change quite abruptly.

Therefore, another major contribution of this work is the use of multiple local GPs. Local GPs have only local influence, so they will fit the data seen under their area of influence. It will be shown that local GPs are able to characterize better areas with distinct length scales. Furthermore, local GPs bring a considerable improvement in the scalability of the algorithm, since the complexity is reduced in global terms from $O(N^3)$ to $O(N^3/M^2)$, with M being the number of local GPs.

Finally, to solve the IPP, the use of deep reinforcement learning (DRL) techniques is done to train deep adaptive policies. In recent years, DRL has begun to be used for multiagent[10] path planning. DRL allows a neural network to optimize a multiagent policy to maximize a long-term objective set in a reward function. The reward function acts as a measurement of the optimality of each action a given an observation of the environment o. Through the interaction of agents and the environment, DRL algorithms such as deep Q-learning (DQL)[11] are able to adapt the fleet behavior of vehicles to improve acquisition in an optimization time horizon. For this application, the use of DRL is also convenient because it will allow behavior specialization of a fleet for a broad set of scalar fields by means of realistic simulators of the environments within the known boundaries of each biological process.

Finally, the framework is validated against other path planning algorithms in the different benchmarks.

In summary, the novelties proposed in this article are as follows. 1) A Local GP model for multimodal environmental scenarios. 2) A DRL framework to maximize the gathered information for a fleet of unmanned vehicles, including the reward function based on the information gain and the observation method. 3) A censoring methodology to avoid agent–agent and agent–environment collisions.

This article is organized as follows. In Section 2, previous approaches and advances in information gathering and path planning with autonomous vehicles are discussed. In Section 3, the problem of IPP is described and the ground truth under monitorization as well. Later, in Section 4, the methodology is explained. First, the local GP proposition is explained, and later, the DRL algorithm and formulation is described. In Section 5, the different results and simulation are described, and also the comparison with other path planning techniques. Finally, in Section 6, the conclusions are presented with future lines of works that shall be addressed.

2. Related Work

The use of autonomous aquatic and aerial vehicles has gained relevance in recent years due to advances in battery autonomy and, above all, to the capability for remote computing and sensing.[12] In the particular case of aquatic autonomous vehicles, they are divided into two types: 1) surface vehicles (ASV) and ii) underwater vehicles (USV). The former has been used especially for monitoring water quality parameters (WQP) in rivers, lakes, and coasts. The use of ASVs for the acquisition of information in natural environments has been gaining momentum lately. In several previous works, it is possible to find examples of multiagent[6] applications for this purpose. In general, these vehicles are equipped with sensor modules for the acquisition of WQPs and can take geographically located samples at one point at a time. These vehicles are particularly convenient because they can be used to obtain a status of the quality of water resources with good GNSS localization capability, maneuverability, and autonomy.[2] USVs have been used for the exploration of underwater biological environments and for remote inspection of shipwrecks.[13] Both types of vehicles are usually equipped with water quality sensors, bathymetry, spectral cameras, etc., for environmental data acquisition. These sensors are going to define the observation capability in monitoring tasks and observability within the patrol optimization problem.

It is common to find in previous works that GPs have been used to model several different types of data into a comprehensible model, from static environmental data as in ref. [14] to dynamic temporal series as in ref. [15]. In the particular task of IPP with ASVs, the work in ref. [14] uses BO to sequentially decide the next sample to be taken with a single vehicle. A GP and the expected improvement acquisition function are used as the decision module. This function must be tuned to reduce the sample acquisition distance and to avoid the high cost of the ASV's movement. In ref. [15], the dynamic case is solved using a similar approach. The IPP considered there uses the predictive uncertainty as points of interest given an expanded definition of the radial basis function (RBF) kernel used in ref. [14] to accommodate a temporal axis. An extension of the methodology proposed in ref. [6] is addressed for the multiagent case. The acquisition space is proposed to be divided by means of a Voronoi tessellation, which, according to the authors, avoids redundant sampling. The methodology is used only to monitor functions with highly spatial-correlated samples, which neglect the existence of dissimilar length scales for the data. Therefore, when addressing environments with zones of high correlations and zones with high correlations between samples or heteroskedastic noise, these approaches cannot efficiently solve the informative task.

Another common approach is the use of algorithms based on particle swarm optimization (PSO).[16] PSO algorithms base agent decisions on swarm behavior. In ref. [16], the use of GPs to improve the classical PSO algorithm is also discussed. In this approach, vehicles update their speeds attracted by the points of highest uncertainty of the GP, the highest individual's, collective, and estimated pollution values. This approach results in explorative–exploitative paths and a very computationally scalable algorithm. This approach is not always adequate due to local gradient continuity fluctuations, especially for the case of algae bloom monitoring where the information gradient is discontinuous.

In relation to the use of DRL for path planning, it is easy to find an upward trend in the number of recent articles such as in ref. [10]. Previous approaches such as those in ref. [17] have focused on solving the informative patrol problem for Ypacaraí Lake itself. This problem consists of continuous monitoring of WQP with a temporal cyclic criterion. In ref. [10], a multiagent version of the double deep Q-learning (DDQL) algorithm of ref. [11] is used. The DDQN algorithm uses two equal neural networks to estimate the cumulative future reward given a state observation o and the set of possible actions a. From this work, it is possible to see that the neural decoder structure with visual observations is used, similar to our proposal. This visual formulation of the state allows for better interpretability and simplifies the feature selection process for the estimation of the estimated future reward. However, unlike our new proposal, the state is completely known a priori, which is unrealistic in an initial exploration scenario such as the one that in this article's proposal.

In ref. [18], a work based on DRL is also presented to solve IPP with multiple agents. The objective is to reduce the estimation error over a relatively small scalar field (10 × 10 pixels) in the minimum possible time. Similar aspects between this approach and ours are: 1) the use of partial and visual observations of the environment and 2) the use of discrete actions as a way of reducing the decision variables of each agent. While they propose a Dueling DQL, this work proposes the proximal policy optimization algorithm which uses rollouts of experiences of the neural policy $\pi(\mathbf{s})$ to update its parameters. A reward function based on the root mean squared error (RMSE) is proposed, which implies that the reward can only be calculated in simulation, when the ground truth is known. Our proposal attempts to decouple the reward from an error function, which is associated with an unknown ground truth, and the error function itself. In this article, several reward laws are proposed based on information gain such as mean transport or uncertainty variance. Moreover, the reward function in ref. [18] does not take into account the measurement redundancy between agents. A similar work was conducted in ref. [19], where the authors implemented a DRL-based IPP algorithm to search for gas leaks. Several reward functions are used in a low-resolution squared map. The conclusions of this last work allows for a better comprehension of the dependency of the uncertainty and the error with the optimal policy. This approach also addressed a case of gradient sparsity benchmarks, similar to the algae bloom case here treated. A similar problem is analyzed in ref. [20], where the *Infotaxis* path planning is addressed. In this approach, the null-gradient case is treated using planning heuristics based on bioinspired algorithms.

Other approaches using DRL for path planning have focused solely on obstacle avoidance. In ref. [21], one can find a promising example of the use of DRL techniques such as advantage actors critic for the generation of obstacle-free paths with a single flying drone. In this article, obstacles can be both dynamic and static. The policy learns to avoid obstacles by internalizing the obstacles. Another example of obstacle avoidance mechanism is found in ref. [22], where DDQL is used to solve the IPP with a single agent.

3. Statement of the Problem

IPP is defined as a sequential decision process in which the objective is to maximize the information $I(t)$ collected over time. For the multiagent case, a set of paths $\boldsymbol{\Psi}:=[\psi_1, \psi_2, \ldots, \psi_N]$that maximize the joint information will be sought with the restriction that these paths have no agent–agent or agent–obstacle collisions.

$$\boldsymbol{\Psi}^{*} = \arg\max_{\boldsymbol{\Psi}} \sum_{\psi \in \boldsymbol{\Psi}} I(t)_{\psi} \tag{1}$$

In our approach, a path will be defined by a series of measurement points X_j^{meas} for each agent j in a fleet of N agents. Each vehicle will take at each instant t an action $\mathbf{a}$ from the possible set of actions A. These actions correspond to the eight vectors of movement $[S, SE, E, NE, N, NW, W, SW]$. Each action involves moving the agent in that direction over a fixed distance d_{meas} and taking a water sample wherever the vehicle is. The paths will have a maximum length of $d_{\max}$ until the battery level reaches a safety threshold that prevents them from returning to the base. Samples are taken from a ground truth that is a static scalar field Y predefined at the start of the mission but unknown except at those points where the agents sample. The measurement model for vehicles is represented as

$$y_p = Y(\mathbf{p}) + \varepsilon_p \tag{2}$$

where $\mathbf{p} = [p_{\text{lat}}, p_{\text{long}}]$ is a vehicle position, ε_p is a noise associated with the variability of sensor measurement, and $Y:\mathbb{R}^2 \rightarrow \mathbb{R}$ is the sought-after (ground truth) function. It will be assumed that no vehicle can take samples or visit areas with obstacles. The navigation limits of the ASVs are indicated by a navigation map $M:\mathbb{R}^2 \rightarrow \mathbb{R}$, where $M(p) = 0$ in the areas that are unreachable. Additionally, no vehicle may be in the same zone at the same time. Two vehicles are considered to be in the same zone when the distance between them is less than d_{safety}. This is a strong constraint to ensure the safety and integrity of the fleet, in addition to reducing redundancy in the measurement. In relation to the initial point, each ASV j has a valid deployment zone on each map $\mathscr{Z}^j$. No vehicle can start outside its zone due to coastal security restrictions. It will be assumed that the values of Y are normalized between 0 and 1, called the normalized contamination index (NCI), for better comparison between applications. In this case, a value of 0 is considered a value outside of any biological risk and 1 a value with high biological risk or high water contamination.

Given this set of hypotheses and statements, it is possible to formulate the problem as a partially observable Markov decision process (POMDP). This type of decision process can be expanded to the multiagent case by defining multiple agents and a partially observable state s_t, only accessible through an observation function $o_j^t = \mathcal{O}(s^t)$ by the agent j. The optimization goal of any POMDP is to find the optimal policy $\mathbf{a}_j^{t+1} = \pi^*(o_j^t)$ that maps an observation o_j^t into an action that maximizes the accumulated reward given by $\sum_{t=0}^{T} R(s^t, \mathbf{a}_j^t)$ and over some optimization time budget T for each agent $j \in [1, N]$ in the fleet. From the complete state s^t that gathers all the information for the environment, the complete ground truth scalar field is ignored, and it is only possible to know at each instant t the positions of the ASVs, the navigation map $\mathscr{M}$, and the samples that have been taken so far by each vehicle $\{Y(X^{\text{meas}}), X^{\text{meas}}\}$.

3.1. Ground Truth Models

This article focuses on two cases of contamination for natural water resources. First, it is the case of monitoring smoothly distributed physicochemical parameters (WQP) such as pH, dissolved oxygen, oxidation–reduction potential (ORP), etc. As described in ref. [6], these parameters can be characterized as mountains and valleys of different heights randomly distributed on the navigable surface (see **Figure 1**a). For the second benchmark,the algae blooms case, the distribution of cyanobacterial clusters is more localized, exhibiting higher variation of gradients at some locations. To obtain this ground truth, we present an Algae Bloom simulator based on a simplified diffusion model of blue–green algae bacteria (see Figure 1b). Up to 3 random algae blooms can appear in anywhere in the waters. We treat these bacteria from the algae bloom as particles with random speeds v_r for each one, to model the diffusion effect of contaminants in the surface of the waters. Additionally, we introduce two speeds related to wind speed v_w and water currents v_c. These three components are weighted to compose a final speed. The position of every particle is updated depending on these speeds by computing the discrete integral with a fixed time step Δt. Finally, to model the effects of the shores, the physical boundaries of the navigable zones exert a pushing-back force to the particles. The algae bloom simulations will be termed static when the blooms are simulated for a random period and then considered stationary. Chlorophyll or turbidity sensors are often used to measure them. These blooms also respond to the dynamics of the tide and wind, acting like surface particles. The framework starts by simulating the dynamics of the algae blooms for a random amount of time prior to any mission to start with any possible state during the algae dissemination process. In the end, it is proposed to model both phenomena in two ground truth generators that provide the learning algorithm with randomly generated scalar functions $f(X)$.

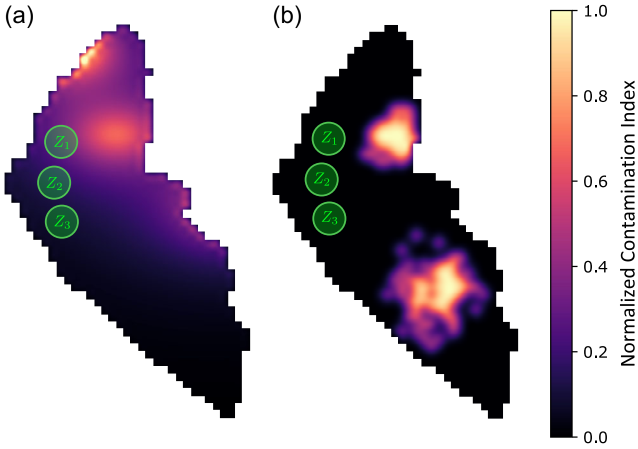

Figure 1. Example of the ground truths used for every mission. a) The WQP map. b) An example of an algae scenario with two blooms. In green, Z_1, Z_2, Z_3 correspond to the initial deployment zones of the vehicles. The initial position of every vehicle is randomly selected within this area.

In both cases, the functions are considered static since the dynamics of the parameters to be measured is much slower than the total time to complete a mission (about 4h).

3.2. Assumptions

The following assumptions are used throughout this article. 1) The navigable waters map will be the same from one episode to another and is obtained by the real navigation map of the water resource under monitorization. 2) This article assumes that vehicles must be homogeneous, with equal movement and measurement capabilities. 3) The vehicles can take the actions and reach the target points without any problem. No moving obstacles are considered within the scenario other than the existence of the other vehicles. 4) Vehicles do not have to end up in the same place as they started. They are considered to have sufficient autonomy to return to shore from any point at the end of the monitoring. 5) It is necessary to have a prior approximate behavioral model of the information. 6) Measurement noise from vehicles is considered to be negligible.

4. Methodology

In this section, the methodology used to 1) perform the model estimation online with local GPs and 2) train the ASVs policy using DRL is described.

4.1. Local Gaussian Process for Estimation

A GP is a stochastic process that is fully specified by a mean function $\mu(x)$ and a covariance function $k(x, x')$. GPs are used in this work, similarly to ref. [6], as online prediction methods to obtain a contamination model. This model will serve, on the one hand, as an estimation of WQP/algae concentration and, on the other hand, as an observer of the hidden state of the POMDP. A GP is denoted by

$$f(\mathbf{x}) \sim GP(\mu(\mathbf{x}), k(\mathbf{x},\ \mathbf{x}')) \tag{3}$$

A GP is a nonparametric probabilistic approach that can be used to estimate an unknown function $f(X)$, given a set of input–output pairs $(\mathbf{X}, \mathbf{y})$.[9] In the case of WQP, $\mathbf{X}$ represents the sampling locations, and $\mathbf{y}$ represents the measured WQPs at these locations. GP assumes that the joint distribution of the output values $\mathbf{y}$ is Gaussian, with a mean function $\mu(\mathbf{x})$ and a covariance function $k(\mathbf{x}, \mathbf{x}')$ that specify the similarity between any $\mathbf{x}$ and $\mathbf{x}'$ in the input space. The GP prediction for a set of measured points $\{\mathbf{X}^{\text{meas}}, \mathbf{x}\}$ is given by

$$\begin{aligned} & \hat{f}(X) = \mathbf{k}_*^T(\mathbf{K} + \sigma_n^2\mathbf{I})^{-1}\mathbf{y} \\ \text{where} \quad & \mathbf{k}_* = [k(\mathbf{x}^*, \mathbf{x}_1^{\text{meas}}), \dots, k(\mathbf{x}^*, \mathbf{x}_n^{\text{meas}})]^T \\ & \mathbf{K} = [k(\mathbf{x}_i^{\text{meas}}, \mathbf{x}_j^{\text{meas}})]_{i,j=1}^n \end{aligned} \tag{4}$$

In this equation, $\mathbf{k}*$ represents the vector of kernel evaluations between the new input point $\mathbf{x}^*$ and all training input points $\mathbf{x}_i^{\text{meas}}$, $i = 1, \dots, n$. $\mathbf{K}$ is the covariance matrix between all training input points, and σ_n^2 is the noise variance of the observations. The GP prediction can be conditioned on observed data by incorporating the training data in the mean and covariance functions. Specifically, given the set of training data $(\mathbf{X}^{\text{meas}}, \mathbf{y})$, where $\mathbf{X} = [\mathbf{x}_1^{\text{meas}}, \dots, \mathbf{x}_n^{\text{meas}}]^T$ and $\mathbf{y} = [y_1, \dots, y_n]^T$, the posterior distribution of the uncertainty values at a new input point $\mathbf{x}*$ can be written as

$$\boldsymbol{\sigma}^2(\mathbf{x}^*) = k(\mathbf{x}^*, \mathbf{x}^*) - \mathbf{k}_*^T(\mathbf{K} + \sigma_n^2\mathbf{I})^{-1}\mathbf{k}_* \tag{5}$$

In this equation, $\boldsymbol{\sigma}^2(\mathbf{x}^*)$ represents the variance of the posterior distribution of the function value at a possible new measurement point.

The choice of the kernel function defines how the input variables are correlated. An immediate choice for the task of WQP monitoring, as explained in ref. [6], is to use an RBF-type kernel, which reduces the correlation exponentially with the distance between samples. This kernel, then, imposes a smooth structure modeled by its lengthscale, which in most cases is sufficient to conform a good model.

$$k_{\text{RBF}}(\mathbf{x}, \mathbf{x}') = \sigma_0^2 \exp\left(\frac{-\|\mathbf{x} - \mathbf{x}'\|^2}{2\ell}\right) \tag{6}$$

In GPs, the hyperparameters $\boldsymbol{\theta} = (\sigma_0^2, l)$ are learnt from the training data by maximizing the type-II log-likelihood function, which is the likelihood of the hyperparameters given the observed data $\{\mathbf{X}^{\text{meas}}, \mathbf{y}\}$.[9]

However, in the classical approach where the whole search space $\mathbf{X}$ is estimated with the same GP, it is implicitly assumed that the same hyperparameters, for example, l defining the smoothness of the estimated function $\hat{f}(\mathbf{x})$, are valid for all regions of the explored environment. This, as will be seen later, may be incorrect for functions with local behaviors.

To deal with benchmark function with different levels of continuity and gradient smoothness, such as those in Figure 1, the use of local GPs is proposed. Local GPs consist of a set of GPs that are only valid on a subset of the total search space $\mathbf{X}_{\text{local}}$. First, to this end, it is possible to define a set of centroids $\mathbf{c}_k$ homogeneously distributed over the map. Each centroid defines a GP and has a radius of action of ν_k. Thus, a model with K local GPs is defined as

$$\begin{aligned} & GP_{\text{local}} := \{GP_1, \dots, GP_K\} \\ \text{where :} \quad & GP_k = GP(\boldsymbol{\mu}, k, \mathbf{c}_k, \nu_k) \end{aligned} \tag{7}$$

To improve consensus between local GPs when fitting the hyperparameters online, several shared zones of influence surge, where a sample $(\mathbf{x}, y)$ is used to fit several processes simultaneously (see **Figure 2**a). Shared areas guarantee better smoothness in the limits of the local areas. The level of redundancy and the granularity of the processes is defined with $(\nu_k, \mathbf{c}_k)$ parameters and will be adjusted for each case.

To compute a total joint mean $\hat{\boldsymbol{\mu}}(\mathbf{x})$ and uncertainty $\hat{\boldsymbol{\sigma}}(\mathbf{x})$ for a point $\mathbf{x}$, a weighted mean among all values assigned by the GPs will be used. The influence of each GP on a value x, will be proportional to the distance from that location to the centroid of that GP. This is convenient since the whole model can easily reach a consensus between GPs because they often see the same data in shared zones, while being robust against outliers or a possible nonconvergence of a local GP where a new sample does not improve a particular GP. This joint model, as depicted in Figure 2a,b, is transparent to the sampling process and produces an output with the same size of a global GP. Thus, the joint mean and uncertainty $(\hat{\boldsymbol{\mu}}(\mathbf{x}), \hat{\boldsymbol{\sigma}}(\mathbf{x}))$ can be defined as

$$\hat{\boldsymbol{\mu}}(\mathbf{x}) = \sum_{i=1}^{K} \mu_i(\mathbf{x}) e^{-\|\mathbf{x}-\mathbf{c}_i\|_2} \left(\sum_{i=1}^{K} e^{-\|\mathbf{x}-\mathbf{c}_i\|_2}\right)^{-1} \tag{8}$$

$$\hat{\boldsymbol{\sigma}}(\mathbf{x}) = \sum_{i=1}^{K} \sigma_i(\mathbf{x}) e^{-\|\mathbf{x}-\mathbf{c}_i\|_2} \left(\sum_{i=1}^{K} e^{-\|\mathbf{x}-\mathbf{c}_i\|_2}\right)^{-1} \tag{9}$$

Any Gaussian model also allows the imposition of a prior on the information obtained. In both ground truths, a prior of mean 0 is imposed, assuming that the measurements are normalized. This makes it possible to establish that, in the absence of measurements and when uncertainty is at a maximum, leading to zero correlation, the estimated value at those points coincides with the prior. The proposed local GPs follow the same rule. This implies some knowledge of the scalar field to be measured. Any other type of ground truth should consider how the information behaves a priori. The same applies to hyperparameters. A range of possible hyperparameter values has to be considered with which to start the optimization of each GP. In our case, the initial value of all processes is chosen to be the maximum possible $l_0 = 10$, indicating that, a priori, the information is smooth. The interval of values imposed is $(l_{\min}, l_{\max}) = (0.1, 10)$, so that in the optimization, l, will be kept bounded. These parameters has been selected according to previous works[14] that addressed the suitable values for WQP-related functions. The selection of the radius $\boldsymbol{\nu}_i$ of the local GPs will be selected accordingly. As the l is a measurement of the spatial correlation between two samples, the diameter of the area must be as large as the maximum expected $l_{\max}$. This value can be selected by simple observation of the benchmark functions under study. In practice, a set of of two samples inside of an influence area that shows a high

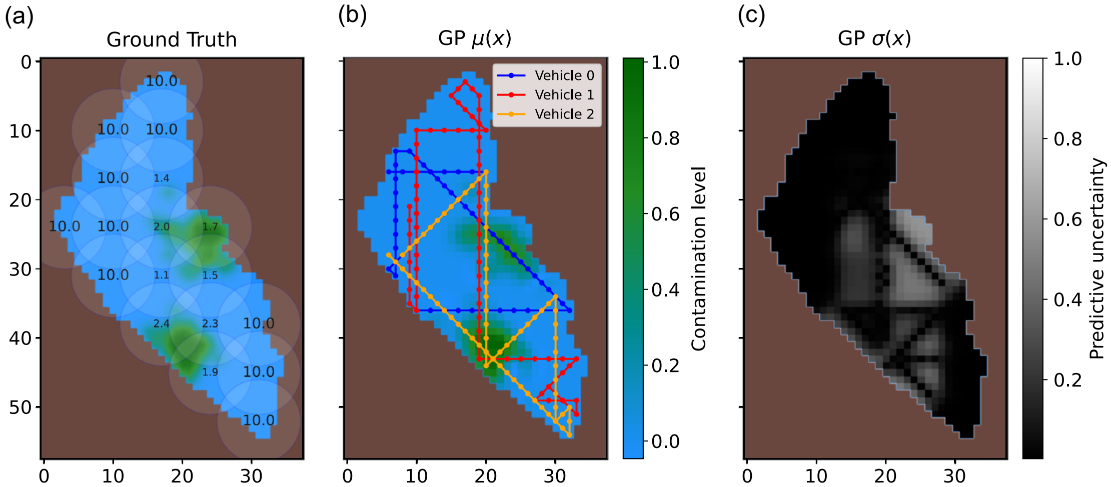

Figure 2. Local GP applied to algae bloom detection with random paths for three ASVs. a) The local GpS influence areas and the ground truth. b) The synthesized model from the local GP $\hat{\mu}(x)$. In c), the joint predictive uncertainty $\hat{\sigma}(x)$.

likelihood for a high length scale will collapse to the maximum and these measurements will be equivalent to uncorrelated or very far away samples, with a distance higher than $l_{\max}$. The selection of the $\boldsymbol{\nu}_i$, in the case of the algae bloom scenario, for example, is imposed by the maximum expected size of the blooms. In any case, the larger the areas of the local GPs given a predefined position of the centroids, the more redundant the model. With regard to the position of the centroids $\mathbf{c}_i$, a homogeneous grid with an equidistant distribution of the centroids (see **Figure 3**) has been imposed. The distance is such that every zone shares a radius of ν_i with every four-connected neighbor zones to allow consensus in the hyperparameter selection.

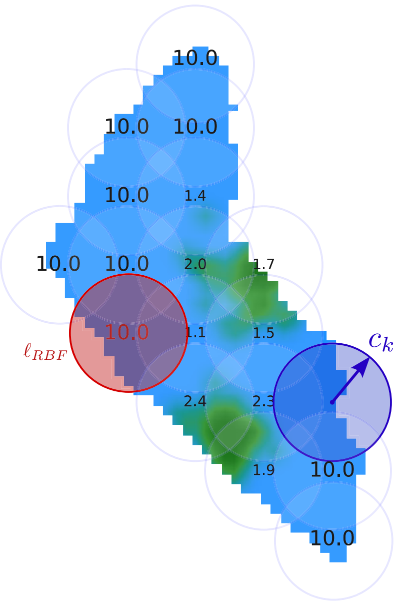

Figure 3. Example of the local GP zones with a radius c_i distributed in the Ypacaraí lake scenario. The different values of every zone represent the posterior value of the RBF kernel ℓ_{RBF} when an homogeneous sampling is applied.

4.2. Deep Reinforcement Learning

DRL is a subfield of machine learning that combines deep neural networks with reinforcement learning to enable agents to learn to make optimal decisions in complex environments.[11] In DRL, an agent interacts with an environment and receives rewards or penalties for its actions. The goal of the agent is to learn a policy that maximizes the expected cumulative reward over time.[8] In this article, the algorithm DDQL is proposed as a common and successful framework to optimize discrete action policies.[10,17,22] DDQL uses a deep neural network to approximate the action-value function $Q(s,\mathbf{a})$.[11] The action-value function is a function that maps a state-action pair to the expected cumulative reward. The Q-learning algorithm uses an iterative process to update the Q-values based on the observed rewards and the discounted future rewards. The updates are given by the Bellman equation.[23]

$$Q(s_t,\mathbf{a}_t) \leftarrow Q(s_t,\mathbf{a}_t) + \alpha\left[r_{t+1} + \gamma \max_{\mathbf{a}'} Q^{\text{target}}(s_{t+1},\mathbf{a}') - Q(s_t,\mathbf{a}_t)\right] \tag{10}$$

where s_t is the state at time t, $\mathbf{a}_t$ is the action taken at time t, r_{t+1} is the reward received at time $t+1$, α is the learning rate, and γ is the discount factor that controls the importance of future rewards. The max operation selects the action with the highest Q-value in the next state. DQL uses a deep neural network $Q(s_t,\mathbf{a};\boldsymbol{\theta})$ with parameters $\boldsymbol{\theta}$ to approximate the action-value function. The network takes a state s as input and outputs Q-values for each action. The loss function for the network is defined as

$$L(\theta) = \mathbb{E}[(r_{t+1} + \gamma \max_{\mathbf{a}'} Q^{\text{target}}(s_{t+1},\mathbf{a}';\boldsymbol{\theta}^-) - Q(s_t,\mathbf{a}_t,\boldsymbol{\theta}))^2] \tag{11}$$

where Q_{target} is a target network with frozen parameters $\boldsymbol{\theta}^-$ that are used to generate the targets for the Q-values.[11] The Q-learning algorithm updates the parameters of the network by minimizing the loss function using stochastic gradient descent.

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \frac{\partial L(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \tag{12}$$

As the agent interacts with the scenario, it will generate $(s^t, \mathbf{a}^t, r^t, s^{t+1})$ experiences that will be stored in a buffer memory that is fed to the optimization algorithms, adjusting the Q value to the new batch at each optimization step. For learning to be effective, an initial exploration phase of the state-action space is required. An ε-greedy policy is used, in which each agent will take with a probability of ε a random action and with a probability $1 - \varepsilon$ the optimal action indicated by the function Q, that is, $\mathbf{a} = \max_{\mathbf{a}'} Q(s, \mathbf{a}'; \boldsymbol{\theta})$. To balance the exploration-exploitation of the network, ε is annealed from 1 (full random) to a minimum value of exploration $\varepsilon_{\min}$ (greedy).

4.3. Observation Function

To define the observation $\mathbf{o}_j^t = \mathcal{O}(s^t)$ of an agent j, this work resorts to a visual description of the scenario. Each map of the scenario is discretized into an $[m, n]$-pixel matrix. Such discretization helps to reduce the complexity of the problem and, as mentioned in ref. [17], it is convenient due to the spatial correlations between vehicle positions and areas of interest. Moreover, these visual states can be directly forwarded by convolutional neural policies as it will be explained later. Thus, the observation will be composed of five channel images. 1) The mean of the local GP $\hat{\boldsymbol{\mu}}(X)$. 2) The predictive uncertainty of the local GP $\boldsymbol{\sigma}(X)$. 3) The navigation map $\mathscr{M}$ where values 1 indicate navigable positions. 4) A null matrix with the position of vehicle j with value 1. 5) A null matrix with the positions of the other vehicles j^- with value 1.

All these images will be min–max normalized to be between 0 and 1.

4.4. Reward Function

The definition of the reward function directly impacts the behavior of the fleet. Therefore, its definition is fundamental for acquiring the desired results.[8] The reward function, $r(s, a)$, quantitatively determines how good or bad an action a is in a state s and must be aligned with the ultimate goal, which is to obtain a model as accurate as possible. To encourage agents to explore the environment, it is necessary to define a metric that evaluates the information gain from one instant to the next. In previous work, as in ref. [6], the utility function is based on the expected improvement of the function. This article proposes to use two different reward functions and compare performance and alignment with the final goal.

First, a reward function similar to the one used in ref. [24] for monitoring forest fire scenarios will be tested. In this work, the reward is directly proportional to discovered ignited cells in a discretized wildfire scenario. Thus, this reward function takes into account the difference between the model at two consecutive timesteps. This article will modify this function to reward the absolute value of the changes from a previous to a later mean of the model $\Delta\mu^t(\mathbf{X}) = \sum |\hat{\boldsymbol{\mu}}^t(\mathbf{X}) - \hat{\boldsymbol{\mu}}^{t-1}(\mathbf{X})|$. This reward provides higher values when the changes between the posterior and the prior increase. This is motivated by the fact that obtaining data that changes its mean with respect to the prior means a better estimation, thus quantifying the quality of an action. The changes in the model are directly related to the Kullback–Leibler (KL) divergence as shown in Equation (13) for a multivariate Gaussian distribution.

A higher KL divergence will is directly related to the degree of change of the GP model experience when adapting new data. Note that in Equation (13) both the changes in uncertainties and means of two distributions impact the divergence. Consequently, a increased divergence would imply that from two consecutive steps, more valuable information is being utilized by the model.

$$KL(GP_1 \to GP_2) = \frac{1}{2}\left[\log\frac{|\Sigma_2|}{|\Sigma_1|} - d + tr\{\Sigma_2^{-1}\Sigma_1\} + (\Delta\mu)^T \Sigma_2^{-1}(\Delta\mu)\right] \tag{13}$$

In addition to this criterion, it is also proposed to use a reward function that benefits those actions that reduce the predictive uncertainty of the GP as much as possible. With the formulation proposed in Section 4.1, the predictive uncertainty will be adjusted through the kernel parameters online by means of likelihood maximization. Where the lengthscale is smaller, reducing the uncertainty requires a larger number of samples and vice versa. Thus, the reward will be proportional to the absolute change in the uncertainty $\Delta\sigma^t(\mathbf{X}) = \sum |\hat{\boldsymbol{\sigma}}^t(\mathbf{X}) - \hat{\boldsymbol{\sigma}}^{t-1}(\mathbf{X})|$. Note that the absolute value is taken because, as the hyperparameters are optimized, the covariance term changes, and the uncertainty could increase or decrease when new data are taken. Therefore, a change in the total uncertainty, regardless of the sign, also has an impact on the model improvement, since this shift in the new hyperparameters also contributes to the model accuracy.

In the multiagent case, it will be necessary to distribute the rewards according to the impact of each action on the total improvement of the model and to avoid that an agent who performs a bad action gets a biased reward for an action of another agent. Here, there are two important components.

First, it is a locality factor in the change, and it is only considered as part of the reward model those changes in an area of influence I of radius R around each agent. This radius is the same as the level of influence of local GPs. Second, a redundancy factor $\rho(\mathbf{x})$ is introduced for each zone within the radius area of influence of the ASVs equal to the radius of the local GPs ν_k. The value $\rho(\mathbf{x})$ is the number of agents who share the changes at that particular location $\mathbf{x}$. This redundancy factor will divide the aforementioned changes in the uncertainty or model, meaning that two ASVs that take the sample too close will receive half of the reward each. In **Figure 4**, the reward parameters are visually depicted.

In summary, the two reward functions for every agent j are

$$r_\mu(s^t, a_j^t) = \sum_{\mathbf{x}\in I_j}\left[\frac{|\Delta\mu^t(\mathbf{x})|}{\rho_j(\mathbf{x})}\right] \tag{14}$$

$$r_\sigma(s^t, a_j^t) = \sum_{\mathbf{x}\in I_j}\left[\frac{|\Delta\sigma^t(\mathbf{x})|}{\rho_j(\mathbf{x})}\right] \tag{15}$$

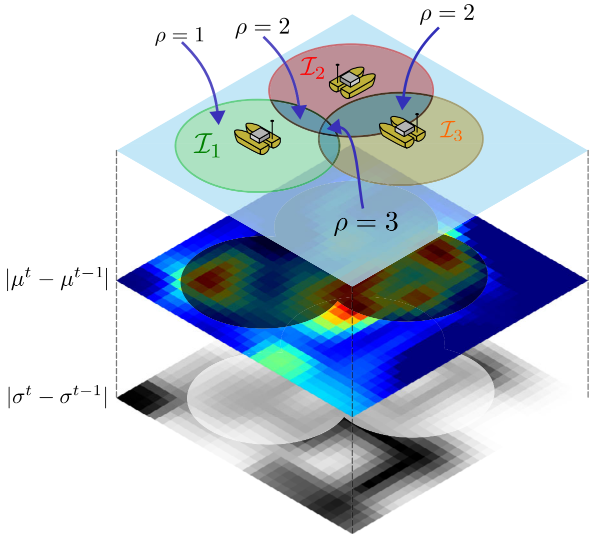

Figure 4. Influence areas *I* for every vehicle and its corresponding redundancy values ρ.

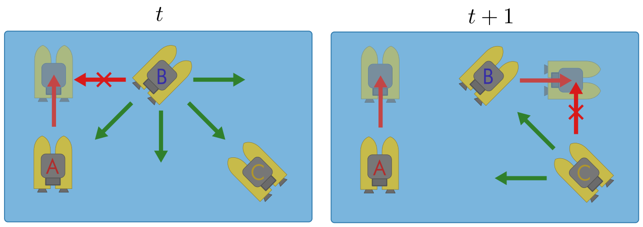

Figure 6. Consensus scheme for the safe action selection. At instant *t*, the agent with higher *Q* chooses its action first. Then, the second agent takes an action rejecting any that causes collision. This is repeated until all agents have decided the next action, and a consensus is reached.

4.5. Deep Safe Policy for Multiagent Training

For this multiagent application, a single neural network for all agents will be used. This technique, called parameter sharing, has been shown to be effective in previous work.[10,24] The difference of this proposal from previous work is that the same neural network is able to accommodate a different number of vehicles due to the egocentric formulation of the observation. As agents are interchangeable in perception and actions, all generated experiences are stored indistinctly in the experience buffer for later use in training.

The proposed neural network is presented in **Figure 5**. This neural network is composed of a first stage in the form of a convolutional encoder, as originally proposed in ref. [11]. This stage extracts the visual features of each observation o_j^t to produce an output $Q(o_j^t, \mathbf{a})$. The three consecutive convolutional layers are followed by three fully connected neural network layers. Following these, the values are unfolded into two heads for the computation of the value function $V(s)$ and the advantage function $A(s, \mathbf{a})$. This way of representing $Q(s, \mathbf{a})$ is based on ref. [25], which allows for a better representation of the cumulative reward. Finally, according to ref. [25], the value of $Q(s, \mathbf{a})$ is calculated as follows.

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

While DRL effectively learns obstacle avoidance,[17] deterministic computation can address actions leading to agent–scenario collisions.[22] However, collision between agents is more complex as simultaneous actions may cause conflicts. This work proposes a heuristic based on conditional decision-making to prevent such situations. Agents are sorted based on the highest joint value of *Q*, with the highest-Q agent taking an action without considering other agents. Subsequent agents consider the new position of the previous one, censoring *Q* values leading to collisions with $-\infty$. Once actions are decided, movements are processed to prevent collisions (see **Figure 6**). This heuristic relies on agent optimism to prioritize actions. In cases of random actions (following ε-greedy policy), only safe actions are considered, avoiding

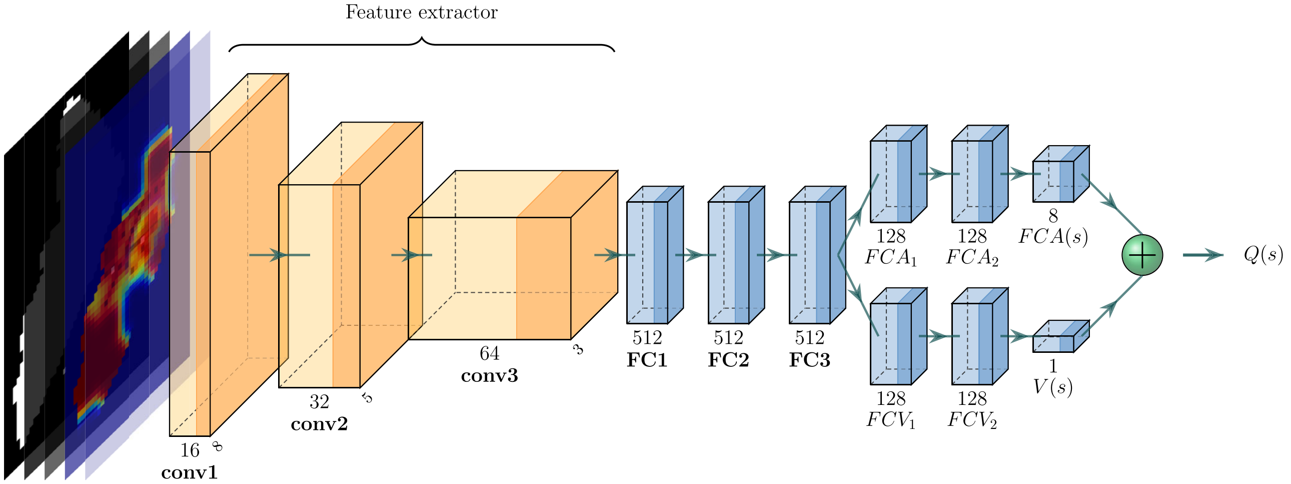

Figure 5. Dueling neural network architecture for the *Q*-function representation. It is composed of an initial visual encoder and two heads: i) the advantage head and the value head. The outputs are the 8 *Q*-values.

collisions. Pseudocode for the safe dueling DQL algorithm is provided in **Algorithm 1**, with the consensus subroutine outlined in **Algorithm 2**.

Algorithm 1. Safe Multiagent DDQL Algorithm.

```
Initialize replay memory $D$ to capacity $|D|$
Initialize target Q-network $Q^{'}$ with weights $\theta' = \theta$
Initialize policy network $Q$ with weights $\theta$
for episode $= 1$ to $E_{\max}$ do
    Reset environment
    Get initial observation $o_0 = O(s_0)$
    for timestep $= 1$ to $T$ do
        $p \sim U(0, 1)$
        if $p < \varepsilon$ then
            $a_j \leftarrow SafeConsensus(U(0,1), \ldots, U(0,1))$
        else
            $a_j \leftarrow SafeConsensus(Q(o_0, a), \ldots, Q(o_N, a))$
        end if
        Execute action $a_j$
        Observe rewards $r_j$ and new observations $o_j^{t+1}$
        Store every transition $(o_j^t, a_j^t, r_j^t, o_j^{t+1})$ in $D$
        Sample random batch $B$ of $(o_j, a_j, r_j, o_{j+1})$ from $D$
        Set $y_j = r_j + \gamma Q^{'}(s_{j+1}, \arg\max_{a} Q(s_{j+1}, a; \theta); \theta')$
        Update weights by minimizing the loss:
            $\mathcal{L}(\theta) = \frac{1}{B}\sum_{j=1}^{B}(y_j - Q(s_j, a_j; \theta))^2$
        $\theta' \leftarrow \theta \times \tau + (1 - \tau) \times \theta'$ ▷ Polyak target update
    end for
    $\varepsilon \leftarrow \min(\varepsilon_{min}, \varepsilon - d\varepsilon)$.
end for
```

Algorithm 2. SafeConsensus algorithm.

```
Input: Positions $P^t = p_1^t, p_2^t, \ldots, p_N^t$ of $N$ agents at time $t$
Input: Values $Q = \{Q_1, Q_2, \ldots, Q_N\}$ that weight each agent's action.
Initialize empty set of future positions $P^{t+1} := \emptyset$
Obtain order of agents' actions in decreasing order of their $Q$ values:
  $j_1, j_2, \ldots, j_N$, such that $\max Q_{j_1} \geq \max Q_{j_2} \geq \ldots \geq \max Q_{j_N}$.
for each agent $j$ in order of actions do
    Select greedy safe action
      $a_j = \arg\max_{a \in A} Q_j(a)$
      subjected to :
      $\|P(p_j^t + a_j) - p' P\|_2 \leq d_{safe} \quad \forall p' \in P^{t+1}$
    $P^{t+1} \leftarrow P^{t+1} \cup (p_j^t + a_j)$ ▷ Update next fleet positions.
    $A_{\text{selected}} \leftarrow A_{\text{selected}} \cup a_j$ ▷ Update consensus actions.
end for
return $A_{\text{selected}}$
```

5. Simulations and Results

In this section, all the experiments and simulations performed to validate the optimality of the algorithm are presented. First, the performance in terms of computation time and accuracy of local GPs along with the classical approach of global GPs is analyzed for both WQP algae bloom benchmarks. Then, the results of the DRL training are presented with both designed rewards and both benchmarks, with a discussion on the scalability of the proposal. Finally, the results are compared with other algorithms used in previous approaches in the literature.

All simulations and training were conducted on an Ubuntu 22.04 server, with 256Gb RAM, Dual Xeon CPU Scalable SP3 HPC, and two different graphic processing units: 1) Nvidia RTX 3090 25GB and 2) Nvidia Quadro A4000 48GB. Python 3.10 and PyTorch were used for policy optimization. All simulation parameters and constraints are summarized in **Table 1**. The code will be available for reproduction of the results in https://github.com/derpberk/.

For the analysis of the results, the comparison between rewards and in three algorithms, the following metrics must be defined.

Sum of Residuals (SoR): This defines the absolute sum of error between the estimated mean of the GP $\hat{\boldsymbol{\mu}}(\mathbf{x})$ and the ground truth value $f(\mathbf{x})$. This metric is intended to be minimized.

$$\text{SoR} = \sum_{\mathbf{x}\in\mathbf{X}} |\boldsymbol{\mu}(\mathbf{x}) - f(\mathbf{x})| \tag{17}$$

Different from other approaches,[6,16] the absolute error will be used to analyze the model accuracy, as the mean squared error does not reflect well small improvements into the model error.

Normalized Sum of Residuals (nSoR): Normalization of the SoR with respect to the amount of information available in ground truth.

$$\text{nSoR} = \frac{\sum_{\mathbf{x}\in\mathbf{X}} |\boldsymbol{\mu}(\mathbf{x}) - f(\mathbf{x})|}{\sum_{\mathbf{x}\in\mathbf{X}} f(\mathbf{x})} \tag{18}$$

Average Error in $f(x)$ **local maxima** (The local maxima is computed in both ground truths by applying a maximum filter with a neighborhood of 1.5 km and taking the locations where the magnitude of the second-order derivative term is 0 using a Sobel

Table 1. Environment and model parameters.

Parameter	Value
Number of GPs (K)	18
Influence radius (ν_k)	1.45 km
l interval ($l_{\min}, l_{\max}$)	(0.1, 10)
Base uncertainty (σ_0)	1.0
Measurement noise (σ_n)	1×10^{-5}
Max. distance ($d_{\max}$)	29 km
Safety distance (d_{safety})	300 m
Movement distance (d_{meas})	580 m
Map size (H, W)	(58, 38) pixels

filter.): This defines the mean error in the local maxima of the ground truth $f(\mathbf{x})$. This metric is useful to obtain a measurement of the error in the most biologically dangerous spots.

$$\text{Avg. SoR} f(\mathbf{x}^*) = \frac{1}{\#\,\text{peaks}\{f(\mathbf{X})\}} \sum_{\mathbf{x}\in\#.\text{peaks}\{f(X)\}} |\boldsymbol{\mu}(\mathbf{x}) - f(\mathbf{x})| \tag{19}$$

Max. Error in $f(\mathbf{x})$ local maxima: This defines the max error in the local maxima of the ground truth $f(\mathbf{x})$. This metric provides a maximum bound of the error in the estimation of $f(\mathbf{x})$.

$$\text{Max. SoR} f(\mathbf{x}^*) = \max\{|\boldsymbol{\mu}(\mathbf{x}) - f(\mathbf{x})|\, \forall x \in \#.\text{peaks}\{|f(\mathbf{X})|\}\} \tag{20}$$

5.1. Local Gaussian Process Performance

Initially, 18 local GPs, distributed 2 km apart (Figure 2), were validated with simulations. The radius of influence ν_k for each process was experimentally set at 1.45 km (5 pixels) based on the granularity of the scalar fields. In **Figure 7**, 50 missions were simulated for the algae bloom scenario with three agents using non-reactive path planners and various regression algorithms (global GPs, k-nearest neighbours, decision tree) to assess local GPs with offline paths. Despite all algorithms collecting the same information at each instant, local GPs, on average, reduce the model estimation error by almost 20 points of SoR, a 33% improvement from 40 samples. With an increasing number of samples, local GPs outperform a single global GP with the same information. Other regressors, like k-nearest neighbors (kNN) (with $k = 5$), exhibit poor performance. Although decision tree accuracy drops with sufficient samples, the early inference process is slow to converge.

In **Figure 8**, computation times are compared, revealing that a global GP experiences cubic growth with the number of samples. The depicted time represents the average cumulative time spent by the model server in model inference. This scalability issue intensifies with more agents and a larger sample size. Conversely, local GPs exhibit linear time growth with the number of samples. While this might not significantly impact small fleets, scalability becomes a concern as the number of samples and vehicles increases. It's important to note that, in this simulation, no concurrency in the optimization of local GPs was implemented. However, the fully parallelizable model optimization process could further reduce computation time.

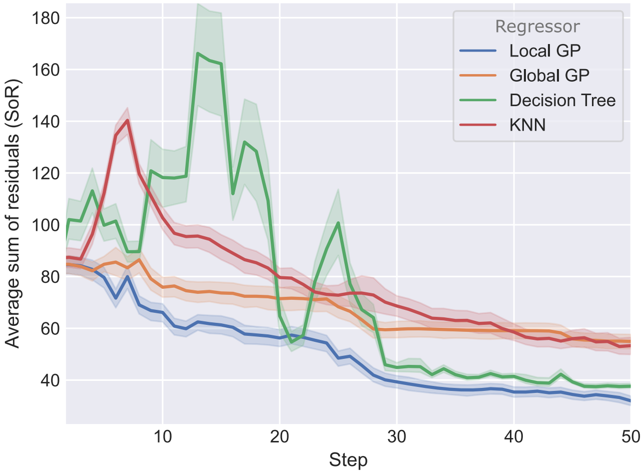

Figure 7. Accumulated inference time, between the proposed local GP and the classic global GP.

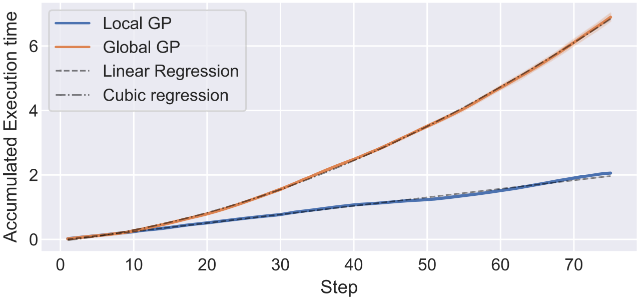

Figure 8. Comparison of error between the proposed local GPs, and other ML algorithms, in the Ypacaraí scenario, for 50 different scenarios using different path planners. The colored area is the standard deviation.

In **Figure 9**, an example of local versus global GPs is presented, giving the same information in the particular case of algae detection. It can be seen that global GPs, as formulated in ref. [6], have estimation problems in areas of higher granularity. This occurs because the global GPs maximize the marginal likelihood of the kernel hyperparameters for the entire sample space. In the case of algae monitoring, a priori it seems to be that there are two distinct zones. Such areas of low pollution concentration with high spatial correlation between samples (posterior l is high) and zones of pollution hotspots with low correlation (posterior l is small). As shown in **Figure 10**, the global GP converges in a small l at the end of the mission, but as there are several highly correlated samples in low contaminated zones, the estimation in higher zones is affected. In the local GP case, the estimation in both zones is isolated due to the locality of the individual GPs.

By constraining a maximum l to 100 (high enough to consider that all possible samples in the scenario are highly correlated), it is possible to obtain a distribution over the length scale l for the conditions of the previous experiment. In Figure 10, the resulting hyperparameter value of the posterior kernel can be observed after maximization of the likelihood at the end of every mission. In the global GP, the maximum likelihood is found in values between 3 and 10. The less correlated samples will cause this length scale to drop to represent both low-correlated and high-correlated data. In the local GP case, the histogram shows a multimodal distribution of the data. The parameter l is found to maximize the marginal likelihood of the local GPs with values lower than 10 and also with values in the limit of 100. This is translated into the environmental task into smaller values around algae blooms and higher length scales in zero-contaminated zones. This shows that the local GP is able to bring up with richer representation of an arbitrary parameter distribution. Without loss of generality, it is reasonable to say that these GPs can be used to estimate more efficiently scalar maps with richer distribution of hyperparameters.

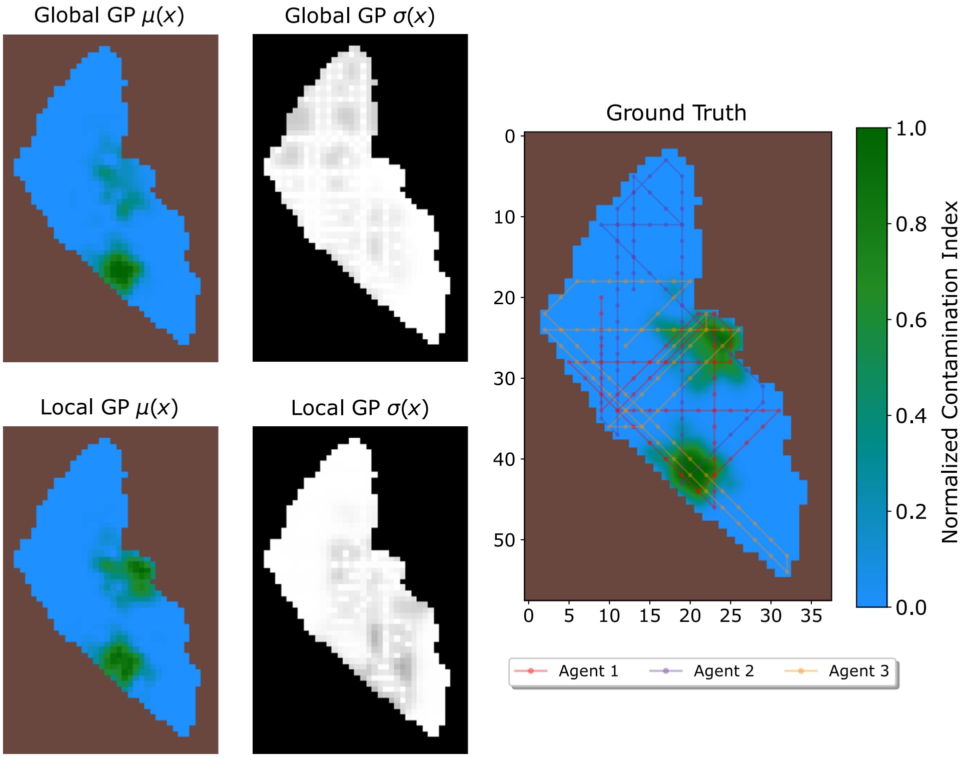

Figure 9. Comparison between the final model $\mu(X)$ and uncertainty $\sigma(X)$ using local GPs (down) and a global GP (up), with random explorative paths.

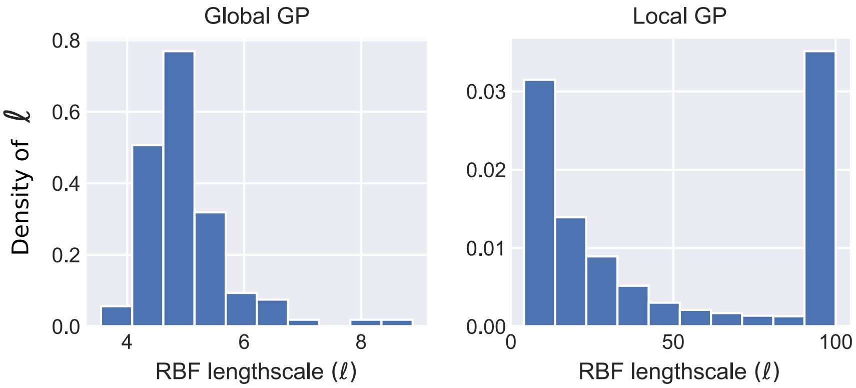

Figure 10. Histogram of RBF kernel length scales l after likelihood optimization with 100 sample points using global GPs (left) and the proposed local GPs (right).

In short, these local GPs generally alleviate the computational complexity with respect to the global counterpart. However, they still have $O(N^3)$ complexity locally, and it may happen that an excessive number of samples in a single GP severely slows down the computation. On the other hand, in terms of the convergence of the processes, the convergence of each local process is still not guaranteed. This is closely related to the choice of the internal structure of the correlation represented by the Kernel. An inadequate selection of the kernel would certainly lead to a convergence of the local process. Although the locality of the proposed GPs limits the effect on global convergence, it is still a fundamental task to select the kernel taking into account how the information we want to measure behaves.

5.2. DRL Fleet Training

For DRL policy training, two reward functions (Section 4.4) were studied across fleets of 1–3 agents, each undergoing 10 000 missions of consistent duration (29 km - 50 steps). The evaluation covered different fleet sizes and benchmarks (WQP and algae bloom monitoring) to validate the approach under varied conditions. Hyperparameters were adopted from previous studies[17]

Table 2. Learning parameters for the DDQL algorithm.

Parameter	Value
Learning rate	1×10^{-4}
Batch size	64
ε_{min}	0.05
ε episode anneal val. $d\varepsilon$	1.9×10^{-4}
τ	1×10^{-4}
Discount factor γ	0.99
Learning rate	1×10^{-4}
Activation Function	ReLU

to streamline the training process. DDQL consistently converges to a similar policy with sufficient training episodes, falling within typical literature hyperparameter ranges. The ε-greedy exploration policy maintains ε values from 1 to ε_{min} (0.05) in 50% of the episodes to balance exploration and exploitation. Neural network training employs a batch size of 64 and a learning rate of 0.0001. In **Table 2**, all training parameters are summarized.

In **Figure 11**, both rewards described in Section 4.4 are compared for every possible combination of fleet size and benchmark, after 10.000 episodes of learning with the same parameters. In general, the reward relating to changes in $\hat{\boldsymbol{\mu}}(\mathbf{X})$ is more aligned with the objective of minimizing the estimation error. In the WQP case, an improvement of ≈26% with $p < 0.05$ using a Wilcoxon ranked test can be observed. With 2 and 3 agents, the improvement is not significant, indicating that both rewards could provide similar performance. Nonetheless, the $\mu(X)$-changes reward obtains slightly better results earlier than the $\sigma(X)$-changes counterpart. The contrary happens in the algae bloom benchmark. The $\mu(X)$-change provides better and earlier performance in three fleet sizes. The error is ≈ 50% better with all fleet sizes with $p < 0.05$. This makes sense if it is considered that in the $\mu(X)$-change reward, higher rewards are received only when the model update results in a significant change. As the model is prone to improve with every new sample, the net changes in the model are an accurate estimator of the error. In the $\sigma(X)$-change reward, agents receive higher rewards even when the prior and posterior models are close to each other.

Figure 12 illustrates the cumulative reward for each fleet against prediction error, enabling the analysis of the correlation between reward and the goal of minimizing error using the R2 score. The $\mu(X)$-changes reward yields an R2 score of -4.54, while the $\sigma(X)$-reward results in an R2 score of -40.42. The $\mu(X)$-reward shows a more linear dependence with decreasing error, in contrast to the $\sigma(X)$-reward, which exhibits plateaus along each reward–error trajectory, indicating that exploring areas of low interest doesn't significantly improve the error. This comparison highlights the effectiveness of the $\mu(X)$-reward in error reduction. **Table 3** displays the average predictive uncertainty values in the map. Encouraging a subtle reduction in predictive uncertainty leads to overconfidence in initially uninteresting areas. Additionally, the $\Delta\sigma$ reward results in a 24% lower final average uncertainty compared to the $\Delta\mu$ reward.

To validate DRL-trained policies and the use of local GPs, we compare online estimation errors with those obtained using a global GP with all collected data at the end (offline GP). **Figure 13** depicts the SoR curves, showing that local GPs achieve results comparable to the global GP with full information. This indicates that, despite minor improvements, local GPs, being more flexible and time efficient, can perform as well as global

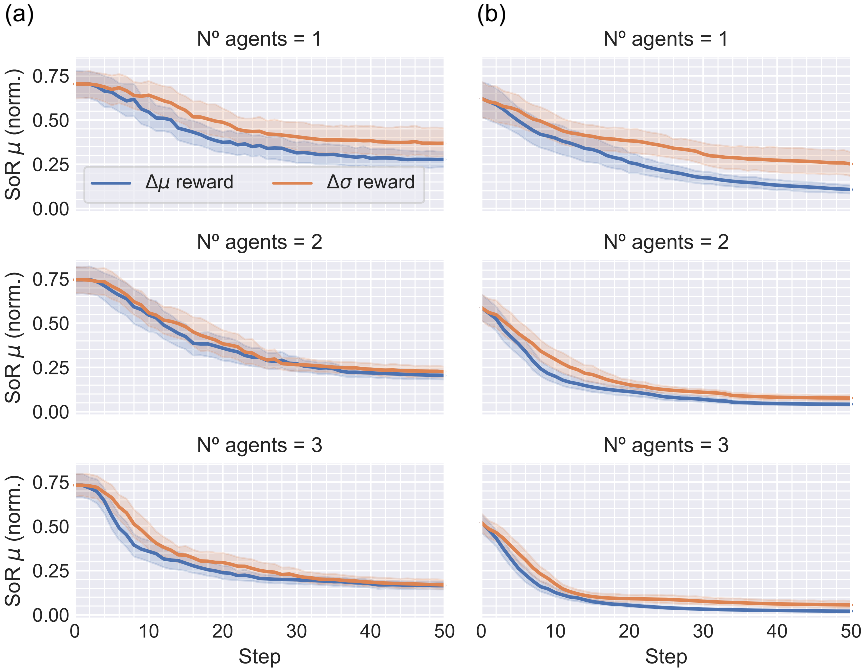

Figure 11. Estimation error (SoR) comparison between final policies trained with μ-change reward (blue) and σ-change reward (orange) in a) WQP and b) algae bloom benchmarks.

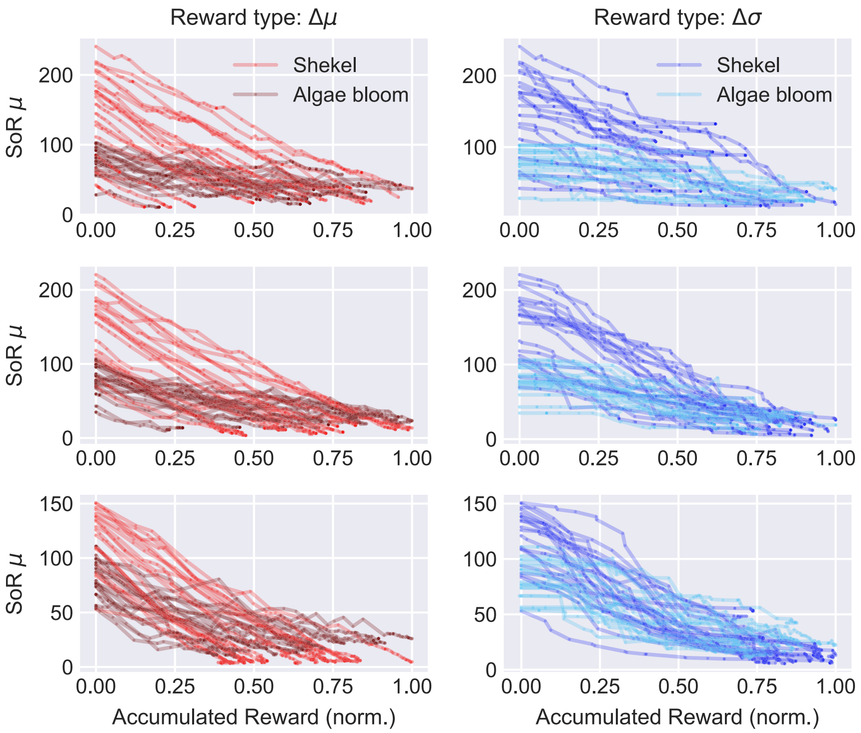

Figure 12. Accumulated reward versus the estimation error for every fleet size, reward type, and benchmark used.

Table 3. Average total uncertainty at the end of the missions, for the two reward types, fleet size and benchmarks.

GT	Reward	N° Agents	Mean $\sigma(X)$
Algae	$\Delta\mu$	1	0.419
		2	0.265
		3	0.235
	$\Delta\sigma$	1	0.338
		2	0.147
		3	0.104
WQP	$\Delta\mu$	1	0.252
		2	0.152
		3	0.024
	$\Delta\sigma$	1	0.259
		2	0.185
		3	0.050

GPs with a proper policy. The synergy between DRL for sequential information gathering and local GPs as an efficient surrogate model generator is evident, as policies demonstrate low errors independent of the model used (**Figure 14**).

5.3. Comparison with Other Algorithms

This work also compared the trained policies with other path planning algorithms to validate the results. For this comparison, the $\Delta\mu$ reward trained policies are used, as they show the best performance in terms of exploration and intensification of high-interest zones. The comparison is made with three different path planners.

Lawn Mower Path Planner (LMPP): The LMPP consists of maximizing the coverage of vehicles by taking samples in parallel lines. Every agent selects a random initial direction to initialize the path. When an obstacle is reached, the agent travels back in the reverse direction in a parallel line. This algorithm will use local GPs as a model for the contamination.

Random Wanderer Path Planner (RWPP): This approach generates random exploratory paths by selecting a direction of exploration. Every agent selects a random free-obstacle direction until a new obstacle is met. Then, the agent selects a direction different from the previous direction to avoid retracing its steps. This algorithm also uses local GPs as a model for the contamination.

GP-Enhanced Particle Swarm Optimization (EG-PSO): This approach is taken directly from ref. [16]. In this approach, every vehicle is a particle that will change its speed proportional to four distances: 1) the distance to the maximum uncertainty, 2) the distance to the maximum sampled value observed by the agent, 3) the distance to the maximum global value samples by the fleet, and 4) the distance to the maximum value predicted by the model.

Up to 300 different simulations were conducted for every ground truth type and with every benchmark. We used six different seeds to reduce the effect of epistemic uncertainty in the results. For a fair comparison, this evaluation set of ground truths will be different from any other episode experimented during training for the DQL. In **Figure 15** and **16**, the online

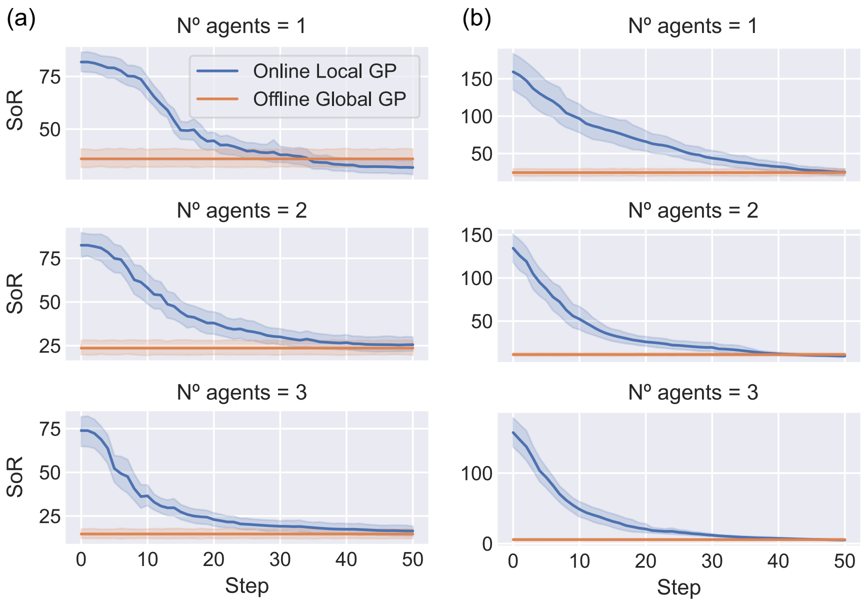

Figure 13. Comparison of the online estimation error using the best μ-change reward with local GPs, and the offline error using a global GP, with the same sample points, at the end of an episode in the a) WQP and b) algae bloom benchmarks.

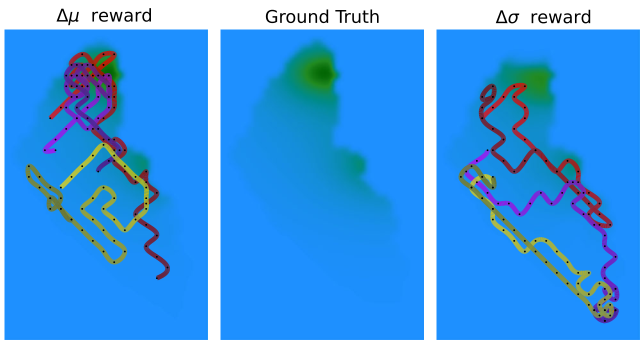

Figure 14. Resulting path of running a simulation with the $\Delta\mu$-reward (left) and the $\Delta\sigma$-reward for the same WQP monitoring scenario (middle).

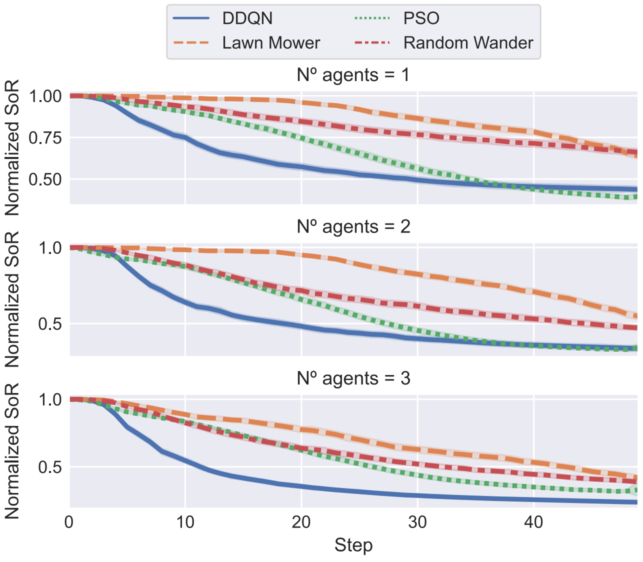

Figure 15. Estimation error between other algorithms (LMPP, RWPP, PSO) and our DDQL trained policies with the μ-change reward, for the WQP monitoring benchmark.

estimation error is represented for the WQP and algae bloom benchmarks, respectively. It is observed that, in general, the DRL is able to obtain better results. In **Table 4** (WQP benchmark) and 5 (algae bloom benchmark), the metrics for the aforementioned simulations are presented, with the mean and standard deviation values of each algorithm (**Figure 17** and 18).

In the WQP task, notable improvement, particularly in the multiagent case, is observed. Offline algorithms like LMPP or RWPP consistently reduce errors over time. DQL demonstrates adaptability, prioritizing actions with higher short- and long-term rewards, resulting in a 45% average improvement over other algorithms. LMPP, while robust with sufficient distance, tends to make inefficient movements in the WQP benchmark due to a lack of trajectory changes in low-interest areas. Random exploration behaves similarly to LMPP, but RWPP, being more exploratory, changes the monitoring front more frequently. These findings affirm that an effective IPP enhances overall modeling accuracy, even though using local GPs provides an advantage. It's reasonable to consider a significant dependence on the modeling method and information richness acquired.

In the PSO model discussed in ref. [16], a significant challenge arises in partitioning the search space among agents. Deployed in close proximity within zones Z_1, Z_2, Z_3 (as shown in Figure 1), agents exhibit a gradient-descent behavior in the multiagent case. Despite resembling a single agent due to similar local gradients caused by the absence of a dispersion mechanism, PSO achieves good convergence, especially in the single-agent scenario and at episode completion (refer to **Figure 19**). Nevertheless, our

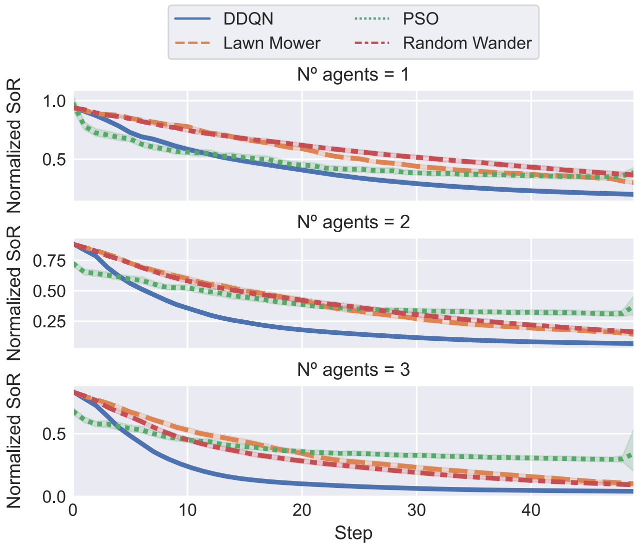

Figure 16. Estimation error between other algorithms (LMPP, RWPP, PSO) and our DDQL trained policies with the μ-change reward, for the algae bloom monitoring benchmark.

proposed algorithm, leveraging the adaptive capacity of the DDQL policy, demonstrates faster convergence even if it doesn't surpass PSO in the single-agent case (see Figure 15, Table 4).

The proposed algorithm presents good properties with respect to the time of exploration and mean error with respect to the other algorithms. This translates into an improvement of the error with the second-best algorithm (EG-PSO), on average for every fleet size, of a 32%, 15%, 6% at 33%, 66%, and 100% respectively, of the distance traveled. In the particular case of $N = 3$, with 43% of the path distance budget traveled (16 steps of 48 samples), the estimation error is 27.75% better than the second-best algorithm at that point (PSO). Regarding the metrics related to the error in the maximums of the benchmark function (Avg. SoR in $f(x^*)$ and Max. SoR in $f(x^*)$), it can be seen that the proposed algorithm is able to reduce the average error in the contamination maxima better than the other algorithms. 27% better estimation in those points has been observed on average with the DQL with respect to the second-best algorithm (RWPP).

In the second benchmark, with algae monitoring, the results also indicate the advantage of using DRL. This second benchmark is more difficult to monitor, and this is reflected in the improved results of DRL over the other algorithms. In terms of estimation at the end of a mission, the results show a 17% improvement over the second-best algorithm (LMPP) on average for all fleet sizes. In this new case, the offline algorithms present a robust but in most cases inefficient result. It can be observed in Figure 16 that during the course of a mission, both present a similar estimation error (no significant difference over a Wilcoxon ranking test). From this it can be inferred that, in this benchmark, the paths have to be less explorative and more exploitative in the search for algae sources. This compounds the need for a more comprehensive route planner that prioritizes high-interest areas in pursuit of a better model. In the end, the synergy between local GPs and an intelligent planner stands out when information is sparse in the search space, by the GPs reaching convergence earlier, in other words, to obtain a good model with less samples and less movements.

The PSO algorithm, on the other hand, due to its high dependence on the local gradients of each agent and the fact that the global maximum uncertainty point is insufficient to guide the fleet, is unable to perform on this benchmark as conceived in ref. [16]. The paths result in the absence of local gradients in a purely random scan unable to find the algae centroids in many cases. When it comes to estimation, PSO utilizes a global GP. However, the initial set of highly correlated samples causes the global lengthscale to quickly reach its upper limit with the first few samples. This, in turn, makes it difficult for the model to converge later on, especially in the presence of new samples.

The DRL algorithm, in the algae bloom benchmark, shows better improvement compared to previous approaches (see **Table 5**). With respect to the second-best result (LMPP), an average improvement of 49%, 55%, 48% at 33%, 66%, and

Table 4. Metric comparison between algorithms for the WQP modeling mission. The highlighted metrics refer to the best performance algorithm.

Algorithm	N_{agents}	SoR(33%)		SoR(66%)		SoR(100%)		Avg.SoR in$f(x^*)$		Max.SoR in$f(x^*)$	
		Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
D-DQL	1	**0.62**	0.18	**0.48**	0.17	0.44	0.17	**0.20**	0.17	**0.35**	0.30
	2	**0.53**	0.18	**0.39**	0.18	**0.34**	0.16	**0.15**	0.14	**0.24**	0.29
	3	**0.40**	0.15	**0.27**	0.09	**0.24**	0.07	**0.09**	0.08	**0.17**	0.16
PSO[16]	1	0.81	0.14	0.52	0.19	**0.40**	0.08	0.37	0.26	0.39	0.30
	2	0.75	0.15	0.41	0.15	0.34	0.06	0.22	0.14	0.24	0.18
	3	0.71	0.13	0.40	0.14	0.33	0.10	0.21	0.15	0.39	0.28
LMPP	1	0.98	0.07	0.84	0.16	0.63	0.19	0.43	0.25	0.67	0.32
	2	0.97	0.08	0.79	0.17	0.54	0.20	0.35	0.25	0.58	0.35
	3	0.83	0.18	0.60	0.22	0.41	0.17	0.24	0.20	0.40	0.31
RWPP	1	0.88	0.16	0.75	0.20	0.66	0.20	0.46	0.26	0.69	0.32
	2	0.77	0.19	0.58	0.19	0.47	0.17	0.28	0.21	0.46	0.31
	3	0.70	0.20	0.49	0.17	0.39	0.13	0.21	0.15	0.39	0.28

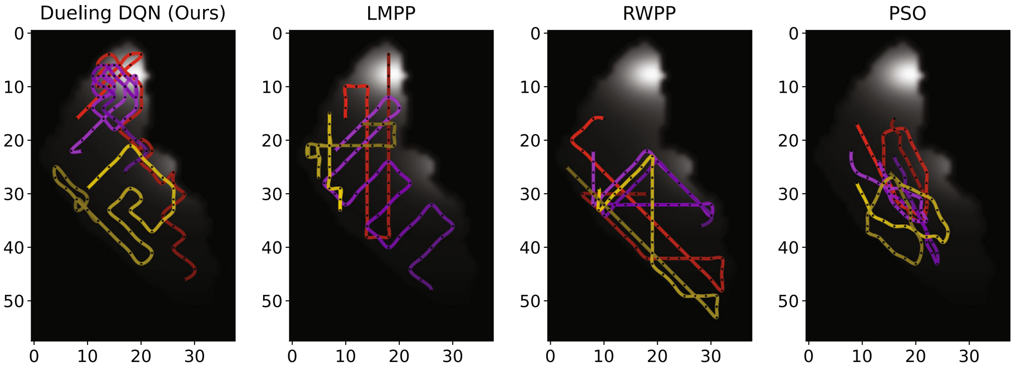

Figure 17. Resulting paths for all other algorithms (LMPP, RWPP, PSO) and our DDQL trained policy with the μ-change reward, for the WQP monitoring benchmark.

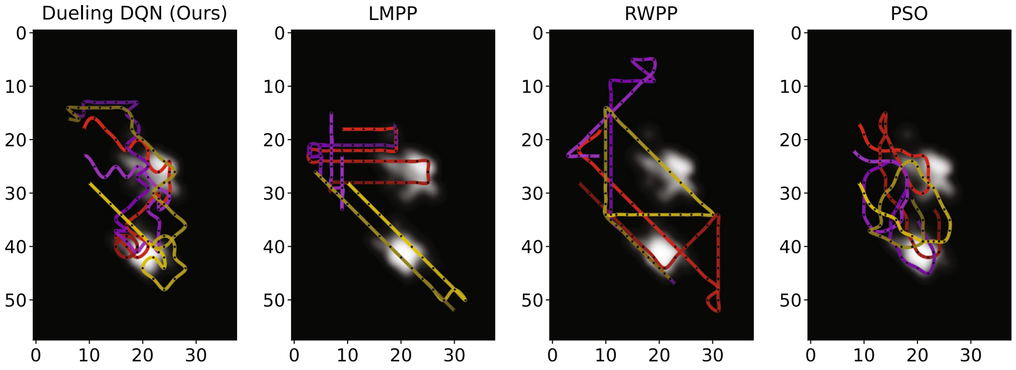

Figure 18. Resulting paths for all other algorithms (LMPP, RWPP, PSO) and our DDQL trained policy with the μ-change reward, for the algae bloom monitoring benchmark.

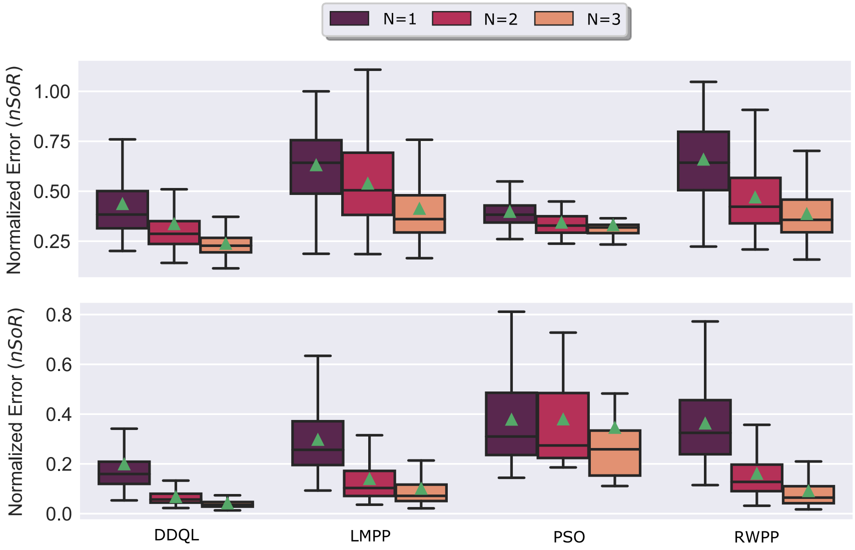

Figure 19. Box plot representation of the final normalized error for 300 experiments with different algorithms. The upper plot corresponds with WQP benchmark. The lower plot corresponds with the algae bloom benchmark.

Table 5. Metric comparison between algorithms for the algae bloom modeling mission. The highlighted metrics refer to the best performance algorithm.

Algorithm	N_{agents}	$SoR(33\%)$		$SoR(66\%)$		$SoR(100\%)$		Avg.SoR$\inf(x^*)$		Max.SoR$\inf(x^*)$	
		Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
D-DQL	1	**0.47**	0.15	0.27	0.13	**0.20**	0.14	**0.15**	0.16	**0.25**	0.27
	2	**0.23**	0.09	**0.10**	0.05	**0.07**	0.04	**0.05**	0.06	**0.09**	0.12
	3	**0.13**	0.06	**0.06**	0.03	**0.04**	0.02	**0.04**	0.04	**0.06**	0.07
PSO[16]	1	0.50	0.23	0.38	**0.18**	0.38	0.22	0.41	0.24	0.53	0.26
	2	0.44	0.25	0.33	0.16	0.38	0.22	0.43	0.22	0.60	0.23
	3	0.39	0.18	0.32	0.15	0.35	0.29	0.26	0.21	0.41	0.35
LMPP	1	0.65	0.16	0.41	0.15	0.30	0.13	0.32	0.27	0.48	0.35
	2	0.49	0.18	0.24	0.15	0.14	0.10	0.18	0.20	0.29	0.28
	3	0.42	0.17	0.21	0.13	0.10	0.08	0.12	0.17	0.21	0.26
RWPP	1	0.66	0.15	0.49	0.17	0.36	0.16	0.37	0.27	0.54	0.34
	2	0.47	0.17	0.28	0.16	0.16	0.11	0.20	0.22	0.30	0.29
	3	0.33	0.16	0.17	0.13	0.09	0.09	0.12	0.18	0.19	0.24

100% respectively, of the distance traveled, among all fleet sizes, is obtained. Improvement is also translated into higher speeds of model convergence. With every agent included in the fleet, the DRL is able to reduce the error earlier (35% faster on average). In the estimation of pollution maxima, the DRL finds the maximums with higher average precision (up to 42% lower error at these points for three vehicles) and with a lower maximum error (up to 40% lower error at these points for three vehicles). This indicates that, in this new benchmark, the performance is robust and provides a good estimate of the errors in the most contaminated areas. This will be convenient when an early warning system requires to track dangerous spots of algae blooms for prevention and bath restrictions.

6. Conclusion

This article presents a framework for training and deploying multiagent fleets of ASVs for environmental monitoring missions. The framework combines local GPs for model estimation and deep policies trained with DRL for decision-making. Two stochastic benchmark simulators were introduced to validate results for different environmental monitoring missions.

Local GPs significantly improve model computation time and yield a 30% average reduction in estimation error with various path planners. These local models excel in estimating scalar fields of varying smoothness and multimodal hyperparameter distributions, demonstrating effectiveness in challenging scenarios like algae bloom monitoring. Combining different local models enhances granularity in estimating scalar functions with distinct local properties, especially beneficial in scenarios with steep gradients, such as algae monitoring.

Deep policies, derived from a DRL algorithm, along with a consensus decision method, yield efficient monitoring policies complying with safety constraints during training. The proposed consensus mechanism is scalable, allowing for the independent adjustment of the number of agents and observations. Studying an appropriate reward function, based on the total net change of the model $\Delta\mu$, improves training efficiency by 26% and achieves 27% average enhancement in benchmarks with other path planning algorithms. Specialization of the DRL algorithm in each mission results in an additional 30% reduction in errors with improved efficiency and measurement redundancy. The combination of GPs with DRL emerges as a superior strategy for this mission type, with the reward function supporting online fleet retraining under real conditions based on the model's convergence estimation rather than the real ground truth.

Future lines of research should be able to extend this work to find policies capable of dealing with the multiobjective case. In the multiobjective case, there are several variables to monitor, and it is necessary to balance the exploration with dissimilar criterion among agents while maintaining cooperation between agents. Another important aspect to be addressed, which has been simplified in this article, is the characterization of sensors with varying noise and different measurement capabilities. In a realistic environment with different variables to be measured, vehicles could present sensors with different noises due to decalibration and manufacturers. One last improvement shall be to address with the generalization capabilities of the policy. When the environmental model is misaligned with the measurements the agents take, it is necessary to design a method to variate not only the paths of the agents, but also switch between different policies trained with other ground truths. Another important aspect will be the dynamic definition of the influence area sizes and positions. As the model is constructed, this method allows for an online redefinition of the local GPs to enhance the computational efficiency and the error minimization.

Acknowledgements

The authors want to specially thank Thomas Wiedemann and the Swarm Exploration Team of the Communications and Navigation Institute of the DLR in Munich for the advice and fruitful discussions about this research.

This work was partially funded by the FPU-2020 Ph.D. grant (Formación del Profesorado Universitario) of Samuel Yanes Luis. This work was also possible, thanks to the research stay of Samuel Yanes Luis in the Institute of Navigation and Communications of the German Aerospace Center (Weißling, Germany) from September to December 2022 under the supervision of Dmitriy Shutin.

This work was also partially funded by the following projects: TED2021-131326B-C21 of the Spanish Ministry of Science MCIN/AEI/10.13039/501100011033, and the European Union with "NextGenerationEU / PRTR" funds, TED2021-131326A-C22 of the Spanish Ministry of Science MCIN/AEI/10.13039/501100011033 and the European Union with "Next Generation EU/PRTR" funds, TED2021-131326A-C22 of the Spanish Ministry of Science PID2021-126921OB-C21 and the European Union with "Next Generation EU/PRTR" funds, and TED2021-131326A-C22 of the Spanish Ministry of Science PID2021-126921OA-C22 and the European Union with "Next Generation EU/PRTR" funds, and the "Junta de Andalucía: Consejería de Universidad, Investigación e Innovación" through the project "Monitorization of Environmental Dangers with Unmanned Surface Agents (MEDUSA)" under grant no. PCM_00019. The authors are members of the research network REINFORCED http://grupos.us.es/reinforced/, funded by the University of Seville through the "Plan Propio de Investigación".

Conflict of Interest

The authors declare no conflict of interest.

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Keywords

deep reinforcement learning, environmental monitoring, multiagent path planning

Received: December 6, 2023
Revised: March 26, 2024
Published online:

[1] J. Álvarez-Rogel, G. G. Barberá, B. Maxwell, M. Guerrero-Brotons, C. Díaz-García, J. J. Martínez-Sánchez, A. Sallent, J. Martínez-Ródenas, M. N. González-Alcaraz, F. J. Jiménez-Cárceles, C. Tercero, R. Gómez, *Ecol. Eng.* **2020**, *158*, 106086.

[2] S. Pieterkosky, A. Ziegwied, C. Cavanagh, L. Thompson, in *OCEANS 2017 – Anchorage*, Alaska **2017**, pp. 1–5.

[3] S. Yuan, Y. Li, F. Bao, H. Xu, Y. Yang, Q. Yan, S. Zhong, H. Yin, J. Xu, Z. Huang, J. Lin, *Sci. Total Environ.* **2023**, *858*, 159741.

[4] Y. Hu, C. Yang, J. Yang, Y. Li, W. Jing, S. Shu, *IOP Conf. Ser.: Earth Environ. Sci.* **2021**, *821*, 012018.

[5] M. J. T. Kathen, P. Johnson, I. J. Flores, D. G. Errez Reina, Aquafel-pso: A monitoring system for water resources using autonomous surface vehicles based on multimodal pso and federated learning, **2022**.

[6] F. Peralta, D. G. Reina, S. Toral, *Mechatronics* **2023**, *91*, 102953.

[7] M. Popovic, T. Vidal-Calleja, G. E. A. Hitz, *Auton. Robot.* **2020**, *44*, 889.

[8] R. S. Sutton, A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed, MIT Press, Cambridge, MA **2018**, http://incompleteideas.net/book/the-book-2nd.html

[9] C. E. Rasmussen, C. K. I. Williams, *Gaussian Processes for Machine Learning*. Adaptive computation and machine learning. MIT Press, Cambridge, MA **2006**.

[10] S. Yanes, D. G. Reina, S. L. T. Marín, *IEEE Access* **2021**, *9*, *17*, 084.

[11] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, D. Hassabis, *Nature* **2015**, *518*, 529.

[12] J. Sánchez-García, J. García-Campos, M. Arzamendia, D. Reina, S. Toral, D. Gregor, *Comput. Commun.* **2018**, *119*, 43.

[13] G. Conte, G. De Capua, D. Scaradozzi, *Robot. Auton. Syst.* **2016**, *76*, 46.

[14] F. Peralta, D. G. Reina, S. L. T. Marín, D. O. Gregor, M. Arzamendia, *IEEE Access* **2021**, *9*, 9163.

[15] L. Booth, S. Carpin, in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Piscataway, NJ **2023**.

[16] M. J. T. Kathen, I. J. Flores, D. G. Reina, *Electronics* **2021**, *10*, 1605.

[17] S. Yanes, D. G. Reina, S. L. T. Marín, *IEEE Access* **2020**, *6*, 1.

[18] A. Viseras, R. Garcia, *IEEE Robot. Autom. Lett.* **2019**, *4*, 3059.

[19] T. Wiedemann, C. Vlaicu, J. Josifovski, A. Viseras, *IEEE Access* **2021**, *9*, 159.

[20] M. Vergassola, E. Villermaux, B. I. Shraiman, *Nature* **2007**, *445*, 406.

[21] S. Zhang, Y. Li, Q. Dong, *Appl. Soft Comput.* **2022**, *115*, 108194.

[22] S. Yanes Luis, D. Gutiérrez-Reina, S. Toral Marín, *Appl. Soft Comput.* **2023**, *132*, 109874.

[23] R. Bellman, *Dynamic Programming*, Dover Publications, New York **1957**.

[24] A. Viseras, M. Meissner, J. Marchal, *IEEE Access* **2021**, *29*, 1.

[25] Z. Wang, N. D. Freitas, M. Lanctot, *CoRR* **2015**, http://arxiv.org/abs/1511.06581