TYPE Original Research
PUBLISHED 05 September 2022
DOI 10.3389/frobt.2022.878246

Estimating spatio-temporal fields through reinforcement learning

Paulo Padrao[1†], Jose Fuentes[1†], Leonardo Bobadilla[1†*] and Ryan N. Smith[2]

[1]Knight Foundation School of Computing and Information Sciences, Florida International University, Miami, FL, United States, [2]Institute for Environment, Florida International University, Miami, FL, United States

EDITED BY
Kostas Alexis,
Norwegian University of Science and Technology, Norway

REVIEWED BY
Elias B. Kosmatopoulos,
Democritus University of Thrace, Greece
Caoyang Yu,
Shanghai Jiao Tong University, China

*CORRESPONDENCE
Leonardo Bobadilla,
bobadilla@cs.fiu.edu

†These authors have contributed equally to this work and share first authorship

SPECIALTY SECTION
This article was submitted to Field Robotics,
a section of the journal
Frontiers in Robotics and AI

RECEIVED 17 February 2022
ACCEPTED 01 August 2022
PUBLISHED 05 September 2022

CITATION
Padrao P, Fuentes J, Bobadilla L and Smith RN (2022), Estimating spatio-temporal fields through reinforcement learning.
Front. Robot. AI 9:878246.
doi: 10.3389/frobt.2022.878246

Prediction and estimation of phenomena of interest in aquatic environments are challenging since they present complex spatio-temporal dynamics. Over the past few decades, advances in machine learning and data processing contributed to ocean exploration and sampling using autonomous robots. In this work, we formulate a reinforcement learning framework to estimate spatio-temporal fields modeled by partial differential equations. The proposed framework addresses problems of the classic methods regarding the sampling process to determine the path to be used by the agent to collect samples. Simulation results demonstrate the applicability of our approach and show that the error at the end of the learning process is close to the expected error given by the fitting process due to added noise.

KEYWORDS

spatio-temporal fields, reinforcement learning, partial differential equations, autonomous navigation, environmental monitoring

1 Introduction

The use of autonomous underwater and surface vehicles (AUVs and ASVs) for persistent surveillance in coastal and estuarine environments has been a topic of increasing interest. Examples of studies enabled by these vehicles include the dynamics of physical phenomena, such as ocean fronts, temperature, the onset of harmful algae blooms, salinity profiles, monitoring of seagrass and coral reefs, and fish ecology.

Due to the stochastic nature of these vital environments and the large spatial and temporal scales of significant processes and phenomena, sampling with traditional modalities (e.g., manned boats, buoys) is sparse and predictive models are necessary to augment decision-making to ensure that robotics assets are at the right time and the right place for sampling. However, no single model provides an informed view or representation of these or any other ocean feature that enables intelligent sampling in a principled manner. Therefore, it is critical to forecasting where a robot should sample in the immediate future so that sufficient information is provided on getting to the desired location within a dynamic environment.

Our ideas are inspired by commonly used underwater vehicles in environmental and infrastructure monitoring problems such as the AUV Ecomapper shown in Figure 1. This vehicle can measure water quality parameters, currents, and bathymetric information. However, its mission endurance is limited to a few hours due to its battery constraints, therefore, efficient sampling strategies are needed.

FIGURE 1
YSI-Ecomapper autonomous underwater vehicle.

The contributions of this paper are the following:

1) A novel framework combining classic methods with reinforcement learning to estimate ocean features, which are modeled as spatio-temporal fields.
2) A technique to get a set of informative samples to estimate spatio-temporal fields, which an agent can collect and process.
3) An extension to the classical partial differential equations fitting methods to estimate models incorporating reinforcement learning.

This paper is an expansion of our preliminary work in Padrao et al. (2022) and extends it to include the estimation of ocean features using partial differential equations. The rest of the paper is organized as follows. In Section 2, we review related work to our approach. Section 3 gives the preliminaries needed to build our method and formulate our problem. Section 4 presents the Reinforcement Learning Methods used to solve our approach, and the results are presented in Section 5. Finally, Section 6 concludes our paper and gives direction for future work.

2 Related work

2.1 Oceanic monitoring sampling

Over the last decade, it has become clear that autonomous marine vehicles will revolutionize ocean sampling. Several researchers have investigated approaches for ASVs and AUVs for adaptive ocean sampling Yuh (2000)-Smith et al. (2010c) and fundamental marine sampling techniques for ASVs and AUVs are discussed in Singh et al. (1997). Besides control algorithms for Oceanic Sampling, an alternative approach is to use static sensor placements to maximize information gathering Zhang and Sukhatme (2008).

2.2 Adaptive sampling with marine vehicles

Our work connects also with research on control design for AUVs for adaptive ocean sampling, Yoerger and Slotine (1985); Frazzoli et al. (2002); Low et al. (2009); Rudnick and Perry (2003); Yuh (2000); Frank and Jónsson (2003); Graver (2005); Barnett et al. (1996); Carreras et al. (2000); Ridao et al. (2000); Rosenblatt et al. (2002); Turner and Stevenson (1991); Whitcomb et al. (1999, 1998); McGann et al. (2008b), McGann et al. (2008a), McGann et al. (2008c), Yoerger and Slotine (1985)-McGann et al. (2008c). Applications of ocean sampling techniques for autonomous vehicles are discussed in Singh et al. (1997)-Eriksen et al. (2001). This body of research differs from the proposed research in that we plan to utilize predictive models in the form of Partial Differential Equations (PDE) to enable effective sampling, navigation, and localization within dynamic features.

2.3 Reinforcement learning in marine robotics

Reinforcement learning in marine robotics, especially model-free methods, is an attractive alternative to finding plans for several reasons. First, executing marine robotics experiments and deployments is expensive, time-consuming, and often risky; controllers learned through RL can represent significant time and cost savings and shorten the time to deployment. Second, system identification can sometimes be challenging in marine environments due to several factors such as unmodeled dynamics and environment's unknowns; for that reason, model-free RL approaches can be an alternative in these scenarios. Examples of approaches that have used RL for ASVs or AUVs include path planning Yoo and Kim (2016), control Cui et al. (2017) and tracking Martinsen et al. (2020).

2.4 Machine learning for partial differential equations

Our ideas are also connected to the use of Machine Learning models in the context of Partial Differential Equations. Due to their usefulness and impact in several domains, there have been efforts to use modern machine learning techniques to solve high dimensional PDEs Han et al. (2018), find appropriate discretizations Han et al. (2018), and control them Farahmand et al. (2017).

3 Preliminaries and problem formulation

3.1 Partial differential equations

Partial differential equations (PDEs) have been used to model water features of interest such as pH, temperature, turbidity, salinity, and chlorophyll-A. Depending on the nature of their motion, they can be modeled through diffusion, advection or a combination of both. It is important to evaluate how they behave given certain initial conditions to understand their evolution in time. We model the ocean features of interest as a scalar field $f\colon \mathbb{R}^2 \times [0,\infty) \to \mathbb{R}$.

3.1.1 Advection equation

The advection equation models how a given ocean feature (e.g., algae bloom, oil spill, chemical contaminants, etc.) is transported by a given flow which goes in the direction of $\mathbf{b} \in \mathbb{R}^2$; it is also called the *transport equation*. The model the space is given by 1.

$$\begin{aligned} \frac{\partial f}{\partial t} + \mathbf{b}\cdot\nabla f &= g(\mathbf{x},t), \text{ for } (\mathbf{x},t)\in\mathbb{R}^2\times(0,\infty) \\ f(\mathbf{x},0) &= h(\mathbf{x}), \text{ for } t=0 \end{aligned} \tag{1}$$

has the solution shown in Evans (1998).

$$f(\mathbf{x},t) = h(\mathbf{x}-t\mathbf{b}) + \underbrace{\int_0^t g(\mathbf{x}+(s-t)\mathbf{b},s)ds}_{\text{By the Duhamel's principle}}. \tag{2}$$

Provided that $g(\mathbf{x},t)\in C(\mathbb{R}^2)$ and has a compact support for each $t \in [0,\infty)$. This function g models if there are sinks or fonts of the ocean feature in the domain. If the sign of $g(\mathbf{x}, t)$ is positive, we consider that point as an ocean feature source; if it is negative, we consider it as an ocean feature sink. On the other hand, $h(\mathbf{x})$ is the initial distribution of the ocean feature at the beginning.

3.2 Estimation of the parameters of a PDE

Once we chose a PDE as a model, it is crucial to estimate the parameters of the PDE to get a reliable model. This problem belongs to the family of inverse problems since those parameters are sensitive to the observations and given initial conditions Richard et al. (2019) Antman et al. (2006). Because of this sensitivity, it is computationally expensive to find the PDE parameters. There are optimization-based techniques to solve this problem. These techniques problems balance the fitting parameter to the observations and the model sensitivity to those parameters. One of most used methods is the Tikhonov regularization technique Nair and Roy (2020) Bourgeois and Recoquillay (2018). It comprises solving a regularized optimization problem to get a regularized solution. It can be highly efficient depending on the regularization norm (especially if the L^2 norm is used). However, it depends on the regularization constant to achieve good results.

Other approaches to solving the PDE estimation problem take advantage of Bayesian theory Xun et al. (2013). In this case, bayesian learning is connected to regularization since the regularization problem coincides with the maximization of the likelihood of the parameters given the observations Bishop (1995). Therefore, Machine Learning techniques have been proposed to take advantage of the capability of the models to discover hidden relationships between the input data and the final estimation Jamili and Dua (2021). Most of those models use the fact that the samples are given in advance. This work proposes a learning mechanism to select samples that can reasonably estimate the model without exploring the complete domain. This principle has been used in numerical integration problems resulting in several quadrature rules, such that Gauss–Kronrod, Gauss-Legendre, or Newton cotes Kincaid et al. (2009). Those methods have proven to be more efficient since they can give reliable estimations using few points. In this work, we employ an intelligent agent capable of sampling the environment, searching for reliable samples, and using them to compute the parameters of a PDE. Also, this allows estimating the ocean feature behavior in the domain according to Eq. 2.

3.3 Model definition

We modeled the marine environment as a 2-D water layer (representing, for example, the surface) denoted as $\mathcal{W}\subset\mathbb{R}^2$ where $\mathcal{W}$ is an open and bounded set. The obstacle-free state space for our robot is represented by $\mathcal{S} = \mathcal{W}\backslash\mathcal{O}$, where $\mathcal{O}$ represents the set of locations that are not accessible to the robot.

To estimate the flow field, we define a scheme of fitting problems based on the known initial conditions of Eq. 1 $h(\mathbf{x})$ and the current samples acquired by the agent. First, we expect to collect samples y_i at the location $\mathbf{x}_i$ and time t_i for $i = 1, \ldots, n$ such that the field minimizes the mean square error of the collected samples. Taking advantage of the closed solution described in the homogeneous version of Eq. 1, the fitting error function $e_f(\boldsymbol{b})$ is defined as

$$\begin{aligned} e_f(\mathbf{b};\mathbf{x}_1,\ldots,\mathbf{x}_n) &= \frac{1}{n}\sum_{i=1}^{n}\left(f(\mathbf{x}_i,t_i)-y_i\right)^2 \\ &= \frac{1}{n}\sum_{i=1}^{n}\left(h(\mathbf{x}_i-t_i\mathbf{b})-y_i\right)^2 \end{aligned} \tag{3}$$

where n is the number of collected samples. Next, we define the fitting error $e_f(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ associated to the locations

$$e_f(\mathbf{x}_1,\ldots,\mathbf{x}_n) = \min_{\mathbf{b}\in\mathbb{R}^2} e_f(\mathbf{b};\mathbf{x}_1,\ldots,\mathbf{x}_n). \tag{4}$$

The fitting error expressed in Eq. 4 measures how well can the best fitted model prediction of the given samples (i.e., predict y_i given $\mathbf{x}_i$ and a parameter vector $\boldsymbol{b}$. It is the "best" in the sense that is the minimum achievable error produced by the model given the samples $\mathbf{x}_1, \ldots, \mathbf{x}_n$). Nevertheless, we can notice that if the locations $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are wrongly chosen, the fitting error can be low, but its capability of estimating the entire field may lead to over-fitting problems. To handle this issue, we add a new error term based on how well one sample can be predicted using the remaining ones. This is known as cross-validation. In this case, we propose the following cross-validation scheme. For each $1 \leq i \leq n$ let $\mathbf{b}_i^*$ defined as

$$\mathbf{b}_i^* = \arg\min_{\mathbf{b}\in\mathbb{R}^2} e_f(\mathbf{b}; \mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_n). \tag{5}$$

We define the cross validation error $e_{cv}(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ as

$$e_{cv}(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \frac{1}{n}\sum_{i=1}^{n}\left(h(\mathbf{x}_i - t_i\mathbf{b}_i^*) - y_i\right)^2, \tag{6}$$

and it measures on average how well the samples can fit a model, which is estimating the remaining sample. This avoids the over-fitting problems and allows to measure how reliable are the taken samples. Lastly, we define the total error $e_{total}(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ or just e_{total} as

$$e_{total}(\mathbf{x}_1, \ldots, \mathbf{x}_n) = e_f(\mathbf{x}_1, \ldots, \mathbf{x}_n) + e_{cv}(\mathbf{x}_1, \ldots, \mathbf{x}_n). \tag{7}$$

This error compound aims to have an equal trade off between the sample estimation measured by $e_f(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ and the reliability of the samples measured by $e_{cv}(\mathbf{x}_1, \ldots, \mathbf{x}_n)$.

The agent is modeled as a rigid body that moves in $\mathbb{R}^2$ and can be described by a non-linear system as

$$\begin{aligned}\dot{\mathbf{x}} &= f(\mathbf{x}, \mathbf{u})\\ \mathbf{z} &= o(\mathbf{x}, \mathbf{r})\end{aligned} \tag{8}$$

Such that $f(\mathbf{x}, \mathbf{u})$ is the motion model of the vehicle, $o(\mathbf{x}, \mathbf{r})$ is the observation model of the vehicle, and $\mathbf{r}$ are additive, zero-mean noise to account for modeling errors and sensor imperfections.

Let $\mathcal{S}$ be the state space, i e., the set of all possible states $\mathbf{x} \in \mathcal{S}$ and $\mathcal{U}$ be the action space, which represents the set of all possible actions. Therefore, a configuration of the vehicle can be described by

$$\begin{aligned}\mathbf{x} &= (x, y, \phi)\\ \mathbf{u} &= (u_v, u_\omega)\end{aligned} \tag{9}$$

In which (x, y) is the position of the vehicle and $\phi \in (-\pi/4, \pi/4)$ is the vehicle's heading; the forward speed v and the angular velocity of the agent orientation ω can be set directly by the action variables u_v and u_ω, respectively. The kinematic model of the agent $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$ is described by Eq. 10.

$$\begin{aligned}\dot{x} &= u_v \cos\phi + v_x\\ \dot{y} &= u_v \sin\phi + v_y\\ \dot{\phi} &= u_\omega\end{aligned} \tag{10}$$

where v_x and v_y account for the velocity components of the environment (flow field) in x and y directions.

Let $\mathbf{x}_S \in \mathcal{S}$ be the initial location of the agent. It is assumed that the agent takes advantage of ocean current dynamics as it drifts and moves forward with or against the currents and rotates clockwise or counterclockwise. Therefore, the action space is defined as

$$\mathcal{U} = [0, v_{\max}/2, v_{\max}] \times (\phi_{\min}, \phi_{\max}) \tag{11}$$

We discretize the action space to obtain a finite subset of $\mathcal{U}$ defined as

$$\begin{aligned}\mathcal{A} = \{&(v_{\max}, 0), (v_{\max}/2, 0), (v_{\max}, -\phi), (v_{\max}, +\phi), (v_{\max}/2, -\phi),\\ &(v_{\max}/2, +\phi), (0, 0)\}\end{aligned} \tag{12}$$

The description of the actions of the agent are summarized in Table 1.

For the observation model, we assume that the vehicle uses an IMU to measure its heading angle ϕ and has access to GPS at surface level. Also, the vehicle can observe its state with uncertainties due to sensor imperfections and the dynamic nature of the underwater environment. The observation space $\mathcal{Z}$, the set of all possible sensor observations $\mathbf{z} \in \mathcal{Z}$, is given by

$$\mathcal{Z} = (x_{\min}, x_{\max}) \times (y_{\min}, y_{\max}) \times (\phi_{\min}, \phi_{\max}). \tag{13}$$

The observation model $o(\mathbf{x})$ is represented by

$$\mathbf{z} = o(\mathbf{x}, \mathbf{r}) = I\mathbf{x} + \mathbf{r} \tag{14}$$

Where $\mathbf{r} \in \mathbb{R}^3$ is noise distributed as $r \sim \mathcal{N}(\mathbf{0}, \Sigma)$ with Σ a diagonal covariance matrix to account for modeling errors and sensor imperfections and I is the identity matrix. It was also considered that the measurement noises of each sensor are uncorrelated and have constant covariance.

These elements allow us to formulate the following problem. **Problem**: *Given an aquatic environment $\mathcal{W}$, the action set of the agent A, the state space $\mathcal{S} = \mathcal{W} \times (\phi_{\min}, \phi_{\max})$, the vehicle's motion model, observations of a given ocean feature in several locations, estimate the flow field (and therefore the ocean feature distribution) by minimizing cross-validation error and error fitting within a given fixed number of steps.*

4 Methods

Because of the computational effort required to tackle problems with large state spaces, tabular learning methods may be unfeasible Sutton and Barto (2018). As a result, combining approximation solutions of reinforcement learning methods with generalization techniques yields a computationally viable solution for real-world problems.

To update the agent policy based on actions taken, we suggest using SARSA(λ) algorithm in conjunction with a linear function approximation technique based on stochastic semi-gradient descent. The agent is in state $\mathbf{s}_t \in \mathcal{S}$, takes action $a_t \in A$, and receives reward r_t at each time step t. In this method, we can estimate the action-value function $\hat{q}(\mathbf{s}, a)$ for the behavior policy π in a systematic way. The SARSA(λ) algorithm also chooses an action based on the ε-greedy approach. Therefore, actions with the highest estimated values are chosen with a high probability,

TABLE 1 Description of the actions of the agent.

$(v_{\max}, 0)$	moving forward with maximum speed $v_{\max}$
$(v_{\max}/2, 0)$	moving forward at half the speed $v_{\max}/2$
$(v_{\max}, -\phi)$	turning clockwise by ϕ and moving with maximum speed
$(v_{\max}, +\phi)$	turning counterclockwise by ϕ and moving with maximum speed
$(v_{\max}/2, +\phi)$	turning clockwise by ϕ and moving at half the maximum speed
$(v_{\max}/2, +\phi)$	turning counterclockwise by ϕ and moving at half the maximum speed

but random actions are picked with a low probability ε independent of their estimated values.

The action-value function approximation is defined as

$$\hat{q}(\mathbf{s}, a, \mathbf{w}) \approx q(\mathbf{s}, a) \tag{15}$$

Where $\mathbf{w} \in \mathbb{R}^d$ is the weight vector of the semi-gradient descent method. The weight vector update is defined by Eq. 16

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha\left[G_t - \hat{q}(\mathbf{s}_t, a_t, \mathbf{w}_t)\right]\nabla\hat{q}(\mathbf{s}_t, a_t, \mathbf{w}_t) \tag{16}$$

where α is the step size, and G_t is the return function. Applying linear function approximation, Eq. 15 can be modified to

$$\hat{q}(\mathbf{s}, a, \mathbf{w}) = \mathbf{w}^\top \mathbf{x}(\mathbf{s}, a) = \sum_{i=1}^{d} w_i x_i(\mathbf{s}, a) \tag{17}$$

Where $\mathbf{x} \in \mathbb{R}^d$ is the feature vector. Each component $x_i(\mathbf{s}, a)$ of the feature vector corresponds to a feature of the state-action pair $(\mathbf{s}, a)$ and maps it to a real value. As a result, the gradient of the approximate action-value function can be modified as $\nabla\hat{q}(\mathbf{s}_t, a_t, \mathbf{w}_t) = \mathbf{x}(\mathbf{s}_t, a_t)$ and Eq. 16 reduces to

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha\left[G_t - \hat{q}(\mathbf{s}_t, a_t, \mathbf{w}_t)\right]\mathbf{x}(\mathbf{s}_t, a_t) \tag{18}$$

4.1 Reward function design

In reinforcement learning problems, designing a reward function is not a trivial task, Ng et al. (1999). To avoid spurious exploration, we defined a terminal condition with a fixed number of observations taken to determine when to reset the environment for a new episode. To encourage the agent to minimize the fitting and cross-validation errors within a given number of steps, we provide a reward that is inversely proportional to the sum of the errors at the terminal state. For each episode, the agent collects 20 observations, and the reward function is defined as

$$r(\mathbf{s}, a) = \begin{cases} \dfrac{100}{e_{total}}, & \text{if the number of observations} = 20 \\ c_2, & \text{if the number of observations} < 5 \\ c_1 c_2^{1.5}, & \text{otherwise} \end{cases} \tag{19}$$

where c_1 is the ratio between total error at previous and current step and $c_2 = 1 - (\frac{e_{total}}{20})^{0.4}$.

4.2 Linear methods and feature construction: tile coding

In reinforcement learning systems, feature construction is critical since it values each state of the agent. The main techniques for feature construction of linear methods are polynomial-based, Fourier basis and tile coding Sherstov and Stone (2005). As such, tile coding is a computationally effective feature design technique that divides the state space into divisions called tiles. Each element in the tiling is referred to as a tile. Different tilings are separated by a fixed-size fraction of the tile width Sutton and Barto (2018). If there are n tilings and each tiling has $m \times m$ tiles, the feature vector is $\mathbf{x}(\mathbf{s}) \in \mathbb{R}^{n \times m \times m}$. One of the main advantages of using tile coding with binary feature vectors is that the weighted sum in the approximate value function (Eq. 17) is easy to compute. Figure 2 shows an example of the representation of tile coding for two-dimensional continuous state space. In this case, $\mathbf{x}(\mathbf{s})$ is a feature vector with twelve components, one for each tile in each tiling. Each component of $\mathbf{x}(\mathbf{s})$ is inactive (zero-valued) except active components $x_0(\mathbf{s})$, $x_4(\mathbf{s})$ and $x_8(\mathbf{s})$ that corresponds to the current location states of the agent. As a consequence, there are n active features in $\mathbf{x}(\mathbf{s})$ because every position in state space falls into precisely one tile in each of the n tilings. Let the weight vector $\mathbf{w} = [w_0, \ldots, w_{11}]^\top$ and the action space be $A = \{a_0, a_1, a_2\}$. The feature vector regarding actions a_0, a_1 and a_2 is $\mathbf{x}(\mathbf{s}, a_0) = \mathbf{x}(\mathbf{s}, a_1) = \mathbf{x}(\mathbf{s}, a_2) = [1,0,0,0,1,0,0,0,1,0,0,0]^\top$. Thus, the action-value function approximation $\hat{q}(\mathbf{s}, a, \mathbf{w})$ described in Eq. 17 is computed as

$$\hat{q}(\mathbf{s}, a, \mathbf{w}) = \sum_{i=1}^{d} w_i \tag{20}$$

for each action in action space.

With tile coding, design issues for discrimination and generalization should be taken into account. The number and size of tiles, for example, affect the granularity of state discrimination, or how far the agent must move in state space to change at least one component of the feature vector. Aside from that, the shape of the tilings and the offset

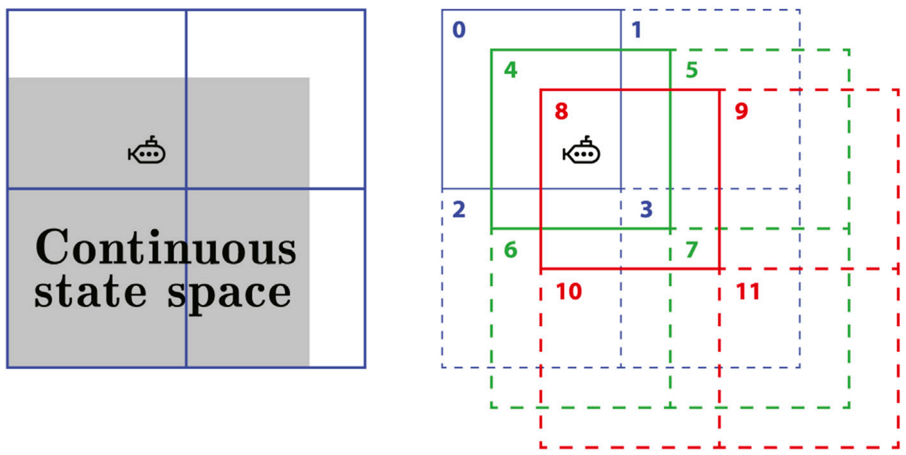

FIGURE 2
An example of tile coding representation of a continuous 2D state space. The agent is a point in the state space to be represented by the active tiles of the three tilings. Active tiles are described by solid lines and have a value of 1. Inactive tiles are described by dashed lines and have a value of 0. Therefore, the feature vector is $\mathbf{x}(s)$ = [1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0].

distance between them have an impact on generalization. As an example, if tiles are stretched along one dimension in state space, generalization will extend to states along that dimension as well Sutton and Barto (2018).

4.3 Eligibility traces in reinforcement learning

In problems with large state spaces, the eligibility trace is a technique to promote computational efficiency of reinforcement learning methods. The eligibility trace is a vector $\mathbf{z}_t \in \mathbb{R}^d$ whose components maintain track of which components of the weight vector $\mathbf{w}_t$ have contributed to recent state values and temporarily records the occurrence of estimated events. Therefore, components of $\mathbf{w}_t$ that most frequently contribute to valuations of previous states are considered *eligible* for an update Singh et al. (1995). Eligibility trace components are updated based on the trace-decay parameter $\lambda \in [0, 1]$, which specifies the pace at which the trace fades away exponentially. In contrast with n-step methods that perform action-value updates after a given number of steps, eligibility traces provide updates continually over the learning process. For this reason, agent behavior can be modified right after a new state is found rather than being delayed n steps.

The action-value return function G_t is a function approximation of the n-step return defined as

$$G_{t:\,t+n} = r_{t+1} + \cdots + \gamma^{n-1}\hat{q}(\mathbf{s}_{t+n}, a_{t+n}, \mathbf{w}_{t+n-1}), \quad t+n<T \tag{21}$$

where γ is the discount rate that regulates the relative importance of near-sighted and far-sighted rewards. Thus, the λ-return G_t^λ is written as

$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{T-t-1}\lambda^{n-1}G_{t:\,t+n} + \lambda^{T-t-1}G_t \tag{22}$$

In this way, the update rule for the weight vector in Eq. 16 is modified as follows

$$\begin{aligned}\mathbf{w}_{t+1} &= \mathbf{w}_t + \alpha\left[G_t^\lambda - \hat{q}(\mathbf{s}_t, a_t, \mathbf{w}_t)\right]\nabla\hat{q}(\mathbf{s}_t, a_t, \mathbf{w}_t)\\ &= \mathbf{w}_t + \alpha\delta_t\mathbf{z}_t\end{aligned} \tag{23}$$

where the action-value estimation error δ_t is defined as

$$\delta_t = r_{t+1} + \gamma\hat{q}(\mathbf{s}_{t+1}, a_{t+1}, \mathbf{w}_t) - \hat{q}(\mathbf{s}_t, a_t, \mathbf{w}_t) \tag{24}$$

The action-value representation of the eligibility trace is defined as

$$\begin{aligned}\mathbf{z}_{-1} &= \mathbf{0}\\ \mathbf{z}_t &= \gamma\lambda\mathbf{z}_{t-1} + \nabla\hat{q}(\mathbf{s}_t, a_t, \mathbf{w}_t), \quad 0 \le t \le T\end{aligned} \tag{25}$$

The complete algorithm for SARSA(λ) is presented in table 1, Sutton and Barto (2018).

Algorithm 1. SARSA(λ) with linear function approximation.

```
Input: $\mathbf{x}(\mathrm{s}, a)$, feature vector from tile coding with the set of active features
Parameters: step size $\alpha > 0$, trace decay rate $\lambda \in [0,1]$, $\varepsilon \in [0,1]$
Initialization: $\mathbf{w} \in \mathbb{R}^d$, $\mathbf{z} \in \mathbb{R}^d$
For each episode:
    Initialize state s
    Choose action $a$ according to $\varepsilon$-greedy
    $\mathbf{z} \leftarrow \mathbf{0}$                                          ▷ Initialize elig. trace
    For each step of the episode:
        Take action $a$, collect reward $r$, update state s′
        $\delta \leftarrow r$
        For $i$ active features in $\mathbf{x}(\mathrm{s}, a)$ :
            $\delta \leftarrow \delta - w_i$
            $z_i \leftarrow 1$                                      ▷ update elig. trace
        If s′ is terminal then:
            $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\mathbf{z}$                                  ▷ Eq. 23
            Go to next episode
        Choose $a'$ according to $\varepsilon$-greedy
        For $i$ active features in $\mathbf{x}(\mathrm{s}', a')$ :
            $\delta \leftarrow \delta + \gamma w_i$                                   ▷ Eq. 24
        $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\mathbf{z}$                                      ▷ Eq. 23
        $\mathbf{z} \leftarrow \gamma\lambda\mathbf{z}$                                           ▷ Eq. 25
        $\mathbf{s} \leftarrow \mathbf{s}'$
        $a \leftarrow a'$
```

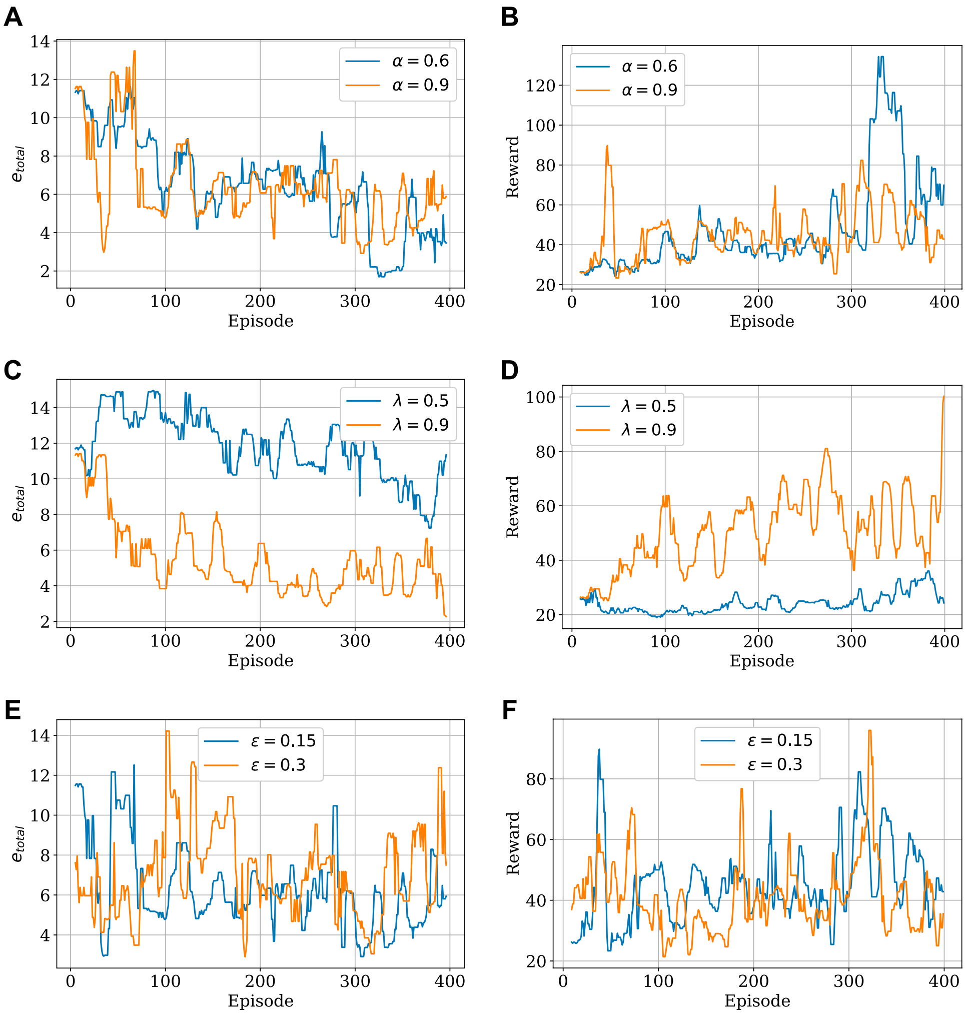

FIGURE 3
Simulation results of the proposed learning framework with the variation of step size α **(A,B)**, trace decay rate λ **(C,D)**, and ϵ-greedy parameter **(E,F)** with respect to the total number of steps per episode and returns per episode.

5 Results and discussion

Simulation results are presented in Figure 3. For each simulation, we ran a set of simulations consisting of 400 episodes with 20 steps each to investigate how the agent behaves under the effect of the flow field and the variation of the step size α and the trace decay rate λ. For tile coding, we used eight tilings, each tiling containing 8×8 tiles. Thus, the feature vector is $\mathbf{x}(\mathbf{s}) \in \mathbb{R}^{8\times 8\times 8}$. Throughout the simulation, the ε-greedy parameter was fixed at 0.15, indicating that actions with the highest estimated returns are selected 75% of the time. In this way, higher values of the ε-greedy parameter can lead to an increase in the exploratory behavior of the agent. Besides, to perform the contaminant estimation we selected the functions, keeping the notation at Eq. 1, as $g(\mathbf{x}, t) = 0$ to mean that there are no more sources of the Ocean feature around the domain and

$$h(\mathbf{x}) = a \cdot \exp\left(-\frac{\|\mathbf{x} - \mathbf{c}\|_q^q}{\sigma^q}\right). \tag{26}$$

$h(\mathbf{x})$ models the initial distribution of the ocean feature. Where $a = 100$ controls the scale, $q = 2$ manages the decay rate, $\| \cdot \|_p$ is the L^p norm defined in $\mathbb{R}^2$ for $1 \le p \le \infty$, $\sigma = 40$ combined with q can be interpreted as the standard deviation of $h(\mathbf{x})$ and c is the point where the ocean feature reaches its maximum. Lastly, each observation was corrupted using Gaussian noise $\epsilon \sim \mathcal{N}(0, 1)$.

Figures 3A,B shows the total estimation error and agent reward with respect to variation of the step size α. The step size is interpreted as the fraction of the way the agent moves towards the target. Smaller values of the step size α provided an increase in rewards through the episodes and a slight decrease in the estimation error. Additionally, the trace decay rate λ was fixed

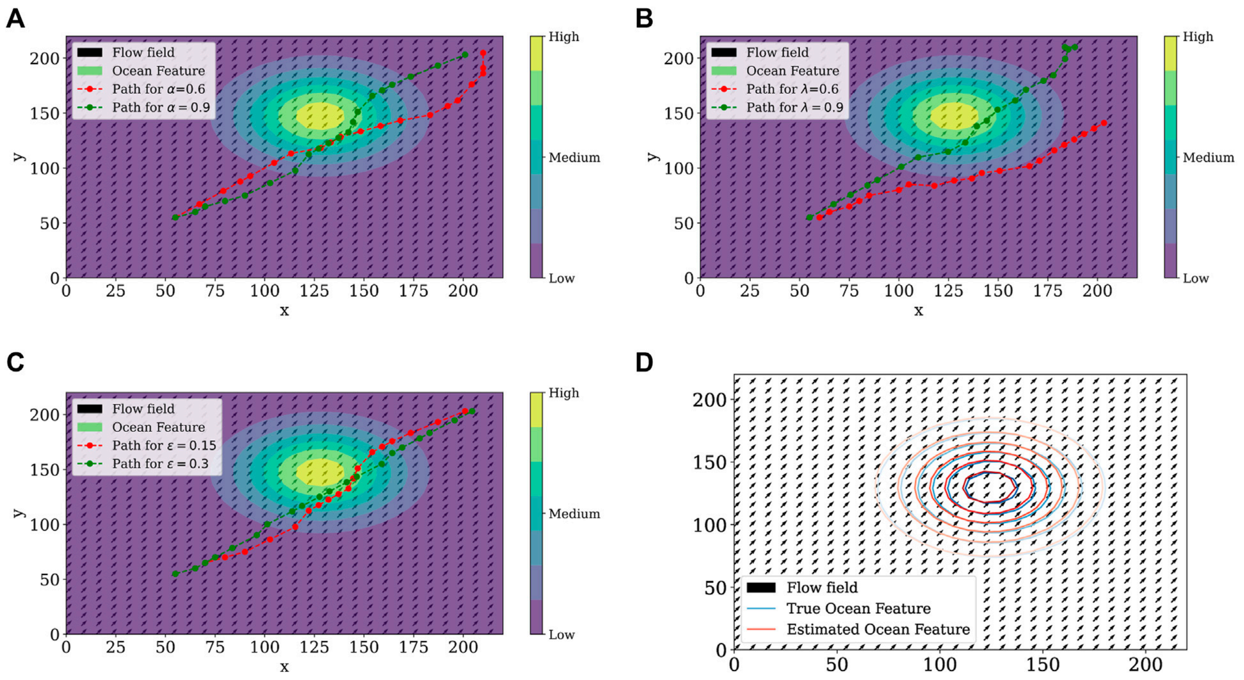

FIGURE 4
Agent path with the variation of step size α **(A)**, trace decay rate λ **(B)**, and ε-greedy parameter **(C)**. **(D)** Difference between the true and the estimated ocean features.

at 0.9. Figures 3C,D shows the total estimation error and agent reward with respect to variation of trace decay rate λ of the eligibility trace $\mathbf{z}_t$ in Eq. 25. Larger values of λ resulted in a significant decrease in the estimation error and an increase in rewards. Figures 3E,F shows the total estimation error and agent reward with respect to variation of the ε-greedy parameter. Although higher values of the ε-greedy parameter can lead to higher exploratory agent behavior, simulation shows similar results with different values of ε.

Figure 4 shows different paths taken by the agent in different simulation scenarios. Circles represent level sets of the ocean feature distribution at the end of the simulation. We notice the highest feature concentration location at the center, and the outer circles represent lower ocean feature levels assuming a radial diffusion. Optimal paths have the characteristic of following the ocean feature and crossing its level sets to obtain information at different levels to estimate the entire field.

Finally, Figure 4D shows the difference between the estimated and the true ocean feature distributions at the final time. Both of them are similar once the parameters for the true flow field is $\mathbf{b} = (5{,}5)^\top$ and the estimated is $\hat{\mathbf{b}} = (4.928, 5.037)^\top$. We notice that $\|\mathbf{b} - \hat{\mathbf{b}}\|_2 \approx 0.0809$, but e_{total} is close to 2 at the end of the reinforcement learning process. This can be explained because the fitting error e_f is on average the difference between the real observation and the corrupted one. If we assume that the true observations $f(\mathbf{x}_i, t_i)$ and the corrupted ones y_i are related by $y_i = f(\mathbf{x}_i, t_i) + \epsilon_i$ for each i, where ϵ_i are i.i.d. Random variables such that $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$ for each i. Then, we can notice Bishop (1995) that both, the fitting error e_f and the cross-validation error e_{cv} approximate the variance σ^2. Since $\mathbb{E}[\epsilon_i{}^2] = \mathrm{Var}(\epsilon_i) + \mathbb{E}[\epsilon_i]^2 = \sigma^2$ and

$$
\begin{aligned}
e_f(\mathbf{x}_1, \ldots, \mathbf{x}_n) &= \min_{\mathbf{b}\in\mathbb{R}^2} \frac{1}{n}\sum_{i=1}^{n}\left(f(\mathbf{x}_i, t_i) - y_i\right)^2 = \frac{1}{n}\sum_{i=1}^{n}\left(h(\mathbf{x}_i - t_i\mathbf{b}) - y_i\right)^2 = \underbrace{\frac{1}{n}\sum_{i=1}^{n}\epsilon_i^2 = \sigma^2}_{\text{as } n\to\infty} \\
e_{cv}(\mathbf{x}_1, \ldots, \mathbf{x}_n) &= \frac{1}{n}\sum_{i=1}^{n}\left(h\left(\mathbf{x}_i - t_i\mathbf{b}_i^*\right) - y_i\right)^2 = \underbrace{\frac{1}{n}\sum_{i=1}^{n}\epsilon_i^2 = \sigma^2}_{\text{as } n\to\infty}
\end{aligned}
\tag{27}
$$

By the large numbers law. Therefore, $e_{total} = e_{cv} + e_f \approx 2$.

To increase the complexity of our simulations, we chose to a double-gyre system; a commonly occurring oceanic feature that is relatively easy to model and analyse Provost and Verron (1987), Wolligandt et al. (2020), Smith et al. (2015), Shadden et al. (2005), Shen et al. (1999). The flow is described by the stream-function

$$\psi(x, y, t) = A\sin\big(\pi f_{dg}(x, t)\big)\sin(\pi y) \tag{28}$$

Where $f_{dg}(x, t) = a(t)x^2 + b(t)x$, $a(t) = \mu\sin(\omega_{dg}t)$, $b(t) = 100, -{,}200\mu\sin(\omega_{dg}t)$ over the domain $(0, 200) \times (0, 100)$. In Eq. 28, A describes the magnitude of the velocity vectors, ω_{dg} is the frequency of gyre oscillation, and μ is the amplitude of motion of the line separating the gyres, Shadden et al. (2005). Then the flow

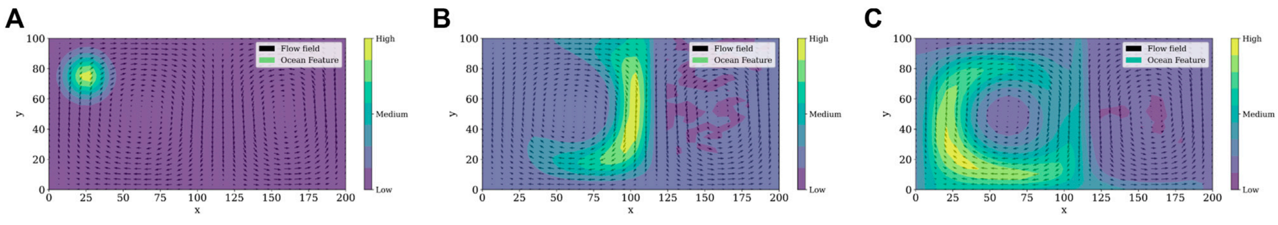

FIGURE 5
Spread of a given ocean feature through time under the influence of a double-gyre flow field with $A = 10$, $\mu = 0.25$, and $\omega_{dg} = \pi/5$ at **(A)** $t = 0$ s **(B)** $t = 5$ s **(C)** $t = 10$ s.

field produced the double gyre is the vectorial field $\mathbf{v}(x, y, t) = \nabla\psi(x, y, t)$.

The PDE (1) considers constant flow fields given by the vector $\mathbf{b}$. For this reason, we need to consider an extension of this equation defined in the bounded domain $\mathcal{W}$ called the *advection-diffusion* equation

$$\begin{aligned} \frac{\partial f}{\partial t} - \rho\Delta f + \nabla\cdot(f\mathbf{v}) &= g(\mathbf{x},t), \text{ for } (\mathbf{x},t) \in \mathcal{W}\times(0,\infty) \\ f(\mathbf{x},0) &= h(\mathbf{x}), \text{ for } t = 0 \\ \frac{\partial f}{\partial \mathbf{n}} &= 0, \text{ for } \mathbf{x} \in \partial\mathcal{W}. \end{aligned} \tag{29}$$

Which considers non-constant flow fields, the addition of the diffusion term $\rho\Delta f$ with a small diffusivity coefficient ρ and the homogeneous Neumann boundary conditions with outer normal vector $\mathbf{n}$ is due to the numerical difficulties found and reported when the pure advection equation is solved by numerical methods Evans (1998).

Figure 5 illustrates the spread of a given ocean feature through time under the influence of a double-gyre flow field.

For the simulation of the reinforcement learning framework and the double-gyre system, we ran a total of 10 episodes with 10 steps each. Although we used tile coding as a computationally effective feature in our reinforcement learning framework, it is still necessary to solve the partial differential equation given in Eq. 29 at each step of each episode. Moreover, to find the fitting and cross-validation errors it is necessary to solve an optimization problem involving the solution of the PDE as a subroutine several times. In order to simulate this computationally intensive optimization algorithm, we took advantage of Florida International University's Phosphorus, a 20-core Intel(R) Xeon(R) Silver 4114 CPU at 2.20 GHz server, and a Bayesian optimization algorithm intended to handle black box functions which are costly to evaluate. True values for the frequency of gyre oscillation ω_{dg} and the amplitude of gyre

TABLE 2 Learning simulation parameters and results. True and learned double-gyre model parameters over 10 learning episodes.

Learning simulation parameters

Number of episodes	10 with 10 steps each
Tile coding	8 tilings with 8 × 8 tiles each
Step size α	0.9
Trace decay rate λ	0.9
ε-greedy parameter	0.15
Double-Gyre Model Parameters and Results	
True μ	0.25
Learned μ	0.2481
True ω_{dg}	$\pi/5 \approx 0.6283$
Learned ω_{dg}	0.6344

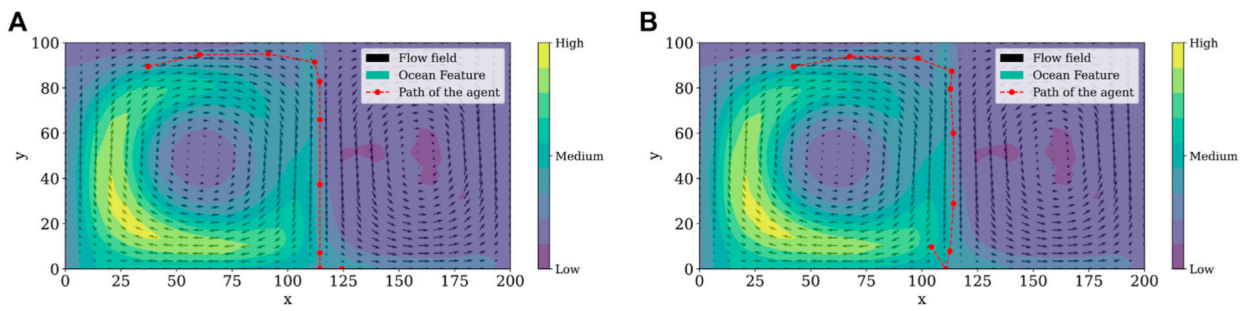

FIGURE 6
Path followed by the agent, double gyre field to estimate the parameters described in Table 2 at different episodes **(A)** episode 3, **(B)** episode 7.

motion μ are set to 0.25 and $\pi/5 \approx 0.6283$, respectively. Considering only 10 episodes, the learned values for ω_{dg} and μ were 0.2481 and 0.6344, respectively, with the smallest estimation error in episode 7. Learned parameters are summarized in Table 2 and Figure 6 shows the paths taken by the agent at different episodes while estimating the flow field. The agent follows the contaminant, but careful examination should be made at the gyre separation line once the agent could take an undesired action, resulting in feature mistracking. This behavior is illustrated when we compared paths in Figures 6A,B.

6 Conclusion

In this work, we presented a novel method for estimating a spatio-temporal field using informative samples taken by a trained agent. This allowed estimating the distribution of the ocean feature, keeping track of its localization and distribution at each time. It was possible to address the problem of selecting meaning samples such that they help to perform the estimation of the field. Therefore, this develops a different perspective in estimation procedures, which has been addressed using other techniques having pre-defined models to show *a priori* which samples should be taken.

Moreover, we proposed combining the classical regularization methods used to estimate parameters in partial differential equations with the optimization processes used to carry out those estimates. We merged machine learning techniques, which are more flexible and capable of learning complex patterns from different sources, to choose the sample locations to keep track of and estimate the ocean feature field.

Future work

For future work, we consider the expansion of the proposed method for 3D environments. This can be accomplished by augmenting the vehicle model (state space, action space, observation space) and validating the proposed framework with deployments in aquatic environments such as in the Biscayne Bay area, Florida, United States. Besides that, it is possible to refine our estimation strategies with cooperative agents. A primary direction for future work is to incorporate a combination of heterogeneous agents in order to provide better estimates of the locations of the ocean feature. In this work, we assume known initial conditions for a given linear, constant flow field. A second direction for future work is to investigate how effective the proposed estimation framework is for time-varying flow fields and actual oceanic data from the Regional Ocean Modeling System (ROMS) Shchepetkin and McWilliams (2005). ROMS data set that provides current velocity prediction data consisting of three spatial dimensions (longitude, latitude, and depth) associated with time. Finally, tracking oceanic features, such as the Lagrangian coherent structures (LCS) contributes to a wide range of applications in ocean exploration Hsieh et al. (2012). Therefore, an additional direction of our work is to expand the current work towards an efficient method for LCS tracking using machine learning techniques.

Data availability statement

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

Author contributions

LB and RS contributed to the conception and design of the study. PP and JF performed simulation experiments and wrote the first draft of the manuscript. LB also wrote sections of the manuscript. All authors contributed to manuscript revision, read, and approved the submitted version.

Funding

This work is supported in part by the NSF grants IIS-2034123, IIS-2024733, and the U.S. Dept. Of Homeland Security grant 2017-ST-062000002.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Antman, S. S., Marsden, J. E., Sirovich, L., Hale, J. K., Holmes, P., Keener, J., et al. (2006). *Inverse problems for partial differential equations.* Second edn. Germany: Springer.

Bachmayer, R., Humphris, S., and Fornari, D. (1998). Oceanographic research using remotely operated underwater robotic vehicles: Exploration. *Mar. Technol. Soc. J.* 32, 37.

Barnett, D., McClaran, S., Nelson, E., and McDermott, M. (1996). "Architecture of the Texas A&M autonomous underwater vehicle controller," in *Proceedings of Symposium on Autonomous Underwater Vehicle Technology* 231–237. doi:10.1109/AUV.1996.532420

Bird, L., Sherman, A., and Ryan, J. P. (2007). "Development of an active, large volume, discrete seawater sampler for autonomous underwater vehicles," in Proc Oceans MTS/IEEE Conference (Vancouver, Canada), Vancouver, Canada, 04 October 2007. doi:10.1109/OCEANS.2007.4449303

Bishop, C. M. (1995). Training with noise is equivalent to tikhonov regularization. *Neural Comput.* 7, 108–116. doi:10.1162/neco.1995.7.1.108

Bourgeois, L., and Recoquillay, A. (2018). A mixed formulation of the tikhonov regularization and its application to inverse pde problems. *ESAIM Math. Model. Numer. Analysis* 52, 123–145. doi:10.1051/m2an/2018008

Carreras, M., Batlle, J., Ridao, P., and Roberts, G. (2000). "An overview on behaviour-based methods for AUV control," in *MCMC2000, 5th IFAC Conference.*

Creed, E. L., Kerfoot, J., Mudgal, C., and Barrier, H. (2004). Transition of slocum electric gliders to a sustained operational system. *OCEANS '04: MTTS/IEEE TECHNO-OCEAN '04* 2, 828–833. doi:10.1109/OCEANS.2004.1405565

Creed, E. L., Mudgal, C., Glenn, S., Schofield, O., Jones, C., and Webb, D. C. (2002). *Using a fleet of slocum battery gliders in a regional scale coastal ocean observatory*. Biloxi, MI, USA: Oceans '02 MTS/IEEE.

Cui, R., Yang, C., Li, Y., and Sharma, S. (2017). Adaptive neural network control of auvs with control input nonlinearities using reinforcement learning. *IEEE Trans. Syst. Man. Cybern. Syst.* 47, 1019–1029. doi:10.1109/tsmc.2016.2645699

Davis, R. E., Ohman, M., Rudnick, D., Sherman, J., and Hodges, B. (2008). Glider surveillance of physics and biology in the southern California current system. *Limnol. Oceanogr.* 53, 2151–2168. doi:10.4319/lo.2008.53.5_part_2.2151

Eriksen, C. C., Osse, T. J., Light, R. D., Wen, T., Lehman, T. W., Sabin, P. L., et al. (2001). Seaglider: A long-range autonomous underwater vehicle for oceanographic research. *IEEE J. Ocean. Eng.* 26, 424–436. doi:10.1109/48.972073

Evans, L. C. (1998). Partial differential equations. *Graduate Stud. Math.* 19, 7.

Farahmand, A.-m., Nabi, S., and Nikovski, D. N. (2017). "Deep reinforcement learning for partial differential equation control," in American Control Conference (ACC), Seattle, WA, USA, 24-26 May 2017 (IEEE), 3120–3127.

Frank, J., and Jónsson, A. (2003). Constraint-based attribute and interval planning. *Constraints* 8, 339–364. doi:10.1023/a:1025842019552

Frazzoli, E., Daleh, M., and Feron, E. (2002). Real-time motion planning for agile autonomous vehicles. *J. Guid. Control Dyn.* 25 (1), 116–129. doi:10.2514/2.4856

Graver, J. (2005). *Underwater gliders: Dynamics, control and design* (Princeton, NJ: Princeton University). Ph.D. thesis.

Griffiths, G., Jones, C., Ferguson, J., and Bose, N. (2007). Undersea gliders. *Feed. Heal. Humans* 2, 64–75.

Han, J., Jentzen, A., and Weinan, E. (2018). Solving high-dimensional partial differential equations using deep learning. *Proc. Natl. Acad. Sci. U. S. A.* 115, 8505–8510. doi:10.1073/pnas.1718942115

Hsieh, M. A., Forgoston, E., Mather, T. W., and Schwartz, I. B. (2012). "Robotic manifold tracking of coherent structures in flows," in IEEE International Conference on Robotics and Automation, Saint Paul, MN, USA, 14-18 May 2012 (IEEE), 4242–4247. doi:10.1109/ICRA.2012.6224769

Jamili, E., and Dua, V. (2021). Parameter estimation of partial differential equations using artificial neural network. *Comput. Chem. Eng.* 147, 107221. doi:10.1016/j.compchemeng.2020.107221

Johnson, K. S., and Needoba, J. A. (2008). Mapping the spatial variability of plankton metabolism using nitrate and oxygen sensors on an autonomous underwater vehicle. *Limnol. Oceanogr.* 53, 2237–2250. doi:10.4319/lo.2008.53.5_part_2.2237

Jones, C., Creed, E. L., Glenn, S., Kerfoot, J., Kohut, J., Mudgal, C., et al. (2005). "Slocum gliders - a component of operational oceanography," in *Autonomous undersea systems institute symposium proceedings.*

Kincaid, D., Kincaid, D. R., and Cheney, E. W. (2009). *Numerical analysis: Mathematics of scientific computing, vol. 2.* Providence, Rhode Island: American Mathematical Soc.

Low, K. H., Dolan, J., and Khosla, P. (2009). "Information-theoretic approach to efficient adaptive path planning for mobile robotic environmental sensing," in Proceedings of the 19th international conference on automated planning and scheduling (ICAPS-09). arXiv:1305.6129v1.

Martinsen, A. B., Lekkas, A. M., Gros, S., Glomsrud, J. A., and Pedersen, T. A. (2020). Reinforcement learning-based tracking control of usvs in varying operational conditions. *Front. Robot. AI* 7, 32. doi:10.3389/frobt.2020.00032

McGann, C., Py, F., Rajan, K., Ryan, J. P., and Henthorn, R. (2008a). "Adaptive control for autonomous underwater vehicles," in AAAI, San Diego, California, USA, 09-12 October 1995 (Chicago, IL: AAAI). doi:10.1109/OCEANS.1995.528563

McGann, C., Py, F., Rajan, K., Thomas, H., Henthorn, R., and McEwen, R. (2008b). "A deliberative architecture for AUV control," in ICRA (Pasadena, CA: ICRA). doi:10.1109/ROBOT.2008.4543343

McGann, C., Py, F., Rajan, K., Thomas, H., Henthorn, R., and McEwen, R. (2008c). "Preliminary results for model-based adaptive control of an autonomous underwater vehicle," in Int. Symp. on Experimental Robotics, Athens, July 13-16, 2008 (Athens, Greece: DBLP).

Nair, M. T., and Roy, S. D. (2020). *A new regularization method for a parameter identification problem in a non-linear partial differential equation*, doi:10.22541/au.159138733.37659934

Ng, A. Y., Harada, D., and Russell, S. (1999). "Policy invariance under reward transformations: Theory and application to reward shaping," in Proceedings of the Sixteenth International Conference on Machine Learning, June 1999 (Burlington, MA, USA: Morgan Kaufmann), 278–287.

Padrao, P., Dominguez, A., Bobadilla, L., and Smith, R. N. (2022). *Towards learning ocean models for long-term navigation in dynamic environments*. Chennai: OCEANS 2022, 1–8.

Paley, D. A., Zhang, F., and Leonard, N. E. (2008). Cooperative control for Ocean sampling: The glider coordinated control system. *IEEE Trans. Control Syst. Technol.* 16, 735–744. doi:10.1109/TCST.2007.912238

Provost, C. L., and Verron, J. (1987). Wind-driven ocean circulation transition to barotropic instability. *Dyn. Atmos. Oceans* 11, 175–201. doi:10.1016/0377-0265(87)90005-4

Richard, A., Brian, B., and Clifford, T. (2019). *Parameter Estimation and inverse problems (candice janco).* third edn.

Ridao, P., Yuh, J., Batlle, J., and Sugihara, K. (2000). *On AUV control architecture*. Kyoto, Japan: IEEE IROS.

Rosenblatt, J., Williams, S., and Durrant-Whyte, H. (2002). A behavior-based architecture for autonomous underwater exploration. *Inf. Sci. (N. Y).* 145, 69–87. doi:10.1016/s0020-0255(02)00224-4

D. L. Rudnick and M. Perry (Editors) (2003). *Alps: Autonomous and Lagrangian platforms and sensors.* Workshop Report, 64. Available at: www.geo-prose.com/ALPS.

Shadden, S., Leigen, F., and Marsden, J. (2005). Definition and properties of Lagrangian coherent structures from finite-time lyapunov exponents in two-dimensional aperiodic flows. *Phys. D. Nonlinear Phenom.* 212, 271–304. doi:10.1016/j.physd.2005.10.007

Shchepetkin, A. F., and McWilliams, J. C. (2005). The regional oceanic modeling system (ROMS): A split-explicit, free-surface, topography-following-coordinate oceanic model. *Ocean. Model.oxf.* 9, 347–404. doi:10.1016/j.ocemod.2004.08.002

Shen, J., Medjo, T., and Wang, S. (1999). On a wind-driven, double-gyre, quasi-geostrophic ocean model: Numerical simulations and structural analysis. *J. Comput. Phys.* 155, 387–409. doi:10.1006/jcph.1999.6344

Sherman, J., Davis, R. E., Owens, W. B., and Valdes, J. (2001). The autonomous underwater glider "Spray. *IEEE J. Ocean. Eng.* 26, 437–446. doi:10.1109/48.972076

Sherstov, A. A., and Stone, P. (2005). "Function approximation via tile coding: Automating parameter choice," in *Abstraction, reformulation and approximation*. Editors J.-D. Zucker and L. Saitta (New York: Springer Berlin Heidelberg), 194–205.

Singh, H., Yoerger, D., and Bradley, A. (1997). Issues in auv design and deployment for oceanographic research. *Proc. 1997 IEEE Int. Conf. Robotics Automation* 3, 1857–1862. doi:10.1109/ROBOT.1997.619058

Singh, S., Sutton, R., and Kaelbling, P. (1995). Reinforcement learning with replacing eligibility traces. *Mach. Learn.* 22, 123–158. doi:10.1023/A:1018012322525

Smith, R. N., Chao, Y., Li, P. P., Caron, D. A., Jones, B. H., and Sukhatme, G. S. (2010a). Planning and implementing trajectories for autonomous underwater vehicles to track evolving ocean processes based on predictions from a Regional Ocean model. *Int. J. Rob. Res.* 29, 1475–1497. doi:10.1177/0278364910377243

Smith, R. N., Das, J., Heidarsson, H., Pereira, A., Cetinić, I., Darjany, L., et al. (2010b). \{USC\} \{CINAPS\} builds bridges: Observing and monitoring the \{S\}outhern \{C\}alifornia \{B\}ight. *IEEE Robot. Autom. Mag.* 17, 20–30. doi:10.1109/mra.2010.935795

Smith, R. N., Das, J., Yi, C., Caron, D. A., Jones, B. H., and Sukhatme, G. S. (2010c). “Cooperative multi-AUV tracking of phytoplankton blooms based on ocean model predictions,” in *MTS/IEEE oceans 2010* (Sydney, Australia: IEEE), 1–10.

Smith, R. N., Heckman, C., Sibley, G., and Hsieh, M. A. (2015). “A representative modeling approach to sampling dynamic ocean structures,” in *Symposium on marine robotics - broadening horizons with inter-disciplinary science & engineering. A. Pascoal (horta, faial island, azores, Portugal)*. Editors K. Rajan and J. Sousa.

Smith, R. N., Schwager, M., Smith, S. L., Jones, B. H., Rus, D., and Sukhatme, G. S. (2011). Persistent ocean monitoring with underwater gliders: Adapting sampling resolution. *J. Field Robot.* 28, 714–741. doi:10.1002/rob.20405

Sutton, R. S., and Barto, A. G. (2018). *Reinforcement learning: An introduction*. Cambridge: MIT press.

Turner, R. M., and Stevenson, R. A. G. (1991). *Orca: An adaptive, context-sensitive reasoner for controlling AUVs. Proc 7th intnl symp. On unmanned untethered submersible tech.* Umhlanga, South Africa: UUST.

Whitcomb, L., Yoerger, D., Singh, H., and Howland, J. (1999). “Advances in underwater robot vehicles for deep ocean exploration: Navigation, control, and survey operations,” in *Proceedings of the ninth international symposium of robotics research* (London: Springer-Verlag Publications).

Whitcomb, L., Yoerger, D., Singh, H., and Mindell, D. (1998). “Towards precision robotic maneuvering, survey, and manipulation in unstructured undersea environments,” in *Robotics research - the eighth international symposium* (London: Springer-Verlag Publications), 45–54.

Wolligandt, S., Wilde, T., Roessl, C., and Theisel, H. (2020). A modified double gyre with ground truth hyperbolic trajectories for flow visualization. *Comput. Graph. Forum* 40, 209–221. doi:10.1111/cgf.14183

Xun, X., Cao, J., Mallick, B., Carroll, R., and Maity, A. (2013). Parameter estimation of partial differential equation models. *J. Am. Stat. Assoc.* 108, 1009–1020. doi:10.1080/01621459.2013.794730

Yoerger, D., and Slotine, J. (1985). Robust trajectory control of underwater vehicles. *IEEE J. Ocean. Eng.* 10 (4), 462–470. doi:10.1109/joe.1985.1145131

Yoo, B., and Kim, J. (2016). Path optimization for marine vehicles in ocean currents using reinforcement learning. *J. Mar. Sci. Technol.* 21, 334–343. doi:10.1007/s00773-015-0355-9

Yuh, J. (2000). Design and control of autonomous underwater robots: A survey. *Aut. Robots* 8, 7–24. doi:10.1023/a:1008984701078

Zhang, B., and Sukhatme, G. S. (2008). “Adaptive sampling with multiple mobile robots,” in *IEEE international conference on robotics and automation*.