2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Observability-based Energy Efficient Path Planning with Background Flow via Deep Reinforcement Learning

Jiazhong Mei, J. Nathan Kutz, and Steven L. Brunton

Abstract— **In many sensor estimation and monitoring tasks, the mobile sensor travels through the state-space under the influence of a complex background flow environment. System observability is commonly used to assess the performance of the sensor-based estimation, although for a mobile sensor there are other important metrics. We consider the path planning problem under the environmental background flow and focus on a cyclic trajectory that (i) maximizes the log determinant of the observability matrix, (ii) minimizes total energy consumption, and (iii) returns close to the initial location at the end of the period. We formulate a reinforcement learning (RL) scheme and define a reward function that justifies multiple objectives. We investigate the performance of a policy-based proximal policy optimization (PPO) algorithm and address the issue of partially observed states with an additional recurrent module. We present our results on two complex unsteady fluid dynamical systems.**

I. INTRODUCTION

Mobile sensors and autonomous vehicles are becoming increasingly important in many geophysical and engineering applications for tackling a wide range of tasks such as estimation, monitoring, and tracking. It is essential to plan optimal sensor trajectories to better guide sensors on these tasks. In many cases, the sensors are moving in an unsteady fluid environment so that the sensor motion is affected by environmental forces such as wind and ocean currents. Although these environmental flows complicate the sensor motion, they also provide opportunities for sensors to exploit the background flows for more efficient navigation [1], [2].

The classical path planning problem has been extensively studied. For instance, in graph-based approaches, the environment is represented in a graph with accessible free space as nodes and energy costs as weighted edges. This decomposition allows the use of graph-related shortest path algorithms such as Dijkstra's algorithm for navigation of the sensors [3]. Alternatively, sampling-based methods are more often applied in a large-scale, high-dimensional environment. The planning is performed on a randomly sampled mapping of the environment. In particular, the rapidly exploring random tree (RRT), and its cyclic variant rapidly exploring random cycle (RRC), are complete and efficient algorithms for path planning by growing a random tree in the environment rooted at the initial location [4], [5]. Madridano et al. [6] give a comprehensive and detailed review of these methods. However, many of these methods only consider static or time-invariant environmental flow fields. Thus, they are not directly applicable in dynamic environments where unsteady background flows are present. In complex, unsteady, and multiscale environments, it is challenging to control the sensors precisely to move from one location to another and efficiently compute trajectories at scale.

J. Mei is with the Department of Applied Mathematics, University of Washington, Seattle, WA 98195, USA jmei@uw.edu

S. L. Brunton is with the Department of Mechanical Engineering, University of Washington, Seattle, WA 98195, USA sbrunton@uw.edu

J. N. Kutz is with the Department of Applied Mathematics, University of Washington, Seattle, WA 98195, USA., and also with the Department of Electrical and Computer Engineering, University of Washington, Seattle, WA 98195, USA kutz@uw.edu

In recent years, researchers have studied path planning in a dynamic flow field using various approaches. A genetic algorithm for path planning in an ocean environment was developed by Alvarez et al. [7]. Some exploit Lagrangian coherent structures [8], [9] in the environment to assist path planning in an improved computation [10], [11]. Krishna et al. [1] established a connection between the trajectories planned by model predictive control and the coherent structures. Subramani and Lermusiaux [12] leveraged stochastic dynamically orthogonal (DO) level-sets from the vehicle speed function.

Advances in deep reinforcement learning (RL) have also helped address some of the challenges of path planning in a complex environment for more optimal and efficient solutions. Although gaining its popularity in the applications of games and robotics [13], [14], RL is useful for solving objectives for path planning seen as a sequence of actions and decisions interacting with the dynamical environment. For example, RL is used as a learnable deterministic method for finding cycles in the RRC approach for improved performance and efficiency [15]. Actor-Critic schemes are used for time-efficient point-to-point navigation, also known as *Zermelo's problem*, of a fixed speed swimmer in complex flows [2], [16]. The same problem is also tackled using other deep RL approaches such as the V-RACER algorithm [17], [18] and the adversarial Q-learning [19].

Many complex systems in the real world exhibit some periodic or quasi-periodic characteristics. It is practically useful for a sensor trajectory to return to a specified location periodically for maintenance and sensor recharging. However, few existing works consider path planning with these additional structural constraints, with most studies considering point-to-point navigation. In this work, we focus on planning a cyclic path for a mobile sensor to maximize system observability under a complex environment with background flow. Historically, system observability has been a popular choice of metric in many sensor placement [20], [21] and planning problems [22], [23], [24]. Not only is observability good for instantaneous spatio-temporal estimation, but it is also a necessary condition that is useful in improving Kalman filter recursive estimation of the complex system [25]. Some alternative choice of metrics include the reconstruction error [26] as well as the *a posteriori* error covariance [27], which are more accurate yet less efficient comparing to the system observability.

In most studies, the task is generally separated from the path planning process. When given a task such as estimation, it is common to first find optimal waypoints and then apply point-to-point path planning methods to efficiently navigate. Shriwastav et al. [28] search over random sequences of optimal waypoints for efficient trajectories. Although the approach simplifies the problem, it can introduce new issues. Splitting the optimization may result in waypoints that are costly to navigate between. As such, it is preferable to directly optimize the task objective and energy-efficient path planning simultaneously. We describe energy efficiency in terms of the energy expenditure within a fixed cycle of the path. The conflicting goals of maximizing observability and minimizing energy cost lead to a multi-objective optimization trade-off. We construct a linear combination of the objectives and investigate the effect of the weights on the resulting trajectories. We apply deep RL algorithms for solving our optimization problem and address the partially observable process with a recurrent component. Finally, we validate the performance on two complex flow environments: the double-gyre flow field and global sea surface temperature data.

II. BACKGROUND

In this section, we first motivate reduced order modeling and system observability concerning the environmental feature of interest. Then, we recall Markov decision processes for mobile sensor path planning to apply RL algorithms.

A. Reduced Order Modeling

Reduced order models (ROMs) are commonly used to represent high-dimensional dynamical systems in a more simple and efficient formulation. The high-dimensional data $\mathbf{x}_t \in \mathbb{R}^n$ from the system is projected to a low-rank representation $\mathbf{z}_t \in \mathbb{R}^m$ $(m < n)$ through a linear basis $\mathbf{\Psi} \in \mathbb{R}^{m\times n}$, $\mathbf{x}_t = \mathbf{\Psi}\mathbf{z}_t$. Often $\mathbf{z}_t$ is assumed to be approximated by linear dynamics $\mathbf{z}_{t+1} = \mathbf{\Lambda}\mathbf{z}_t + \mathbf{w}_t$, where $\mathbf{w}_t \in \mathbb{R}^m$ is a vector accounting for system disturbance and nonlinearity. Then, the high-dimensional data can be approximately decomposed as $\mathbf{x}_t \approx \mathbf{\Psi}\mathbf{\Lambda}^t z_0$.

Classical approximation approaches to find the projection basis include the Fourier or wavelet transforms, as well as the proper orthogonal decomposition (POD). However, these approaches generally do not guarantee that there is a simple dynamical system in the low-rank representation. Suppose that the high-dimensional dynamics is known, with linear dynamics represented by a matrix. Then one could use a truncated matrix eigenvalue decomposition to obtain a linear basis and low-rank dynamics with good approximation. However, in most cases the dynamics of the high-dimensional system is not given directly nor is it linear. Modern Koopman theory provides conditions under which a nonlinear system can be rewritten as an infinite-dimensional linear operator, and dynamic mode decomposition (DMD) is a data-driven approach to obtain a low-rank approximation of the model from data [29], [30], [31], [32]. In particular, the DMD modes constitute the linear projection basis from high-dimensional data to the low-rank representation. The DMD eigenvalues form a diagonal matrix capturing the dynamics of the low-rank system. We use in our experiments optimized DMD that fits a debiased Koopman decomposition from the data [33].

B. Observability

In a complex system, it is usually not possible to collect measurements from all locations in the state space. Additionally, what we observe can be a transformation of the original system states. Observability defines how well the measurements we observe may be used to reconstruct and estimate the system state. It is typically examined through the observability Gramian or the observability matrix. In a time-varying setting, the measurements can be defined as $\mathbf{y}_t = \mathbf{C}_t\mathbf{x}_t$, where the selection matrix $\mathbf{C}_t$ depends on time. In particular, $\mathbf{C}_t$ has standard unit vectors as columns if directly measured from the state space. We define the observability matrix of a time-varying system $\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t, \mathbf{y}_t = \mathbf{C}_t\mathbf{x}_t$ as:

$$\mathbf{O}_t = \begin{bmatrix} \mathbf{C}_t \\ \mathbf{C}_{t+1}\mathbf{A} \\ \cdots \\ \mathbf{C}_{t+n-1}\mathbf{A}^{n-1} \end{bmatrix}.$$

The system is observable if and only if the observability matrix has full (column) rank. In many problems, it is helpful to look to the conditionality of the observability matrix, such as condition number, trace, or determinant, as a more continuous measure of system observability. The system is considered to be better observed when the observability matrix is well-conditioned.

C. Markov Decision Process

The motion and decisions of the mobile sensor can be described as a Markov decision process (MDP). It is represented by the state space $\mathcal{S}$, the action space $\mathcal{A}$, and the observation space Ω. In a partially observable MDP (POMDP), the agent (sensor) cannot observe the full state but rather a function $o: \mathcal{S} \to \Omega$. At each step, the agent chooses an action based on its observation through a policy function $\pi: \Omega \to \mathcal{A}$. The state transition model $\mathcal{T}: \mathcal{S}\times\mathcal{A} \to \mathcal{S}$ determines the next state from the current state and action. In general, the agent receives rewards $r: \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to \mathbb{R}$ for its actions, the current state, and the next state. The objective is to optimize the total expected rewards over a time horizon T, $\mathbb{E}[\sum_{i=0}^{T-1} \gamma^t r(s_t, a_t)]$, where $0 \le \gamma \le 1$ is a discount factor. With this framework, we apply RL algorithms to optimize the path planning of the mobile sensor.

III. PROBLEM FORMULATION

A. Path Planning

In this paper, we focus on planning a trajectory for a single mobile sensor in an unsteady background flow. We model the environmental feature with a low-rank representation and examine the log determinant of the observability matrix along the path. Specifically, consider the following low-rank representation of the large-scale spatio-temporal data $\mathbf{x}_t$:

$$\begin{aligned} \mathbf{x}_t &= \mathbf{\Psi}\mathbf{z}_t, \quad \mathbf{z}_{t+1} = \mathbf{\Lambda}\mathbf{z}_t + \mathbf{w}_t, \\ \mathbf{y}_t &= \mathbf{C}_t\mathbf{x}_t + \mathbf{v}_t = \mathbf{C}_t\mathbf{\Psi}\mathbf{z}_t + \mathbf{v}_t, \end{aligned} \tag{1}$$

with the observability matrix

$$\mathbf{O} = \begin{bmatrix} \mathbf{C}_0\mathbf{\Psi} \\ \mathbf{C}_1\mathbf{\Psi}\mathbf{\Lambda} \\ \cdots \\ \mathbf{C}_{T-1}\mathbf{\Psi}\mathbf{\Lambda}^{T-1} \end{bmatrix}. \tag{2}$$

$\mathbf{C}_t$ is defined by the location of mobile sensor at time t. The dynamic feature of interest $\mathbf{x}_t$ can be the unsteady flow field, but it can also be other environmental features in general such as sea surface temperature in the ocean model. Regardless, the unsteady background flow affects the motion of the mobile sensor in terms of state transition in the RL algorithm, which we discuss in detail in Section III-B.

We aim to find a sensor path that: (i) maximizes the log determinant of the observability matrix of the system; (ii) minimizes the energy cost; (iii) completes a cyclic path by returning to the region close to the initial location at the end of the period; and (iv) is collision-free in an environment with obstacles. We incorporate these objectives into the RL formulation next.

B. RL Formulation

We formulate the path planning problem as a POMDP. We define the trajectory to be the sequence of states and actions $\{\mathbf{s}_0, \mathbf{a}_0, \mathbf{s}_1, \ldots, \mathbf{a}_{T-1}, \mathbf{s}_T\}$ of the sensor. The time horizon T is set to be the predefined cycle length. Based on the study of Gunnarson et al. [18] showing velocity information are effective in navigation in flow fields, we give the agent access to the position of the mobile sensor, local background velocity, as well as time. The continuous action space represents the velocity vector of the sensor. It can be unbounded in general or closed within an interval to provide limits to the sensor speed. The transition of the mobile sensor is influenced by the unsteady background flow. We define $\mathbf{u}(\mathbf{s}_t, t)$ as the vector function of background flow, which can either be a precise equation or an interpolation from data. Then, the transition follows the equation of motion $\dot{\mathbf{s}}_t = \mathbf{u}(\mathbf{s}_t, t) + \mathbf{a}_t$.

It is non trivial to translate our objectives listed in the previous section into a reward function to guide the learning of the RL agent. With different objectives acting together, it is important to formulate a reward that is learnable and convergent. Typically, a reward that is sparse in the process is harder to learn by the RL agent. For example, the observability matrix is not completed until the end of the process, and the return penalty depends on the final state of the mobile sensor. Instead, it is better to design the reward function to track the step-wise changes, while the total reward is still representative of the objectives. Therefore, we define a reward function with the following components:

$$r_t = r_t^1 + r_t^2 + r_t^3 + r_t^4, \tag{3a}$$

$$r_t^1 = \log\det(\mathbf{O}_{:t+1}) - \log\det(\mathbf{O}_{:t}), \tag{3b}$$

$$r_t^2 = -\lambda\|\mathbf{a}_t\|^2, \tag{3c}$$

$$r_t^3 = -\rho(d(\mathbf{s}_{t+1}, \mathbf{s}_0) - d(\mathbf{s}_t, \mathbf{s}_0)). \tag{3d}$$

Each component denotes a marginal change with respect to our objectives. Conceptually, r_t^1 represents the marginal information gain of observability from the new location compared to the previous step, r_t^2 is the energy cost weighted by a hyperparameter λ, and r_t^3 is the step-wise change in return distance weighted by a hyperparameter ρ. d is the distance function, which we take as the squared Euclidean distance between the locations. A hard constraint on the precise return of the mobile sensor by objective (iii) is generally a difficult task in a dynamic environment, so we relax it to be a soft penalty to minimize the distance between the starting and ending location of the sensor within the time horizon.

r_t^4 denotes all other sparse situational penalties to ensure the validity of the resulting path such as collision avoidance. For example, when moving in an environment with obstacles, r_t^4 can be defined to give a constant negative reward for hitting the obstacle and 0 otherwise. A good path solution should completely eliminate this component and have r_t^4 to be 0. With a discount factor of $\gamma = 1$, the total reward becomes

$$\sum_{i=0}^{T-1} r_t = \log\det(\mathbf{O}) - \lambda\sum_{i=0}^{T-1}\|\mathbf{a}_t\|^2 - \rho d(\mathbf{s}_T, \mathbf{s}_0) + \sum_{i=0}^{T-1} r_t^4, \tag{4}$$

which matches all of our objectives.

This problem is a POMDP. r_t^1 depends not only on the current and next sensor locations but all previously visited locations in the trajectory as well, resulting in a partially observed process. One solution is to append the location history to the state so that the process is Markovian, but the additional dimensions increase model and computation complexity. Alternatively, a common approach to tackle this issue is to include a recurrent component such as a recurrent neural network (RNN) or long short-term memory (LSTM) module in the model to internally memorize the previous states in the recurrent hidden layer [34], [35], [36].

Additionally, objective (i) and (ii) are usually conflicting with each other, given that the optimal locations with the most information are typically far apart from one another, resulting in higher energy costs. In this case, the optimal solution over the objectives is not unique. A path that uses more energy to explore farther regions and achieves better system observability may have the same total reward as a path that only observes in a close neighborhood but consumes minimal energy. These equivalent solutions make up a Pareto front through multi-objective optimization. Finding the entire Pareto optimal set is extremely difficult, which has been addressed in the past [37]. For simplicity, we focus on efficiently finding one solution in the Pareto front of these objectives.

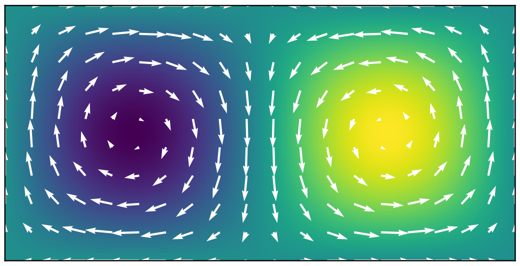

Fig. 1. Double-gyre flow at $t = 0$. The arrows are the velocity field, and the colored background represents vorticity values.

C. Deep Reinforcement Learning Algorithms

Modern reinforcement learning algorithms generally fall under one of the two main approaches: Q-learning or policy optimization. In Q-learning, the RL agent learns a Q-function that evaluates each state-action pair and chooses the best actions based on Q values. In policy optimization, the RL agent directly learns a policy function that chooses an action given the state. Typically, when the state and action space are continuous, it is tedious to learn a good Q-function, and policy methods are usually preferred.

Therefore, in this paper, we consider the framework of Proximal Policy Optimization (PPO) [14] for path planning. PPO is commonly seen in intelligent control for its simplicity and easy convergence. It uses the Actor-Critic architecture where we have two neural networks, the Actor and the Critic. The Actor network is in charge of learning the optimal policy and actions, while the Critic network estimates a value function to evaluate the states.

PPO uses a clipped surrogate objective:

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[\min(r_t(\theta)\hat{A}_t, r_t^c(\theta)\hat{A}_t) \right]. \tag{5}$$

$r_t(\theta) := \frac{\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)}{\pi_{\theta_{old}}(\mathbf{a}_t|\mathbf{s}_t)}$ is the probability ratio of choosing the action under the current policy versus the previous policy. $r_t^c(\theta) = \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)$ clips the probability ratio outside the interval $[1-\epsilon, 1+\epsilon]$. $\hat{A}_t$ is an estimator of the advantage function that evaluates the chosen action. This clipped objective prevents any large policy changes from the old policy and adds stability and reliability to the algorithm. Along with an MSE loss of the value function and an entropy term, they contribute to the final loss function to optimize over. Typically, during training, PPO assumes a stochastic policy under a Gaussian distribution with a mean learned from the algorithm and variable standard deviations. While in testing, a deterministic policy is used to find a single trajectory by choosing the mean action generated from PPO. Implementation details matter in PPO, and we follow the study by Engstrom et al. [38] when setting up our algorithm in the experiments.

IV. EXPERIMENTS

We examine the use of the RL framework for efficient path planning on two complex systems. For simplicity, we focus on the scenario where the initial location is known and fixed. However, the initial time is randomly set so the time-dependent background flow that the sensor experiences is different in each iteration. We apply PPO and PPO-LSTM algorithms to both experiments. The reinforcement learning agents are trained on an NVIDIA A40 GPU, while the environment state transitions are computed with a differential equation solver on CPU. The code is available at `github.com/frankmei33/DRLPathPlanning`.

A. Double-Gyre Flow

A double-gyre flow (Figure 1) is a flow pattern that is often seen in many geophysical flows and well studied for its coherent structures [39], [40]. It is described by the following stream function

$$\begin{aligned} \psi(x,y,t) &= A\sin(\pi f(x,t))\sin(\pi y), \\ f(x,t) &= \epsilon\sin(\omega t)x^2 + x - 2\epsilon\sin(\omega t)x, \end{aligned} \tag{6}$$

defined on a closed and bounded domain $[0,2]\times[0,1]$. The background flow is given by the velocity field

$$\begin{aligned} \mathbf{v}(x,y,t) &= \begin{bmatrix} -\frac{\partial\psi}{\partial y} \\ \frac{\partial\psi}{\partial x} \end{bmatrix} \\ &= \begin{bmatrix} -\pi A\sin(\pi f(x,t))\cos(\pi y) \\ -\pi A\cos(\pi f(x,t))\sin(\pi y)\frac{df}{dx} \end{bmatrix} \end{aligned} \tag{7}$$

We take the parameters with values $A = 0.5, \omega = 2\pi, \epsilon = 0.25$ such that the flow has a period 1 and a max velocity of $\pi A \approx 1.57$. We model the vorticity measurements (curl of velocity field) on a 201×101 discretized grid with a step size of 0.01. The discrete time step is 0.1. The vorticity field is highly compressible, so we find a low-rank representation by fitting an optDMD approximation with rank $r = 10$ to the sampled measurements. The action space is bounded in a box region with $\|\mathbf{a}\|_\infty \le 1$ to ensure that the sensor velocity is on the same scale as the background flow field velocity.

We aim to find energy-efficient, cyclic trajectories of length $T = 20$ using PPO and PPO-LSTM. The Actor-Critic networks are set with 2 hidden layers of size 16. A hyperbolic tangent activation function is used to connect between layers. In PPO-LSTM, a separate LSTM module with one hidden layer is appended before the regular Actor and Critic networks. The initial weights are set by orthogonal initialization. The continuous actions are sampled with annealing state-independent standard deviation and clipped to be smaller than the max flow velocity. We fix the initial sensor location to be at $(1, 0.8)$. Then, as the initial time varies, the horizontal component of the background flow changes direction as the gyres oscillate in the x-direction. We set the hyperparameters $\lambda = 1, \rho = 100$.

We first examine energy-efficient trajectories for an active mobile sensor. For references, we compare them with the free-flowing trajectories in Figure 2. The background flow carries the sensor under a path in the left or right half of the region depending on the starting time. However, the observability under free flowing trajectories is far from ideal as shown in Figure 4. In comparison, PPO and PPO-LSTM agents are able to find trajectories that explore the same half of the region and follow a similar shape as the free flowing trajectories in

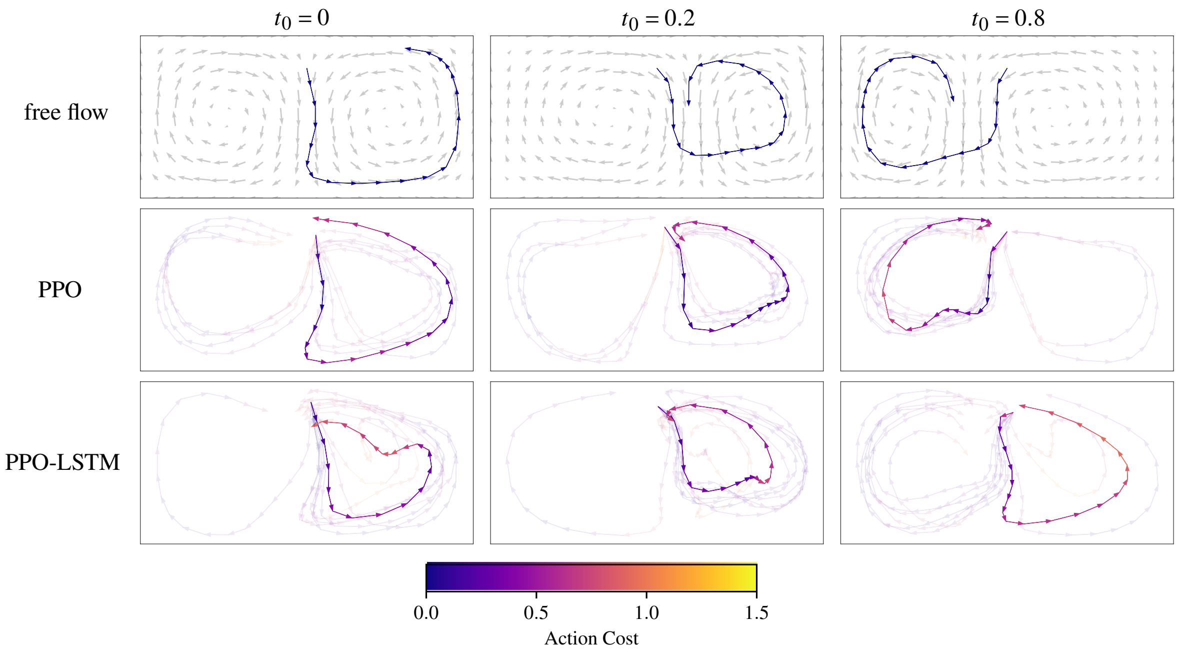

Fig. 2. Sensor trajectories in double-gyre flow. In each column, the sensor starts at a different initial time, which affects the background flow and the resulting planned path by PPO and PPO-LSTM. The trajectories are colored by the energy spent. Those with higher transparency are from repeated experiments.

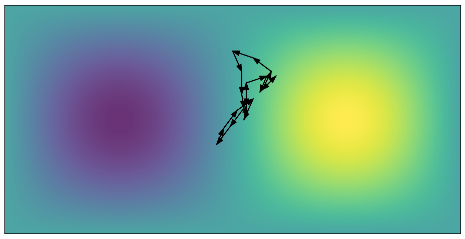

Fig. 3. Greedy sensor trajectory ignoring the effect and energy cost of the background flow. This is found by optimizing only the log determinant of the observability matrix using a greedy algorithm.

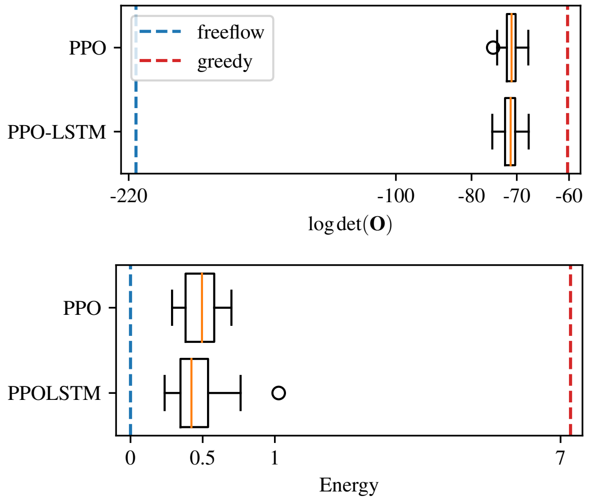

Fig. 4. Distribution of $\log\det(\mathbf{O})$ and energy cost along PPO and PPO-LSTM trajectories. The blue dashed lines represent values from a free-flowing, and the red dashed lines represent values from a greedy trajectory.

most cases of the repeated experiments. And with minimal learned actions, these trajectories return much closer to the initial location and achieve much better observability along the path.

We also generate a trajectory from optimizing only the conditionality of the observability matrix with a greedy QR column pivoting approach and ignoring background flow [25], with the max distance between time steps to be 0.1 in consistent with mobile sensor speed constraint (Figure 3). The resulting path is then used as waypoints in a model predictive control (MPC) optimization to efficiently navigate from one point to the next with a quadratic cost function. To successfully reach within a close distance of each waypoint, the sensor requires much larger speed and energy than the given constraint. In comparison, the trajectories planned by the RL agents achieve similar observability to the greedy trajectory, while consuming much less energy (Figure 4). Interestingly, the additional recurrent structure in PPO-LSTM only results in marginal improvement in terms of total rewards, path observability, and total energy spent. The difference is further narrowed as the weights increase. Since the partially observed component is only present in r_t^1, as the other objectives gain larger weights, the RL algorithms are less likely to be affected by the partially observed process.

We then compare the trajectories by adjusting the hyperparameter weight of the energy cost in Figure 5. As λ increases, it is more costly to exert large controls on the sensor, so we see that the actions are smaller in magnitude and more frequently in the same direction as the background flow. The direction of

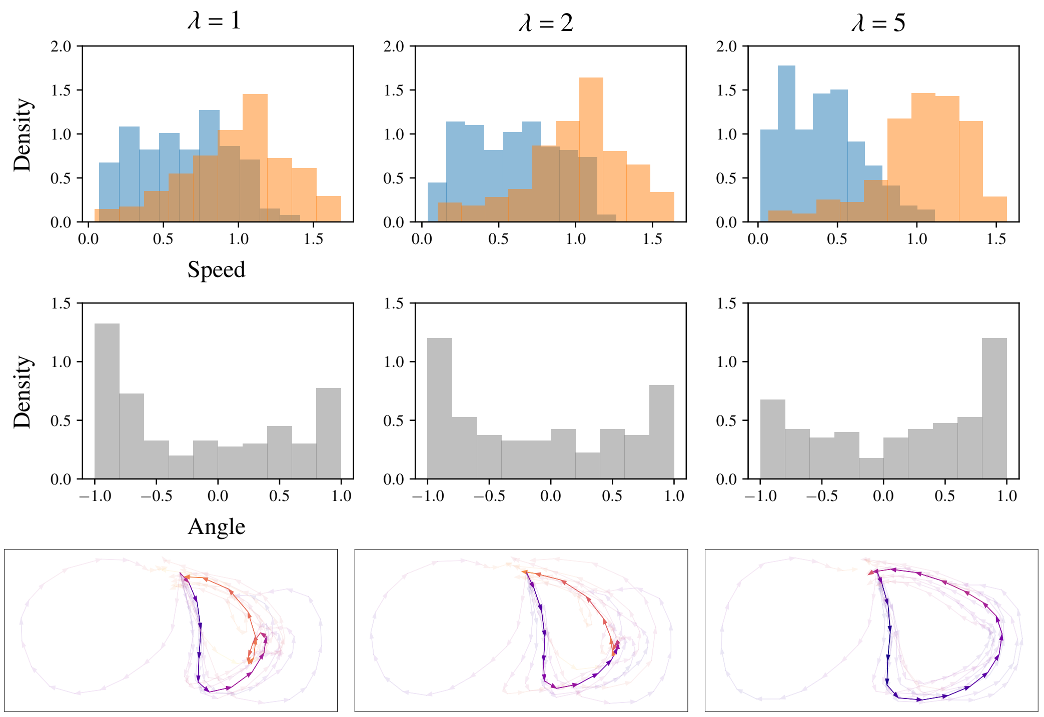

Fig. 5. The relative velocity of mobile sensor versus the background flow velocity with different regularization weights on energy cost. The top row of histograms are the distributions of the magnitude of actions (in blue) and the magnitude of background velocity (in orange) taken at the sensor locations. The second row of histograms are the distributions of the sensor orientation against background flow. The trajectories planned by PPO-LSTM are shown in the bottom row.

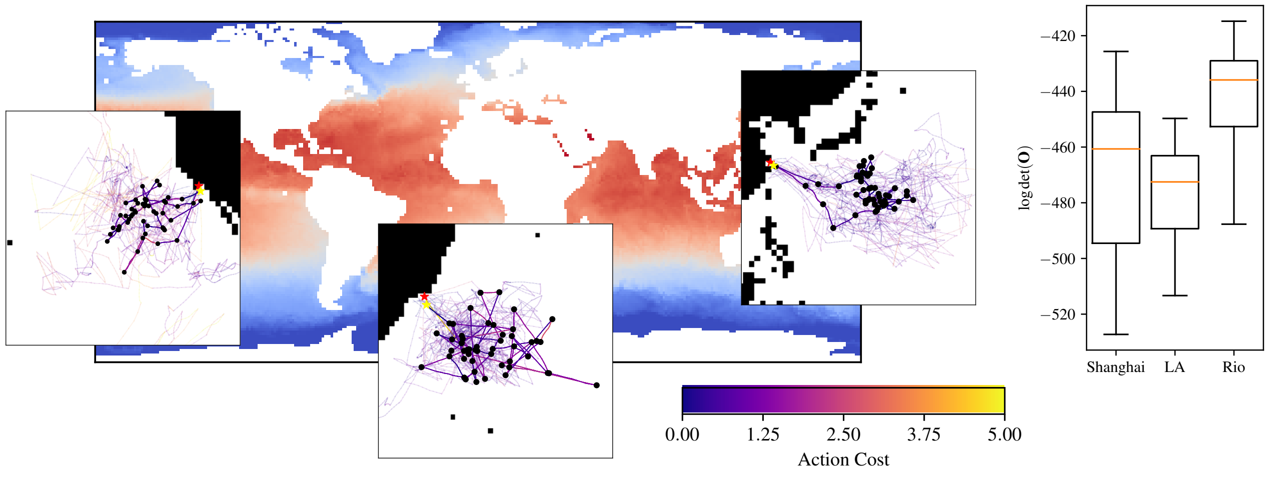

Fig. 6. (Left) A snapshot of HYCOM sea surface temperature in the background. Zoomed-in map and planned sensor trajectories starting near Los Angeles (left), Rio (middle), and Shanghai (right). Higher transparency paths are from repeated experiments. (Right) Box plots of log determinant of SST observability along the learned trajectories. ($\lambda = 1, \rho = 10$)

the control is more often in the same or opposite direction of the flow for reduced variation. When the sensor moves directly along or against the flow, its future location is more predictable. Moreover, the sensor spends extra energy exploring the center of the right gyre for better observability when the λ is small, while staying close to the perimeter moving along the flow with larger value of λ.

B. Sea Surface Temperature

Next, we study a real-world application concerning the sea surface temperature and find efficient trajectories under oceanic current flow. We acquire the ocean data from the HYbrid Coordinate Ocean Model (HYCOM) including sea surface temperature, eastward velocity, and northward velocity. The daily data is collected on a uniform 1.2-degree lat/lon grid (134×300) from 2001 to 2012. The weekly sea surface

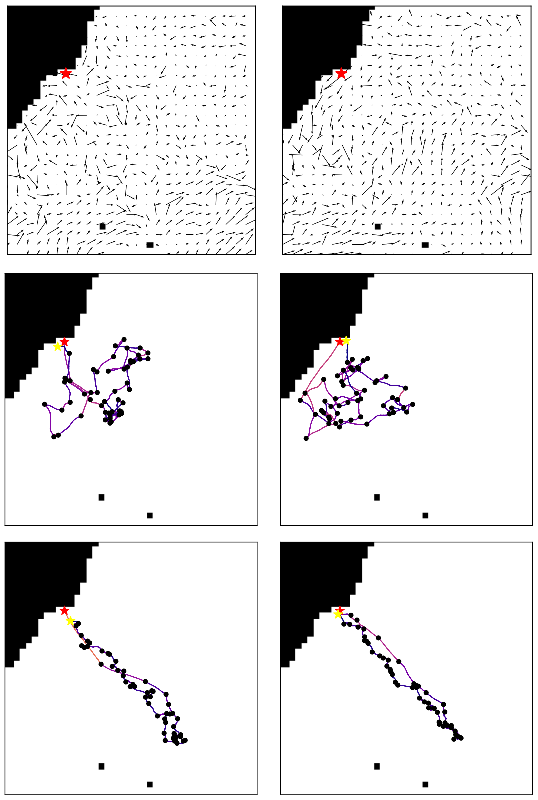
Fig. 7. Path planning from Rio starting on the same day of different years. The first row shows the HYCOM ocean flow velocity field at initialization, the middle row is the paths planned by PPO, and the last row is the paths planned by PPO-LSTM. ($\lambda = 5, \rho = 10$)

temperature data is used to fit a low-rank representation with optDMD approximation with rank $r = 52$. The ocean flow is very chaotic and difficult to model as shown in Figure 7, so we fit a continuous interpolation from the data for the transition model. Historically, sensors are deployed in the ocean to provide high-quality observations and validations to the satellite-measured sea surface temperature. These sensors can be carried on in situ moorings, drifting buoys, as well as ships. In our experiment, to obtain large spatial coverage for the estimation of global sea surface temperature, we set a bound on the action space with $||\mathbf{a}||_\infty \leq 5$ degrees per day, or equivalently 23 knots, to model the speed of the sensor on a ship. Additionally, the complex geographical layout requires a safe trajectory of the mobile sensor that avoids crashing into the land. We define r_4^t to be -100 if the sensor hits land and 0 otherwise.

We consider a trajectory with a length of 1 year (52 weeks) that matches the natural cycle of the ocean. The Actor-Critic networks are set with 2 hidden layers of size 32. Other model setups remain similar to the double-gyre flow experiment. We focus on a few major coastal cities and ports in the world on different continents and oceans, such as Rio, Los Angeles, and Shanghai, as the bases and initial locations for the mobile sensor. The planned trajectories are shown in Figure 6. In most cases, we observe that the RL agent first directs the sensor away from the land and into the ocean to avoid the possibility of land

TABLE I

TESTING RESULTS FROM REPEATED EXPERIMENTS USING PPO AND PPO-LSTM PLANNING TRAJECTORIES STARTING FROM RIO. ($\lambda = 5, \rho = 10$)

Model	PPO	PPO-LSTM
$\log\det(\mathbf{O})$	$-484.19(\pm 35.10)$	$\mathbf{-482.78(\pm 32.94)}$
Energy	$12.24(\pm 5.94)$	$\mathbf{7.15(\pm 2.75)}$
Return Distance	$\mathbf{2.61(\pm 2.59)}$	$6.62(\pm 11.09)$

collision. This behavior is most significant when the path starts near Shanghai where there are many lands in close proximity with the risk of collision. By our setup of the max speed limit, the ship carrying the sensor can overcome the background flow quite easily with a larger speed. Although the ocean consists of features with natural cycles, the changes in the ocean flow field in time are more complex as they do not repeat in a pattern like the double-gyre system. If we consider the span of multiple cycles, the background flow behaves differently within each cycle, even though the sensor starts its path at the same time of the year. The RL agent adjusts with the flow and plans a different trajectory in each cycle as shown in Figure 7 to ensure a similar performance in the objectives.

Through repeated experiments, we do not see significant difference in total rewards learned using PPO or PPO-LSTM algorithm, but the converged paths exhibit different patterns with focuses on different component of the reward function. While paths from PPO are mostly clustered and close to shore, those from PPO-LSTM more often extend further into the ocean. Aided by the historical information stored in the recurrent layer, PPO-LSTM is more confident in moving away and returning from a farther location compared with PPO. Additionally, we observe that in general PPO-LSTM finds paths with better system observability and energy cost than PPO, while returning further away from the staring location. The results are summarised in Table I.

In our experiments, the initial location of the sensor is fixed. We observe from repeated experiments that the initial sensor location affects the overall observability of the planned trajecotory (Figure 6). In future work, a more general RL agent can be trained to optimally place the initial location of the sensor in the first step and followed by forming a cyclic path around the chosen location.

V. CONCLUSION

In this work, we used PPO and the recurrent variant PPO-LSTM as RL agents to plan cyclic paths for a mobile sensor to maximize system observability while minimizing energy consumption navigating in the environment with unsteady background flow. We acknowledged the partially observable nature of the path planning problem and adopted a recurrent policy to account for the history of observations. We demonstrated with numerical experiments on double-gyre system and realistic ocean model that RL agents learn to plan efficient trajectories starting at a fixed location as the background flow changes. They achieve near-optimal system observability while significantly reducing total energy consumption comparing to the strategy of finding optimal waypoints and planning efficient paths separately. We may extend to a more general objective with complex objectives and learnable initial locations for future work. The framework can also be adapted to a multi-sensor setup where we shall consider the effect of local information and communication among multiple sensors on the performance of RL agents.

VI. ACKNOWLEDGMENTS

The authors acknowledge support from the National Science Foundation AI Institute in Dynamic Systems (grant number 2112085). SLB acknowledges support from the Air Force Office of Scientific Research (FA9550-21-1-0178). JNK acknowledges support from the Air Force Office of Scientific Research (FA9550-19-1-0011).

References

[1] K. Krishna, Z. Song, and S. L. Brunton, "Finite-horizon, energy-efficient trajectories in unsteady flows," *Proceedings of the Royal Society A*, vol. 478, no. 2258, p. 20210255, 2022.

[2] L. Biferale, F. Bonaccorso, M. Buzzicotti, P. Clark Di Leoni, and K. Gustavsson, "Zermelo's problem: optimal point-to-point navigation in 2d turbulent flows using reinforcement learning," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 29, no. 10, p. 103138, 2019.

[3] X. Bai, W. Yan, M. Cao, and D. Xue, "Distributed multi-vehicle task assignment in a time-invariant drift field with obstacles," *IET Control Theory & Applications*, vol. 13, no. 17, pp. 2886–2893, 2019.

[4] S. M. LaValle, "Rapidly-exploring random trees : a new tool for path planning," *The annual research report*, 1998.

[5] X. Lan and M. Schwager, "Rapidly exploring random cycles: Persistent estimation of spatiotemporal fields with multiple sensing robots," *IEEE Transactions on Robotics*, vol. 32, no. 5, pp. 1230–1244, 2016.

[6] Á. Madridano, A. Al-Kaff, D. Martín, and A. de la Escalera, "Trajectory planning for multi-robot systems: Methods and applications," *Expert Systems with Applications*, vol. 173, p. 114660, 2021.

[7] A. Alvarez, A. Caiti, and R. Onken, "Evolutionary path planning for autonomous underwater vehicles in a variable ocean," *IEEE Journal of Oceanic Engineering*, vol. 29, no. 2, pp. 418–429, 2004.

[8] G. Haller, "Lagrangian coherent structures," *Annual review of fluid mechanics*, vol. 47, pp. 137–162, 2015.

[9] S. C. Shadden, F. Lekien, and J. E. Marsden, "Definition and properties of lagrangian coherent structures from finite-time lyapunov exponents in two-dimensional aperiodic flows," *Physica D: Nonlinear Phenomena*, vol. 212, no. 3-4, pp. 271–304, 2005.

[10] A. Ramos, V. García-Garrido, A. Mancho, S. Wiggins, J. Coca, S. Glenn, O. Schofield, J. Kohut, D. Aragon, J. Kerfoot *et al.*, "Lagrangian coherent structure assisted path planning for transoceanic autonomous underwater vehicle missions," *Scientific reports*, vol. 8, no. 1, pp. 1–9, 2018.

[11] T. Salam, V. Edwards, and M. A. Hsieh, "Learning and leveraging features in flow-like environments to improve situational awareness," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2071–2078, 2022.

[12] D. N. Subramani and P. F. Lermusiaux, "Energy-optimal path planning by stochastic dynamically orthogonal level-set optimization," *Ocean Modelling*, vol. 100, pp. 57–77, 2016.

[13] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[14] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[15] J. Chen, T. Shu, T. Li, and C. W. de Silva, "Deep reinforced learning tree for spatiotemporal monitoring with mobile robotic wireless sensor networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 11, pp. 4197–4211, 2019.

[16] M. Buzzicotti, L. Biferale, F. Bonaccorso, P. Clark di Leoni, and K. Gustavsson, "Optimal control of point-to-point navigation in turbulent time dependent flows using reinforcement learning," in *International Conference of the Italian Association for Artificial Intelligence*. Springer, 2021, pp. 223–234.

[17] G. Novati and P. Koumoutsakos, "Remember and forget for experience replay," in *International Conference on Machine Learning*. PMLR, 2019, pp. 4851–4860.

[18] P. Gunnarson, I. Mandralis, G. Novati, P. Koumoutsakos, and J. O. Dabiri, "Learning efficient navigation in vortical flow fields," *Nature Communications*, vol. 12, no. 1, pp. 1–7, 2021.

[19] J. K. Alageshan, A. K. Verma, J. Bec, and R. Pandit, "Machine learning strategies for path-planning microswimmers in turbulent flows," *Physical Review E*, vol. 101, no. 4, p. 043110, 2020.

[20] S. Sinha, U. Vaidya, and R. Rajaram, "Operator theoretic framework for optimal placement of sensors and actuators for control of nonequilibrium dynamics," *Journal of Mathematical Analysis and Applications*, vol. 440, no. 2, pp. 750–772, 2016.

[21] K. Manohar, J. N. Kutz, and S. L. Brunton, "Optimal sensor and actuator selection using balanced model reduction," *IEEE Transactions on Automatic Control*, vol. 67, no. 4, pp. 2108–2115, 2021.

[22] L. DeVries, S. J. Majumdar, and D. A. Paley, "Observability-based optimization of coordinated sampling trajectories for recursive estimation of a strong, spatially varying flowfield," *Journal of intelligent & robotic systems*, vol. 70, no. 1, pp. 527–544, 2013.

[23] A. B. Asghar, S. T. Jawaid, and S. L. Smith, "A complete greedy algorithm for infinite-horizon sensor scheduling," *Automatica*, vol. 81, pp. 335–341, 2017.

[24] M. Rafieisakhaei, S. Chakravorty, and P. Kumar, "On the use of the observability gramian for partially observed robotic path planning problems," *arXiv preprint arXiv:1801.09877*, 2018.

[25] J. Mei, S. L. Brunton, and J. N. Kutz, "Mobile sensor path planning for kalman filter spatiotemporal estimation," *arXiv preprint arXiv:2212.08280*, 2022.

[26] B. Li, H. Liu, and R. Wang, "Efficient sensor placement for signal reconstruction based on recursive methods," *IEEE Transactions on Signal Processing*, vol. 69, pp. 1885–1898, 2021.

[27] V. Tzoumas, A. Jadbabaie, and G. J. Pappas, "Sensor placement for optimal kalman filtering: Fundamental limits, submodularity, and algorithms," in *2016 American control conference (ACC)*. IEEE, 2016, pp. 191–196.

[28] S. Shriwastav, G. Snyder, and Z. Song, "Dynamic compressed sensing of unsteady flows with a mobile robot," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 11 910–11 915.

[29] C. W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, and D. S. Henningson, "Spectral analysis of nonlinear flows," *Journal of fluid mechanics*, vol. 641, pp. 115–127, 2009.

[30] P. J. Schmid, "Dynamic mode decomposition of numerical and experimental data," *Journal of fluid mechanics*, vol. 656, pp. 5–28, 2010.

[31] J. H. Tu, C. W. Rowley, D. M. Luchtenburg, S. L. Brunton, and J. N. Kutz, "On dynamic mode decomposition: theory and applications," *Journal of Computational Dynamics*, vol. 1, no. 2, pp. 391–421, 2014.

[32] J. N. Kutz, S. L. Brunton, B. W. Brunton, and J. L. Proctor, *Dynamic mode decomposition: data-driven modeling of complex systems*. SIAM, 2016.

[33] T. Askham and J. N. Kutz, "Variable projection methods for an optimized dynamic mode decomposition," *SIAM Journal on Applied Dynamical Systems*, vol. 17, no. 1, pp. 380–416, 2018.

[34] M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps," in *2015 aaai fall symposium series*, 2015.

[35] J. Yuan, H. Wang, C. Lin, D. Liu, and D. Yu, "A novel gru-rnn network model for dynamic path planning of mobile robot," *IEEE Access*, vol. 7, pp. 15 140–15 151, 2019.

[36] B. Li and Y. Wu, "Path planning for uav ground target tracking via deep reinforcement learning," *IEEE access*, vol. 8, pp. 29 064–29 074, 2020.

[37] K. Van Moffaert and A. Nowé, "Multi-objective reinforcement learning using sets of pareto dominating policies," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3483–3512, 2014.

[38] L. Engstrom, A. Ilyas, S. Santurkar, D. Tsipras, F. Janoos, L. Rudolph, and A. Madry, "Implementation matters in deep policy gradients: A case study on ppo and trpo," *arXiv preprint arXiv:2005.12729*, 2020.

[39] M. Michini, M. A. Hsieh, E. Forgoston, and I. B. Schwartz, "Robotic tracking of coherent structures in flows," *IEEE Transactions on Robotics*, vol. 30, no. 3, pp. 593–603, 2014.

[40] B. T. Nadiga and B. P. Luce, "Global bifurcation of shilnikov type in a double-gyre ocean model," *Journal of physical oceanography*, vol. 31, no. 9, pp. 2669–2690, 2001.