Exploring Learning-based Control Policy for Fish-like Robots in Altered Background Flows

Xiaozhu Lin[1], Wenbin Song[1], Xiaopei Liu[1], Xuming He[1] and Yang Wang[1]

***Abstract*— The study of motion control for the fish-like robots in complex fluid fields is of great importance in improving the performance of underwater vehicles, due to its strong maneuverability, propulsion efficiency, and deceptive visual appearance. In this article, a novel learning-based control framework is first proposed to autonomously explore efficient control policies that are capable of performing motion control tasks in *non-quiescent and unknown* background flows. First, we utilize a high-fidelity simulation system, named FishGym, to generate various uniform flows. Next, a DRL-based algorithm is incorporated with the FishGym to train the fish-like robot to control its motion to optimally complete a delicately designed task (Approaching Target and Stay) in both quiescent and uniform flow. Then, the obtained control policy together with an online estimator is directly applied to a Path-Following Task. The proposed framework well balances the simulation accuracy and the computational efficiency, which is of crucial importance for effective coupling with the learning algorithm. The simulation results indicate that, via the proposed learning framework, the robot successfully acquired a swimming strategy that can be used to adapt to different background flows and tasks. Furthermore, we also observe some adaptation behavior of the robot, such as rheotaxis, that is similar to the fish in nature, which gains us more insight into the mechanism underlying the adaptation behavior of fish in a complex environment.**

I. Introduction

Due to the huge potential to be used in engineering, the robotic society has long been interested in comprehending and taking advantage of the motion control strategy used by animals in nature [1][2]. Understanding the control strategy adopted by animals and insects is of great importance in improving the performance of man-made robots. In recent years, Deep Reinforcement Learning (DRL), has been proven to be a powerful paradigm to study bio-inspired robots, especially for those with complex dynamics, such as the snake robots [3], the flapping wing hummingbird robots [4][5], the quadruped robots [6][7], and underwater robots [8][9]. However, for those robots, obtaining an implementable DRL policy through real-world interactions is not only time-consuming but also likely to damage the robot. As a result, the paradigm that transfers learned policies from the simulation platform, such as MuJoCo [10] and ISAAC [11], to the actual robot is frequently used [12][13]. Despite the impressive results produced by the combination of DRL-based approaches and simulation platforms, one obvious challenge with using such a methodology is that, in order to obtain a robust control policy for every specific task, the learning agent must repeatedly explore a wide range of different possible actions in many different environmental states, which in turn poses a great challenge on the computational efficiency and accuracy of the simulator [14].

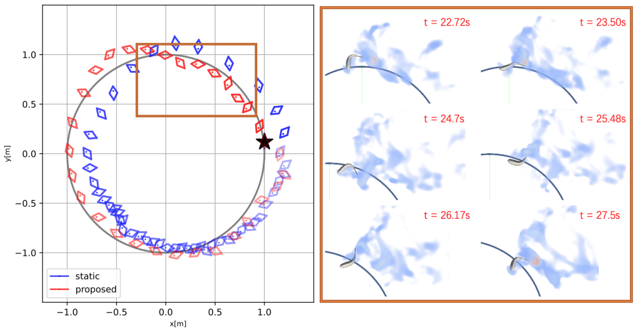


Fig. 1: **Comparison of the static flow trained approach and the proposed approach in a flow field environment.** The left part shows the trajectories of the fish-like robot for the path-following control task with two different policies in a nonstatic flow field. The direction of the flow is from left to right and the speed is $0.2m/s$. The diamond pattern represents the fish-like robot, and the dot in the diamond pattern means the head direction of the robot. The star pattern means the start position of the path, and the direction of the path is clockwise. The right part shows the specific posture of the fish-like robot.

In this work, among all kinds of bio-inspired robots, we focus on the fish-like robots [15][16][17] that have drawn intensive investigation thanks to their strong maneuverability, propulsion efficiency, and deceptive appearance. Following the sim-to-real paradigm, significant efforts [15][18][19] have been devoted to developing learning-based control policies for the fish-like robots to complete a specific task. To accurately reflect reality, a straightforward idea is to directly combine the DRL with Computational Fluid Dynamics (CFD) simulators [20]. But traditional numerical solutions based on the Navier–Stokes (NS) equations [9][8] or dynamic overset unstructured grid method [21][22] will lead to an unacceptable rising in the time cost of the training process, even just in 2D space. One widely used solution is to use a data-driven (surrogate) model [23][24][25] generated from real-world interaction to partially or completely replace the original CFD simulator. But such an approach limits the obtained control policy to the particular agent in a particular environment, as shown in Section III. Furthermore, the majority of the DRL-based control research works for the fish-like robots mentioned above are restricted to a quiescent flow field in 2D space. Very few works consider an non-quiescent

[1]Xiaozhu Lin, Wenbin Song, Xiaopei Liu, Xuming He and Yang Wang are with the School of Information Science and Technology, ShanghaiTech University, Shanghai, China {linxzh, songwb, liuxp, hexm, wangyang4}@shanghaitech.edu.cn

This work was supported by National National Science Foundation of China (No. 62072310).

flow field [26][27][28] and/or 3D environment [25], where either the control task is reduced to a rather simple one (e.g. single angle attitude control for position fixed fish-like robot [26].) or the model accuracy is dramatically sacrificed for the computation cost.

In summary, the difficulties of efficiently simulating and designing a motion controller for fish-like robots mainly stem from the absence of an efficient control framework along with a fast simulated environment that enables the fish-like robot to learn swimming techniques, especially in unknown and non-quiescent 3D flow environments. Inspired by the above observations, this paper addresses a nontrivial control problem for multi-articulated fish-like robots in *unknown and altered flow fields*.

To solve the problem, we propose a learning-based adaptive control framework, which combines a DRL-based controller, a Supervised Learning (SL)-based estimator, and a high-performance 3D simulator to enable the robots to quickly adapt to the different environments and complete the task. Here in this work, the environment with non-quiescent background flow fields is simulated by a tailored FishGym [29] simulator that achieves excellent performance and efficiency [30][31][32] in simulating the two-way interaction dynamic between fish-like robots and non-stationary surrounding fluid in a 3D space. As a result, it enables us to generate a variety of flow fields and combined them with reinforcement learning methods in a reasonable time (see Section II for more detail). Instead of end-to-end learning, we break down the training task and obtain the control policy in two steps. First, the flow-related knowledge is assumed to be available and a full-information controller is trained in various uniform flows generated by the tailored FishGym. Secondly, we use the data generated in the first stage to train an observer that can estimate the flow-related knowledge given the current and recent history information of the robot itself. Then, the trained controller and estimator are combined to form a control algorithm for the fish-like robot to execute in an environment with altered flow fields. Finally, the effectiveness of the proposed scheme is verified by two typical motion control tasks, namely an Approaching Target and Stay (ATS) task and a Path-Following Control (PFC) mission, both in altered flows.

It should be noted that we acknowledge that the learning methods employed in the presented scheme are not completely original, since proposing new learning techniques is not our focus. Instead, the main contribution of this work is to develop an interpretable and generalizable control framework that enables researchers to combine advanced learning-based methods with a high-fidelity simulator to solve a high-dimensional, nonconvex, and nonlinear control problem for a robot operating in a complex flow environment. Specifically for fish-like robots, to the best of our knowledge, this is the first work that obtains a learning-based adaptive control policy for a fish-like robot that is capable of quickly adapting to altered flows in simulation. This preliminary work opens the possibility to produce proper control policies for fish-like robots swimming in a more complex (unsteady and even vortical) flow field, with promising applications for better sim-to-real transfer.

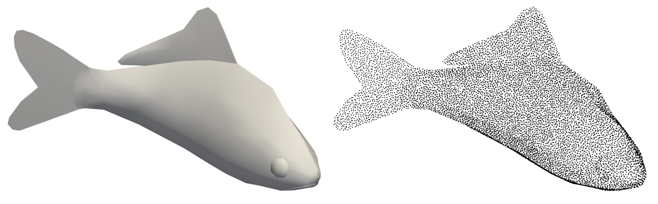

(a) A fish-like robot model (b) Sampling points for IBM

Fig. 2: **Illustration of a fish-like robot example.** (a) A skinned fish-like robot model, which is created from the Computer Aided Design (CAD) software. (b) The generated sampling points of a fish-like robot for IBM. The sampling size is 10000.

The rest of this article is organized as follows. Section II introduces the modified FishGym simulation and the multi-link robotic fish briefly. In Section III, the motivation for this work has been demonstrated. In Section IV, we propose a learning-based framework that combines the DRL-based controller and the SL-based estimator. Simulation results are presented in Section V. Finally, Section VI concludes the article with some discussion.

II. Fish-like Robot and Experimental Platform

A. Fish-like Robot

Inherited from the FishGym, we model a fish-like robot by using an articulated rigid body with surface skinning. An example of a fish-like robot model is shown in Fig. 2(a). The skeleton motion of a fish-like robot is driven by the articulated rigid body dynamics [33]. Given a specific set of joint angles, the surface shape of a fish is determined by the linear blend skinning method [34]. In this way, a fish-like robot can modify its skeleton pose by just changing its joint angles, which consequently determines its surface shape.

B. FishGym Simulator with Flow Fields

FishGym is a high-performance underwater robot simulation platform based on a recently proposed GPU-optimized lattice Boltzmann solver [32]. Compared with traditional CFD-based simulators, FishGym can greatly accelerate the learning process for fish-like robots control. To support generating altered flows, some modifications have been made to the FishGym. First of all, the simulation is conducted in a large fixed domain, rather than in a moving local domain as in [29]. Next, an inlet boundary condition is set for the stream source. Given a desired inflow velocity $v(x_{\text{inlet}}, t)$, the discrete distribution of the inlet, $f_i(x_{\text{inlet}}, t)$, is directly set as the Maxwell-Boltzmann equilibrium distribution. For other domain boundaries, Neumann boundary conditions are applied. Immersed boundary method (IBM) is used for fluid-structure interaction. To calculate the penalty force in IBM, we uniformly sample points on the surface of the solid object by using the method proposed in [35], which is an efficient algorithm to generate Poisson disk sample sets with a desired size. Fig. 2(b) shows an example of the sampling points for a fish-like robot. To speed up the sampling for IBM, Open Multi-Processing (OpenMP) is used to make sampling parallel.

III. Key Observation

In this section, we will demonstrate that considering the flow information in the first place of designing a motion controller for the fish-like robot is essential. It has long been observed that fish can adapt to various flow fields and successfully achieve their goal. The adaptation behaviors, such as rheotaxis[2] [36][37] and Karman gaiting [38], outperform what is currently possible with an artificial autonomous underwater vehicle. The mechanism underlying these adaptation behaviors is complex and far from being fully understood. A key issue in understanding the underlying mechanism is the clarification of the role of the flow environment. However, the majority of the learning framework in the literature did not consider the nonstatic background flow field that can dramatically change the two-way interaction dynamic between fish-like robots and the surrounding fluid.

Fig. 1 compares the performance of two different policies in completing a path-following task in a uniform flow field: the blue one is generated by a DRL approach trained in static flow while the red one is driven by the proposed method. The result clearly shows that the uniform background flow degrades the performance of the static policy and causes the robot unable to adapt its motion to the environment change and compensate for the flow-induced displacement. Moreover, for our policy, special attention needs to be paid to the behavior during the last seconds (highlighted in an orange box and magnified in the left-half part of Fig.1), where the robot not only keeps its position close to the desired trajectory but also chooses to turn around to face the oncoming flow. Note that, we did not use any angle-related term in the learning process (more details will be given later in Section IV). This behavior is known as rheotaxis. Thus, we believe the proposed framework allows us to study the control policy of fish swimming in complex environments, with which researchers are able to understand the design concept and control strategy of fish, and put them into the production of man-made underwater vehicles.

IV. Hybrid Framework for Control Problems of Fish-like Robot in Unknown Flow Fields

In this section, we intend to utilize the modified FishGym simulator described in Section II to develop a learning-based control policy that allows the fish-like robot to complete several control tasks in uniform flow fields with unknown velocity and direction.

The overall structure of the proposed learning framework is depicted in Fig. 3, which consists of two modules: the Full Information Controller (FIC) and the Flow Characteristic Estimator (FCE). The FIC takes all the state of the robot, task and flow-related knowledge as the inputs and outputs of an action vector to the robot, while the flow-related knowledge is provided by FCE. Instead of employing any extra sensors such as Artificial Lateral Line Systems (ALLS), the FCE tends to identify the flow-related information using merely the sequential history of the state-action pairs. In what follows, we will introduce how we obtain each module via the presented learning framework in detail.

[2]which is a tendency of the fish to directly face into an oncoming current to capture the food carried by the flow

A. Formulating the Control Problem

The control problem is considered to be a Markov Decision Process (MDP), which can be described as the tuple $(\mathcal{S}, \mathcal{A}, r, f_\mu)$. At each time step k in an episode, the robot observes a state $\mathbf{s}_k \in \mathcal{S}$, executes an action $\mathbf{a}_k \in \mathcal{A}$ according to a control policy $\pi(\mathbf{a}_k|\mathbf{s}_k)$, and receives a scalar reward $\mathbf{r}_k = r(\mathbf{s}_k, \mathbf{a}_k)$, in which reward function $r(\cdot,\cdot)$ is used to evaluate how well the robot accomplishes the task. Then, the state $\mathbf{s}_k$ changes to $\mathbf{s}_{k+1}$ according to environmental dynamics $f_\mu(\mathbf{s}_k, \mathbf{a}_k)$ which is parameterized by a constant vector μ characterizing the uniform flow field, reads as follows

$$\mu = [\theta_{flow}, v_{flow}] \tag{1}$$

where θ_{flow}, v_{flow} is the direction and speed of the uniform flow, respectively. At the end of each episode, we get a trajectory $\tau(\pi) = [\mathbf{s}_0, \mathbf{a}_0, \mathbf{r}_0, \mathbf{s}_1, \mathbf{a}_1, \mathbf{r}_1, \cdots]$ under π. The objective for the robot is to learn a control policy π^* that maximizes the expectation of the discounted cumulative reward defined as

$$\pi^* = \arg\max_\pi \mathbb{E}_{\tau(\pi)}[\sum_{k=0}^{\infty} \gamma^t \mathbf{r}_k], \tag{2}$$

where $\gamma \in [0, 1)$ is the discount factor.

To be specific, the state $\mathbf{s}_k$ concatenate three portions is given by

$$\mathbf{s}_k = [\mathbf{s}_k^{robot}, \mathbf{s}_k^{task}, \mathbf{s}_k^{flow}] \tag{3}$$

where the $\mathbf{s}_k^{robot}$ is the state of the robot is measured by proprioceptive sensors, which is defined as:

$$\mathbf{s}_k^{robot} = [\theta_{robot}, v_{robot}, \theta_{joints}, v_{joints}] \tag{4}$$

where θ_{robot}, v_{robot} is the orientation and linear velocity of the fish-like robot, respectively. The θ_{joints}, v_{joints} refer to the angle and velocity of the robot's joints. The $\mathbf{s}_k^{task}$ stands for some task-related information depending on the specific task. The action $\mathbf{a}_k$ is the torque input to the joints of the robot, and the reward r_k is task-specific and will be introduced with tasks in Section V.

In the proposed framework, we ensemble an extra state vector $\mathbf{s}_k^{flow}$ to represent the flow-related knowledge. Choosing an appropriate state vector $\mathbf{s}_k^{flow}$ to describe the flow, especially for the complex fluid fields, is again a challenging issue that remains unsolved. Nevertheless, since the flow field in this work is altered and uniform flow field, the straightforward idea is to use a 2D vector to characterize the flow-related state s_k^{flow} that is defined as

$$s_k^{flow} = [\Delta\theta, \Delta v] \tag{5}$$
$$\Delta\theta = \theta_{flow} - \theta_{robot}, \quad \Delta v = v_{flow} - v_{robot}$$

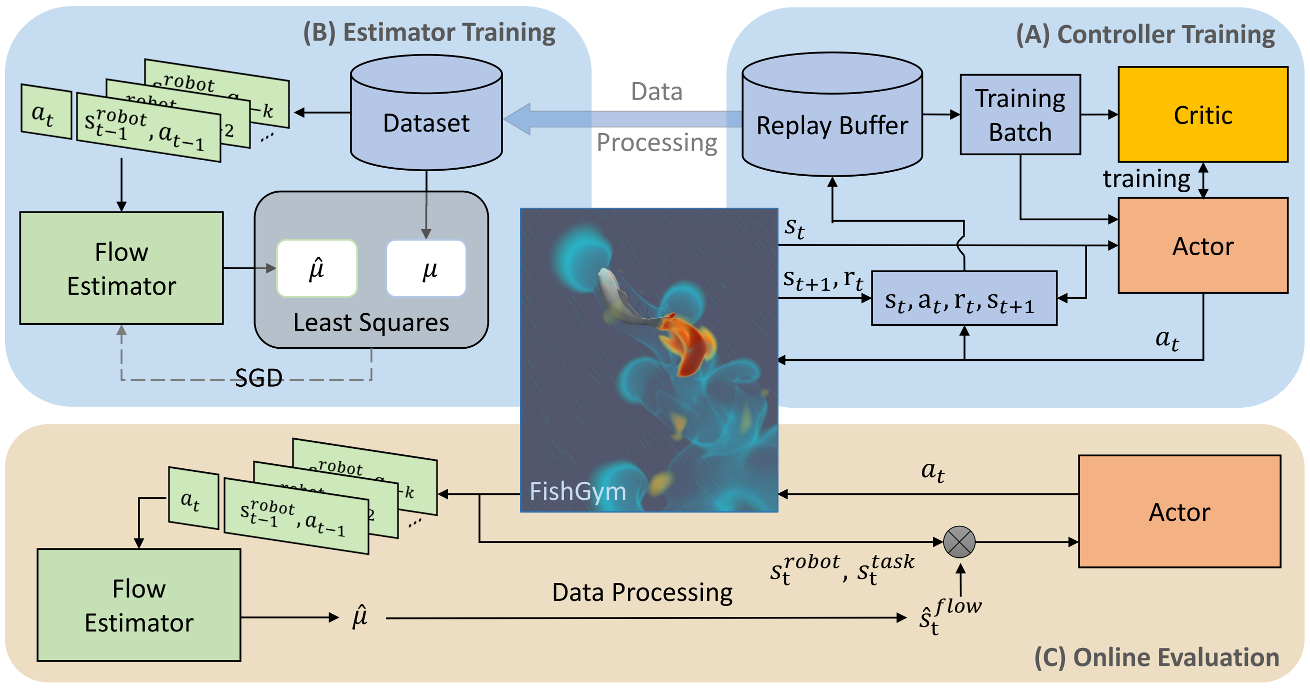


Fig. 3: **Proposed framework.** The central part is the tailored FishGym simulator which can generate altered uniform flow. (A) The controller training process for the full information controller. The actor network interacts with FishGym and stores the interaction data in the replay buffer for critic network and actor network learning. (B) The training process for flow characteristic estimator. The dataset is processed from the replay buffer. The input of the flow estimator is the current action concatenates with the history of state-action pairs and the output of the estimator is the vector μ used to describe the flow characteristics. (C) The online evaluation in unknown flow fields. Once trained, the weights of the flow estimator network and actor network are fixed. The flow estimator use the history interact data to estimate $\hat{\mu}$, then process $\hat{\mu}$ into $\hat{s}^{flow}_t$ and concatenate it with the current s^{robot}_t and s^{task}_tas input for actor network (i.e. full information controller)

where $\Delta\theta$ is the relative angle between the direction of the robot θ_{robot} and the direction of the flow θ_{flow}, Δv is the difference between the velocity of the background flow v_{flow} and the velocity of the robot.

B. Full Information Controller

Now, we formally introduce the training procedure for the FIC. To break the mutual dependence between the controller and estimator, we first assume that the flow characteristic vector μ is available in training the FIC, that is the state $\mathbf{s}^{flow}_k$ can be directly calculated by (1) and (5).

In this work, we use the Soft Actor-Critic (SAC) [39] algorithm to train the FIC. SAC is one of the actor-critic methods, it takes the entropy of the action into consideration and automatically uses the temperature coefficient α to adjust the weight of the entropy term. These features can greatly improve the sampling efficiency of the algorithm and shorten the data required for training. The usage of this maximum entropy framework leads to robust policies, that do not collapse into a single successful trajectory but explore the complete range of successful trajectories. For more details of the SAC algorithm, the reader is referred to [39]. In addition, SAC is an off-policy algorithm, which makes it more sample efficient than on-policy algorithms like PPO [40]. This feature makes the SAC favorable in this work from the computation efficiency perspective. Nevertheless, the success of the proposed framework does not entirely rely on SAC, which can be replaced by many other more advanced DRL algorithms as long as the algorithm complexity does not dramatically increase. After sufficient training, we fixed the actor network as our FIC.

C. Flow Characteristic Estimator

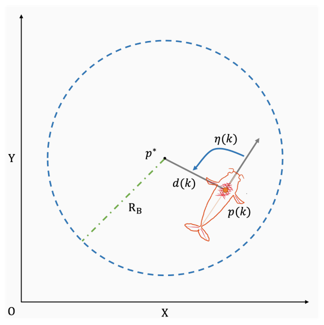
To remove the unrealistic assumption on the accessibility of the flow characteristic vector μ in FIC, in this section, we will train an FCE to provide an estimate $\hat{\mu}$ with sufficient accuracy, and further obtain the estimated flow-related information $\hat{\mathbf{s}}^{flow}_k$ by (5). As we mentioned before, the ALLS [26][41][27] sensor which usually is used for fish-like robots in non-static flow is not available. Therefore, we used the history of state-action pairs to extract flow-related information.

To this end, we formulate a regression problem to train a flow characteristic estimator ϕ that can estimate the flow-related information $\hat{\mu}$, given the current action and the history of state-action pairs, which is described as:

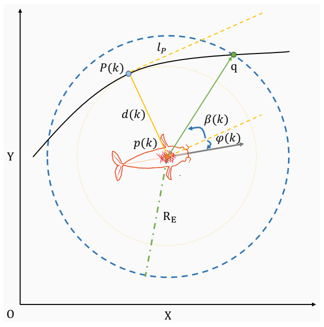
$$\hat{\mu}(k) = \phi((\mathbf{s}^{robot}_{k-n}, \mathbf{a}_{k-n}), \cdots, (\mathbf{s}^{robot}_{k-1}, \mathbf{a}_{k-1}), \mathbf{a}_k) \quad (6)$$

where n is the length of history of state-action pairs. Then, we can get the estimated information $\hat{\mathbf{s}}^{flow}_k$ by (1) and (5).

To train the FCE, we use the interaction data in the replay buffer which is generated by the DRL training process of FIC as shown in Fig. 3. In this way, we avoid generating



(a)



(b)

Fig. 4: **Schematic of the two control tasks of the fish-like robot.** (a) ATS task. The p^* is the target point and the R_B is the exploration bound that prevents the agent from getting too far away from the target. $p(k)$ is the position of robot and $\phi(k)$ is the orientation of the robot at the time step k. $d(k) := ||p(k) - p^*|| \in \mathbb{R}$ is the distance between the centroid of the fish-like robot and the target point p^* the at time step k, $\eta(k)$ is the relative angle between its current orientation θ_{robot} and the ray from $p(k)$ to p^*. (b) PFC task. Given a path, the $P(k)$ is the closest point to $p(k)$ in the path and the $d(k)$ is the length of the vector pointing from $p(k)$ to $P(k)$. The R_E is the exploration range for the fish-like robot and the q is the farthest point in the given path that the fish-like robot can detect. The tangent line l_p at the $P(k)$ in the given path point to the direction of the given path. The $\beta(k)$ is the relative angle between $p(k)q$ and l_p. The $\phi(k)$ is the relative angle between l_p and the direction of the fish-like robot.

new training data in the simulator again, which remarkably reduces the training workload for the whole framework.

V. Evaluations

In this section, we evaluate the proposed framework in two control tasks. The first one is the Approaching Target and Stay (ATS) task which requires the fish-like robot to approach a particular position in altered flows with out overshoot and hold it for a certain time. The second application is a Path-Following Control (PFC) task [23][21][29] which demands the fish-like robot to accurately follow a predefined trajectory.

Simulations are deployed on the tailored FishGym simulator in Section II with a simulation timestep set to $0.004s$. The domain size is $4\text{m} \times 4\text{m} \times 1\text{m}$ (length $\times$ width$\times$ height), and the grid resolution is $200 \times 200 \times 50$. The size of the fish-like robot is $0.32\text{m} \times 0.05\text{m} \times 0.08\text{m}$. The fish-like robot has 4 joints and the control timestep of the fish-like robot is $0.2s$.

All the simulations are conducted on a personal computer with an Intel Core i9-12900K CPU and NVIDIA GeForce RTX 3080 Ti GPU. Due to the GPU parallel computational feature of FishGym, the training process by FishGym is dramatically quicker than the traditional CFD environment [9][8][21][22]. For instance, [21] needs 16 days for training just 50 episodes. However, as we mention after, the whole learning process in this paper includes 4000 episodes, but only cost 55 hours.

A. Formulating the Approaching Target and Stay Task

As shown in Fig. 4, the fish-like robot is placed in a circular area. Its goal is to reach the center of the circular

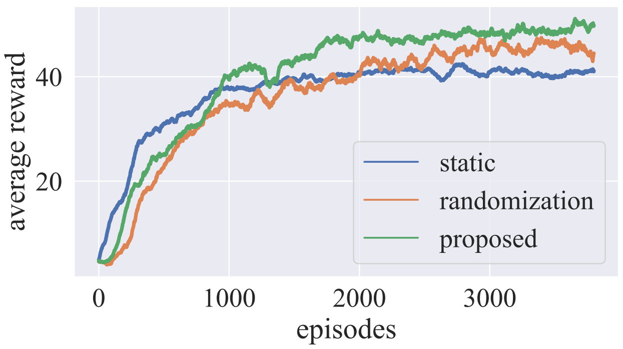


Fig. 5: **Training process for proposed approach and two baselines.** The static controller is trained in static background flow. The randomization controller and the proposed controller are trained in the altered flow field. Each controller has trained 4000 episodes. The average reward in this figure is smoothed by sliding windows for clear visualization.

and try to stay nearby. Referring to the definition of ATS task, the task-related state $\mathbf{s}_k^{task}$ is defined as:

$$s_k^{task} = [d(k), \eta(k)]. \tag{7}$$

which indicates that only local information is known to the robot, i.e. the robot can obtain only the relative information from its actual position and pose to the target position and pose. Action $\mathbf{a}_k$ is set to be a vector of the joints' torque. The reward function received by the robot after executing the action $\mathbf{a}_k$ is specified as

$$r(k) = 1/(1 + \delta d(k)) \tag{8}$$

with $\delta = 18$ is a hyper-parameter. Each episode will terminate when the distance exceeds the exploration bound $R_B = 0.5\text{m}$ or the timestep reaches the maximum value $k_{max} = 100$, i.e. 20 seconds.

Note that, the majority of the state-of-art literature [20][42] uses the Point-to-Point Navigation (PTPN) task to train the fish-like robot in which the episode will terminate once the fish-like robot arrived at the target point. However, this type of task usually teaches the robot to swim fast and results in a large overshoot after reaching the target point. Thus, the fish-like robot cannot learn how to sense the dynamic of the flow field and properly use such information to swim smartly. We want to emphasize that, different from the PTPN task [43] with only one objective, the delicately designed ATS not only takes the position control into consideration but also the pose requirement which is given implicitly in the reward function. Thus, ATS enables the robot to learn a more sophisticated control policy, such as the rheotaxis behavior exhibited in Fig. 1, to cope with the effect brought by the background flow.

B. Training for FIC

In this subsection, we will describe the learning process and results of FIC for the ATS task using the FishGym simulator and our DRL-based training methodology proposed in IV.

1) Training Process: At the beginning of each training episode, the fish-like robot is randomly located in the circle centered around p^* satisfying $d(0) < R_B$ and $\psi(0) \in [-\pi, \pi]$. As for the uniform background flow, the direction and velocity will be randomly chosen from the set $v_{flow} \in \{0.0, 0.1, 0.2, 0.3\}$ and $\theta_{flow} \in \{0, 0.5\pi, \pi, 1.5\pi\}$, separately. Then, the characteristic vector μ of the background flow will remain constant in the current episode. Each episode continues until the fish-like robot either goes out of bounds (i.e. $d(k) \geq R_B$) or the time step reaches its maximum value. The proposed FIC is implemented using fully connected layers. The structure of the actor-network is $15 \times 256(\text{ReLU}) \times 256(\text{ReLU}) \times 4$, while that of the critic-network is $19 \times 256(\text{ReLU}) \times 256(\text{ReLU}) \times 1$. The learning rate of the actor network, critic network and temperature parameter α are 0.0003, 0.001 and 0.001, respectively. The reward discount factor γ is 0.9. The whole learning process costs us 55 hours for 4000 episodes. The obtained FIC is referred to as $\pi^o_{ATS}(\mu)$.

2) Training Results: Fig.5 shows the learning curves smoothed by a sliding window. To further demonstrate the effectiveness of the proposed framework, we also trained two additional control policies for comparison: a static flow DRL policy π^s_{ATS} and a randomization policy π^r_{ATS}. The π^r_{ATS} policy is obtained through the exact same learning procedure (including the task, the background flow, and the network) as that of $\pi^o_{ATS}(\mu)$. The only difference is that policy π^r_{ATS} is trained without the given flow-related state s^{flow}_k. The static DRL policy π^s_{ATS} is trained only in static background flow, which is similar to the policy presented in our previous paper [29]. As shown in Fig. 5, three policies have a similar rate of growth in their average reward, but the proposed FIC $\pi^o_{ATS}(\mu)$ achieves a higher reward in the later stage. This indicates that the robot has learned how to take advantage of the background flow to reach its destination faster and thus gain a higher reward.

C. Training for FCE

Even with the ability to adapt to different background flows, the FIC $\pi^o_{ATS}(\mu)$ can only succeed at a task when given accurate flow characteristic vector μ, and this oracle knowledge is typically not readily available without any extra sensors like ALLS. As we promised before, here we tend to train an online estimator that provides a sufficiently accurate estimate $\hat{\mu}$ such that we can substitute $\hat{\mu}$ into the FIC $\pi^o_{ATS}(\mu)$ to obtain an implementable FCE-based FIC, namely $\pi^o_{ATS}(\hat{\mu})$. We would like to emphasize that, for the FCE, the training data comes from the latest 10000 data in the replay buffer which is obtained during the FIC training process. This is based on one key observation that FCE only needs to be accurate for the trajectories that are likely to be observed when performing the tasks of interest. Hence, the training for FCE does not involve any CFD simulation, and in turn, tremendously improves the computational efficiency of the proposed algorithm.

The training of the FCE network is conducted according to the SL algorithm given in Section IV, The FCE network is realized with fully connected layers where the structure of the network is $80 \times 256(ReLU) \times 256(ReLU) \times 2$ and the number of the historical information used n is 4. The loss is calculated by the MSELoss function to complete the gradient descent to update the network weights. The results are neglected due to space limitations.

D. Testing in ATS Task

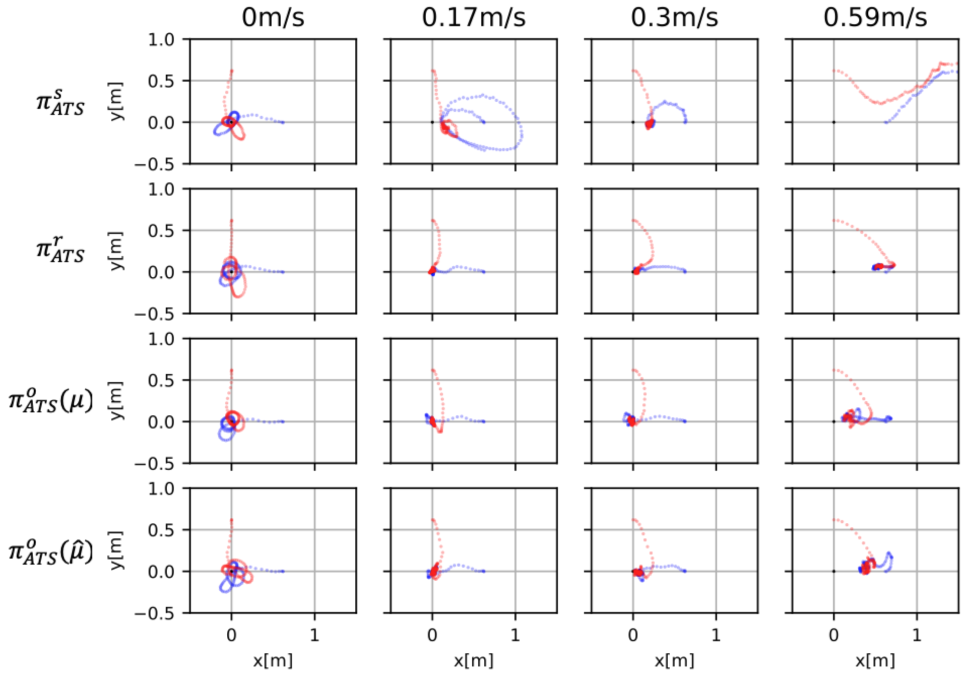
Now we deploy the aforementioned four policies (π^s_{ATS}, π^r_{ATS}, $\pi^o_{ATS}(\mu)$, $\pi^o_{ATS}(\hat{\mu})$) to the fish-like robot to complete the ATS task in different background uniform flows generated by our tailored FishGym simulator. In the test, the speed of the background flow v_{flow} varies from 0 to 0.59, but the direction θ_{flow} is fixed to be zero degrees (moving from left to right). However, the initial positions and orientations of the robot are different, with one being orthogonal to the direction flow while the other one directly facing the oncoming flow. Hence, the flow direction from the standpoint of the robot is not the same, thus can comprehensively show the ability of our proposed method.

The resulting trajectories of the fish-like robot are shown in Fig. 6(a). The behaviors of the four algorithms in the quiescent flow are comparable. However, once the environment has a nonstatic background flow, the π^s_{ATS} obtained in the quiescent environment quickly failed, which again illustrates the significance of considering the influence of background flow for the motion control of the underwater vehicle. When facing the flows with speed (0.17m/s and 0.3 m/s) that are covered in training, all the rest of the methods succeeded in completing the ATS task which indicates that the DRL-based algorithm indeed enable the robot to learn an adaptive policy to compensate for the influence brought by the background flow. However, once the flow speed exceeds the training region, the performance of π^r_{ATS} and $\pi^o_{ATS}(\hat{\mu})$ degrades. Nevertheless, it is worth pointing out that, the $\pi^o_{ATS}(\mu)$ policy can better cope with the unforeseen background flow. Therefore, we believe the reason behind the failure of the randomization policy π^r_{ATS} and the proposed one $\pi^o_{ATS}(\hat{\mu})$ are essentially different, where the former one reveals the limitation of an end-to-end DRL approach and the later one implies the critical role of an accurate estimator. These phenomena motivate us to improve our FCE in future work, but on the other hand, also imply the explainability and generalizability of the presented framework.

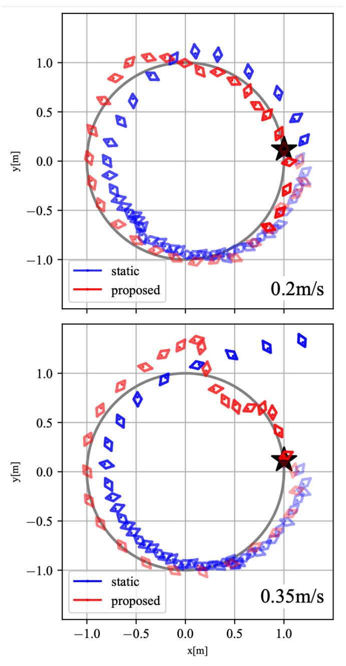
E. Transfer to PFC task

Now, to showcase the generalizability of the presented framework, we directly deploy the FCE-based FIC obtained in the ATS task, namely $\pi^o_{ATS}(\hat{\mu})$, to a classical PFC task [44] that requires a robot to accurately follow a predefined geometric path.

As shown in Fig. 4(b), the robot is said to be able to follow a path if the distance $d(k)$ is sufficiently small. But to drive the robot to swim forward and facilitate the $\pi^o_{ATS}(\hat{\mu})$, the point q instead of the $P(k)$ is chosen as the target position p^* of the ATS task. The rest of the states in s_k are identical to the original one. The simulation results are given in Fig.



(a)



(b)

Fig. 6: **Evaluation in two control tasks.** (a) Approaching target and stay task. We test the proposed policy with other policy in flow fields with different flow velocity. The fish-like robot with two different initial pose (i.e. position and orientation) blue one and red one, respectively. The test in the same column means the same velocity of background flow. The test in the same row means conducted by the same controller. (b) Path-following control task. We test the proposed policy and the static flow policy in two different flow fields, top one is $0.2m/s$ and bottom one is $0.35m/s$. The direction of the flow is from left to right. We perform this task by transfer the PFC task to ATS task without any policy re-train. The desired path is the circle with $1m$ radius.

6(b). Again, the $\pi^o_{ATS}(\hat{\mu})$ outperforms the policy obtained in a static environment π^s_{ATS} in both two nonquiescent background flows. In addition, we would like to draw the reader's attention to the last few seconds of the PFC task, where we clearly observe the turn-around action of the fish-like robot under our policy. Since the background flow flows from left to right, such behavior known as rheotaxis for fish in nature enables the fish to face the oncoming flow and hold its position with minimal effort.

VI. Conclusion and Discussions

This work develops a numerical study framework for underwater robots operating in non-quiescent background flows by utilizing a DRL and an immersed boundary-lattice Boltzmann method (IB-LBM)-based simulator. As mentioned before, to facilitate successful learning-based policy exploration, we need two essential features to form the whole framework: first, computational efficiency, which is crucial for the agent to learn in a reasonable time, and second, simulation accuracy, which can reflect not only the nonlinear fluid-structure interaction during swimming but also the complex background flow field that the robot may encounter in reality.

Based on the above considerations, in the presented framework, the fish swimming in a viscous incompressible flow is simulated with an IB-LBM-based fluid solver, known as FishGym in this paper, which has been validated in our previous publications [29]. Based on that, we carefully tailored FishGym to generate various uniform flows without slowing down the computation of fluid dynamics or losing accuracy. Then, a learning methodology is designed to train an adaptive control policy for the robot to perform a specific control task. Unlike the end-to-end approach, our control policy is formed by coupling a full information controller with an online flow characteristic estimator. But the coupling only exists in the implementation stage rather than during the whole training process. Thus, we significantly reduce the computation cost while ensuring the effectiveness of the estimator-based controller by explicitly integrating the estimated parameters into the input state. Finally, we use two typical motion control tasks (ATS and PFC) to verify our algorithm. In this way, we believe the proposed framework achieves state-of-the-art simulation accuracy and control effectiveness, which is crucial for sim-to-real transfer currently under investigation. In future work, the effectiveness of the proposed learning-based control policy will be verified via a real fish-like robot in an experiment environment.

References

[1] J. C. Liao, "A review of fish swimming mechanics and behaviour in altered flows," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 362, no. 1487, pp. 1973–1993, 2007.

[2] P. Chirarattananon, K. Y. Ma, and R. J. Wood, "Adaptive control of a millimeter-scale flapping-wing robot," *Bioinspiration & biomimetics*, vol. 9, no. 2, p. 025004, 2014.

[3] G. Sartoretti, W. Paivine, Y. Shi, Y. Wu, and H. Choset, "Distributed learning of decentralized control policies for articulated mobile robots," *IEEE Transactions on Robotics*, vol. 35, no. 5, pp. 1109–1122, 2019.

[4] Z. Tu, F. Fei, and X. Deng, "Bio-inspired rapid escape and tight body flip on an at-scale flapping wing hummingbird robot via reinforcement learning," *IEEE Transactions on Robotics*, vol. 37, no. 5, pp. 1742–1751, 2021.

[5] F. Fei, Z. Tu, J. Zhang, and X. Deng, "Learning extreme hummingbird maneuvers on flapping wing robots," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 109–115, IEEE, 2019.

[6] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, p. eaau5872, 2019.

[7] N. Rudin, H. Kolvenbach, V. Tsounis, and M. Hutter, "Cat-like jumping and landing of legged robots in low gravity using deep reinforcement learning," *IEEE Transactions on Robotics*, vol. 38, no. 1, pp. 317–328, 2021.

[8] G. Novati, S. Verma, D. Alexeev, D. Rossinelli, W. M. Van Rees, and P. Koumoutsakos, "Synchronisation through learning for two self-propelled swimmers," *Bioinspiration & biomimetics*, vol. 12, no. 3, p. 036001, 2017.

[9] S. Verma, G. Novati, and P. Koumoutsakos, "Efficient collective swimming by harnessing vortices through deep reinforcement learning," *Proceedings of the National Academy of Sciences*, vol. 115, no. 23, pp. 5849–5854, 2018.

[10] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5026–5033, IEEE, 2012.

[11] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, *et al.*, "Isaac gym: High performance gpu-based physics simulation for robot learning," *arXiv preprint arXiv:2108.10470*, 2021.

[12] P. Kormushev, S. Calinon, and D. G. Caldwell, "Reinforcement learning in robotics: Applications and real-world challenges," *Robotics*, vol. 2, no. 3, pp. 122–148, 2013.

[13] K. Alam, T. Ray, and S. G. Anavatti, "Design optimization of an unmanned underwater vehicle using low-and high-fidelity models," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 11, pp. 2794–2808, 2015.

[14] S. L. Brunton, B. R. Noack, and P. Koumoutsakos, "Machine learning for fluid mechanics," *Annual review of fluid mechanics*, vol. 52, pp. 477–508, 2020.

[15] J. Yu, C. Wang, and G. Xie, "Coordination of multiple robotic fish with applications to underwater robot competition," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 2, pp. 1280–1288, 2015.

[16] A. Raj and A. Thakur, "Fish-inspired robots: design, sensing, actuation, and autonomy—a review of research," *Bioinspiration & biomimetics*, vol. 11, no. 3, p. 031001, 2016.

[17] R. K. Katzschmann, J. DelPreto, R. MacCurdy, and D. Rus, "Exploration of underwater life with an acoustically controlled soft robotic fish," *Science Robotics*, vol. 3, no. 16, p. eaar3449, 2018.

[18] J. Liu, L. E. Parker, and R. Madhavan, "Reinforcement learning for autonomous robotic fish," *Mobile Robots: The Evolutionary Approach*, pp. 121–135, 2007.

[19] S. K. Rajendran and F. Zhang, "Learning based speed control of soft robotic fish," in *Dynamic Systems and Control Conference*, vol. 51890, p. V001T04A005, American Society of Mechanical Engineers, 2018.

[20] R. Tian, L. Li, W. Wang, X. Chang, S. Ravi, and G. Xie, "Cfd based parameter tuning for motion control of robotic fish," *Bioinspiration & Biomimetics*, vol. 15, no. 2, p. 026008, 2020.

[21] T. Zhang, R. Tian, H. Yang, C. Wang, J. Sun, S. Zhang, and G. Xie, "From simulation to reality: A learning framework for fish-like robots to perform control tasks," *IEEE Transactions on Robotics*, vol. 38, no. 6, pp. 3861–3878, 2022.

[22] P. M. Carrica, R. V. Wilson, R. W. Noack, and F. Stern, "Ship motions using single-phase level set with dynamic overset grids," *Computers & fluids*, vol. 36, no. 9, pp. 1415–1433, 2007.

[23] T. Zhang, R. Tian, C. Wang, and G. Xie, "Path-following control of fish-like robots: A deep reinforcement learning approach," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 8163–8168, 2020.

[24] S. Yan, Z. Wu, J. Wang, M. Tan, and J. Yu, "Efficient cooperative structured control for a multijoint biomimetic robotic fish," *IEEE/ASME Transactions on Mechatronics*, vol. 26, no. 5, pp. 2506–2516, 2020.

[25] L. Yan, X. Chang, R. Tian, N. Wang, L. Zhang, and W. Liu, "A numerical simulation method for bionic fish self-propelled swimming under control based on deep reinforcement learning," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 234, no. 17, pp. 3397–3415, 2020.

[26] J. Zheng, T. Zhang, C. Wang, M. Xiong, and G. Xie, "Learning for attitude holding of a robotic fish: An end-to-end approach with sim-to-real transfer," *IEEE Transactions on Robotics*, vol. 38, no. 2, pp. 1287–1303, 2021.

[27] M. Kruusmaa, P. Fiorini, W. Megill, M. De Vittorio, O. Akanyeti, F. Visentin, L. Chambers, H. El Daou, M.-C. Fiazza, J. Ježov, *et al.*, "Filose for svenning: A flow sensing bioinspired robot," *IEEE Robotics & Automation Magazine*, vol. 21, no. 3, pp. 51–62, 2014.

[28] T. Salumäe, I. Ranó, O. Akanyeti, and M. Kruusmaa, "Against the flow: A braitenberg controller for a fish robot," in *2012 IEEE International Conference on Robotics and Automation*, pp. 4210–4215, IEEE, 2012.

[29] W. Liu, K. Bai, X. He, S. Song, C. Zheng, and X. Liu, "Fishgym: A high-performance physics-based simulation framework for underwater robot learning," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 6268–6275, IEEE, 2022.

[30] X. Liu, W.-M. Pang, J. Qin, and C.-W. Fu, "Turbulence simulation by adaptive multi-relaxation lattice boltzmann modeling," *IEEE transactions on visualization and computer graphics*, vol. 20, no. 2, pp. 289–302, 2012.

[31] W. Li, K. Bai, and X. Liu, "Continuous-scale kinetic fluid simulation," *IEEE transactions on visualization and computer graphics*, vol. 25, no. 9, pp. 2694–2709, 2018.

[32] W. Li, Y. Chen, M. Desbrun, C. Zheng, and X. Liu, "Fast and scalable turbulent flow simulation with two-way coupling," *ACM Transactions on Graphics*, vol. 39, no. 4, pp. Art–No, 2020.

[33] R. Weinstein, J. Teran, and R. Fedkiw, "Dynamic simulation of articulated rigid bodies with contact and collision," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 3, pp. 365–374, 2006.

[34] T. Magnenat, R. Laperrière, and D. Thalmann, "Joint-dependent local deformations for hand animation and object grasping," tech. rep., Canadian Inf. Process. Soc, 1988.

[35] C. Yuksel, "Sample elimination for generating poisson disk sample sets," in *Computer Graphics Forum*, vol. 34, pp. 25–32, Wiley Online Library, 2015.

[36] J. C. Montgomery, C. F. Baker, and A. G. Carton, "The lateral line can mediate rheotaxis in fish," *Nature*, vol. 389, no. 6654, pp. 960–963, 1997.

[37] C. Baker and J. Montgomery, "The sensory basis of rheotaxis in the blind mexican cave fish, astyanax fasciatus," *Journal of Comparative Physiology A*, vol. 184, pp. 519–527, 1999.

[38] G. Toming, L. D. Chambers, and M. Kruusmaa, "Experimental study of hydrodynamic forces acting on artificial fish in a von kármán vortex street," *Underwater Technology*, vol. 32, no. 2, pp. 81–91, 2014.

[39] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

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

[41] D. S. Jung, P. P. Pott, T. Salumäe, and M. Kruusmaa, "Flow-aided path following of an underwater robot," in *2013 IEEE International Conference on Robotics and Automation*, pp. 4602–4607, IEEE, 2013.

[42] Y. Zhu, J.-H. Pang, and F.-B. Tian, "Point-to-point navigation of a fish-like swimmer in a vortical flow with deep reinforcement learning," *Frontiers in Physics*, p. 237, 2022.

[43] L. Li, A. Liu, W. Wang, S. Ravi, R. Fu, J. Yu, and G. Xie, "Bottom-level motion control for robotic fish to swim in groups: modeling and experiments," *Bioinspiration & biomimetics*, vol. 14, no. 4, p. 046001, 2019.

[44] Y. A. Kapitanyuk, A. V. Proskurnikov, and M. Cao, "A guiding vector-field algorithm for path-following control of nonholonomic mobile robots," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 4, pp. 1372–1385, 2017.