Model-Reference Reinforcement Learning Control of Autonomous Surface Vehicles

Zhang, Qingrui; Pan, Wei; Reppa, Vasso

DOI
10.1109/CDC42340.2020.9304347

Publication date
2020

Document Version
Final published version

Published in
Proceedings of the 59th IEEE Conference on Decision and Control, CDC 2020

Citation (APA)
Zhang, Q., Pan, W., & Reppa, V. (2020). Model-Reference Reinforcement Learning Control of Autonomous Surface Vehicles. In *Proceedings of the 59th IEEE Conference on Decision and Control, CDC 2020* (pp. 5291-5296). IEEE. https://doi.org/10.1109/CDC42340.2020.9304347

2020 59th IEEE Conference on Decision and Control (CDC)
Jeju Island, Republic of Korea, December 14-18, 2020

Model-Reference Reinforcement Learning Control of Autonomous Surface Vehicles

Qingrui Zhang[1,2], Wei Pan[2], and Vasso Reppa[1]

Abstract— **This paper presents a novel model-reference reinforcement learning control method for uncertain autonomous surface vehicles. The proposed control combines a conventional model-based control method with deep reinforcement learning. With the conventional model-based control, we can ensure the learning-based control law provides closed-loop stability for the trajectory tracking control of the overall system, and increase the sample efficiency of the deep reinforcement learning. With reinforcement learning, we can directly learn a control law to compensate for modeling uncertainties. In the proposed control, a nominal system is employed for the design of a baseline control law using a conventional control approach. The nominal system also defines the desired performance for uncertain autonomous vehicles to follow. In comparison with traditional deep reinforcement learning methods, our proposed learning-based control can provide stability guarantees and better sample efficiency. We demonstrate the performance of the new algorithm via extensive simulation results.**

I. INTRODUCTION

Autonomous surface vehicles (ASVs) have been employed in many applications, such as resource exploration [1], shipping [2], environmental monitoring [3], and many more. Successful launch of ASVs in real life requires accurate tracking control along a desired trajectory [4]. However, it is challenging for accurate tracking control of ASVs, as they are subject to uncertain nonlinear hydrodynamics [5]. Hence, tracking control of highly uncertain ASVs has received extensive research attention [6], [7].

Control algorithms for uncertain systems including ASVs mainly lie in four categories: 1) robust control that is the "worst-case" design for bounded uncertainties and disturbances [8]; 2) adaptive control that adapts to system uncertainties with parameter estimations [9], [10]; 3) disturbance observer (DO)-based control that compensates uncertainties and disturbances in terms of the observation technique [11]; and 4) reinforcement learning (RL) that learns a control law from data samples [12]. The first three algorithms follow a model-based control approach, while the last one is data-driven. Model-based control can ensure closed-loop stability, but a system model is indispensable. In robust control, uncertainties and disturbances are assumed to be bounded with known boundaries [13]. As a consequence, robust control will lead to conservative high-gain control laws that reduce the control performance (i.e., overshoot, settling time, and stability margins) [14]. Adaptive control can handle varying uncertainties with unknown boundaries, but system uncertainties are assumed to be linearly parameterized with known structure and unknown constant parameters [15], [16]. DO-based control can adapt to both uncertainties and disturbances with unknown structures and without assuming systems to be persistently excited [11]. However, the frequency information of uncertainty and disturbance signals is necessary for choosing proper gains for the DO-based control, otherwise, it is highly possible to end up with a high-gain control law [17]. Besides, the DO-based control can only address matched uncertainties and disturbances that act on systems through the control channel [18]. In general, comprehensive modeling and analysis of systems are essential for all model-based methods.

In comparison with model-based methods, RL is capable of learning a control law from data samples using much less model information [19], [20]. Hence, it is more promising in controlling systems subject to massive uncertainties and disturbances as ASVs [21], given the sufficiency and good quality of collected data. Nevertheless, it is challenging for RL to ensure closed-loop stability, though some research attempts have been made [22]. Model-based RL with stability guarantee has been studied by introducing a Lyapunov constraint into the objective function [23]. However, the model-based RL with stability guarantee requires an admissible control law—a control law that provides asymptotic stability—for the initialization. Both the Lyapunov candidate function and system dynamics are assumed to be Lipschitz continuous with known Lipschitz constants for the construction of the Lyapunov constraint. It is challenging to find the Lipschitz constant of an uncertain system. Hence, the introduced Lyapunov constraint function is restrictive, as it is established based on the worst-case consideration [23].

With the consideration of the merits and limitations of existing RL methods, we propose a novel learning-based control algorithm for uncertain ASVs by combining a conventional control method with deep RL in this paper. Such a design method has several advantages over both conventional model-based methods and pure deep RL methods. First of all, in relation to the "model-free" feature of deep RL, we can learn a control law directly to compensate for uncertainties without exploiting their structures, boundaries, or frequencies. In the new design, uncertainties are not necessarily matched, as deep RL seeks a control law like direct adaptive control [24]. Second, the overall control law can ensure closed-loop stability without the usage of the Lipschitz constant of the uncertain system. Lastly, the proposed design is more sample efficient than RL that learns from scratch—that is, fewer data samples are needed for training. In RL, a system learns from

[1]Department of Maritime and Transport Technology, Delft University of Technology, Delft, the Netherlands `Qingrui.Zhang@tudelft.nl; V.Reppa@tudelft.nl`

[2]Department of Cognitive Robotics, Delft University of Technology, Delft, the Netherlands `Wei.Pan@tudelft.nl`

978-1-7281-7447-1/20/$31.00 ©2020 IEEE

mistakes so a lot of trials and errors are demanded. Fortunately, in our proposed design, the baseline control can help to exclude unnecessary mistakes, so it provides a good starting point for the RL training. The main contributions of this paper include: 1) a novel model reference RL algorithm is developed for the tracking control of ASVs; 2) The performance of the proposed algorithm is analyzed.

The rest of the paper is organized as follows. In Section II, we present the ASV dynamics. Section III provides basic concepts on model reference control and RL. The detailed design of the model reference RL is presented in Section IV. Algorithm analysis is given in Section V. In Section VI, numerical simulation results are provided. Conclusions are given in Section VII.

II. System dynamics

In most scenarios, we are interested in controlling the horizontal motioins of ASVs in this paper [25], thus ignoring the vertical, rolling, and pitching motions by default. Let x and y be the horizontal position coordinates of an ASV in the inertial frame and ψ the heading angle. As shown in Figure 1, let u and v be the linear velocities in surge (x-axis) and sway (y-axis), respectively. Let r be the heading angular rate. The general 3-DOF nonlinear dynamics of an ASV are

$$\begin{cases} \dot{\boldsymbol{\eta}} = \boldsymbol{R}(\boldsymbol{\eta})\boldsymbol{\nu} \\ \boldsymbol{M}\dot{\boldsymbol{\nu}} + \left(\boldsymbol{C}(\boldsymbol{\nu}) + \boldsymbol{D}(\boldsymbol{\nu})\right)\boldsymbol{\nu} + \boldsymbol{G}(\boldsymbol{\nu}) = \boldsymbol{\tau} \end{cases} \tag{1}$$

where $\boldsymbol{\eta} = [x, y, \psi]^T \in \mathbb{R}^3$ is a generalized coordinate vector, $\boldsymbol{\nu} = [u, v, r]^T \in \mathbb{R}^3$ is the speed vector, $\boldsymbol{M}$ is the inertia matrix, $\boldsymbol{C}(\boldsymbol{\nu})$ denotes the matrix of Coriolis and centripetal terms, $\boldsymbol{D}(\boldsymbol{\nu})$ is the damping matrix, $\boldsymbol{\tau} \in \mathbb{R}^3$ represents the control forces and moments, $\boldsymbol{G}(\boldsymbol{\nu}) = [\boldsymbol{g}_1(\boldsymbol{\nu}), \boldsymbol{g}_2(\boldsymbol{\nu}), \boldsymbol{g}_3(\boldsymbol{\nu})]^T \in \mathbb{R}^3$ denotes unmodeled dynamics due to gravitational and buoyancy forces and moments [5], and $\boldsymbol{R} = \begin{bmatrix} \cos\psi, & -\sin\psi, & 0 \\ \sin\psi, & \cos\psi, & 0 \\ 0, & 0, & 1 \end{bmatrix}$ is a rotation matrix. The inertia matrix $\boldsymbol{M} = \boldsymbol{M}^T > 0$ is

$$\boldsymbol{M} = [M_{ij}] = \begin{bmatrix} M_{11} & 0 & 0 \\ 0 & M_{22} & M_{23} \\ 0 & M_{32} & M_{33} \end{bmatrix} \tag{2}$$

where $M_{11} = m - X_{\dot{u}}$, $M_{22} = m - Y_{\dot{v}}$, $M_{33} = I_z - N_{\dot{r}}$, and $M_{32} = M_{23} = mx_g - Y_{\dot{r}}$. The matrix $\boldsymbol{C}(\boldsymbol{\nu}) = -\boldsymbol{C}^T(\boldsymbol{\nu})$ is

$$\boldsymbol{C} = [C_{ij}] = \begin{bmatrix} 0 & 0 & C_{13}(\boldsymbol{\nu}) \\ 0 & 0 & C_{23}(\boldsymbol{\nu}) \\ -C_{13}(\boldsymbol{\nu}) & -C_{23}(\boldsymbol{\nu}) & 0 \end{bmatrix} \tag{3}$$

where $C_{13}(\boldsymbol{\nu}) = -M_{22}v - M_{23}r$, $C_{23}(\boldsymbol{\nu}) = -M_{11}u$. The damping matrix $\boldsymbol{D}(\boldsymbol{\nu})$ is

$$\boldsymbol{D}(\boldsymbol{\nu}) = [D_{ij}] = \begin{bmatrix} D_{11}(\boldsymbol{\nu}) & 0 & 0 \\ 0 & D_{22}(\boldsymbol{\nu}) & D_{23}(\boldsymbol{\nu}) \\ 0 & D_{32}(\boldsymbol{\nu}) & D_{33}(\boldsymbol{\nu}) \end{bmatrix} \tag{4}$$

where $D_{11}(\boldsymbol{\nu}) = -X_u - X_{|u|u}|u| - X_{uuu}u^2$, $D_{22}(\boldsymbol{\nu}) = -Y_v - Y_{|v|v}|v| - Y_{|r|v}|r|$, $D_{23}(\boldsymbol{\nu}) = -Y_r - Y_{|v|r}|v| - Y_{|r|r}|r|$, $D_{32}(\boldsymbol{\nu}) = -N_v - N_{|v|v}|v| - N_{|r|v}|r|$, $D_{33}(\boldsymbol{\nu}) = -N_r - N_{|v|r}|v| - N_{|r|r}|r|$, and $X_{(\cdot)}$, $Y_{(\cdot)}$, and $N_{(\cdot)}$ are hydrodynamic coefficients [5]. Accurate numerical models of the nonlinear dynamics (1) are rarely available. Major uncertainty sources come from $\boldsymbol{M}$, $\boldsymbol{C}(\boldsymbol{\nu})$, and $\boldsymbol{D}(\boldsymbol{\nu})$ due to hydrodynamics, and $\boldsymbol{G}(\boldsymbol{\nu})$ due to gravitational and buoyancy forces and moments.

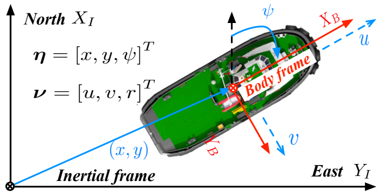


Fig. 1: Coordinate systems of an ASV

III. Model-Reference Reinforcement Learning

A. Model-reference control scheme

Let $\boldsymbol{x} = \left[\boldsymbol{\eta}^T, \boldsymbol{\nu}^T\right]^T$ and $\boldsymbol{u} = \boldsymbol{\tau}$, so (1) can be rewritten as

$$\dot{\boldsymbol{x}} = \begin{bmatrix} 0 & \boldsymbol{R}(\boldsymbol{\eta}) \\ 0 & \boldsymbol{A}(\boldsymbol{\nu}) \end{bmatrix}\boldsymbol{x} + \begin{bmatrix} 0 \\ \boldsymbol{B} \end{bmatrix}\boldsymbol{u} + \begin{bmatrix} 0 \\ \boldsymbol{M}^{-1}\boldsymbol{G}(\boldsymbol{\nu}) \end{bmatrix} \tag{5}$$

where $\boldsymbol{A}(\boldsymbol{\nu}) = \boldsymbol{M}^{-1}\left(\boldsymbol{C}(\boldsymbol{\nu}) + \boldsymbol{D}(\boldsymbol{\nu})\right)$, and $\boldsymbol{B} = \boldsymbol{M}^{-1}$. Assume an accurate model (5) is not available, but it is possible to get a nominal model expressed as

$$\dot{\boldsymbol{x}}_m = \begin{bmatrix} 0 & \boldsymbol{R}(\boldsymbol{\eta}) \\ 0 & \boldsymbol{A}_m \end{bmatrix}\boldsymbol{x}_m + \begin{bmatrix} 0 \\ \boldsymbol{B}_m \end{bmatrix}\boldsymbol{u}_m \tag{6}$$

where $\boldsymbol{A}_m$ and $\boldsymbol{B}_m$ are the known system matrices. Assume that there exists a control law $\boldsymbol{u}_m$ allowing the states of the nominal system (6) to converge to a reference signal $\boldsymbol{x}_r$, i.e., $\|\boldsymbol{x}_m - \boldsymbol{x}_r\|_2 \to 0$ as $t \to \infty$.

The objective is to design a control law allowing the state of (5) to track state trajectories of the nominal model (6). As shown in Figure 2, the overall control law for the ASV system (5) has the following expression.

$$\boldsymbol{u} = \boldsymbol{u}_b + \boldsymbol{u}_l \tag{7}$$

where $\boldsymbol{u}_b$ is a baseline controller, and $\boldsymbol{u}_l$ is a control policy from the deep RL module shown in Figure 2. The baseline control $\boldsymbol{u}_b$ is employed to ensure some basic performance, (i.e., local stability), while $\boldsymbol{u}_l$ is introduced to compensate for all system uncertainties. The baseline control $\boldsymbol{u}_b$ can be designed based on any existing model-based method based on the nominal model (6). Hence, we ignore the design process of $\boldsymbol{u}_b$, and mainly focus on the development of $\boldsymbol{u}_l$ using RL.

B. Reinforcement learning

In RL, system dynamics are characterized using a Markov decision process denoted by a tuple $\mathcal{MDP} := \left\langle \mathcal{S}, \mathcal{U}, \mathcal{P}, R, \gamma \right\rangle$, where $\mathcal{S}$ is the state space, $\mathcal{U}$ specifies the action/input space, $\mathcal{P} : \mathcal{S} \times \mathcal{U} \times \mathcal{S} \to \mathbb{R}$ defines a transition probability, $R : \mathcal{S} \times \mathcal{U} \to \mathbb{R}$ is a reward function, and $\gamma \in [0, 1]$ is a discount factor. Note that the state vector $\boldsymbol{s}$ contains all available signals affecting the RL control $\boldsymbol{u}_l$. In this paper, such signals include $\boldsymbol{x}$, $\boldsymbol{x}_m$, $\boldsymbol{x}_r$, and $\boldsymbol{u}_b$, where $\boldsymbol{x}_m$ performs like a target state for system (5) and $\boldsymbol{u}_b$ is a function of $\boldsymbol{x}$ and $\boldsymbol{x}_r$. Hence, we choose $\boldsymbol{s} = \{\boldsymbol{x}_m, \boldsymbol{x}, \boldsymbol{u}_b\}$. Assume that we can sample input and state data from system

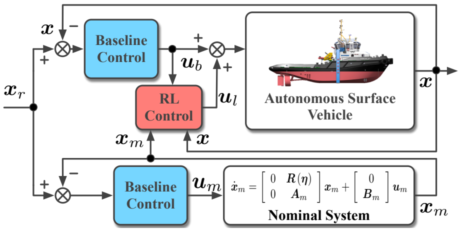


Fig. 2: Model-reference reinforcement learning control

(5) at discrete time steps. Let $\boldsymbol{x}_t$, $\boldsymbol{u}_{b,t}$, and $\boldsymbol{u}_{l,t}$ be the ASV state, the baseline control action, and the control action from the RL at a time step t, respectively. The state signal $\boldsymbol{s}$ at the time step t is, therefore, denoted by $\boldsymbol{s}_t = \{\boldsymbol{x}_{m,t}, \boldsymbol{x}_t, \boldsymbol{u}_{b,t}\}$.

For each state $\boldsymbol{s}_t$, we define a value function $V_{\boldsymbol{\pi}}(\boldsymbol{s}_t)$ as an expected accumulated return described as

$$V_{\boldsymbol{\pi}} = \sum_{t}^{\infty} \sum_{\boldsymbol{u}_{l,t}} \boldsymbol{\pi}(\boldsymbol{u}_{l,t}|\boldsymbol{s}_t) \sum_{\boldsymbol{s}_{t+1}} \mathcal{P}_{t+1|t} \big(R_t + \gamma V_{\boldsymbol{\pi}}(\boldsymbol{s}_{t+1})\big) \quad (8)$$

where $R_t = R(\boldsymbol{s}_t, \boldsymbol{u}_{l,t})$, $\mathcal{P}_{t+1|t} = \mathcal{P}(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{u}_{l,t})$, and $\boldsymbol{\pi}(\boldsymbol{u}_l|\boldsymbol{s})$ is a policy in RL, denoting the probability of choosing an action $\boldsymbol{u}_l \in \mathcal{U}$ at a state $\boldsymbol{s} \in \mathcal{S}$ [19]. In this paper, the reward function R_t is defined as

$$R_t = -(\boldsymbol{x}_t - \boldsymbol{x}_{m,t})^T \boldsymbol{H}_1 (\boldsymbol{x}_t - \boldsymbol{x}_{m,t}) - \boldsymbol{u}_{l,t}^T \boldsymbol{H}_2 \boldsymbol{u}_{l,t} \quad (9)$$

where $\boldsymbol{H}_1 \geq 0$ and $\boldsymbol{H}_2 > 0$ are positive definite matrices. Accordingly, the action-value function (a.k.a., Q-function) is

$$Q_{\boldsymbol{\pi}}(\boldsymbol{s}_t, \boldsymbol{u}_{l,t}) = R_t + \gamma \sum_{\boldsymbol{s}_{t+1}} \mathcal{P}_{t+1|t} V_{\boldsymbol{\pi}}(\boldsymbol{s}_{t+1}) \quad (10)$$

The objective of the RL in this paper is to find an optimal policy $\boldsymbol{\pi}^*$ to maximize the Q-function $Q_{\boldsymbol{\pi}}(\boldsymbol{s}_t, \ \boldsymbol{u}_{l,t})$, namely

$$\boldsymbol{\pi}^* = \arg\max_{\boldsymbol{\pi}} Q_{\boldsymbol{\pi}}(\boldsymbol{s}_t, \boldsymbol{u}_{l,t}) \quad \forall \boldsymbol{s}_t \in \mathcal{S} \quad (11)$$

IV. Deep Reinforcement Learning Control Design

The deep RL control in this paper is developed based on the soft actor-critic (SAC) algorithm which provides both sample efficient learning and convergence [26]. In SAC, an entropy term is added to the objective function in (11) to regulate the exploration performance at the training stage. The objective of (11) is thus rewritten as

$$\begin{aligned} \boldsymbol{\pi}^* = &\arg\max_{\boldsymbol{\pi}} \big(R_t + \gamma \mathbb{E}_{\boldsymbol{s}_{t+1}} [V_{\boldsymbol{\pi}}(\boldsymbol{s}_{t+1}) \\ &+ \alpha \mathcal{H}(\boldsymbol{\pi}(\boldsymbol{u}_{l,t+1}|\boldsymbol{s}_{t+1}))]\big) \end{aligned} \quad (12)$$

where $\mathbb{E}_{\boldsymbol{s}_{t+1}}[\cdot] = \sum_{\boldsymbol{s}_{t+1}} \mathcal{P}_{t+1|t}[\cdot]$ is an expectation operator, $\mathcal{H}(\boldsymbol{\pi}(\boldsymbol{u}_{l,t}|\boldsymbol{s}_t)) = -\sum_{\boldsymbol{u}_{l,t}} \boldsymbol{\pi}(\boldsymbol{u}_{l,t}|\boldsymbol{s}_t) \ln(\boldsymbol{\pi}(\boldsymbol{u}_{l,t}|\boldsymbol{s}_t)) = -\mathbb{E}_{\boldsymbol{\pi}}[\ln(\boldsymbol{\pi}(\boldsymbol{u}_{l,t}|\boldsymbol{s}_t))]$ is the entropy of the policy, and α is a temperature parameter.

Training of SAC repeatedly executes policy evaluation and policy improvement. In the policy evaluation, the Q-function is computed by applying a Bellman operation $Q_{\boldsymbol{\pi}}(\boldsymbol{s}_t, \boldsymbol{u}_{l,t}) = \mathcal{T}^{\pi} Q_{\boldsymbol{\pi}}(\boldsymbol{s}_t, \boldsymbol{u}_{l,t})$ where

$$\begin{aligned} \mathcal{T}^{\pi} Q_{\boldsymbol{\pi}}(\boldsymbol{s}_t, \boldsymbol{u}_{l,t}) = R_t + \gamma \mathbb{E}_{\boldsymbol{s}_{t+1}} \{ &\mathbb{E}_{\boldsymbol{\pi}} [Q_{\boldsymbol{\pi}}(\boldsymbol{s}_{t+1}, \boldsymbol{u}_{l,t+1}) \\ &- \alpha \ln(\boldsymbol{\pi}(\boldsymbol{u}_{l,t+1}|\boldsymbol{s}_{t+1}))]\} \end{aligned} \quad (13)$$

In the policy improvement, the policy is updated by

$$\boldsymbol{\pi}_{new} = \arg\min_{\boldsymbol{\pi}'} \mathscr{D}_{KL} \left(\boldsymbol{\pi}'(\cdot|\boldsymbol{s}_t) \middle\| Z^{\boldsymbol{\pi}_{old}} e^{Q^{\boldsymbol{\pi}_{old}}(\boldsymbol{s}_t, \cdot)} \right) \quad (14)$$

where $\boldsymbol{\pi}_{old}$ denotes the policy from the last update, $Q^{\boldsymbol{\pi}_{old}}$ is the Q-value of $\boldsymbol{\pi}_{old}$. $\mathscr{D}_{KL}$ denotes the Kullback-Leibler (KL) divergence, and $Z^{\boldsymbol{\pi}_{old}}$ is a normalization factor. Via mathematical manipulations, the objective for the policy improvement is transformed into

$$\boldsymbol{\pi}^* = \arg\min_{\boldsymbol{\pi}} \mathbb{E}_{\boldsymbol{\pi}} \big[\alpha \ln(\boldsymbol{\pi}) - Q(\boldsymbol{s}_t, \boldsymbol{u}_{l,t})\big] \quad (15)$$

More details on how (15) is obtained can be found in [26]. Both the policy $\boldsymbol{\pi}$ and value function $Q_{\boldsymbol{\pi}}(\boldsymbol{s}_t, \boldsymbol{u}_{l,t})$ will be parameterized using fully connected multiple layer perceptrons (MLP) with 'ReLU' nonlinearities as the activation functions. The Q-function is parameterized using θ and denoted by $Q_\theta(\boldsymbol{s}_t, \boldsymbol{u}_{l,t})$. The parameterized policy is denoted by $\boldsymbol{\pi}_\phi(\boldsymbol{u}_{l,t}|\boldsymbol{s}_t)$, where ϕ is the parameter set to be trained. Note that both θ and ϕ are a set of parameters whose dimensions are determined by the deep neural network setup. Details on the design of $Q_\theta(\boldsymbol{s}_t, \boldsymbol{u}_{l,t})$ and $\boldsymbol{\pi}_\phi(\boldsymbol{u}_{l,t}|\boldsymbol{s}_t)$ can be found in [27]. The deep neural network for Q_θ is called critic, while the one for $\boldsymbol{\pi}_\phi$ is called actor [27].

In this paper, $\boldsymbol{\pi}_\phi(\boldsymbol{u}_{l,\phi}|\boldsymbol{s}) = \mathcal{N}(\boldsymbol{u}_{l,\phi}(\boldsymbol{s}), \boldsymbol{\sigma})$ with $\mathcal{N}(\cdot,\cdot)$ denoting a Gaussian distribution. Hence, at training, we have

$$\bar{\boldsymbol{u}}_{l,\phi} = \boldsymbol{u}_{l,\phi} + \boldsymbol{\sigma}_\phi \odot \boldsymbol{\xi} \quad (16)$$

where $\boldsymbol{u}_{l,\phi}$ is the control law to be implemented after the training, $\boldsymbol{\xi} \sim \mathcal{N}(0, \boldsymbol{I})$ is the exploration noise, $\boldsymbol{\sigma}_\phi$ denotes the standard deviation of the exploration noise, and "$\odot$" is the Hadamard product. Note that the exploration noise $\boldsymbol{\xi}$ is only applied to the training stage. Once the training is done, $\boldsymbol{u}_l$ in Figure 2 will be approximated by $\boldsymbol{u}_{l,\phi}$ in the implementation.

The whole training process will be offline. At each time step $t+1$, we collect data samples, such as an input $\boldsymbol{u}_{l,t}$, a state $\boldsymbol{s}_t$, a reward R_t, and a current state $\boldsymbol{s}_{t+1}$. Those data samples are stored as a tuple $(\boldsymbol{s}_t, \boldsymbol{u}_{l,t}, R_t, \boldsymbol{s}_{t+1})$ at a replay memory $\mathcal{D}$ [28]. At the policy evaluation or improvement, we randomly sample a batch of historical data $\mathcal{B}$, from the replay memory $\mathcal{D}$ for the training of the parameters θ and ϕ. Starting the training, we apply the baseline control policy $\boldsymbol{u}_b$ to an ASV system to collect the initial data $\mathcal{D}_0$ as shown in Algorithm 1. The initial data set $\mathcal{D}_0$ is used for the initial fitting of Q-value functions. When the initialization is over, we execute both $\boldsymbol{u}_b$ and the latest updated RL policy $\boldsymbol{\pi}_\phi(\boldsymbol{u}_{l,t}|\boldsymbol{s}_t)$ to run the ASV system. The entire algorithm is summarized in Algorithm 1, where ι_Q and ι_π are positive learning rates (scalars), and $\kappa > 0$ is a constant scalar.

At the policy evaluation step, the parameters θ are trained to minimize the following Bellman residual.

$$J_Q(\theta) = \mathbb{E}_{(\boldsymbol{s}_t, \boldsymbol{u}_{l,t}) \sim \mathcal{D}} \Big[\frac{1}{2} (Q_\theta(\boldsymbol{s}_t, \boldsymbol{u}_{l,t}) - Y_{target})^2\Big] \quad (17)$$

where $(\boldsymbol{s}_t, \boldsymbol{u}_{l,t}) \sim \mathcal{D}$ denotes that data samples $(\boldsymbol{s}_t, \boldsymbol{u}_{l,t})$ are randomly picked from the replay memory $\mathcal{D}$, and

$$Y_{target} = R_t + \gamma \mathbb{E}_{\boldsymbol{s}_{t+1}} \big[\mathbb{E}_{\boldsymbol{\pi}} [Q_{\bar{\theta}}(\boldsymbol{s}_{t+1}, \boldsymbol{u}_{l,t+1}) - \alpha \ln(\boldsymbol{\pi}_\phi)]\big]$$

Algorithm 1 Model reference reinforcement learning control

```
Initialize parameters $\theta$ for the critic $Q_\theta$ and $\phi$ for the actor in (16). Set $\bar{\theta}$ by $\bar{\theta} \leftarrow \theta$
Get data set $\mathcal{D}_0$ by running $\boldsymbol{u}_b$ on (5) with $\boldsymbol{u}_l = \mathbf{0}$
Turn off the exploration and train an initial critic parameter $\theta^0$ using $\mathcal{D}_0$ according to (17)
Initialize the replay memory $\mathcal{D} \leftarrow \mathcal{D}_0$
Assign initial values to the critic parameter $\theta \leftarrow \theta^0$ and its target $\bar{\theta} \leftarrow \theta^0$
repeat
    for each data collection step do
        Choose an action $\boldsymbol{u}_{l,t}$ according to $\boldsymbol{\pi}_\phi(\boldsymbol{u}_{l,t}|\boldsymbol{s}_t)$
        Collect $\boldsymbol{s}_{t+1} = \{\boldsymbol{x}_{t+1}, \boldsymbol{x}_{m,t+1}, \boldsymbol{u}_{b,t+1}\}$
        $\mathcal{D} \leftarrow \mathcal{D} \bigcup \{\boldsymbol{s}_t, \boldsymbol{u}_{l,t}, R(\boldsymbol{s}_t, \boldsymbol{u}_{l,t}), \boldsymbol{s}_{t+1}\}$
    end for
    for each gradient update step do
        Sample a batch of data $\mathcal{B}$ from $\mathcal{D}$
        $\theta \leftarrow \theta - \iota_Q \nabla_\theta J_Q(\theta)$
        $\phi \leftarrow \phi - \iota_\pi \nabla_\phi J_{\boldsymbol{\pi}}(\phi)$
        $\bar{\theta} \leftarrow \kappa\theta + (1-\kappa)\bar{\theta}$
    end for
until convergence (i.e. $J_Q(\theta) <$ a small threshold)
```

where $\bar{\theta}$ is the target parameter to be updated slowly. Applying a stochastic gradient descent technique (ADAM [29] in this paper) to (17) on a data batch $\mathcal{B}$ with a fixed size, we obtain

$$\nabla_\theta J_Q(\theta) = \sum \frac{\nabla_\theta Q_\theta \big(Q_\theta(\boldsymbol{s}_t, \boldsymbol{u}_{l,t}) - Y_{target}\big)}{|\mathcal{B}|}$$

where $|\mathcal{B}|$ is the batch size.

At the policy improvement step, the objective function defined in (15) is represented using data samples from the replay memory $\mathcal{D}$ as

$$J_\pi(\phi) = \mathbb{E}_{(\boldsymbol{s}_t, \boldsymbol{u}_{l,t}) \sim \mathcal{D}} \Big(\alpha \ln(\boldsymbol{\pi}_\phi) - Q_\theta(\boldsymbol{s}_t, \boldsymbol{u}_{l,\phi})\Big) \tag{18}$$

V. Performance analysis

For the convergence analysis, Algorithm 1 is recapped as a policy iteration (PI) technique [27]. The following two lemmas are provided without proofs [26].

Lemma 1 (Policy evaluation): Let $\mathcal{T}^\pi$ be the Bellman backup operator under a fixed policy $\boldsymbol{\pi}$ and $Q^{k+1}(\boldsymbol{s}, \boldsymbol{u}_l) = \mathcal{T}^\pi Q^k(\boldsymbol{s}, \boldsymbol{u}_l)$. The sequence $Q^{k+1}(\boldsymbol{s}, \boldsymbol{u}_l)$ will converge to the soft Q-function $Q^{\boldsymbol{\pi}}$ of the policy $\boldsymbol{\pi}$ as $k \to \infty$.

Lemma 2 (Policy improvement): Let $\boldsymbol{\pi}_{old}$ be an old policy and $\boldsymbol{\pi}_{new}$ be a new policy obtained according to (14). There exists $Q^{\boldsymbol{\pi}_{new}}(\boldsymbol{s}, \boldsymbol{u}_l) \geq Q^{\boldsymbol{\pi}_{old}}(\boldsymbol{s}, \boldsymbol{u}_l)$ $\forall \boldsymbol{s} \in \mathcal{S}$ and $\forall \boldsymbol{u} \in \mathcal{U}$.

In terms of (1) and (2), Theorem 1 is presented to show the convergence of the SAC algorithm.

*Theorem 1 (**Convergence**):* If one repeatedly applies the policy evaluation and policy improvement steps to any control policy $\boldsymbol{\pi}$, the control policy $\boldsymbol{\pi}$ will converge to an optimal policy $\boldsymbol{\pi}^*$ such that $Q^{\boldsymbol{\pi}^*}(\boldsymbol{s}, \boldsymbol{u}_l) \geq Q^{\boldsymbol{\pi}}(\boldsymbol{s}, \boldsymbol{u}_l)$ $\forall \boldsymbol{\pi} \in \Pi$, $\forall \boldsymbol{s} \in \mathcal{S}$, and $\forall \boldsymbol{u} \in \mathcal{U}$, where Π denotes a policy set.

Proof: Let $\boldsymbol{\pi}_i$ be the policy obtained from the i-th policy improvement with $i = 0, 1, \ldots, \infty$. According to Lemma 2, one has $Q^{\boldsymbol{\pi}_i}(\boldsymbol{s}, \boldsymbol{u}_l) \geq Q^{\boldsymbol{\pi}_{i-1}}(\boldsymbol{s}, \boldsymbol{u}_l)$, so $Q^{\boldsymbol{\pi}_i}(\boldsymbol{s}, \boldsymbol{u}_l)$ is monotonically non-decreasing with respect to the policy iteration step i. In addition, $Q^{\boldsymbol{\pi}_i}(\boldsymbol{s}, \boldsymbol{u}_l)$ is upper bounded according to the definition of the reward given in (9), so $Q^{\boldsymbol{\pi}_i}(\boldsymbol{s}, \boldsymbol{u}_l)$ will converge to an upper limit $Q^{\boldsymbol{\pi}^*}(\boldsymbol{s}, \boldsymbol{u}_l)$ with $Q^{\boldsymbol{\pi}^*}(\boldsymbol{s}, \boldsymbol{u}_l) \geq Q^{\boldsymbol{\pi}}(\boldsymbol{s}, \boldsymbol{u}_l)$ $\forall \boldsymbol{\pi} \in \Pi$, $\forall \boldsymbol{s} \in \mathcal{S}$, and $\forall \boldsymbol{u}_l \in \mathcal{U}$. ■

Next, we will show the closed-loop stability of the ASV with the overall control law around its desired state trajectory defined by $\boldsymbol{x}_m$. The control objective is to ensure $\|\boldsymbol{x}_t - \boldsymbol{x}_{m,t}\|_2 \to 0$ as $t \to \infty$, so we define $\boldsymbol{e}_t = \boldsymbol{x}_t - \boldsymbol{x}_{m,t}$ as the tracking error. The following assumption is made for the baseline control developed using the nominal system (6).

Assumption 1: The baseline control law $\boldsymbol{u}_b$ can ensure that the overall uncertain ASV system is stable around its desired state trajectory $\boldsymbol{x}_m$– that is, there exists a positive definite function $\mathbb{V}(\boldsymbol{e}_t)$ associate with $\boldsymbol{u}$ such that $\mathbb{V}(\boldsymbol{e}_{t+1}) - \mathbb{V}(\boldsymbol{e}_t) \leq 0$, $\forall \boldsymbol{e}_t \in \Omega$, where Ω defines the state domain.

With Assumption 1, the Q-value function $Q(\boldsymbol{s}, \boldsymbol{u}_l)$ is ensured to be finite at the beginning of the training. In the stability analysis, we will ignore the entropy term $\mathcal{H}(\boldsymbol{\pi})$, as it is only introduced to regulate the exploration magnitude at the learning stage. Now, Theorem 2 is presented to show the closed-loop stability of the ASV system (5).

*Theorem 2 (**Stability**):* Suppose Assumption 1 holds. The overall control law $\boldsymbol{u}^i = \boldsymbol{u}_b + \boldsymbol{u}_l^i$ canstabilize the ASV system (5) around the desired state defined by $\boldsymbol{x}_m$, where $\boldsymbol{u}_l^i$ are the RL control law from i-th iteration, and $i = 0$, 1, 2, ... ∞.

Proof: In our proposed algorithm, we start the training/learning using the baseline control law $\boldsymbol{u}_b$. According to Lemma 1, we are able to obtain the corresponding Q value function for the baseline control law $\boldsymbol{u}_b$. At the beginning, the Q value function be $Q^0\big(\boldsymbol{s}, \boldsymbol{u}_l^0\big)$ where $\boldsymbol{u}_l^0$ is a function of $\boldsymbol{s}$. The learning-based control law $\boldsymbol{u}_l^0$ is initialized to be around 0. With Assumption 1, the reward function R_t^0 is ensure to be bounded. The following equation exists for $Q^0(\boldsymbol{s}, \boldsymbol{u}_l)$.

$$Q^0\left(\boldsymbol{s}_t, \boldsymbol{u}_{l,t}^0\right) = R_t^0 + \gamma Q^0\left(\boldsymbol{s}_{t+1}, \boldsymbol{u}_{l,t+1}^0\right) \tag{19}$$

According to the definitions of the reward function in (9) and Q^0 in (19), we can choose the continuous function of $\boldsymbol{e}_t$ as $\mathbb{V}^0(\boldsymbol{e}_t) = -\gamma^t Q^0\big(\boldsymbol{s}_t, \boldsymbol{u}_{l,t}^0\big)$. Hence, one has

$$\begin{aligned}\mathbb{V}^0(\boldsymbol{e}_t) &= -\gamma^t R_t^0 - \gamma^{t+1} Q^0(\boldsymbol{s}_{t+1}, \boldsymbol{u}_{l,t+1}) \\ &= -\gamma^t R_t^0 + \mathbb{V}^0(\boldsymbol{e}_{t+1})\end{aligned} \tag{20}$$

Hence, $\mathbb{V}^0(\boldsymbol{e}_{t+1}) - \mathbb{V}^0(\boldsymbol{e}_t) = \gamma^t R_t^0 \leq 0$, $\forall \boldsymbol{e}_t \in \Omega$. According to LaSalle's invariance principle [30], the tracking error $\boldsymbol{e}_t$ will converge to a set where $\mathbb{V}^0(\boldsymbol{e}_{t+1}) - \mathbb{V}^0(\boldsymbol{e}_t) = 0$. There are two scenarios for $\mathbb{V}^0(\boldsymbol{e}_{t+1}) - \mathbb{V}^0(\boldsymbol{e}_t) = 0$. One is $\boldsymbol{e}_t = \mathbf{0}$, which corresponds to asymptotic stability. The other is the case where R_t^0 is finite as $t \to \infty$, which is input-to-state stability. If Assumption 1 holds, R_t^0 is guaranteed to be bounded by the baseline control law $\boldsymbol{u}_b$. Hence, the tracking error $\boldsymbol{e}_t$ is ensured to be at least ultimately bounded by the overall control law $\boldsymbol{u}_t^0 = \boldsymbol{u}_{b,t} + \boldsymbol{u}_{l,t}^0$.

In the policy improvement, the control law is updated by

$$\boldsymbol{u}_l^1 = arg \min_{\boldsymbol{u}_l} \left(-R_t + \gamma \mathbb{V}^0(\boldsymbol{e}_{t+1})\right) \tag{21}$$

For any nonlinear system $\boldsymbol{e}_{t+1} = \boldsymbol{f}(\boldsymbol{e}_t) + \boldsymbol{g}(\boldsymbol{e}_t)\boldsymbol{u}_t$, a necessary condition for the existence of (21) is

$$\boldsymbol{u}_l^1 = -\frac{\gamma}{2}\boldsymbol{H}_2^{-1}\boldsymbol{g}^T(\boldsymbol{e}_t)\frac{\partial \mathbb{V}^0(\boldsymbol{e}_{t+1})}{\partial \boldsymbol{e}_{t+1}} \tag{22}$$

Substituting (22) back into (9) yields $R_t^1 = -\boldsymbol{e}_t^T\boldsymbol{H}_1\boldsymbol{e}_t - \frac{\gamma^2}{4}\left(\frac{\partial \mathbb{V}^0(\boldsymbol{e}_{t+1})}{\partial \boldsymbol{e}_{t+1}}\right)^T \boldsymbol{g}(\boldsymbol{e}_t)\boldsymbol{H}_2^{-1}\boldsymbol{g}^T(\boldsymbol{e}_t)\frac{\partial \mathbb{V}^0(\boldsymbol{e}_{t+1})}{\partial \boldsymbol{e}_{t+1}}$. According to the optimality of principle of dynamic programming, one has that $R_t^0 \leq R_t^1 \leq 0$. Since R_t^0 is bounded, R_t^1 is bounded as well. After the policy evaluation on $\boldsymbol{u}_l^1$, one can obtain

$$\mathbb{V}^1(\boldsymbol{e}_t) = -\gamma^t R_t^1 + \mathbb{V}^1(\boldsymbol{e}_{t+1}) \tag{23}$$

Hence, one can easily obtain that the new control law $\boldsymbol{u}^1 = \boldsymbol{u}_b + \boldsymbol{u}_l^1$ can ensure the tracking error ($\boldsymbol{e}_t$ is ultimately bounded—-that is, the ASV system (5 is stabilized by $\boldsymbol{u}^1$. In terms of $\mathbb{V}^1(\boldsymbol{s_t})$, (20), and (21), we can show that $\boldsymbol{u}^2$ also stabilizes the ASV system (5). Repeating the aforementioned analysis for all $i = 1, 2, \ldots,$ we can prove that all $\boldsymbol{u}^i$ can stabilize the ASV system (5), if Assumption 1 holds. ■

VI. Simulation

In this section, the proposed learning-based control algorithm is implemented to the trajectory tracking control of a supply ship model presented in [25]. Model parameters can be found in [27]. The unmodeled dynamics in the simulations are given by $g_1 = 0.279uv^2 + 0.342v^2r$, $g_2 = 0.912u^2v$, and $g_3 = 0.156ur^2 + 0.278urv^3$, respectively. The baseline control $\boldsymbol{u}_b$ is designed based on a nominal model in (24) in terms of the backstepping control method [30].

$$\boldsymbol{M}_m\dot{\boldsymbol{\nu}}_m = \boldsymbol{\tau} - \boldsymbol{D}_m\boldsymbol{\nu}_m \tag{24}$$

where $\boldsymbol{M}_m = diag\{M_{11}, M_{22}, M_{33}\}$. $\boldsymbol{D}_m = diag\{-X_v, -Y_v, -N_r\}$. The reference signal is assumed to be produced by the following motion planner.

$$\dot{\boldsymbol{\eta}}_r = \boldsymbol{R}(\boldsymbol{\eta}_r)\boldsymbol{\nu}_r \quad \dot{\boldsymbol{\nu}}_r = \boldsymbol{a}_r \tag{25}$$

where $\boldsymbol{\eta}_r = [x_r, y_r, \psi_r]^T$ is the generalized reference position vector, $\boldsymbol{\nu}_r = [u_r, 0, r_r]^T$ is the generalized reference velocity vector, and $\boldsymbol{a}_r = [\dot{u}_r, 0, \dot{r}_r]^T$. In the simulation, the initial position vector $\boldsymbol{\eta}_r(0)$ is chosen to be $\boldsymbol{\eta}_r(0) = \left[0, 0, \frac{\pi}{4}\right]^T$, and we set $u_r(0) = 0.4\ m/s$ and $r_r(0) = 0\ rad/s$. The reference acceleration $\dot{u}_r$ and angular rates are chosen to be

$$\dot{u}_r = \begin{cases} 0.005\ m/s^2 & \text{if } t < 20\ s \\ 0\ m/s^2 & \text{otherwise} \end{cases} \tag{26}$$

$$\dot{r}_r = \begin{cases} \frac{\pi}{600}\ rad/s^2 & \text{if } 25\ s \leq t < 50\ s \\ 0\ rad/s^2 & \text{otherwise} \end{cases} \tag{27}$$

At the training stage, we run the ASV system for 100 s, and the repeat the training processes for 1000 times (i.e., 1000 episodes). Detailed training setup can be found in [27]. Figure 3 shows the learning curves of the proposed algorithm (red) and the RL algorithm without baseline control (blue). The learning curves demonstrate that both of the two algorithms will converge in terms of the long term returns. However, our proposed algorithm results in a larger return (red) in comparison with the RL without baseline control (blue). Hence, the introduction of the baseline control helps to increase the sample efficiency significantly, as the proposed algorithm (blue) converges faster to a higher return value.

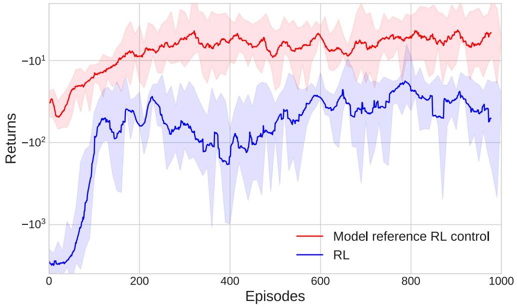
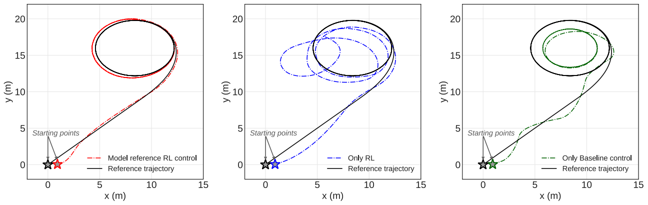


Fig. 3: Learning curves of two RL algorithms at training (One episode is a training trial, and 1000 time steps per episode)



(a) Model reference RL (b) Only deep RL (c) Only baseline control

Fig. 4: Trajectory tracking results of the three algorithms

At the evaluation stage, we run the ASV system for 200 s to demonstrate whether the control law can ensure stable trajectory tracking. Note that we run the ASV for 100 s at training. The trajectory tracking performance of the three algorithms (our proposed algorithm, the baseline control $\boldsymbol{u}_0$, and only RL control) is shown in Figures 4. As observed from Figure 4.b, the control law learned merely using deep RL fails to ensure stable tracking performance. It implies that only deep RL cannot ensure the closed-loop stability. In addition, the baseline control itself fails to achieve acceptable tracking performance mainly due to the existence of system uncertainties. By combining the baseline control and deep RL, the trajectory tracking performance is improved dramatically, and the closed-loop stability is also ensured. The position tracking errors are summarized in Figure 5 and 6. Figure 7 shows the absolute distance errors used to compare the tracking accuracy of the three algorithms. The introduction of the deep RL increases the tracking performance substantially. More simulation results can be found in [27].

VII. Conclusions

In this paper, we presented a novel learning-based algorithm for the control of uncertain ASV systems by combining a model-based control method with deep RL. With the model-based control, we ensured the overall closed-loop stability of the learning-based control and increase the sample efficiency of the deep RL. With deep RL, we learned to compensate for the model uncertainties, and thus increased the trajectory tracking performance. In future works, we will extend the

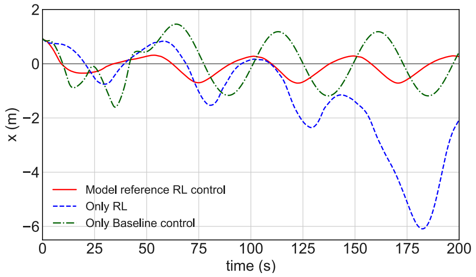


Fig. 5: Position tracking errors (e_x)

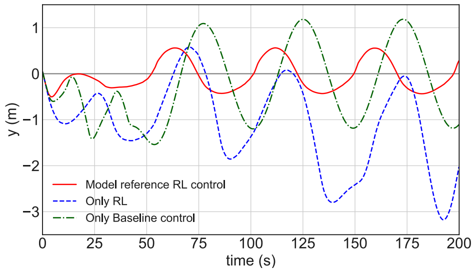


Fig. 6: Position tracking errors (e_y)

results with the consideration of environmental disturbances. The theoretical results will be further verified via experiments instead of simulations. The sample efficiency of the proposed algorithm will also be analyzed.

References

[1] J. Majohr and T. Buch, *Advances in Unmanned Marine Vehicles*. Institution of Engineering and Technology, 2006, ch. Modelling, simulation and control of an autonomous surface marine vehicle for surveying applications Measuring Dolphin MESSIN.

[2] O. Levander, "Autonomous ships on the high seas," *IEEE Spectrum*, vol. 54, no. 2, pp. 26–31, 2017.

[3] D. O.B.Jones, A. R.Gates, V. A.I.Huvenne, A. B.Phillips, and B. J.Bett, "Autonomous marine environmental monitoring: Application in decommissioned oil fields," *Science of The Total Environment*, vol. 668, no. 10, pp. 835– 853, 2019.

[4] C. R. Sonnenburg and C. A. Woolsey, "Integrated optimal formation control of multiple unmanned aerial vehicles," *Journal of Field Robotics*, vol. 3, no. 30, pp. 371–398, May/Jun. 2013.

[5] T. I. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley & Sons, Inc., 2011.

[6] R. A. Soltan, H. Ashrafiuon, and K. R. Muske, "State-dependent trajectory planning and tracking control of unmanned surface vessels," in *Proceedings of 2009 American Control Conference*. St. Louis, MO, USA: IEEE, Jun. 2009.

[7] N. Wang, J.-C. Sun, M. J. Er, and Y.-C. Liu, "A novel extreme learning control framework of unmanned surface vehicles," *IEEE Transactions on Cybernetics*, vol. 46, no. 5, pp. 1106–1117, May 2016.

[8] R. Yu, Q. Zhu, G. Xia, and Z. Liu, "Sliding mode tracking control of an underactuated surface vessel," *IET Control Theory & Applications*, vol. 6, no. 3, pp. 461– 466, 2012.

[9] K. Do and J. Pan, "Global robust adaptive path following of underactuated ships," *Automatica*, vol. 42, no. 10, pp. 1713–1722, Oct. 2006.

[10] G. Chowdhary, T. Yucelen, M. Mühlegg, and E. N. Johnson, "Concurrent learning adaptive control of linear systems with exponentially convergent bounds," *International Journal of Adaptive Control and Signal Processing*, vol. 27, no. 4, pp. 280–301, May 2013.

[11] Q. Zhang and H. H. Liu, "UDE-based robust command filtered backstepping control for close formation flight," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 11, pp. 8818–8827, Nov. 2018.

[12] W. Shi, S. Song, C. Wu, and C. L. P. Chen, "Multi pseudo q-learning-based deterministic policy gradient for tracking control of autonomous underwater vehicles," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 12, pp. 3534–3546, Dec. 2019.

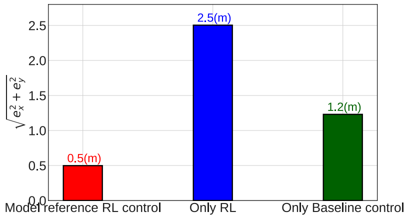


Fig. 7: Mean absolute distance errors ($\sqrt{e_x^2+e_y^2}$)

[13] T. Shen and K. Tamura, "Robust H_∞ control of uncertain nonlinear system via state feedback," *IEEE Transactions on Automatic Control*, vol. 40, no. 4, pp. 766–768, Apr. 1995.

[14] X. Liu, H. Su, B. Yao, and J. Chu, "Adaptive robust control of a class of uncertain nonlinear systems with unknown sinusoidal disturbances," in *Proceedings of 2008 47th IEEE Conference on Decision and Control*. Cancun, Mexico, USA: IEEE, Dec. 2008.

[15] P. A. Ioannou and J. Sun, *Robust Adaptive Control*. Prentice-Hall, Inc., 1996.

[16] Q. Zhang and H. H. Liu, "Aerodynamic model-based robust adaptive control for close formation flight," *Aerospace Science and Technology*, vol. 79, pp. 5–16, 2018.

[17] B. Zhu, Q. Zhang, and H. H. Liu, "Design and experimental evaluation of robust motion synchronization control for multivehicle system without velocity measurements," *International Journal of Robust and Nonlinear Control*, vol. 28, no. 7, pp. 5437–5463, 2018.

[18] X. Zhang, H. Li, B. Zhu, and Y. Zhu, "Improved ude and lso for a class of uncertain second-order nonlinear systems without velocity measurementss," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 7, pp. 4076–4092, Jul. 2020.

[19] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introductions*, 2nd ed. The MIT Press, 2018.

[20] B. Kiumarsi, K. G. Vamvoudakis, H. Modares, and F. L. Lewis, "Optimal and autonomous control using reinforcement learning: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 6, pp. 2042–2062, Jun. 2018.

[21] J. Woo, C. Yu, and N. Kim, "Deep reinforcement learning-based controller for path following of an unmanned surface vehicle," *Ocean Engineering*, vol. 183, no. 1, pp. 155–166, Dec. 2019.

[22] M. Han, Y. Tian, L. Zhang, J. Wang, and W. Pan, "H_∞ model-free reinforcement learning with robust stability guarantee," in *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*, Vancouver, Canada, Dec. 2019.

[23] F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause, "Safe model-based reinforcement learning with stability guarantees," in *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, CA, USA, Dec. 2017.

[24] R. Sutton, A. Barto, and R. Williams, "Reinforcement learning is direct adaptive optimal control," *IEEE Control Systems Magazine*, vol. 12, no. 2, pp. 19–22, Apr. 1992.

[25] R. Skjetne, T. I. Fossen, and P. V. Kokotović, "Adaptive maneuvering, with experiments, for a model ship in a marine control laboratory," *Mathematics of Operations Research*, vol. 41, pp. 289–298, 2005.

[26] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proceedings of the 35th International Conference on Machine Learning (ICML 2018)*, vol. 80, Stockholmsmässan, Stockholm Sweden, Jul. 2018, pp. 1861–1870.

[27] Q. Zhang, W. Pan, and V. Reppa, "Model-reference reinforcement learning control of autonomous surface vehicles with uncertainties," *arXiv preprint arXiv:2003.13839*, 2020.

[28] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, C. B. Stig Petersen, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the 3rd International Conference for Learning Representations (ICLR 2015)*, San Diego, USA, May 2015.

[30] H. K. Khalil, *Nonlinear Systems*, 3rd ed. Prentice Hall, 2001.