Multi-AUV Pursuit-Evasion Game in the Internet of Underwater Things: An Efficient Training Framework via Offline Reinforcement Learning

Jingzehua Xu, *Student Member, IEEE*, Zekai Zhang, Jingjing Wang, *Senior Member, IEEE*,
Zhu Han, *Fellow, IEEE*, and Yong Ren, *Senior Member, IEEE*

***Abstract*—In this article, we investigate the pursuit-evasion game of multiple autonomous underwater vehicles (AUVs) in a complex ocean environment. The pursuer AUVs need to optimize their trajectories to avoid obstacles and dangerous vortex regions in the environment in order to pursue the escaper AUV. Both the pursuer and escaper can sense each other with limited detection capabilities for further pursuit or escape. As the underwater pursuit-evasion (UPE) game is a high-dimensional NP-hard problem, we innovatively transform it into a finite-horizon Markov game process and propose a decentralized training and decentralized execution efficient training framework based on the offline reinforcement learning. During the training process, we propose multiagent independent soft actor–critic to facilitate policy improvement and generate the offline data set, and propose multiagent independent decision transformer for model training in the UPE game. Extensive simulations demonstrate the scalability and generalization ability of our proposed training framework, which can achieve excellent performance in the UPE games under different conditions and environments with only a few AUVs participating in policy improvement to generate the high-quality offline data set.**

***Index Terms*—Autonomous underwater vehicle (AUV), decision transformer (DT), finite-horizon Markov game process (FMGP), offline reinforcement learning (ORL), pursuit-evasion game.**

I. Introduction

DUE TO its strong mobility, wide range of activities, and strong concealment, autonomous underwater vehicles (AUVs) have been widely used in typical Internet of Underwater Things (IoUT) tasks, such as resource survey [1], information collection [2], [3], real-time search and rescue [4], and underwater target search is the key to efficient implementation of these tasks. Considering the high maneuverability and unknown escape strategy of the target and the limited perception ability of AUV, it is urgent to study the underwater pursuit-evasion (UPE) game between the multiple AUV and the escape target [5]. In addition, the complex and unknown underwater environment makes the decision-making process of this game full of challenges.

The multiagent pursuit-evasion game requires pursuers and escapers to continuously adjust their behaviors and policies according to their own observable environmental information, simultaneously or successively, in order to maximize their own interests and finally pursue the escaping target successfully. In the process, the target is highly maneuvering, with unknown motion state and certain escape policy [4]. There is a special imbalance in the research of the multiagent pursuit-evasion game, which mainly focuses on the unmanned aerial vehicles and unmanned ground vehicles. The research of the UPE game is still in the initial stage due to the complex underwater environment and the difficulty in describing the underwater dynamic game process. Most UPE game decision schemes are derived from the migration of space-based and roadbed robots. Mainstream pursuit-evasion methods include the heuristic algorithms [6], the neural network [7], the game theory [8], etc. Although they have achieved good results in specific tasks, such model-based methods require a large amount of prior information, and the control parameters need to be constantly adjusted according to the changes of the environment, and consequently the performance of the methods deteriorates significantly. Due to their limited scalability and adaptability, they are not suitable for highly dynamic multiagent UPE game tasks.

Manuscript received 22 April 2024; revised 28 May 2024; accepted 16 June 2024. Date of publication 19 June 2024; date of current version 25 September 2024. This work of Jingjing Wang was supported in part by the National Natural Science Foundation of China under Grant 62071268 and Grant 62222101; in part by the Young Elite Scientist Sponsorship Program by the China Association for Science and Technology under Grant 2020QNRC001; and in part by the Fundamental Research Funds for the Central Universities. This work of Zhu Han was supported in part by the NSF, U.S. Department of Transportation, Toyota under Grant CNS-2107216, Grant CNS-2128368, Grant CMMI-2222810, and Grant ECCS-2302469; and in part by the Amazon and Japan Science and Technology Agency (JST) Adopting Sustainable Partnerships for Innovative Research Ecosystem (ASPIRE) under Grant JPMJAP2326. This work of Yong Ren was supported in part by the National Natural Science Foundation of China under Grant 62127801; and in part by the National Key Research and Development Program of China under Grant 2020YFD0901000. *(Jingzehua Xu and Zekai Zhang contributed equally to this work.) (Corresponding author: Jingjing Wang.)*

Jingzehua Xu and Zekai Zhang are with the Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen 518055, China (e-mail: xjzh23@mails.tsinghua.edu.cn; zhangzej21@mails.tsinghua.edu.cn).

Jingjing Wang is with the School of Cyber Science and Technology, Beihang University, Beijing 100191, China, and also with the Hangzhou Innovation Institute, Beihang University, Hangzhou 310051, China (e-mail: drwangjj@buaa.edu.cn).

Zhu Han is with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX 77004 USA, and also with the Department of Computer Science and Engineering, Kyung Hee University, Seoul 446-701, South Korea (e-mail: hanzhu22@gmail.com).

Yong Ren is with the Department of Electronic Engineering, Tsinghua University, Beijing 100084, China (e-mail: reny@tsinghua.edu.cn).

Digital Object Identifier 10.1109/JIOT.2024.3416616

2327-4662 © 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Zhejiang University. Downloaded on October 01,2024 at 09:11:10 UTC from IEEE Xplore. Restrictions apply.

Thanks to the excellent feature expression ability and interactive feedback ability with the environment, deep reinforcement learning (DRL) has become a feasible solution to the UPE games of various multiagent unmanned systems. In the literature, Zhang et al. [9] extended the multiagent deep deterministic policy gradient (MADDPG) algorithm to build an efficient target prediction network and studied the pursuit-evasion game of multiquadcopter in the environment of obstacles. Wang et al. proposed a scalable DRL method scalable-MADDPG for cooperative target invasion in the multiple unmanned surface vehicles (USVs) system. This method can change the scale of the multi-USV system at any time to help the multi-USV system adapt to the complex marine environment [10]. Wei et al. [4] proposed a differential game-based DRL method to study the differential game problem of underwater target hunting. Unfortunately, the traditional DRL methods are not suitable for the dynamic and unstable multiagent UPE game environments, which can be effectively mitigated by using multiagent reinforcement learning (MARL), but the existing MARL methods are often based on the centralized training and decentralized execution (CTDE), which makes the algorithm less scalable [11]. In addition, due to the complexity of the task and the lack of prior information, agents need to interact frequently with the environment to collect a large amount of data, which requires a lot of time and computing resources [12]. Therefore, it is very important to improve the sampling rate of RL. The emergence of offline reinforcement learning (ORL) effectively solves this problem, which uses the preexisting offline data set for training, and improves the training efficiency while saving computing resources and time [13]. However, there are three shortcomings (bootstrap, off-policy, and approximation) in traditional ORL due to the introduction of time difference (TD), which affect the stability and performance of training [14].

Based on the above analysis, it can be seen that the existing UPE game works do not carefully consider the impact of complex ocean environment on AUV movement, and the hypothesis of game confrontation between the pursuers and escapers is relatively ideal. At the same time, the existing training strategies have some problems, such as inapplicability to dynamic multiagent environment, poor scalability, low sampling efficiency, unstable training, and unsatisfactory performance. Therefore, this work analyses the environment of the UPE game, and proposes an efficient training framework, named multiagent independent soft actor–critic (MAISAC) and the multiagent independent decision transformer (MAIDT)-based ORL training strategy (MMOTS). The main contributions are as follows.

1) To the best of our knowledge, this is the first work that investigates the multi-AUV UPE game in complex environments with obstacles and ocean currents, aiming to plan AUV's trajectory to safely and reliably pursue targets. Considering that the trajectory optimization is a high-dimensional NP-hard problem, we define the multi-AUV UPE game as a the finite-horizon Markov game process (FMGP) to solve it.
2) We propose an efficient training framework MMOTS based on the decentralized training and decentralized execution (DTDE), in which proposed MAISAC is first used to realize policy improvement and make the offline data set, and then ORL is used to train the model and apply it to the multi-AUV UPE game. To overcome the training instability and low efficiency of ORL, we introduce decision transformer (DT) into ORL and extend it to MAIDT.
3) Extensive experiments demonstrate the superiority and adaptability of our proposed MMOTS, allowing it to accommodate UPE games under various conditions and environments. Compared to the other state of the art algorithms, MAISAC exhibits significantly higher training efficiency, while MAIDT shows enhanced generalization capability and stability.

The remainder of this article is organized as follows. In Sections II and III, the related work and system model are given in detail. The constrained optimization problem and algorithm design are then introduced in detail in Sections IV and V, respectively. In Section VI, the numerical simulation experiments are carried out to verify the effectiveness of MMOTS, followed by the conclusion in Section VII.

II. Related Work

The study of the UPE game is of great significance to underwater detection, underwater precision guidance, underwater target tracking, and other application fields [1], [2]. AUV is used to search and capture targets because of its intelligence, flexibility, and controllability [15]. In [16], [17], and [18], researchers use sensors, such as multibeam forward-looking sonar and underwater cameras mounted on the AUV to determine the location of the target and predict its trajectory based on this, and then dispatch the AUV to keep track of the target. However, this target tracking method that relies on the sensing ability of the AUV, has a low success rate due to the unknown escape strategy of the target and the limited detection range due to the harsh underwater environment [19]. Therefore, the focus of current research is to track or hunt targets by means of multi-AUV coordination to make up for the defects of limited AUV sensing range and low search efficiency. For example, Zhao et al. [20] proposed a minimum rigid graph-based tracking strategy based on collaboration between the AUVs to improve target tracking accuracy. Zhang et al. [8] proposed a contract network-based allocation framework to achieve multi-AUV formation target hunting. Unfortunately, the highly dynamic and complex underwater game environment presents a challenge to the multi-AUV system control methods [21].

Presently, the common methods of multi-AUV pursuit-evasion game mainly include neural networks [5], control models [22], game theory [23], etc. In [24], topologically organized biological neural networks based on the grid diagrams are used to characterize dynamic game environments, guiding AUVs to search for the targets and avoid obstacles in a 3-D underwater environment. In [6], particle swarm optimization algorithm is applied to real-time rescue assignment of multi-AUV systems. In [5], fractional order recurrent neural network (RNN) is constructed to optimize anti-game maneuvering strategies based on the Karush–Kuhn–Tucker

Authorized licensed use limited to: Zhejiang University. Downloaded on October 01,2024 at 09:11:10 UTC from IEEE Xplore. Restrictions apply.

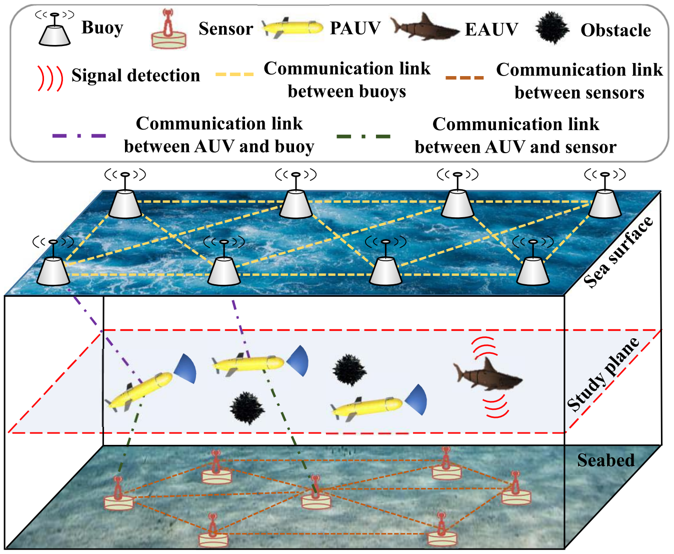

Buoy
Sensor
PAUV
EAUV
Obstacle
Signal detection
Communication link between buoys
Communication link between sensors
Communication link between AUV and buoy
Communication link between AUV and sensor
Sea surface
Study plane
Seabed

Fig. 1. Illustration of the IoUT-assisted UPE game scenario.

(KKT) optimality conditions for the anti-game problems in dynamic target scenarios. In [25] and [26], based on the game theory, the interactive process between the multi-AUV systems and targets was analysed and hunting strategies were derived. However, in practical applications, the above model-based multi-AUV control strategies require a large amount of prior environmental information, and need to adjust the control parameters in real time according to the changes in the environment, which is not suitable for the highly dynamic underwater game environment.

The multi-AUV control strategy based on the MARL has excellent performance in the UPE game. Wei et al. [27] proposed a MARL strategy for multi-AUV underwater target hunting task based on differential game. Xia et al. [28] proposed an end-to-end MARL scheme for multiagent target tracking, which improved the success rate of target tracking. However, the above MARL-based solutions have the problems of unstable training and low sampling efficiency, so they cannot train an efficient hunting model. This kind of solution does not have scalability because it uses the CTDE framework. On the other hand, the game scenarios considered by the above schemes are relatively simple and ideal, and does not take into account complex ocean environment and the escape strategy of the target, which lacks practicability.

To sum up, different from the previous work, we study the multi-AUV UPE game scenario, not only considering the interference of obstacles and vortexes on AUV movement but also considering the escape strategy of the target, and propose an innovative ORL-based training framework named MMOTS, which adopts the DTDE mode with strong applicability and scalability.

III. System Model

We consider the IoUT-assisted UPE game scenario as shown in Fig. 1, which includes an IoUT network composed of buoys on the sea surface and sensor nodes laid on the seabed, and N AUVs participating in the UPE game. The buoys can communicate with the shore-based stations or satellites through the electromagnetic signals to obtain their location and time [20], while the sensor nodes utilize acoustic communication to directly interact with the buoys for self-localization and clock synchronization [29]. The set of AUVs is defined as $\mathcal{N}=\{1, 2, \ldots, N\}(N > 1)$, where the type of the first N-1 AUVs is the pursuer AUV (PAUV), and the Nth AUV is identified as the escape AUV (EAUV). PAUVs and EAUV perform the UPE game on the study plane with a fixed depth h, the position of PAUV i at time t can be denoted as $\boldsymbol{P}_i(t) = [x_i(t), y_i(t), h]^T$, and the position of EAUV can be denoted as $\boldsymbol{P}_T(t) = [x_T(t), y_T(t), h]^T$. During the whole game process, the motion state of the EAUV is unpredictable, yet its position can be captured by the sensor nodes or buoys and reported to PAUVs via the acoustic communication methods [8]. In addition, there are obstacles and vortexes in the environment, and PAUVs and EAUV need to avoid these hazards as much as possible. Each AUV is equipped with the sonar for underwater detection and a horizontal acoustic Doppler current profiler (HADCP) [30] for current velocity measurement, which manufacturers an accuracy of 1% of measured velocity ±5 mm/s and can be used to measure water velocity on a horizontal line hundreds of meters ahead [31]. With the HADCP, AUVs can sense the location of surrounding vortex centers and avoid them through the trajectory scheduling. The AUV dynamics model, underwater detection model, and ocean current model are given in detail in Sections III-A–III-C, respectively.

A. AUV Dynamics Model

Since, PAUVs pursue the EAUV in the horizontal plane, without the loss of generality, their dynamic models can be expressed by the three-degree of freedom underdrive model, in which AUV i has the body reference frame $\boldsymbol{\nu}_i = [\nu_{i,x}(t), \nu_{i,y}(t), \omega_i]^T$, and the world reference frame $\boldsymbol{\eta}_i = [x_i(t), y_i(t), \theta_i]^T$, where $\nu_{i,x}(t)$, $\nu_{i,y}(t)$, ω_i and θ_i are the surge velocity, sway velocity, yaw angular velocity, and yaw angle, respectively. According to the Fossen's motion equation [27], the dynamic model of AUV i considering hydrodynamics and hydrostatic forces is

$$\dot{\boldsymbol{\eta}}_i = \boldsymbol{J}(\boldsymbol{\eta}_i)\boldsymbol{\nu}_i \tag{1}$$

$$\boldsymbol{M}_U\dot{\boldsymbol{\nu}}_i + \boldsymbol{C}_U(\boldsymbol{\nu}_i)\boldsymbol{\nu}_i + \boldsymbol{D}_U(\boldsymbol{\nu}_i)\boldsymbol{\nu}_i + \boldsymbol{G}_U(\boldsymbol{\eta}_i) = \boldsymbol{\tau}_i \tag{2}$$

where $\boldsymbol{M}_U$ represents the inertia matrix, including the additional mass of AUV, while $\boldsymbol{C}_U$ is the Coriois centripetal force matrix of AUV. Moreover, $\boldsymbol{D}_U$ is the damping matrix describing the viscous fluid force and $\boldsymbol{G}_U$ is the composite matrix of gravity and buoyancy. $\boldsymbol{\tau}_i$ denotes the control input of AUV i. We define $\boldsymbol{J}(\boldsymbol{\eta}_i)$ as the transformation matrix, and we have

$$\boldsymbol{J}(\boldsymbol{\eta}_i) = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 \\ \sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3}$$

For practical applications, the above kinematic and dynamic equations need to be separated at any time

$$\boldsymbol{\eta}_{t+1} = \boldsymbol{\eta}_t + \Delta T \cdot \boldsymbol{J}(\boldsymbol{\eta}_t)\boldsymbol{\nu}_t \tag{4}$$

$$\boldsymbol{\nu}_{t+1} = \boldsymbol{\nu}_t + \Delta T \cdot \boldsymbol{M}_U^{-1} F(\boldsymbol{\eta}_t, \boldsymbol{\nu}_t) \tag{5}$$

Authorized licensed use limited to: Zhejiang University. Downloaded on October 01,2024 at 09:11:10 UTC from IEEE Xplore. Restrictions apply.

where ΔT is the time interval and $F(\boldsymbol{\eta}_t, \boldsymbol{v}_t)$ can be given by $F(\boldsymbol{\eta}_t, \boldsymbol{v}_t) = \boldsymbol{\tau}_t - \boldsymbol{C}_U(\boldsymbol{v}_t)\boldsymbol{v}_t - \boldsymbol{D}_U(\boldsymbol{v}_t)\boldsymbol{v}_t - \boldsymbol{G}_U(\boldsymbol{\eta}_t)$.

B. Underwater Detection Model

AUVs use the sonar to detect the environment in a limited range, such as detecting surrounding obstacles and keeping track of the target. This process can be uniformly modeled using the active sonar equation [32] as

$$EM = SL - 2TL(f, d) + TS - NL(f) + DI - DT. \tag{6}$$

The unit of all the parameters in (6) is dB, where *SL*, *TL*, *TS*, *NL*, and *DI* represent the emission sound strength, transmission loss, target strength related to the target reflection area, environmental noise level, and directionality index, respectively. Furthermore, *DT* and *EM* represent the detection threshold and echo margin of active sonar, respectively. Furthermore, *TL* is related to the detection radius d and the center acoustic frequency f, namely

$$TL = 20\log(d) + d \times a(f) \times 10^{-3} \tag{7a}$$
$$a(f) = 0.11\frac{f^2}{1+f^2} + 44\frac{f^2}{4100+f^2} + 2.75 \times 10^{-4}f^2 + 0.003 \tag{7b}$$

where $a(f)$ is the attenuation coefficient of sound wave in water. Environmental noise *NL* composed of turbulence noise *Nt*, shipping noise *Ns*, wind noise *Nw*, thermal noise *Nth*, and the environmental noise [33] can be represented as

$$NL(f) = Nt(f) + Ns(f) + Nw(f) + Nth(f). \tag{8}$$

The noise components in (8) are

$$\begin{cases} 10\log N_t(f) = 17 - 30\log f \\ 10\log N_s(f) = 30 + 20s + \log\big(f^{26}/(f+0.03)^{60}\big) \\ 10\log N_w(f) = 50 + 7.5\omega^{1/2} + 20\log\big(f/(f+0.4)^2\big) \\ 10\log N_{th}(f) = -15 + 20\log f \end{cases} \tag{9}$$

where s and w represent the shipping activity factor and wind speed (m/s), respectively, $s \in [0, 1]$. Since, *EM* and d show a monotonically decreasing relationship, when the frequency f is given, the maximum detection radius r_c of the AUV is

$$r_c = \arg\max_d\{EM(d) \geq 0\}. \tag{10}$$

C. Ocean Current Model

The motion of AUV in the UPE game needs to take into account the influence of ocean turbulent environment because the intensity of ocean current on the horizontal plane is much greater than that on the vertical plane under the influence of the Earth rotation, the ocean flow can be approximately 2-D [34]. We refer to the work in [30] to use 2-D Navier–Stokes equations [32] to model the ocean turbulent environment as

$$\frac{\partial \varpi}{\partial t} + (\boldsymbol{V}_c\nabla)\varpi = \zeta\Delta\varpi \tag{11}$$

where $\boldsymbol{V}_c = (V_x, V_y)$ is the velocity of the current field, ϖ and ζ are the vorticity of the current and the viscosity of the fluid, and ∇ and Δ are the gradient operators and the Laplacian operators, respectively. To simplify the Navier–Stokes equation, the numerical equation of the ocean current model is represented by the superposition of several viscous vortex functions, which are described as follows:

$$V_x(\boldsymbol{P}_i(t)) = -\Gamma \cdot \frac{y - y_0}{2\pi\|\boldsymbol{P}_i(t) - \boldsymbol{P}_0\|_2^2} \cdot \left(1 - e^{-\frac{\|\boldsymbol{P}_i(t)-\boldsymbol{P}_0\|_2^2}{\delta^2}}\right) \tag{12a}$$
$$V_y(\boldsymbol{P}_i(t)) = -\Gamma \cdot \frac{x - x_0}{2\pi\|\boldsymbol{P}_i(t) - \boldsymbol{P}_0\|_2^2} \cdot \left(1 - e^{-\frac{\|\boldsymbol{P}_i(t)-\boldsymbol{P}_0\|_2^2}{\delta^2}}\right) \tag{12b}$$

$$\varpi(\boldsymbol{P}_i(t)) = \frac{\Gamma}{\pi\delta^2} \cdot e^{-\frac{\|\boldsymbol{P}_i(t)-\boldsymbol{P}_0\|_2^2}{\delta^2}} \tag{13}$$

where $\boldsymbol{P}_i(t)$ and $\boldsymbol{P}_0$ are the current position of the AUV i and the coordinate vector of the Lamb vortex center, $V_x(\boldsymbol{P}_i(t))$ and $V_y(\boldsymbol{P}_i(t))$ are the velocities of the ocean current on the X and Y axis perceived by AUV i at the position $\boldsymbol{P}_i(t)$ at time t, respectively. While δ and Γ are the radius and intensity of the vortex, respectively.

IV. Problem Formulation

In this section, we first model the UPE game between the PAUVs and the EAUV as an FMGP. Then, the constrained optimization problem is presented and the reward function is designed in detail.

A. Finite-Horizon Markov Game Process Modeling

In the UPE game, the goal is to train PAUVs to navigate in an underwater environment with currents and obstacles to find and then pursue the EAUV. The game belongs to the multiagent task category, considering the interaction between the multiagent system and the underwater environment, this game can be modeled as an FMGP, and the process has a termination state, allowing AUVs to end the current event. Similar to the Markov decision process [28], FMGP's main elements include the state space $\boldsymbol{S}_i$, action space $\boldsymbol{\mathcal{A}}_i$, and reward function $\mathcal{R}_i$.

1) State Space $\boldsymbol{S}_i$: In FMGP, the states of each AUV are observable, and the ith AUV's state $\boldsymbol{s}_i(t)$ of belongs to the state space $\boldsymbol{S}_i$, which can be expressed as

$$\boldsymbol{s}_i(t) = \big[\boldsymbol{l}_i(t), l_{(p-e)_i}(t), \min(\boldsymbol{l}_i(t)), V_x(\boldsymbol{P}_i(t)), V_y(\boldsymbol{P}_i(t)) \\ \alpha_{o_i}(t), \alpha_{(p-e)_i}(t), D_i(t)\big] \tag{14}$$

where $\boldsymbol{l}_i(t)$ denotes the ambient distance detected by the sonar, while $l_{(p-e)_i}(t)$ represents the distance between the AUV i and the EAUV if the AUV i is a PAUV, or the distance between the nearest PAUV and the EAUV if the AUV i is an EAUV. Then, $\alpha_{o_i}(t)$ and $\alpha_{(p-e)_i}(t)$ are the orientation angle of AUV i, and the yaw angle from the PAUV i to the EAUV or the yaw angle from the EAUV to the nearest PAUV, respectively. While the termination state is represented by $D_i(t) \in \{True, \ False\}$, indicating whether the episode has concluded or not.

Authorized licensed use limited to: Zhejiang University. Downloaded on October 01,2024 at 09:11:10 UTC from IEEE Xplore. Restrictions apply.

2) Action Space $\boldsymbol{\mathcal{A}}_i$: In FMGP, at each step, the AUV i needs to determine its next action based on the feedback from the environment. According to the AUV dynamics model in the previous section, the action space $\boldsymbol{\mathcal{A}}_i$ and action $\mathfrak{a}_i(t)$ of the AUV i can be expressed as

$$\boldsymbol{\mathcal{A}}_i = [v_{\min}, v_{\max}] \times [\omega_{\min}, \omega_{\max}] \tag{15}$$

$$\mathfrak{a}_i(t) = [\boldsymbol{v}_i(t), \omega_i(t)] \tag{16}$$

where $\|\boldsymbol{v}_i(t)\| = \sqrt{v_{i,x}(t)^2 + v_{i,y}(t)^2} \in [v_{\min}, v_{\max}]$ and $\|\omega_i(t)\| \in [\omega_{\min}, \omega_{\max}]$. The actions chosen by the ith AUV interact with the environment to produce the next state according to the state transition function, and $\boldsymbol{s}_i(t) \times \mathfrak{a}_{1_i} \times \mathfrak{a}_{2i} \times \cdots \times \mathfrak{a}_{N_i} \longmapsto \boldsymbol{s}_i(t+1)$ denotes the transition from the state $\boldsymbol{s}_i(t)$ to the next state $\boldsymbol{s}_i(t+1)$.

3) Reward Function $\mathcal{R}_i$: The agent adjusts its policy according to the reward obtained by the current action, and so it is crucial to design an appropriate reward function. The design of the reward function need to take the engineering practice and the complexity of the UPE game into account, and the specific design is given in the following section.

B. Problem Formulation

In this section, we summarize several engineering constraints to be considered in the UPE game, and formulate a constrained optimization problem, whose goal is to optimize the policy of each AUV (π_{θ_i}) to maximize the total expected reward. The constrained optimization problem can be expressed as

$$\max_{\pi_{\theta_i}} J(\theta_i) = \max_{\pi_{\theta_i}} E\left[\sum_{t'=t}^{T=\infty} \gamma^{t'-t} R_{i,t'}\big(\boldsymbol{s}_i, \pi_{\theta_i}(\boldsymbol{a}_i \mid \boldsymbol{s}_i)\big)\right] \tag{17a}$$

$$\text{s.t. } l_{ij}(t) \geq l_{\min}^{i\leftrightarrow j}, l_{iT}(t) \geq l_{\min}^{i\leftrightarrow T} \quad \forall i, j \in \mathcal{N}, i \neq j \tag{17b}$$

$$v_{\min} \leq \|\boldsymbol{v}_i(t)\| \leq v_{\max}, \omega_{\min} \leq \|\omega_i(t)\| \leq \omega_{\max} \quad \forall i \in \mathcal{N} \tag{17c}$$

where $\gamma \in (0, 1]$ represents the discount factor, while $\pi_{\theta_i}(\mathfrak{a}_i \mid \boldsymbol{s}_i)$ denotes the policy, indicating the probability of choosing action $\mathfrak{a}_i$ in the state $\boldsymbol{s}_i$ for AUV i, and θ_i is the parameters of the policy. Equation (17b) considers collision avoidance between the PAUVs and between each PAUV and EAUV. Considering the size and structure limitations in practice, (17c) restricts the velocity and angular velocity range of each AUV. In addition, based on the assistance of the underwater sensor network, PAUVs have better pursuit ability.

C. Reward Function Design

The design of the reward function needs to consider obstacle avoidance, encouraging PAUVs to approach the EAUV, encouraging the EAUV to escape and approach the target point, and guiding each AUV to avoid dangerous areas near the vortex centers. Therefore, we outline them as follows.

Collision Avoidance: To ensure the safety of PAUVs pursuing the EAUV, and the policy for the EAUV to escape when PAUVs approach the EAUV, it is essential to set a minimum distance between the AUVs as well as between the AUVs and obstacles. Imperatively, we set $l_{\min}^{i\leftrightarrow j}$ as the safe distance imperative to prevent collisions. Based on the above analysis, the reward function $r_{C_i}(t)$ is designed as

$$r_{C_i}(t) = -400\,\text{ceil}\Big(l_{\min}^{i\leftrightarrow j} / \min(\boldsymbol{l}_i(t))\Big) \quad i = 1, \ldots, N \tag{18}$$

where ceil(x) is the binary function, which means that the ceil(x) equals to 1 when $x \geq 1$, and equals to 0 when $x \leq 1$. More intuitively, (18) denotes that when the nearest distance is less than or equal to $l_{\min}^{i\leftrightarrow j}$, the AUV will receive a penalty of 400.

Encourage Pursuit/Evasion: Drawing on the insight that the PAUVs should be guided away from aimlessly wandering during exploration and interaction within the environment, we leverage $r_{E_{1_i}}$ to incentivize each PAUV to actively move toward the EAUV, and encourage the EAUV to get to the target point

$$r_{E_{1_i}}(t) = \begin{cases} 0.25, & l^{i\leftrightarrow T}(t-1) > l^{i\leftrightarrow T}(t) \\ -0.25, & l^{i\leftrightarrow T}(t-1) < l^{i\leftrightarrow T}(t), \quad i = 1, \ldots, N. \end{cases} \tag{19}$$

Meanwhile, in order to maintain the consistent performance of each AUV during the entire process, we define $l_{\max}^{i\leftrightarrow T}$ as the target distance and provide rewards based on the results of each AUV

$$r_{E_{2_i}}(t) = 900\,\text{ceil}\big(l_{\max}^{i\leftrightarrow T} / l^{i\leftrightarrow T}(t)\big) \quad i = 1, \ldots, N \tag{20}$$

where $l^{i\leftrightarrow T}$ is the distance between the AUV i and the EAUV for PAUVs, while the distance between the EAUV and the target point for the EAUV.

Vortex Avoidance: Owing to the presence of vortexes, the current velocity increases as the AUV gets closer to the vortex center. This can lead to deviations from the preplanned route and direction, subsequently impacting the AUV's decision making ability. Fortunately, the AUV is equipped with HADCP to measure the current velocity at its position. Therefore, it is crucial to apply penalties to the AUV based on the measured current velocity

$$r_{T_i}(t) = -400\,\text{ceil}\Big(\|\boldsymbol{V}_c(\boldsymbol{P}_i(t))\| / V_{\max}^{i\leftrightarrow C}\Big) \quad i = 1, \ldots N \tag{21}$$

where $V_{\max}^{i\leftrightarrow C}$ denotes the safe current velocity to help AUVs avoid vortex centers, while $\|\boldsymbol{V}_c(\boldsymbol{P}_i(t))\| = \sqrt{V_x(\boldsymbol{P}_i(t))^2 + V_y(\boldsymbol{P}_i(t))^2}$ represents the value of the current velocity.

To summarize, the total reward $\mathcal{R}_i(t)$ can be represented as follows:

$$\mathcal{R}_i(t) = \delta_C r_{C_i}(t) + \delta_{E_1} r_{E_{1_i}}(t) + \delta_{E_2} r_{E_{2i}}(t) + \delta_T r_{T_i}(t) \tag{22}$$

where δ_C, δ_{E_1}, δ_{E_2}, and δ_T represent the weight coefficients associated with the respective reward functions $r_{C_i}(t)$, $r_{E_{1_i}}(t)$, $r_{E_{2i}}(t)$, and $r_{T_i}(t)$.

V. Algorithm Design

In this section, we first introduce the MMOTS framework for the UPE game, which consists of two main stages: 1) policy improvement and 2) model training. Then, we introduce MAISAC and the MAIDT algorithm utilized in MMOTS in detail.

Authorized licensed use limited to: Zhejiang University. Downloaded on October 01,2024 at 09:11:10 UTC from IEEE Xplore. Restrictions apply.

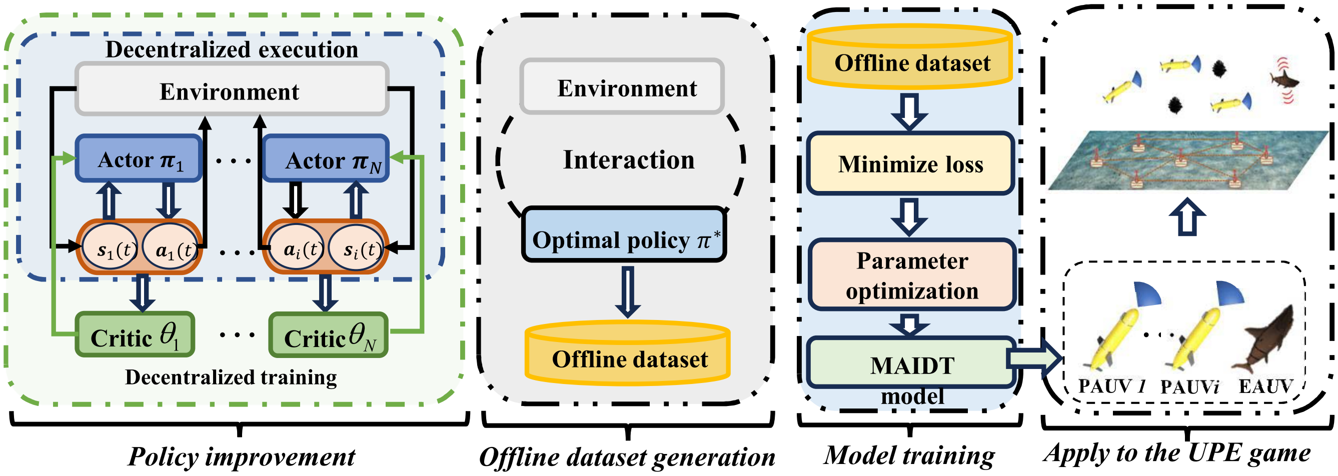


Fig. 2. Framework of MMOTS, which consists of two main stages, policy improvement and model training. In the first stage, we propose and utilize the MAISAC algorithm to train the AUVs for policy improvement, aiming to obtain the expert policy, which is then utilized for data collection to generate an offline data set. Subsequently, in the second stage, MAIDT algorithm in the ORL is utilized for each AUV to learn from the existing offline data set through the model training. Finally, the trained model will be employed in each AUV for the UPE game.

A. Framework of MMOTS for UPE Game

Due to its inability to adapt to the highly dynamic UPE game environment, traditional RL has shortcomings, such as low training efficiency, poor scalability, and complex calculation when solving the constrained optimization problem in the previous section. Therefore, we propose MMOTS, the framework is shown in Fig. 2. First, DTDE is used to extend SAC algorithm to MAISAC for parallel and independent training of AUVs, so that they can perform their own tasks in unknown dynamic environment because the SAC algorithm lacks asymptotic convergence guarantee, the further policy improvement is necessary. We designate the optimal policy solved by (17a) as the expert policy, and all the trajectories under the expert policy are saved as the offline data set, defined as τ_i

$$\tau_i = \left(\boldsymbol{s}_{1_i}, \mathfrak{a}_{1_i}, r_{1_i}, \boldsymbol{s}_{2_i}, \boldsymbol{s}_{2_i}, \mathfrak{a}_{2_i}, r_{2_i}\boldsymbol{s}_{3_i} \ldots, \boldsymbol{s}_{T_i}, \mathfrak{a}_{T_i}, r_{T_i}, \boldsymbol{s}_{T+1_i}\right). \tag{23}$$

Then, the MAIDT model is trained based on the obtained offline data set to achieve policy improvement for each AUV in the UPE game. The trained MAIDT model can be used to predict the real-time action of each AUV based on the initial state and expected total reward. The optimal policy of MAIDT can be obtained according to (24) as

$$\max_{\pi_{\theta_i'}} J'\left(\theta_i'\right) = \max_{\pi_{\theta_i'}} E\left[\sum_{t=1}^{T=\infty} r_{t_i}\right] \tag{24}$$

where $\pi_{\theta_i'}$ denotes policy of AUV i and θ_i' denotes the parameters of the policy, which depends on the model training via MAIDT.

B. Multiagent Independent Soft Actor-Critic

Inspired by SAC [35], MAISAC involves modeling two action value functions Q_{1_i} and Q_{2_i}, along with a policy function π_{θ_i} for each AUV. To address the issue of the Q value overestimation, we utilize two critic networks Θ_{1_i} and Θ_{2_i}, as well as their respective target networks, $\Theta_{1_i}^-$ and $\Theta_{2_i}^-$. The selection of the network with a smaller Q value mitigates the overestimation problem. Consequently, the loss function of Q can be formulated as

$$L_{Q_{1_i}}\left(\Theta_{1_i}\right) = E_{(\boldsymbol{s}_t, \mathfrak{a}_t, r_t, \boldsymbol{s}_{t+1})\sim\mathcal{D}_i}\left[1/2\left(Q_{\Theta_{1_i}}(\boldsymbol{s}_t, \mathfrak{a}_t) - \left(r_t + \gamma V_{\Theta_{1_i}^-}(\boldsymbol{s}_{t+1})\right)\right)^2\right] \tag{25}$$

$$L_{Q_{2_i}}\left(\Theta_{2_i}\right) = E_{(\boldsymbol{s}_t, \mathfrak{a}_t, r_t, \boldsymbol{s}_{t+1})\sim\mathcal{D}_i}\left[1/2\left(Q_{\Theta_{2_i}}(\boldsymbol{s}_t, \mathfrak{a}_t) - \left(r_t + \gamma V_{\Theta_{2_i}^-}(\boldsymbol{s}_{t+1})\right)\right)^2\right] \tag{26}$$

where $\mathcal{D}_i$ denotes the replay buffer to store the collected data, while $V_{\Theta_{1_i}^-}(\boldsymbol{s}_t)$ and $V_{\Theta_{2_i}^-}(\boldsymbol{s}_t)$ represent the state value function with the parameters $\Theta_{1_i}^-$ and $\Theta_{2_i}^-$, respectively. To prevent the AUV i from getting stuck in the local optimal policy, we introduce the entropy regularization and express $V_{\Theta_{1_i}^-}(\boldsymbol{s}_{t+1})$ and $V_{\Theta_{2_i}^-}(\boldsymbol{s}_{t+1})$ as follows:

$$V_{\Theta_{1_i}^-}(\boldsymbol{s}_{t+1}) = \min_{j=1,2} Q_{\Theta_{j_i}^-}(\boldsymbol{s}_{t+1}, \mathfrak{a}_{t+1}) - \alpha_i \log \pi_{\theta_i}(\mathfrak{a}_{t+1} \mid \boldsymbol{s}_{t+1}) \tag{27}$$

$$V_{\Theta_{2_i}^-}(\boldsymbol{s}_{t+1}) = \min_{j=1,2} Q_{\Theta_{j_i}^-}(\boldsymbol{s}_{t+1}, \mathfrak{a}_{t+1}) - \alpha_i \log \pi_{\theta_i}(\mathfrak{a}_{t+1} \mid \boldsymbol{s}_{t+1}) \tag{28}$$

where α_i stands for the regularization coefficient, determining the weight placed on the entropy in the policy. Subsequently, the policy's loss function can be derived from the simplified KL divergence

$$L_{\pi_{\theta_i}}(\theta_i) = E_{\boldsymbol{s}_t\sim\mathcal{D}_i, \mathfrak{a}_t\sim\pi_{\theta_i}}\left[\alpha_i \log\left(\pi_{\theta_i}(\mathfrak{a}_t \mid \boldsymbol{s}_t)\right) - \min_{j=1,2} Q_{\theta_{j_i}}(\boldsymbol{s}_t, \mathfrak{a}_t)\right]. \tag{29}$$

To address the issue of nondifferentiability when sampling actions from the Gaussian distribution $\mathfrak{n}$, the reparameterization trick is introduced, allowing the policy function to be expressed as $\mathfrak{a}_t = f_{\theta_i}(\epsilon_t; \boldsymbol{s}_t)$, where ϵ_t represents a noise

Authorized licensed use limited to: Zhejiang University. Downloaded on October 01,2024 at 09:11:10 UTC from IEEE Xplore. Restrictions apply.

random variable. By considering two action value functions simultaneously, the policy's loss function can be reformulated as follows:

$$L_{\pi_{\theta_i}}(\theta_i) = E_{\boldsymbol{s}_t \sim \mathcal{D}_i, \epsilon_t \sim \mathfrak{n}}\Big[\alpha_i \log\big(\pi_{\theta_i}\big(f_{\theta_i}(\epsilon_t; \boldsymbol{s}_t) \mid \boldsymbol{s}_t\big)\big) - \min_{j=1,2} Q_{\Theta_{j_i}}\big(\boldsymbol{s}_t, f_{\theta_i}(\epsilon_t; \boldsymbol{s}_t)\big)\Big]. \tag{30}$$

To automatically adjust the entropy regularization term, the goal of RL can be reformulated as a constrained optimization problem

$$\max_{\pi_{\theta_i}} E_{\pi_{\theta_i}}\left[\sum_t r_{t_i}\right] \text{ s.t. } E_{\boldsymbol{s}_t \sim \mathcal{D}_i, \mathfrak{a}_t \sim \pi_{\theta_i}}\big[-\log\big(\pi_{\theta_i}(\mathfrak{a}_t \mid \boldsymbol{s}_t)\big)\big] \geq H_0. \tag{31}$$

More intuitively, the objective is to maximize the expected total reward while ensuring that the entropy mean exceeds H_0. By simplifying (31), we can derive the loss function for α_i as

$$L(\alpha_i) = E_{\boldsymbol{s}_t \sim \mathcal{D}_i, \mathfrak{a}_t \sim \pi_{\theta_i}}\big[-\alpha_i \log \pi_{\theta_i}(\mathfrak{a}_t \mid \boldsymbol{s}_t) - \alpha_i H_0\big]. \tag{32}$$

Equations (31) and (32) imply that if the policy entropy is below the desired value H_0, the training target $L(\alpha_i)$ will raise the value of α_i. Consequently, it will amplify the significance of the corresponding term in the policy entropy during the process of minimizing the loss function $L_{\pi_{\theta_i}}(\theta_i)$. Conversely, if the policy entropy exceeds H_0, $L(\alpha_i)$ will lower α_i, thereby directing the policy training toward prioritizing value improvement.

C. Multiagent Independent Decision Transformer

The traditional training mode based on the TD algorithm is faced with challenges, such as low efficiency and especially overestimation. The primary reason for overestimation stems from the tendency to maximize the Q value, which becomes more pronounced when the action space expands.

Theorem 1: Maximization leads to overstimation.

Proof: See Appendix A. ■

Theorem 2: Assume that there is no difference in the expected return of all the actions in the state $\boldsymbol{s}$, i.e., $Q^*(\boldsymbol{s}, \mathfrak{a}) = V^*(\boldsymbol{s})$, It is also assumed that the neural network estimation error $Q_{\omega^-}(\boldsymbol{s}, \mathfrak{a}) - V^*$ obeys the uniform independent distribution of $[-1, 1]$. Suppose the size of the action space is n, then for any state $\boldsymbol{s}$

$$E\left[\max_{\mathfrak{a}} Q_{\omega^-}(\boldsymbol{s}, \mathfrak{a}) - \max_{\mathfrak{a}'} Q^*\big(\boldsymbol{s}, \mathfrak{a}'\big)\right] = \frac{n-1}{n+1} \tag{33}$$

that is, the larger the action space is, the more seriously the Q value is overestimated.

Proof: See Appendix B. ■

To avoid overestimation caused by the TD algorithm, we refer to the utilization of DT to convert the offline RL problem into the seq2seq problem, and we extend DT to MAIDT to make simultaneous training of multi-AUV possible. MAIDT is a DT-based framework that incorporates the insights from [36] on the transformer structure. Transformers have multiple self-attention layers with residual connections, as in [36]. Each layer represents input tokens as embeddings ($\{x_i\}_{i=1}^n$) and outputs embeddings ($\{z_i\}_{i=1}^n$), maintaining the original dimensions. This is achieved by mapping tokens to the key (k_i), query (q_i), and value (v_i) through linear transformations. The self-attention layer calculates the output for each token by weighting values based on the dot product between the query and key. This mechanism establishes associations between the states and returns by assigning "credit" based on the similarity

Algorithm 1 MMOTS Framework

```
Initialize the training environment, including the replay buffer $\mathcal{D}_i$, critic network and corresponding target network, policy network parameters, and entropy regularization $\Theta_{1_i}$, $\Theta_{2_i}$, $\bar{\Theta}_{1_i}$, $\bar{\Theta}_{2_i}$, $\phi_i$, $\alpha_i$ of AUV $i$.
for each episode $k$ do
    Reset the training environment and total reward.
    for each time step $t$ do
        Sample an action according to the policy:
        $\mathfrak{a}_{t_i} \sim \pi_{\theta_i}\big(\mathfrak{a}_{t_i} \mid \boldsymbol{s}_{t_i}\big)$;
        Collect the next state from environment:
        $\boldsymbol{s}_{t+1_i} \sim \mathcal{P}\big(\boldsymbol{s}_{t+1_i} \mid \boldsymbol{s}_{t_i}, \mathfrak{a}_{t_i}\big)$;
        Calculate reward $r_{t_i}$ by (18) $\sim$ (22);
        Store sampling tuple $\big(\boldsymbol{s}_{t_i}, \mathfrak{a}_{t_i}, r_{t_i}, \boldsymbol{s}_{t+1_i}\big)$ into $\mathcal{D}_i$.
        Extract $N$ batches tuple of data from $\mathcal{D}_i$.
        $\Theta_{j_i} \leftarrow \Theta_{j_i} - \lambda_{\Theta_{j_i}} \nabla_{\Theta_{j_i}} J_{\Theta_{j_i}}\big(\Theta_{j_i}\big), j = 1, 2$.
        $\theta_i \leftarrow \theta_i - \lambda_{\theta_i} \nabla_{\theta_i} J_{\theta_i}(\theta_i)$.
        $\alpha_i \leftarrow \alpha_i - \lambda_{\alpha_i} \nabla_{\alpha_i} J_{\alpha_i}(\alpha_i)$.
        $\bar{\Theta}_{j_i} \leftarrow \kappa \Theta_{j_i} + (1 - \kappa)\bar{\Theta}_{j_i}, j = 1, 2$
    end for
end for
Collect trajectories using expert policy obtained by (31).
Modify the trajectories and generate offline data sets $\tau_i'$ by (35).
Sample $n$ batches of sequence length K from the offline data set $\tau_i'$ by (36).
for each gradient step $j$ do
    Update the models of MAIDT using Adam through updating on $\theta_i'$ via $L_{\mathrm{MSE}}\big(\theta_i'\big)$ by (37).
end for
```

$$z_i = \sum_{j=1}^{n} \operatorname{softmax}\Big(\big\{< q_i, k_{j'} >\big\}_{j'=1}^{n}\Big)_j \cdot v_j. \tag{34}$$

Then, the offline data set obtained by (23) is utilized to train the model via MAIDT for each AUV in the UPE game. Moreover, to accurately predict action $\hat{\mathfrak{a}}_i(t)$ during the UPE game, the MAIDT model requires modeling the reward and reshaping the trajectory in the offline data set to align with the autoregressive training and action prediction [36]. This modified trajectory is denoted as τ_i'

$$\tau_i' = \big(\hat{r}_{1_i}, \boldsymbol{s}_{1_i}, \mathfrak{a}_{1_i}, \hat{r}_{2_i}, \boldsymbol{s}_{2_i}, \mathfrak{a}_{2_i}, \ldots, \hat{r}_{T_i}, \boldsymbol{s}_{T_i}, \mathfrak{a}_{T_i}\big) \tag{35}$$

where $\hat{r}_{t_i} = \sum_{t'=t}^{T} r_{t_i'}$ denotes the expected total reward of AUV i.

During the model training, n batches of sequences (τ_s) with the length K are randomly selected from the offline data set

$$\tau_{s_j} = \big(\hat{r}_{1_j}, \boldsymbol{s}_{1_j}, \mathfrak{a}_{1_j}, \hat{r}_{2_j}, \boldsymbol{s}_{2_j}, \mathfrak{a}_{2_j}, \ldots, \hat{r}_{K_j}, \boldsymbol{s}_{K_j}, \mathfrak{a}_{K_j}\big) \quad (j = 1, 2, \ldots, n). \tag{36}$$

Authorized licensed use limited to: Zhejiang University. Downloaded on October 01,2024 at 09:11:10 UTC from IEEE Xplore. Restrictions apply.

The training objective of the prediction head for the input token $\boldsymbol{s}_i(t)$ is to minimize the mean-squared error L_{MSE} for the actions aiming to predict action $\hat{\mathfrak{a}}_i(t)$ for AUV i. The error for each timestep is averaged, as illustrated in

$$\max_{\pi_{\theta_i'}} J'\left(\theta_i'\right) = \min_{\pi_{\theta_i'}} L_{\mathrm{MSE}}\left(\theta_i'\right) = \min_{\pi_{\theta_i'}} \left[-\frac{1}{N} \sum_{j=1}^{N} \left(\mathfrak{a}_j - \hat{\mathfrak{a}}_j\right)^2 \right]. \tag{37}$$

VI. Simulation Result and Discussion

In this section, we aim to validate the proposed MMOTS through a two-stage simulation of training multi-AUV for the UPE game. First we present the experiment's settings, followed by a detailed description of the entire process. Subsequently, we analyze and discuss the results of the experiments, focusing on the performance of MMOTS.

A. Experiment Settings

During the simulation, we employ two distinct sets of parameters: 1) the simulation environment and 2) algorithm parameters. These sets of parameters are considered comprehensively to ensure an effective evaluation.

1) Simulation Environment Parameters: The simulation is carried out on a 400 m × 400 m area with a water depth of −200 m, on which obstacles and vortices are randomly distributed. At the beginning, the positions of the AUVs are randomly distributed. The AUVs know their positions and can obtain the surrounding current velocity through the equipped HADCP. The area boundaries act as obstacles to restrict the AUVs in the specified area. Considering the engineering practice, the speed parameters of AUVs are $\|\boldsymbol{v}_i(t)\| \in [0.0, 3.0]$ m/s and $\|\omega_i(t)\| \in [0.0, 2.0]$ rad/s. In addition, the AUVs' quantity, and the spatial distribution of vortex centers and obstacles will change according to the two different stages of MMOTS.

2) Algorithm Parameters: The implementation of MMOTS incorporates various parameters and settings. In the first stage, MAISAC is employed to optimize the policy and critic networks. The learning rate λ for these networks is set to 3×10^{-4}, while the discount factor γ is assigned a value of 0.99. To facilitate network updates, a soft update coefficient κ of 0.01 is utilized, while the regularization coefficient of entropy α is initialized to 0.2. For efficient training, a replay buffer size C of 5×10^5 is employed, and the batch size for network parameters updating is set to 256. During each episode, a maximum of 6000 steps T are allowed, with a simulation time step Δt of 0.25 s. The training process comprises a total of 140 episodes ε, and in terms of network architecture, a hidden layer size of 256 is utilized. Moving on to the second stage of MMOTS, MAIDT is employed and the parameters are mainly referred to DT [36]. However, certain modifications are made, such as setting the expected total reward to 18 817 and adjusting the number of steps per iteration to 5000. The parameters mentioned above are detailed in Table I for a summary.

TABLE I
Parameters of Simulation Experiment

Parameters	Values
Maximum velocity $\mathrm{V}_{\max}$	3.0 m/s
Max angular velocity $\omega_{\max}$	2.0 rad/s
Experimental site size	400m × 400m
Number of PAUVs	2(first stage) / 4(second stage)
Number of EAUVs	2(first stage) / 1(second stage)
Locations of obstacles	(287m, 300m), (321m, 100m)
Locations of vortex centers	(155m, 146m), (212m, 245m)
Strength of the vortex Γ	8
Radius of the vortex δ	160 m
Safe distance $l_{\min}^{i\leftrightarrow j}$	15 m
Target distance $l_{\max}^{i\leftrightarrow T}$	25 m
Safe current velocity $V_{\max}^{i\leftrightarrow C}$	1.8 m/s
Sample batch size B	256
Maximum steps per episode T	6000
Time step per episode Δt	0.25
Training episodes ε	140
Hidden layer size	256
Expected total reward	18817
Number of steps per iteration (DT)	5000

B. Process of Experiment and Results Analysis

According to the training process of MMOTS, MAISAC is first used for policy improvement to select the optimal policy for the offline data set generation. In order to reduce training difficulty and improve efficiency, the simulation setup is simplified to two PAUVs pursuing two EAUVs navigating at low velocity, respectively, with two obstacles and vortexes in the environment. When the distance between the PAUV and the corresponding EAUV is less than $l_{\max}^{i\leftrightarrow e}$, the pursuit is considered successful, and the EAUV's position will be randomly reset, which prompts the PAUV to continue the pursuit, and each AUV can get the reward in real time in the process. When the maximum number of steps (6000) is reached, or the AUVs encounter obstacles and travel near the center of the vortex, the episode terminates and the indicator variable $D_i(t)$ is set to True, which triggers an environmental reset for the next episode. Then, we compare the proposed MAISAC with the independent proximal policy optimization (IPPO) algorithm, and repeat the experiment three times with different random seeds to mitigate experimental contingency. Subsequently, the comparative analysis are undertaken to investigate how the varying maximum velocity (Vmax) influences the training dynamics. Finally, the outcomes of these investigations are depicted in Figs. 3 and 4, respectively.

Fig. 3 shows the smoothed average total reward curves of the two AUVs using two different algorithms for policy improvement. With the increase of training episodes, the reward presents the gradual upward trend, and it is observed that the reward curves of the MAISAC algorithm rises faster than that of IPPO. In addition, after 140 training episodes, the MAISAC curves reach a stable state, while the IPPO curves have not converged. In addition, it can be seen from Fig. 4 that as the maximum speed increases, the converged value of the average total reward curves also increases. To be intuitive, we also present the relationship between the average total reward and Vmax ranging from 1.8 to 4.2 m/s in Fig. 5. The curves in Fig. 5 demonstrate that as Vmax rises, the average total rewards for both the PAUVs 1 and 2 show a

Authorized licensed use limited to: Zhejiang University. Downloaded on October 01,2024 at 09:11:10 UTC from IEEE Xplore. Restrictions apply.

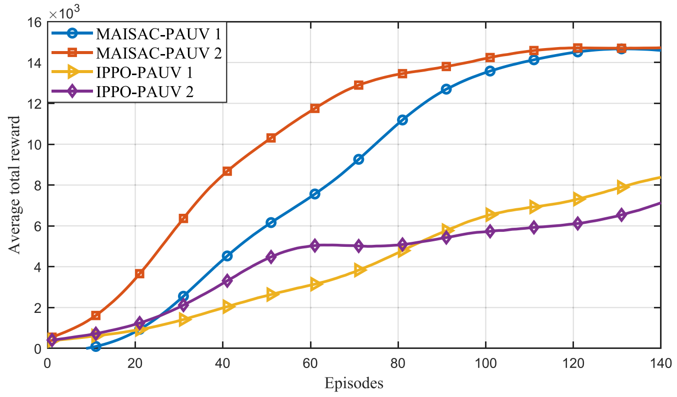


Fig. 3. Smoothed average total reward curves of each PAUV relying on MAISAC and IPPO for policy improvement.

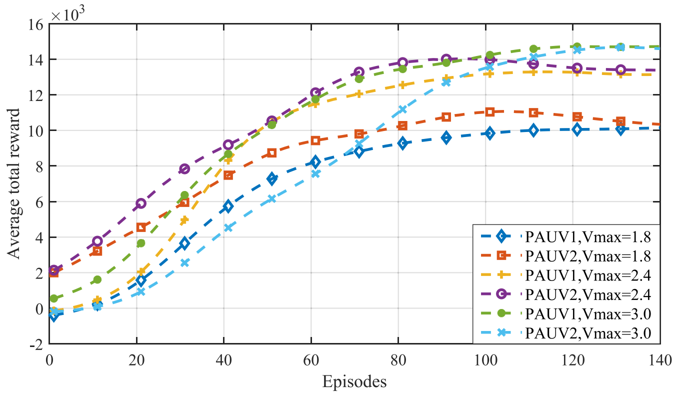


Fig. 4. Smoothed average total reward curves of each PAUV relying on the MAISAC algothrim for policy improvement with V_{max} ranging from 1.8 to 3.0 m/s.

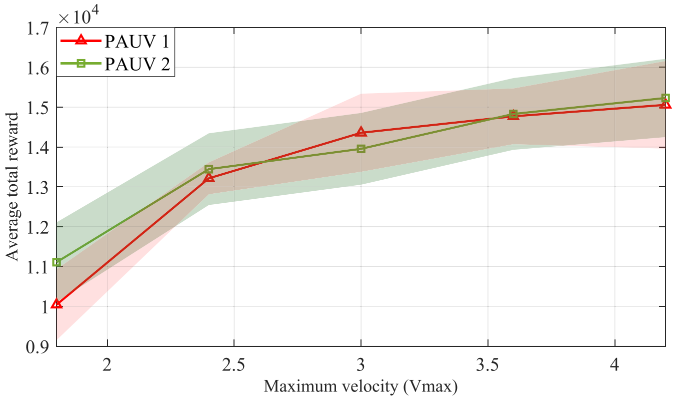
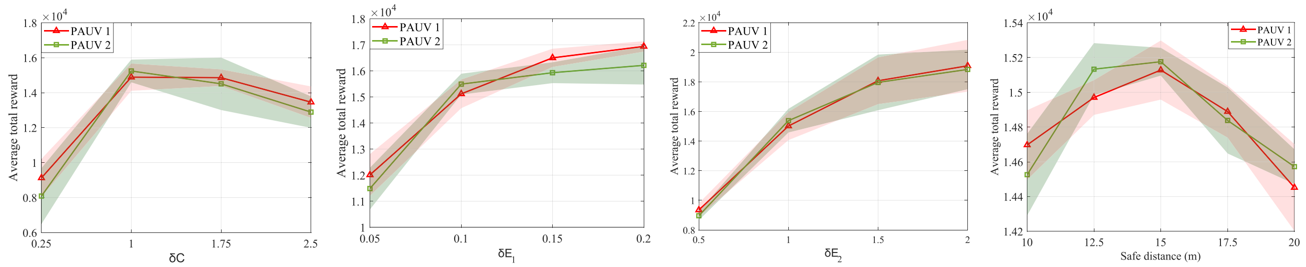


Fig. 5. Average total reward curves of each PAUV relying on the MAISAC algothrim for policy improvement with V_{max} ranging from 1.8 to 4.2 m/s.

progressive uplift. However, this upward trend diminishes over time and eventually stabilizes. Preliminary analysis suggests that since the action space encompasses both the velocity and angular velocity, and the maximum angular velocity becomes the primary factor that limits the PAUV's mobility when the velocity of each PAUV crosses a certain threshold. Therefore, the average total rewards for PAUVs 1 and 2 begin to plateau as Vmax increases.

Furthermore, we also investigate the influence of varying weight coefficients and safe distance on the MAISAC algorithm performance. Specifically, we conduct ablation experiments utilizing δC ranging from 0.25 to 2.50, δE_1 ranging from 0.05 to 0.20, δE_2 ranging from 0.5 to 2.0, and safe distance ranging from 10 to 20 m, respectively, with the results depicted in Fig. 6. Observations from Fig. 6(a) demonstrate that as δC increases, the average total rewards for PAUVs 1 and 2 initially rise before showing a decline. A preliminary analysis suggests this phenomenon is due to δC's role in penalizing AUVs upon colliding obstacles. With a minimal δC, the penalty is significantly less than the rewards for successful pursuit, leading to insufficient deterrence. As a result, each PAUV prioritizes tracking its target EAUV over avoiding obstacles, making obstacle avoidance a secondary concern, which leads to the frequent obstacle collision and low average total reward. Conversely, when δC achieve a high value, such as 2.5, the obstacle penalty of obstacle collision greatly exceeds the rewards for successful pursuit making the deterrence overly harsh. This situation causes PAUVs to adopt a more conservative policy, reluctant to explore the environment aggressively, which leads them to settle for suboptimal solutions and thus the MAISAC algorithm underperforms. Then, as shown in Fig. 6(b) and (c), with the increase of δE_1 and δE_2, the average total reward of PAUVs 1 and 2 gradually increases. Since, δE_1 and δE_2 are both intended to award each PAUV to approach the target EAUV, when δE_1 is very small, the absolute value of reward is much smaller than the reward or penalty obtained when successfully pursuit or colliding obstacles, making the reward obtained by PAUVs approximately sparse reward. As a result, in the early stage of training via MAISAC, it is difficult to obtain effective information from the replay buffer data for training the policy and critic networks, so the training and convergence speed is reduced, resulting in a low total average reward and suboptimal policies. With the increase of δE_1, the value of reward also increases, and PAUVs may gradually obtain effective information from the data collected through each step, thus speeding up the training process. However, the increase of the total average reward gradually slows down, which may be attributed to the increase of δE_1, leading to the saturation of effective information obtained from the ollected data. On the other hand, when δE_2 is very small, the absolute value of reward is much smaller than the penalty obtained when colliding obstacles. As a result, each PAUV prioritizes avoiding obstacles over tracking its target EAUV, making target pursuit a secondary concern. Conversely, when δE_2 rises to 2.0, the corresponding reward value is much smaller than the reward value obtained after successful pursuit, resulting in PAUVs being more inclined to approach the target EAUV, while ignoring obstacle avoidance, finally resulting in the low total average reward and suboptimal policies. Moreover, as can be seen from the Fig. 6(d), with the increase of safe distance, the average total reward of both the PAUV shows a trend of first increasing and then decreasing. Upon analyzing the phenomenon, it can be indicated that when the safe distance is too low, PAUV will take a shorter path to complete the pursuit, but at the same time, it will also cause

Authorized licensed use limited to: Zhejiang University. Downloaded on October 01,2024 at 09:11:10 UTC from IEEE Xplore. Restrictions apply.



(a)

(b)

(c)

(d)

Fig. 6. Average total reward curves of each PAUV relying on the MAISAC algorithm for policy improvement with varying reward weight coefficients and safe distance, respectively. (a) δC varies from 0.25 to 2.5. (b) δE_1 varies 0.05 to 0.2. (c) δE_2 varies from 0.5 to 2. (d) Safe distance varies from 10 to 20 m.

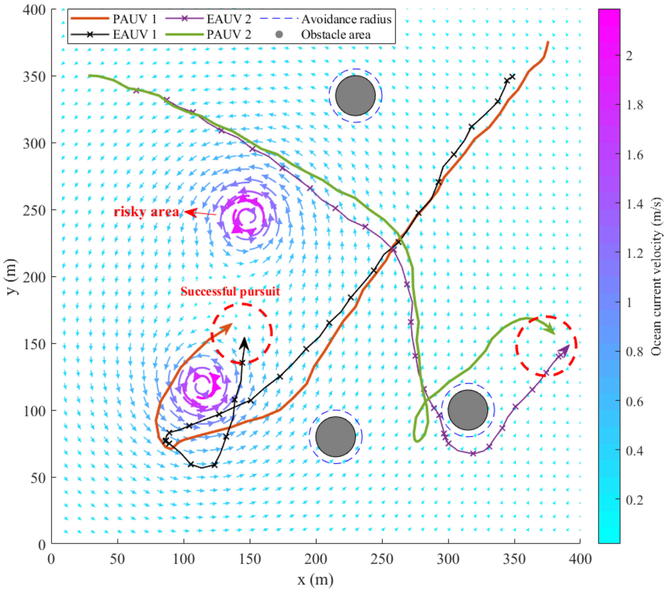


Fig. 7. Trajectories of two PAUVs and two corresponding target EAUVs in one successful pursuit, using MAISAC for policy improvement.

the short-sightedness of obstacle avoidance, which makes the obstacle avoidance not timely enough, especially in the dynamic environment, thus weakening the overall performance of MAISAC to a certain extent. On the contrary, when the safe distance gets too large, in order to avoid obstacles, PAUVs will aggravate the distance from the obstacles, which also weakens the performance of MAISAC. These results and analysis highlight the superior training efficiency of MAISAC and the significance of parameter settings for the first stage of MMOTS, helping to reduce computational and time costs in RL training.

In order to visualize the training process of MAISAC more intuitively, we draw the trajectories of each AUV in one successful pursuit as depicted in Fig. 7. As illustrated in Fig. 7, each PAUV tracks its corresponding EAUV while avoiding the obstacles and vortices, and finally complete a successful pursuit, showcasing the effectiveness of the first stage of MMOTS. While at episode 140 in the training process of MAISAC, where the reward curves reach a plateau, the AUVs' policies are deemed to have reached expert level. The policy corresponding to the highest total reward (18 817) is selected to generate an offline data set. Leveraging the collaboration of the multi-AUV within the UPE game, the offline data set is subsequently employed to train the MAIDT models, which encompasses the models of four PAUVs and a single EAUV. This process yields training loss curves as depicted in Fig. 8. The initial loss values of 0.7938, 0.8129, 0.8226, 0.8103, and 0.7822 are subsequently reduced to 0.0747, 0.0778, 0.0769, 0.0767, and 0.0772, respectively, indicating successful completion of the model training process.

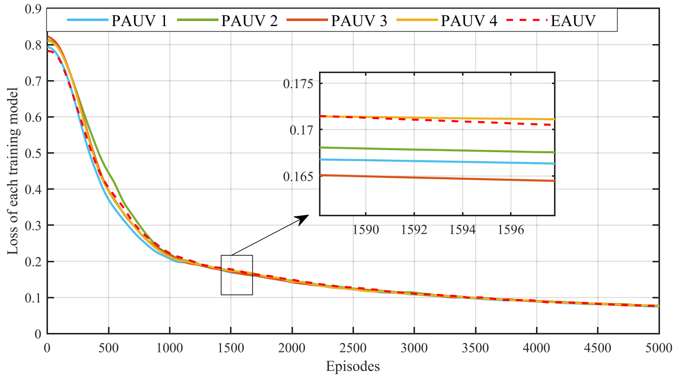


Fig. 8. Model loss curves of each AUV relying on the MAIDT algorithm for training.

For the remainder in the second stage of MMOTS, we employ the trained MAIDT model for each AUV in the UPE game. By inputting the highest expected total reward and the initial state of each AUV into the model, it can accurately predict the next action based on the current expected total reward and state. The AUVs initiate navigation simultaneously to fulfill their respective roles and travel around obstacles and vortices. After conducting 1500 steps, we observe the trajectories of each AUV in a successful pursuit episode, depicted in Fig. 9.

Furthermore, we also conduct comparative experiments to explore the impact of the offline data set quality and environment complexity on the ultimate training results of MMOTS, respectively. On the one hand, for the offline data set quality, we utilize a suboptimal policy derived from employing MAISAC for policy improvement, and subsequently, the MAIDT algorithm is employed to train the model, which is then employed in each AUV. On the other hand, we improve environment complexity by introducing more obstacles and

Authorized licensed use limited to: Zhejiang University. Downloaded on October 01,2024 at 09:11:10 UTC from IEEE Xplore. Restrictions apply.

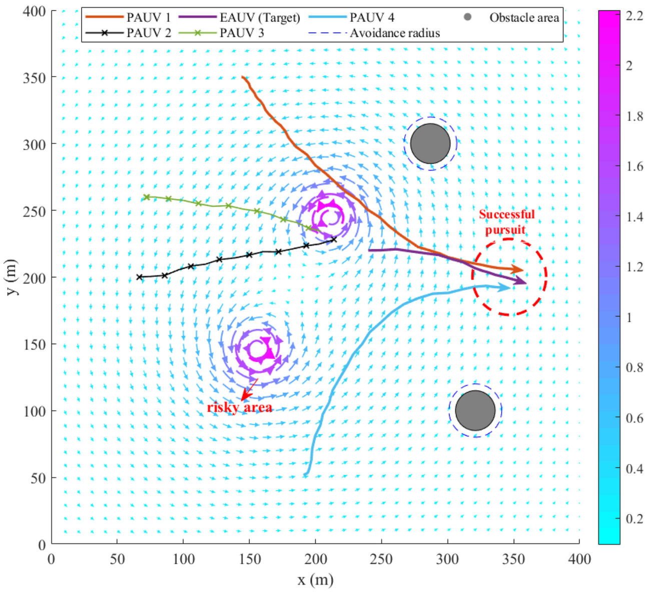


Fig. 9. Trajectories of the UPE game with four PAUVs and one EAUV in a successful pursuit episode in the simple environment, using the offline data set with optimal quality for model training.

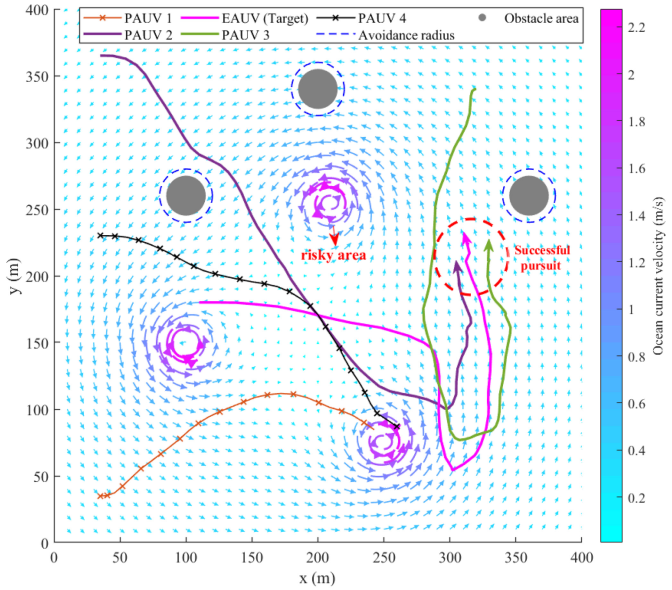


Fig. 11. Trajectories of the UPE game with four PAUVs and one EAUV in a successful pursuit episode in the complex environment, using the offline data set with optimal quality for model training.

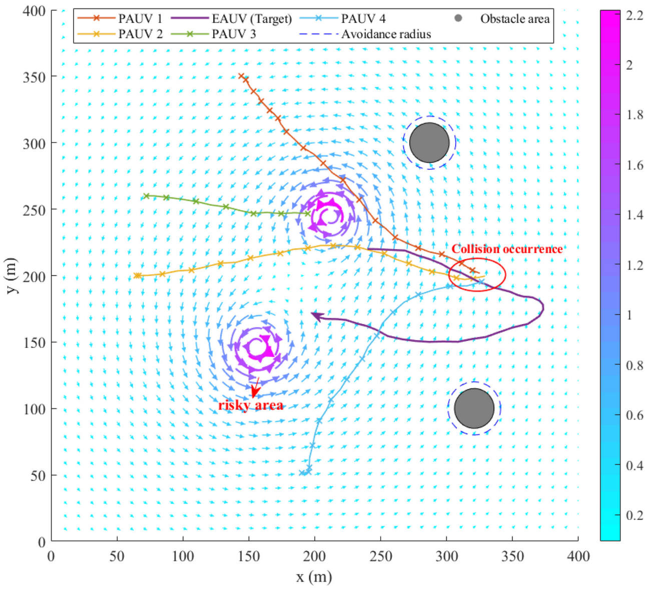


Fig. 10. Trajectories of the UPE game with four PAUVs and one EAUV in a failed pursuit episode in the simple environment, using the offline data set with suboptimal quality for model training.

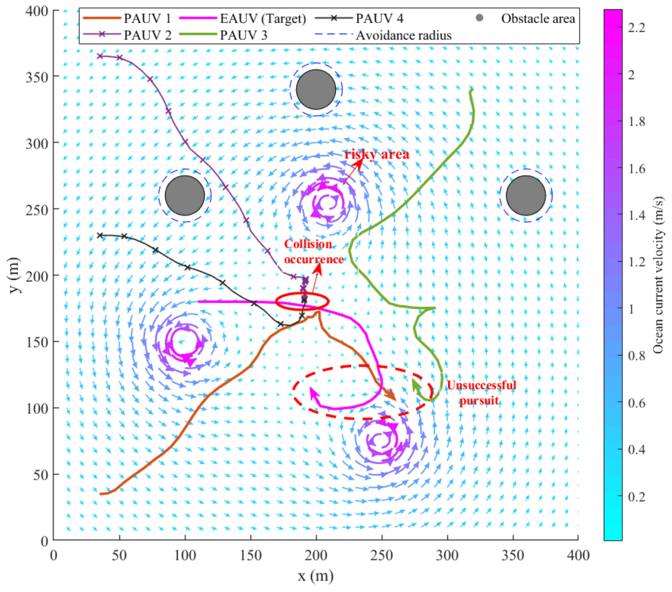


Fig. 12. Trajectories of the UPE game with four PAUVs and one EAUV in a failed pursuit episode in the complex environment, using the offline data set with suboptimal quality for model training.

vortices. The corresponding resulting trajectories in the UPE game are depicted in Figs. 10–12, respectively.

Compared with Figs. 9 and 11, respectively, trajectories from Figs. 10 and 12 indicate that the pursuit performance of PAUVs is unsatisfactory, resulting in a collision involving three PAUVs and two PAUVs, respectively. These outcomes underscore the significance of the offline data set's quality in the training results of MMOTS. Enhancements in the quality of the offline data set are correlated with advancements in the AUVs' policy and intelligence within the UPE game. Furthermore, compared with Fig. 9, the trajectories in Fig. 11 indicate that the PAUVs can also successfully complete the pursuit of EAUV and the UPE game, showcasing the excellent robustness and generalization of proposed MMOTS.

Moreover, to prove MAIDT's superior performance over baselines, and the improvement on the overestimation of the Q value, we compare it with behavior cloning (BC) [37] and conservative Q-learning (CQL) [14], two classical offline RL algorithms based on the supervised learning and TD, respectively. The mean and variance of the total reward of MAIDT and CQL algorithms across the 140 episodes are compared in Fig. 13 with respect to the number of PAUVs ranging from 1 to 4.

Authorized licensed use limited to: Zhejiang University. Downloaded on October 01,2024 at 09:11:10 UTC from IEEE Xplore. Restrictions apply.

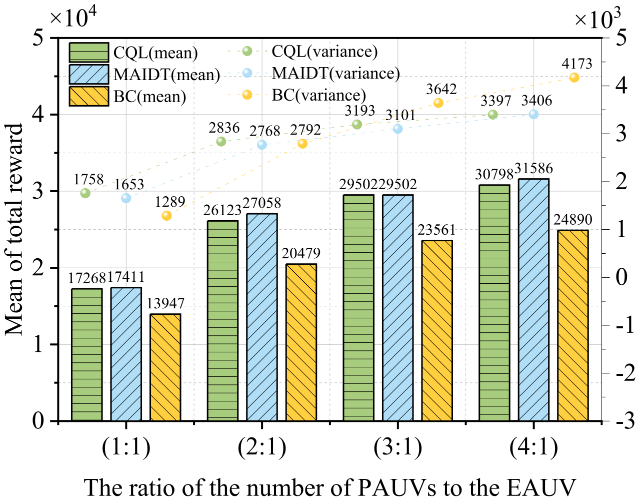


Fig. 13. Mean and variance of total reward curves of PAUVs using CQL, MAIDT, and BC for model training with the PAUVs number ranging from 1 to 4.

Upon analysing Fig. 13, it becomes apparent that as the number of PAUVs increases, the mean and variance of the total reward increases as well, with the mean rising from 17 268, 17 411, 13 947 to 30 798, 31 586, and 24 890, respectively. Similarly, the variance also sees an increase from 1785, 1635, 1289 to 3397, 3406, and 4173, respectively. Furthermore, it can be observed that the except in cases involving three PAUVs, MAIDT generally has higher total reward than the CQL, and at any cases than the BC. And similarly, except in the cases involving four PAUVs, MAIDT typically has lower variance than the CQL, and at any cases than the BC.

The above simulation results demonstrate the superior performance of MMOTS. In its first stage, the MAISAC algorithm effectively implements policy improvement for AUVs and obtains the expert policy in an unstable environment, thereby reducing training convergence difficulties and improving efficiency. Additionally, in the second stage, the MAIDT algorithm enables AUVs to accomplish the UPE game solely from the existing offline data sets, reducing computational and time costs associated with environment interaction while achieving favorable results. Furthermore, the scalability of MMOTS is validated as they require only a small number of AUVs for the policy improvement and offline data set generation, allowing for extension to more quantity of PAUVs in the UPE game.

VII. Conclusion

In this article, we present MMOTS, a novel training framework for the IoUT-assisted UPE game in complex ocean environments. First, we deduce the motion model and detection model of AUV and introduced the ocean current model to characterize the ocean turbulent environment. Considering that the UPE game is a high-dimensional NP-hard problem, we formulate the UPE game as FMGP and design the appropriate reward function. Subsequently, the DTDE framework-based MAISAC is used to train the multiple AUVs to make policy improvements and generate the offline data sets. Finally, we utilize the data set to realize the model training via MAIDT, enabling a larger number of AUVs to learn the expert policy for the UPE game. Extensive experiments confirm that the MAISAC has significant training efficiency, and the high-quality offline data sets are crucial for MAIDT model training. The excellent performance in the UPE games under different conditions and environments reflects that our proposed MMOTS framework has good practicability and extensibility. The future work can focus on the multitarget pursuit-evasion game while considering the intelligence level differences between the pursuer and evaders. Furthermore, efforts are needed to reduce the gap between the simulation environment and the real environment to address the challenges of transferring from simulation to reality.

Appendix A
Proof of Theorem 1

For all the actions $\mathfrak{a} \in \boldsymbol{\mathcal{A}}$ and states $\boldsymbol{s} \in \boldsymbol{S}$, assume that the critic network's output is the true value $Q_*(\boldsymbol{s}_t, \mathfrak{a}_t)$ combined with the random noise ϵ, which has a mean value of 0

$$Q(\boldsymbol{s}_t, \mathfrak{a}_t) = Q_*(\boldsymbol{s}_t, \mathfrak{a}_t) + \epsilon. \tag{38}$$

Obviously, $Q(\boldsymbol{s}_t, \mathfrak{a}_t)$ is an unbiased estimate of the true value $Q_*(\boldsymbol{s}_t, \mathfrak{a}_t)$. However, there are following inequalities:

$$E_\epsilon\left[\max_{\mathfrak{a}\in A} Q(\boldsymbol{s}_t, \mathfrak{a}_t)\right] \geq \max_{\mathfrak{a}\in\boldsymbol{\mathcal{A}}} Q_*(\boldsymbol{s}_t, \mathfrak{a}_t). \tag{39}$$

Equation (39) highlights that although the critic network provides an unbiased estimation of the true value, maximizing it will inevitably lead to an overestimation of the actual value. In summary, the TD algorithm computes the target as follows:

$$\hat{y}_t = r_t + \gamma \cdot \underbrace{\max_{\mathfrak{a}\in\boldsymbol{\mathcal{A}}} Q(\boldsymbol{s}_{t+1}, \mathfrak{a})}_{\text{overestimate } \max_{\mathfrak{a}\in\boldsymbol{\mathcal{A}}} Q_*(s_{t+1}, \mathfrak{a})}. \tag{40}$$

The provided equation demonstrates that the TD target, denoted as $\hat{y}_t$, often exceeds the true value $Q_*(\boldsymbol{s}_t, \mathfrak{a}_t)$. Consequently, the TD algorithm incentivizes $Q(\boldsymbol{s}_t, \mathfrak{a}_t)$ to converge toward $\hat{y}_t$, leading to an overestimation of $Q_*(\boldsymbol{s}_t, \mathfrak{a}_t)$. ■

Appendix B
Proof of Theorem 2

We can write the estimate error $\epsilon_{\mathfrak{a}}$ as: $\epsilon_{\mathfrak{a}} = Q_{\omega^-}(\boldsymbol{s}, \mathfrak{a}) - \max_{\mathfrak{a}'} Q^*(\boldsymbol{s}, \mathfrak{a}')$. Considering the estimation error for different action is independent, thus there are

$$P\left(\max_{\mathfrak{a}} \epsilon_{\mathfrak{a}} \leq x\right) = \prod_{\mathfrak{a}=1}^{n} P(\epsilon_{\mathfrak{a}} \leq x) \tag{41}$$

where $P(\epsilon_{\mathfrak{a}} \leq x)$ is the cumulative distribution function (CDF) of $\epsilon_{\mathfrak{a}}$, which can be concretely written as

$$P(\epsilon_{\mathfrak{a}} \leq x) = \begin{cases} 0, & \text{if } x \leq -1 \\ \frac{1+x}{2}, & \text{if } x \in (-1, 1) \\ 1, & \text{if } x \geq 1. \end{cases} \tag{42}$$

Authorized licensed use limited to: Zhejiang University. Downloaded on October 01,2024 at 09:11:10 UTC from IEEE Xplore. Restrictions apply.

Therefore, we obtain the CDF for $\max_{\mathfrak{a}} \epsilon_{\mathfrak{a}}$ as

$$P\Big(\max_{\mathfrak{a}} \epsilon_{\mathfrak{a}} \leq x\Big) = \prod_{\mathfrak{a}=1}^{n} P(\epsilon_{\mathfrak{a}} \leq x) = \begin{cases} 0, & \text{if } x \leq -1 \\ \left(\frac{1+x}{2}\right)^{n}, & \text{if } x \in (-1, 1) \\ 1, & \text{if } x \geq 1. \end{cases} \tag{43}$$

This gives us the CDF of the random variable $\max_{\mathfrak{a}} \epsilon_{\mathfrak{a}}$, whose expectation can be written as an integral

$$E\Big[\max_{\mathfrak{a}} \epsilon_{\mathfrak{a}}\Big] = \int_{-1}^{1} x g_m(x) dx \tag{44}$$

where g_m denotes the probability density function (PDF), defined as the derivative of the CDF: $g_m(x) = (d/dx)P(\max_{\mathfrak{a}} \epsilon_{\mathfrak{a}} \leq x)$, so that for $x \in [-1, 1]$, we have $g_m(x) = (n/2)[(1 + x)/2]^{n-1}$. Finally, we can get

$$\begin{aligned} E\Big[\max_{\mathfrak{a}} \epsilon_{\mathfrak{a}}\Big] &= \int_{-1}^{1} x \frac{d}{dx} P\Big(\max_{\mathfrak{a}} \epsilon_{\mathfrak{a}} \leq x\Big) dx \\ &= \left[\left(\frac{1+x}{2}\right)^{n} \frac{nx-1}{n+1}\right]_{-1}^{1} \\ &= \frac{n-1}{n+1}. \end{aligned} \tag{45}$$

■

References

[1] Z. Wang, Z. Zhang, J. Wang, C. Jiang, W. Wei, and Y. Ren, "AUV-assisted node repair for IoUT relying on multiagent reinforcement learning," *IEEE Internet Things J.*, vol. 11, no. 3, pp. 4139–4151, Feb. 2024, doi: 10.1109/JIOT.2023.3298522.

[2] Y. Li, L. Liu, W. Yu, Y. Wang, and X. Guan, "Noncooperative mobile target tracking using multiple AUVs in anchor-free environments," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9819–9833, Oct. 2020.

[3] S. Guan, J. Wang, C. Jiang, R. Duan, Y. Ren, and T. Q. S. Quek, "MagicNet: The maritime giant cellular network," *IEEE Commun. Mag.*, vol. 59, no. 3, pp. 117–123, Mar. 2021.

[4] W. Wei, J. Wang, J. Du, Z. Fang, C. Jiang, and Y. Ren, "Underwater differential game: Finite-time target hunting task with communication delay," in *Proc. IEEE Int. Conf. Commun.*, 2022, pp. 3989–3994.

[5] L. Liu, S. Zhang, L. Zhang, G. Pan, and J. Yu, "Multi-UUV maneuvering counter-game for dynamic target scenario based on fractional-order recurrent neural network," *IEEE Trans. Cybern.*, vol. 53, no. 6, pp. 4015–4028, Jun. 2023.

[6] J. Wu, C. Song, J. Ma, J. Wu, and G. Han, "Reinforcement learning and particle swarm optimization supporting real-time rescue assignments for multiple autonomous underwater vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 6807–6820, Jul. 2022.

[7] X. Cao, L. Ren, and C. Sun, "Dynamic target tracking control of autonomous underwater vehicle based on trajectory prediction," *IEEE Trans. Cybern.*, vol. 53, no. 3, pp. 1968–1981, Mar. 2023.

[8] M. Zhang, H. Chen, and W. Cai, "Hunting task allocation for heterogeneous multi-AUV formation target hunting in IoUT: A game theoretic approach," *IEEE Internet Things J.*, vol. 11, no. 5, pp. 9142–9152, Mar. 2024, doi: 10.1109/JIOT.2023.3322197.

[9] R. Zhang, Q. Zong, X. Zhang, L. Dou, and B. Tian, "Game of drones: Multi-UAV pursuit-evasion game with online motion planning by deep reinforcement learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 10, pp. 7900–7909, Oct. 2023.

[10] C. C. Wang, Y. L. Wang, P. Shi, and F. Wang, "Scalable-MADDPG-based cooperative target invasion for a multi-USV system," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Sep. 7, 2023, doi: 10.1109/TNNLS.2023.3309689.

[11] K. Xu, N. Van Huynh, and G. Y. Li, "Distributed-training-and-execution multi-agent reinforcement learning for power control in HetNet," *IEEE Trans. Commun.*, vol. 71, no. 10, pp. 5893–5903, Oct. 2023.

[12] Y. Qiu, Y. Jin, L. Yu, J. Wang, Y. Wang, and X. Zhang, "Improving sample efficiency of multi-agent reinforcement learning with non-expert policy for flocking control," *IEEE Internet Things J.*, vol. 10, no. 16, pp. 14014–14027, Aug. 2023.

[13] P. Rashidinejad, B. Zhu, C. Ma, J. Jiao, and S. Russell, "Bridging offline reinforcement learning and imitation learning: A tale of pessimism," *IEEE Trans. Inf. Theory,* vol. 68, no. 12, pp. 8156–8196, Dec. 2022.

[14] R. Yang, C. Bai, X. Ma, Z. Wang, C. Zhang, and L. Han, "RORL: Robust offline reinforcement learning via conservative smoothing," in *Proc. 36th Conf. Neural Inf. Process. Syst.*, 2022, pp. 23851–23866.

[15] J. Kim, "Tracking controllers to chase a target using multiple autonomous underwater vehicles measuring the sound emitted from the target," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 7, pp. 4579–4587, Jul. 2021.

[16] J. Wang, C. Feng, L. Wang, G. Li, and B. He, "Detection of weak and small targets in forward-looking sonar image using multi-branch shuttle neural network," *IEEE Sens. J.*, vol. 22, no. 7, pp. 6772–6783, Apr. 2022.

[17] J. Yan, K. You, W. Cao, X. Yang, and X. Guan, "Binocular vision-based motion planning of an AUV: A deep reinforcement learning approach," *IEEE Trans. Intell. Veh.*, early access, Oct. 4, 2023, doi: 10.1109/TIV.2023.3321884.

[18] R. Ren, L. Zhang, L. Liu, and Y. Yuan, "Two AUVs guidance method for self-reconfiguration mission based on monocular vision," *IEEE Sens. J.*, vol. 21, no. 8, pp. 10082–10090, Apr. 2021.

[19] X.-F. Liu, Y. Fang, Z.-H. Zhan, Y.-L. Jiang, and J. Zhang, "A cooperative evolutionary computation algorithm for dynamic multiobjective multi-AUV path planning," *IEEE Trans. Ind. Inf.*, vol. 20, no. 1, pp. 669–680, Jan. 2024, doi: 10.1109/TII.2023.3268760.

[20] H. Zhao, J. Yan, X. Luo, and X. Guan, "Ubiquitous tracking for autonomous underwater vehicle with IoUT: A rigid-graph-based solution," *IEEE Internet Things J.*, vol. 8, no. 18, pp. 14094–14109, Sep. 2021.

[21] Y. Shou, B. Xu, A. Zhang, and T. Mei, "Virtual guidance-based coordinated tracking control of multi-autonomous underwater vehicles using composite neural learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 12, pp. 5565–5574, Dec. 2021.

[22] K. Zhang, H. Wang, H. Zhang, N. Luo, and J. Ren, "Target tracking of UUV based on máximum correntropy high-order UGHF," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–16 Oct. 2023.

[23] H. Wang, G. Han, W. Lai, Y. Hou, and C. Lin, "A multi-round game-based source location privacy protection scheme with AUV enabled in underwater acoustic sensor networks," *IEEE Trans. Veh. Technol.*, vol. 72, no. 6, pp. 7728–7742, Jun. 2023.

[24] X. Cao, D. Zhu, and S. X. Yang, "Multi-AUV target search based on bioinspired neurodynamics model in 3-D underwater environments," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 11, pp. 2364–2374, Nov. 2016.

[25] A. Signori, F. Chiariotti, F. Campagnaro, and M. Zorzi, "A game-theoretic and experimental analysis of energy-depleting underwater jamming attacks," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9793–9804, Oct. 2020.

[26] A. Signori et al., "A geometry-based game theoretical model of blind and reactive underwater jamming," *IEEE Trans. Wireless Commun.*, vol. 21, no. 6, pp. 3737–3751, Jun. 2022.

[27] W. Wei, J. Wang, J. Du, Z. Fang, Y. Ren, and C. L. P. Chen, "Differential game-based deep reinforcement learning in underwater target hunting task," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Oct. 27, 2023, doi: 10.1109/TNNLS.2023.3325580.

[28] Z. Xia et al., "Multi-agent reinforcement learning aided intelligent UAV swarm for target tracking," *IEEE Trans. Veh. Technol.*, vol. 71, no. 1, pp. 931–945, Jan. 2022.

[29] M. T. Isik and O. B. Akan, "A three dimensional localization algorithm for underwater acoustic sensor networks," *IEEE Trans. Wireless Commun.*, vol. 8, no. 9, pp. 4457–4463, Sep. 2009.

[30] B. Garau, A. Alvarez, and G. Oliver, "AUV navigation through turbulent ocean environments supported by onboard H-ADCP," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2006, pp. 3556–3561.

[31] Z. Zeng, K. Sammut, A. Lammas, F. He, and Y. Tang, "Efficient path re-planning for AUVs operating in spatiotemporal currents," *J. Intell. Robot. Syst.*, vol. 79, no. 1, pp. 135–153, Jul. 2015.

[32] X. Hou, J. Wang, T. Bai, Y. Deng, Y. Ren, and L. Hanzo, "Environment-aware AUV trajectory design and resource management for multi-tier underwater computing," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 2, pp. 474–490, Feb. 2023.

Authorized licensed use limited to: Zhejiang University. Downloaded on October 01,2024 at 09:11:10 UTC from IEEE Xplore. Restrictions apply.

[33] M. Stojanovic, “On the relationship between capacity and distance in an underwater acoustic communication channel,” *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 11, no. 4, pp. 34–43, Oct. 2007.

[34] S. Shuai and M. H. Kasbaoui, “Accelerated decay of a Lamb–Oseen vortex tube laden with inertial particles in Eulerian–Lagrangian simulations,” *J. Fluid Mech.*, vol. 936, p. A8, Feb. 2022.

[35] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, “Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor,” in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 1861–1870.

[36] L. Chen et al., “Decision transformer: Reinforcement learning via sequence modeling,” in *Proc. 35th Conf. Neural Inf. Process. Syst.*, 2021, pp. 15084–15097.

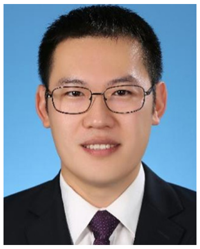
[37] G. Li, Z. Ji, S. Li, X. Luo, and X. Qu, “Driver behavioral cloning for route following in autonomous vehicles using task knowledge distillation,” *IEEE Trans. Intell. Veh.*, vol. 8, no. 2, pp. 1025–1033, Feb. 2023.

Jingzehua Xu (Student Member, IEEE) was born in Xuzhou, Jiangsu, China, in 2001. He received the B.S. degree in marine science and the B.E. degree in electronic science and technology from Zhejiang University, Hangzhou, China, in 2023. He is currently pursuing the M.S. degree in electronic information from Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen, China.

He is an outstanding graduate with Zhejiang University. His main research interests include reinforcement learning, large language models, and underwater robots.

Zekai Zhang was born in Nanjing, Jiangsu, China, in 2000. He received the B.S. degree in electronic engineering from the North University of China, Taiyuan, China, in 2021. He is currently pursuing the M.S. degree in electronic information with Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen, China.

His research interests include robot simulation technology, multiagent cooperation, and industrial applications.



Jingjing Wang (Senior Member, IEEE) received the B.S. degree in electronic information engineering from Dalian University of Technology, Dalian, China, in 2014, and the Ph.D. degree in information and communication engineering from Tsinghua University, Beijing, China, in 2019, both with the highest honors.

From 2017 to 2018, he visited the next generation wireless group chaired by Prof. Hanzo with the University of Southampton, Southampton, U.K. He is currently a Professor with the School of Cyber Science and Technology, Beihang University, Beijing, China, and also a Researcher with Hangzhou Innovation Institute, Beihang University, Hangzhou, China. He has published over 100 IEEE journal/conference papers. His research interests include AI enhanced next-generation wireless networks, UAV networking, and swarm intelligence.

Dr. Wang was a recipient of the Best Journal Paper Award of IEEE ComSoc Technical Committee on Green Communications and Computing in 2018 and the Best Paper Award of the IEEE ICC and IEEE IWCMC in 2019. He is currently serving as an Editor for IEEE WIRELESS COMMUNICATIONS LETTER and IEEE Open Journal of the Communications Society. He has served as a Guest Editor for IEEE INTERNET OF THINGS JOURNAL.

Zhu Han (Fellow, IEEE) received the B.S. degree in electronic engineering from Tsinghua University, Beijing, China, in 1997, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Maryland, College Park, MD, USA, in 1999 and 2003, respectively.

From 2000 to 2002, he was an Research and Development Engineer with JDSU, Germantown, MD. From 2003 to 2006, he was a Research Associate with the University of Maryland. From 2006 to 2008, he was an Assistant Professor with Boise State University, Boise, ID, USA. He is currently a John and Rebecca Moores Professor with the Electrical and Computer Engineering Department as well as with the Computer Science Department, University of Houston, Houston, TX, USA. His main research targets on the novel game-theory related concepts critical to enabling efficient and distributive use of wireless networks with limited resources. His other research interests include wireless resource allocation and management, wireless communications and networking, quantum computing, data science, smart grid, carbon neutralization, and security and privacy.

Dr. Han received an NSF Career Award in 2010, the Fred W. Ellersick Prize of the IEEE Communication Society in 2011, the EURASIP Best Paper Award for the Journal on Advances in Signal Processing in 2015, the IEEE Leonard G. Abraham Prize in the field of Communications Systems (Best Paper Award in IEEE JSAC) in 2016, the IEEE Vehicular Technology Society 2022 Best Land Transportation Paper Award, and several best paper awards in IEEE conferences. He is also the winner of the 2021 IEEE Kiyo Tomiyasu Award (an IEEE Field Award), for outstanding early to mid-career contributions to technologies holding the promise of innovative applications, with the following citation: “For Contributions to Game Theory and Distributed management of Autonomous Communication Networks.” He is an 1% Highly Cited Researcher according to Web of Science since 2017. He was an IEEE Communications Society Distinguished Lecturer from 2015 to 2018 and an ACM Distinguished Speaker from 2022 to 2025. He has been an ACM Distinguished Member since 2019. He has been a Fellow of AAAS since 2019 and ACM since 2024.

Yong Ren (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electronic engineering from Harbin Institute of Technology, Harbin, China, in 1984, 1987, and 1994, respectively.

He worked as a Postdoctoral Fellow with the Department of Electronics Engineering, Tsinghua University, Beijing, China, from 1995 to 1997, where he is currently a Professor with the Department of Electronics Engineering and the Director of the Complexity Engineered Systems Lab. He holds 60 patents and has authored or co-authored more than 300 technical papers in the behavior of computer network, P2P network, and cognitive networks. His current research interests include complex systems theory and its applications to the optimization and information sharing of Internet, Internet of Things, ubiquitous network, cognitive networks, and cyber–physical systems.

Prof. Ren has served as a reviewer of *IEICE Transactions on Communications*, *Digital Signal Processing*, *Chinese Physics Letters*, *Chinese Journal of Electronics*, *Chinese Journal of Computer Science and Technology*, and *Chinese Journal of Aeronautics*.

Authorized licensed use limited to: Zhejiang University. Downloaded on October 01,2024 at 09:11:10 UTC from IEEE Xplore. Restrictions apply.