Digital Communications and Networks xxx (xxxx) xxx

Contents lists available at ScienceDirect

Digital Communications and Networks

journal homepage: www.keaipublishing.com/dcan

A digital twins enabled underwater intelligent internet vehicle path planning system via reinforcement learning and edge computing

Jiachen Yang [a], Meng Xi [a,*], Jiabao Wen [a], Yang Li [a], Houbing Herbert Song [b]

[a] School of Electrical and Information Engineering, Tianjin University, Tianjin, 300072, China
[b] Security and Optimization for Networked Globe Laboratory (SONG Lab), Embry-Riddle Aeronautical University, Daytona Beach, FL, 32114, USA

ARTICLE INFO

Keywords:
Digital twins
Reinforcement learning
Edge computing
Underwater intelligent internet vehicle
Path planning

ABSTRACT

The Autonomous Underwater Glider (AUG) is a kind of prevailing underwater intelligent internet vehicle and occupies a dominant position in industrial applications, in which path planning is an essential problem. Due to the complexity and variability of the ocean, accurate environment modeling and flexible path planning algorithms are pivotal challenges. The traditional models mainly utilize mathematical functions, which are not complete and reliable. Most existing path planning algorithms depend on the environment and lack flexibility. To overcome these challenges, we propose a path planning system for underwater intelligent internet vehicles. It applies digital twins and sensor data to map the real ocean environment to a virtual digital space, which provides a comprehensive and reliable environment for path simulation. We design a value-based reinforcement learning path planning algorithm and explore the optimal network structure parameters. The path simulation is controlled by a closed-loop model integrated into the terminal vehicle through edge computing. The integration of state input enriches the learning of neural networks and helps to improve generalization and flexibility. The task-related reward function promotes the rapid convergence of the training. The experimental results prove that our reinforcement learning based path planning algorithm has great flexibility and can effectively adapt to a variety of different ocean conditions.

1. Introduction

The ocean occupies the vast majority of the earth and plays a pivotal role in the entire ecology. The diversity of marine life helps maintain the stability of the biosphere and the ocean environment provides inexhaustible resources for human society. It is of great significance to the protection and rational development of the ocean. With the development of intelligent devices and Internet of Things (IoT) technology, people are committed to constructing a smart ocean system [1] and Internet of Underwater Things (IoUT) [2]. Among the numerous underwater intelligent internet vehicles, the Autonomous Underwater Glider (AUG) presents unique superiorities, large scale, low cost, and long endurance. It has become a favorable tool for ocean exploration and has been widely used in many industrial applications [3,4]. Path planning is the basis of extensive applications [5], which specifies an optimal path according to the characteristics of the specific task, such as collision-free or minimum time consumption.

The path planning system includes two steps: establishing an environment model, which summarizes the physical characteristics of the environment, and designing a path planning algorithm, which satisfies the application requirements. There are several common models to describe the environment. For example, the grid method divides the physical world into regular grids and uses coordinates to indicate the location of entities. The Voronoi diagrams method abstracts the environment as a mathematical geometric figure and divides it into different areas based on distance. At present, many path planning algorithms have emerged, such as Dijkstra algorithm [6], A* algorithm [7], D* algorithm [8], which search for the shortest path based on the global information. However, there are some defects, such as high complexity, poor flexibility, and unstable performance. Some other intelligent simulation algorithms imitate organisms in nature and learn from their structure, group behavior or evolutionary mechanism, such as Ant Colony Optimization (ACO) [9], Particle Swarm Optimization (PSO) [10], Genetic Algorithm (GA) [11]. Although they have made great progress in flexibility, they are facing other challenges, local optima, time consumption, computational burden, and so on.

Recently, artificial intelligence and data-driven applications have become the mainstream trend of modern industry [12,13]. With the rise

* Corresponding author.
E-mail address: ximeng@tju.edu.cn (M. Xi).

https://doi.org/10.1016/j.dcan.2022.05.005
Received 15 September 2021; Received in revised form 21 April 2022; Accepted 10 May 2022
Available online xxxx

Please cite this article as: J. Yang et al., A digital twins enabled underwater intelligent internet vehicle path planning system via reinforcement learning and edge computing, Digital Communications and Networks, https://doi.org/10.1016/j.dcan.2022.05.005

of the concept of Industry 4.0, which advocates exploiting information and digitization to promote the industrial revolution, we resort to emerging digital technologies to solve the current dilemma [14,15]. Digital twins is a major digital technology and presents great superiority in industrial manufacturing [16], engineering control [17], smart city [18] and other fields. It constructs a virtual digital space that completely replicates the real world so that events can be simulated in it. The virtual space of digital twins reduces the gap between the simulation and the real world, so it is more complete and reliable. Edge computing is another burgeoning technology, in which the response services are provided by terminals close to the data source. Thus, edge computing is conducive to promoting digital implementation rapidly, efficiently, and has good prospects in Industrial Internet of Things (IIoT) [19], resource scheduling [20], intelligent terminals [21,22], etc. In addition, a new machine learning paradigm, reinforcement learning, has attracted our attention with its good performance in robot control [23], resource allocation [24], intelligent transportation [25], etc. It imitates the way human beings have mastered skills from scratch, and has good strong learning capabilities, flexible adaptability, and convenient realization. These emerging digital technologies and methods provide us with a new solution to the intelligent underwater vehicle path planning system.

In the industrial practice, the underwater intelligent internet vehicles are distributed to specific sea areas to autonomously detect marine information and are controlled by the central cloud as Fig. 1. To plan the optimal path for each AUG, there are many challenges.

1. Environmental completeness. The motion of AUG is largely dependent on ocean conditions, which is a combined result of multiple factors and is difficult to predict, such as the currents and waves. Under extreme conditions, AUG may be damaged or lost, causing economic losses. It is of practical significance to completely describe the ocean environment and then simulate the planned route. However, most of the existing work mainly uses simple mathematical function models, which lack authenticity and reliability.
2. Communication interference. Considering the limitations of physical constraints and economic benefits, the underwater intelligent internet vehicles usually adopt the acoustic signal for information transmission. But the acoustic signal is easily affected by the background noise and echoes, which cause serious interference to the information transmission between the terminal AUG and the central cloud.
3. Algorithm flexibility. The distinguishing feature of the ocean environment is uncertainty. Traditional path planning algorithms rely on exact environmental models. When environmental elements change, they cannot adapt and even cause performance degradation. The uncertainty of the marine environment puts forward higher requirements for the flexibility of the underwater intelligent vehicle path planning algorithm.

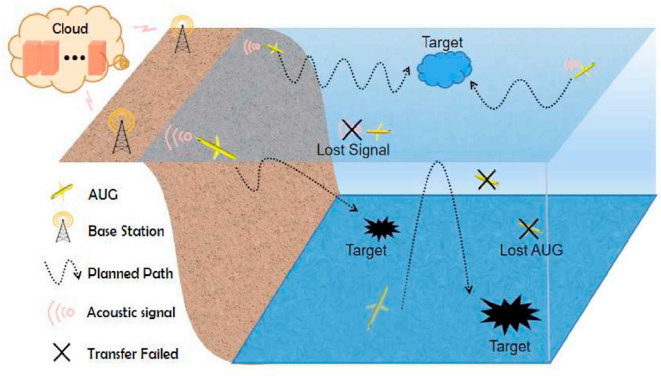

Fig. 1. AUG path planning controlled by cloud.

To solve these problems, we propose a digital twins enabled underwater intelligent internet vehicle path planning system, which exploits edge computing and reinforcement learning. First of all, it integrates the real ocean information from sensors, including topography, currents, etc., and uses the grid method to construct a completely virtual space, which is a digital mapping of the real ocean environment. Then we design a motion function to describe the simulated motion of underwater vehicles at the combined action of dynamic system and ocean conditions. The simulation environment models are deployed in the terminal vehicles to control the process of autonomous path planning through edge computing. A closed-loop model responds to sensor data to control the simulation process. Finally, we design the value-based reinforcement learning algorithms to plan the optimal path. The well-designed reward function utilizes the relative position information and the current characteristics to guide the underwater vehicles to move quickly towards the target points.

The main contributions of our work are as follows.

1. We construct a virtual digital space with the sensor data, which is a mapping of the real ocean. A motion function describes the motion of the AUG under the combined action of the internal dynamic system and the external environment. This complete digital virtual space provides a real and reliable environment for underwater intelligent internet vehicle path planning algorithm.
2. A data-driven response is deployed on the terminal AUG, which is responsible for regulating data collection, processing, modeling, and planning. There is no need for centralized control of the cloud, which avoids transmission loss and improves the efficiency of the path planning system.
3. We propose a flexible AUG path planning method using the value-based reinforcement learning algorithm. The state space integrates location information and sensor data, and sufficient feature input effectively improves the generalization ability of the network. The reward function combines time and space information, which speeds up the learning rate and improves accuracy.

The rest of this paper is organized as follows. We introduce the background knowledge in Section 2. The methodology of our system is explained in Section 3. Section 4 analyzes and discusses the experiments in detail. In Section 5, we introduce related work and summarize our work in Section 6.

2. Background knowledge

2.1. Underwater intelligent internet vehicle path planning

AUG is a new type of ocean observation platform equipped with a variety of sensors for collecting basic ocean data, mapping seabed topography, and detecting underwater resources. The remarkable feature of AUG is low energy consumption, which moves under the joint action of the internal power system and external environment. It obtains the driving force by adjusting the center of gravity and net buoyancy to float and dive alternately, so that its path is zigzag. The AUG communicates with the ground base station through the acoustic signal, receives the control signal from the cloud, and adjusts the attitude according to the path planned as Fig. 1.

The environment plays a pivotal role, especially the currents, which provide external impetus. Ideally, following proper currents will quickly reach the target point; conversely, extreme conditions will cause equipment damage or even loss, resulting in economic losses. Therefore, the simulation of AUG path planning is of great significance, which can find the optimal path and avoid the extreme situation. Existing studies usually utilize simple models to build the marine environment. For example, the typical two-dimensional current models $V(v_x, v_y)$, where

$$v_y(P) = \lambda \frac{x - x_0}{2\pi(P - P_0)^2}\left[1 - e^{-\left(\frac{P-P_0}{\xi}\right)^2}\right] \tag{1}$$

$$v_x(P) = -\lambda \frac{y - y_0}{2\pi(P - P_0)^2}\left[1 - e^{-\left(\frac{P-P_0}{\xi}\right)^2}\right] \tag{2}$$

$P(x, y)$ and $P_0(x_0, y_0)$ represent the two-dimensional space and the center position of the current vertex. λ and ξ are the hyper-parameters to adjust the strength and radius of the current. The mathematical function fitting ocean current ignores the physical constraints in the real scene. The simulation environment from the single mathematical models is not complete and lacks representation and reliability, resulting in a large gap with practical application. Therefore, we introduce the digital twins into the ocean environment modeling to map the physical world, which not only constructs a more reliable simulation environment, but also provides real data support for path planning algorithm.

2.2. Reinforcement learning

Based on the idea of trial and error, reinforcement learning constructs an agent, which imitates the human learning mechanism. Good behavior obtains positive feedback, reward; bad behavior receives negative feedback, punishment. In particular, the reward is designed to be task-related. Then the agent learns towards the direction of increasing mathematical expectation of cumulative discount reward, so as to finally complete the task. Reinforcement learning can be represented by the Markov Decision Process model (MDP). It is a mathematical model of sequential decision-making, which is used to simulate the random strategies and benefits that an agent can achieve in a Markov environment. There are two basic components: agent and environment, and five elements: status, action, state transfer probability, reward, discount factor, which can be denoted by $\langle S, A, P, R, \gamma \rangle$. The whole process can be summarized as follows. The environment gives the initial state s_0; the agent executes the action a_0; then the environment transfers to the next state s_1 according to the probability P and a_0, and gives the feedback r_1; and repeats the above process until the end state s_T, which means that the AUG has successfully reached the target point. Such a process is called a round and forms a trajectory, expressed as $\tau(s_0, a_0, r_0, s_1, a_1, r_1, \ldots s_T)$, in which (s_t, a_t, r_t, s_{t+1}) is called a transition. Just as human beings learn from past experience, reinforcement learning has an Experience Replay $\mathcal{D}$ to store the historical transition, which is also called experience. The corresponding discount reward is

$$G_t = \sum_{i=t}^{N} \gamma^{i-t} r_t \tag{3}$$

where N is the maximum number of states and γ is the discount factor. During the training process, the agent updates the parameters in the direction of maximizing the discount reward. Compared with the traditional path planning method, the mechanism of reinforcement learning ensures an end-to-end learning process and outstanding flexibility.

3. Methodology

3.1. Virtual digital space

Based on digital twins, we construct a virtual digital space, which is the mapping of the real physical world, and then carry out the motion simulation of AUG in this virtual space. The ocean is a comprehensive and complex three-dimensional environment, including seasons, climate, ocean currents, topography, etc., which can be detected by sensors attached to the terminal AUG. The study sea area is $142°E - 142.5°E$, $11°N - 11.5°N$, $-10000\ M$ below the sea-surface. We use the grid method to characterize the relevant factors and visualize this virtual digital space as Fig. 2, including the corresponding currents, topography, and other observation data. The blue arrows represent the currents, they point to the directions of the current, and the lengths indicate the current values. In practical applications, the virtual digital environment can be updated in real-time according to the sensor data carried on the AUG. In the virtual model, the ocean current presents the following characteristics. With the rise and fall of the terrain, the current also presents the characteristics of stratification. In addition, as the depth increases, the current value will be significantly smaller. Digital twins realizes the mapping between virtual digital space and real space, especially using real data. Therefore, compared with the traditional methods, our system shows great advantages in the representation and reliability of the environment, which is the basis of the subsequent path planning algorithm.

3.2. Terminal closed-loop interaction

Limited by poor communication conditions and network congestion, cloud centralized control greatly reduces the efficiency of AUG path planning. Edge computing migrates computing burdens from the cloud to terminal devices, which utilize the local information. Therefore, based on edge computing, we design a closed-loop interactive control model, in which the data collection, processing, modeling, and planning are integrated into the terminal AUG as Fig. 3. A virtual digital ocean environment space is constructed based on the ocean information data collected by the sensors mounted on the AUG. The virtual space includes some basic elements, starting point, present position, target point, and currents. Subsequently, AUG can implement a path planning algorithm to find an optimal path. In the terminal closed-loop interaction, the agent converts the observation information into the input state value s of the reinforcement learning algorithm, expressed by

$$s = (P - P_{Goal}, C_p) \tag{4}$$

The state s is a six-dimensional continuous vector, in which $P(x, y, z)$ is the present position, P_{Goal} is the target point, and $C_P(c_x, c_y, c_z)$ is the current value of position P. Then, the agent issues motion control instructions, action a. Considering the motion characteristics of the AUG, undulating zigzag shape, the action space $a = [a^1, a^2] \in \mathcal{A}$ is divided into two layers. a^1 is a four-dimensional discrete vector indicating the direction, along the longitude or latitude. a^2 is a two-dimensional discrete vector indicating the floating and diving. The environment receives the actions given by the algorithm and gives feedback including observations and rewards. The reward function R is closely related to the specific task. To reach the target point quickly, R contains time, distance and goal reward, as

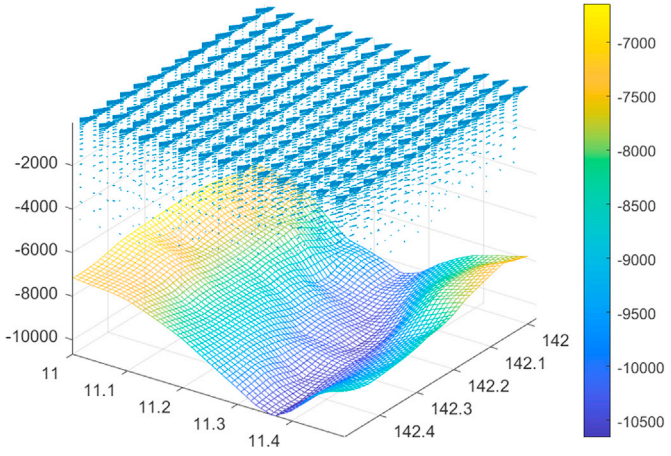

Fig. 2. The visualization of ocean environment model based on digital twins.

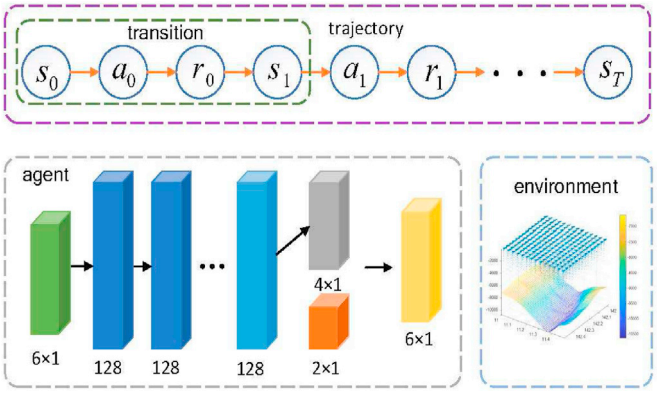

Fig. 3. The closed-loop interaction integrated on the terminal AUG.

$$R = \varpi_1 r_T + \varpi_2 r_D + r_{GOAL} \tag{5}$$

1. Time Reward r_T: MDP can be regarded as a sequential decision problem. During the interaction between the agent and the environment, the AUG conducts the action, then the environment transmits to the next state and gives feedback. Such a process forms a step and repeats until arriving at the target point. Each step is performed at a fixed time interval, so that the number of steps reflects the time of simulation.
2. Distance Reward r_D: The goal of the path planning task is to guide the AUG to the target point, so the distance between the present position and the target point is an important index to guide learning.
3. Goal Reward r_{GOAL}: The ultimate goal of reinforcement learning is to obtain the maximum cumulative discount reward. As mentioned above, the task of path planning is to reach the target point. Therefore, the target point should be the most attractive to the agent, and reaching the target point will be given a huge positive reward r_{GOAL}.

This closed-loop interaction is integrated on the terminal AUG based on edge computing, which significantly reduces the transmission cost of communication with the cloud. In addition, the low-delay of information meets the real-time requirements of path planning.

3.3. Path planning algorithm

For the terminal AUG, it is necessary to plan an optimal path to the destination combined with the sensor data. The traditional path planning algorithm is highly dependent on the environment, resulting in poor flexibility. When the environment changes, it needs to rebuild the map, which is time-consuming and laborious. Therefore, with the advantage of reinforcement learning, we design a path planning algorithm based on DQN [26]. It's an end-to-end way of learning, updating using the experience from the Experience Replay. A neural network outputs the predictive value of the action directly. As we mentioned above, the input is a six-dimensions continuous vector, and the output is a discrete vector. Therefore, we choose the fully connected network, followed by the ReLU activation function, which has a good nonlinear fitting ability for the state transition relationship of the environment, as

$$\begin{bmatrix} \theta_1^{k+1} \\ \theta_2^{k+1} \\ \vdots \\ \theta_{n^{k+1}}^{k+1} \end{bmatrix} = \begin{bmatrix} \vartheta_{11}^k & \vartheta_{12}^k & \dots & \vartheta_{1n^k}^k \\ \vartheta_{21}^k & \vartheta_{22}^k & \dots & \vartheta_{2n^k}^k \\ \vdots & \vdots & \vdots & \vdots \\ \vartheta_{n^{k+1}1}^k & \vartheta_{n^{k+1}2}^k & \dots & \vartheta_{n^{k+1}n^k}^k \end{bmatrix} \begin{bmatrix} \theta_1^k \\ \theta_2^k \\ \vdots \\ \theta_{n^k}^k \end{bmatrix} + \begin{bmatrix} \varsigma_1^k \\ \varsigma_2^k \\ \vdots \\ \varsigma_{n^k}^k \end{bmatrix} \tag{6}$$

$$\theta^{k+1} = \varpi^k \theta^k + \sigma^k \tag{7}$$

$\theta^k = (\theta_1^k, \theta_2^k, \dots, \theta_{n^k}^k)^T$ denotes n^k neuronal nodes of layer K, and $\theta^{k+1} = (\theta_1^{k+1}, \theta_2^{k+1}, \dots, \theta_{n^{k+1}}^{k+1})^T$ denotes n^{k+1} neuronal nodes of layer $K+1$. ϖ^k is a matrix of n^{k+1} times n^k, and $\sigma^k = (\varsigma_1^k, \varsigma_2^k, \dots, \varsigma_{n^k}^k)^T$ is the weighting bias term of neuron nodes, as a nonlinear factor.

The specific flow of the algorithm is as Algorithm 1. Firstly, we randomly initialize two fully connected neural networks, Current network $N_C(s, a; \theta_C)$ and Target network $N_T(s, a; \theta_T)$, $\theta_T = \theta_C$, and establish a Experience Replay $\mathcal{D}$. Secondly, in each round, the virtual digital space is established, and the closed-loop controls the path simulation in terminal AUG. Based on the reinforcement learning, the agent selects action and the environment moves to the next state and gives reward. A simulation round interaction information forms a trajectory $\tau(s_0, a_0, r_0, s_1, a_1, r_1, \dots s_{target})$, and pieces of experience (s_t, a_t, r_t, s_{t+1}) are stored into the Experience Replay. Thirdly, the Current network updates by a minibatch of experience in the Experience Replay. The target of updating is the estimated value given by the Target network as (8). The parameters of network nodes $\theta^k = (\theta_1^k, \theta_2^k, \dots, \theta_{n^k}^k)^T$ are updated by

$$y_t = r_t + \gamma \max_a N_T(s_{t+1}, a; \theta_T) \tag{8}$$

$$\theta_{C,t+1} = \theta_{C,t} + \alpha[y_t - N_C(s_t, a_t; \theta_{C,t})] \nabla N_C(s, a; \theta_{C,t}) \tag{9}$$

Algorithm 1 Deep Q-Learning based Algorithm

Initialize the Experience Replay $\mathcal{D}$ with the capacity N.
Initialize the Current network $N_C(s, a; \theta_C)$, and the Target network $N_T(s, a; \theta_T)$, $\theta_T = \theta_C$.
for *round* = 1, M **do**
  done = *None*
  while NOT *done* **do**
    Agent excutes a_t in the ocean environment.
    Environment gives r_t, s_{t+1}, and *done*.
    Store the transition (s_t, a_t, r_t, s_{t+1}) into $\mathcal{D}$.
  end while
  while *round* MOD η_C **do**
    Sample minibatch transitions from $\mathcal{D}$.
    Target $y_t = r_t + \gamma \max_a N_T(s_{t+1}, a; \theta_T)$
    $\theta_{C,t+1} = \theta_{C,t} + \alpha [y_t - N_C(s_t, a_t; \theta_{C,t})] \nabla N_C(s, a; \theta_{C,t})$
  end while
  while *round* MOD η_T **do**
    $\theta_T = \theta_C$
  end while
end for

Algorithm 2 Double Q-Learning based Algorithm

Update Current network
Randomly sample minibatch transitions from $\mathcal{D}$.
Estimate the value of the next state-action pair and find the best action $a_{t+1} = \max_a N_C(s_{t+1}, a; \theta_{C,t})$.
Target $y_t = r_t + \gamma N_T(s_{t+1}, a_{t+1}; \theta_T)$
$\theta_{C,t+1} = \theta_{C,t} + \alpha [y_t - N_C(s_t, a_t; \theta_{C,t})] \nabla N_C(s, a; \theta_{C,t})$

Algorithm 3 Dueling DQN based Algorithm

Estimate state-action pair value
Initialize the Current network $N_C(s, a; (\theta_C^s, \theta_C^a))$, and the target network $N_T(s, a; (\theta_T^s, \theta_T^a))$, $\theta_T^s = \theta_C^s, \theta_T^a = \theta_C^a$.
Estimate the state value $V(s; \theta_T^s)$
Estimate the action value $V(s, a; \theta_T^a)$
Calculate the value of the next state-action pair $V(s, a; (\theta_T^s, \theta_T^a)) = V(s; \theta_T^s) + V(s, a; \theta_T^a) - \frac{1}{|A|} \sum_{a \in A} V(s, a; \theta_T^a))$
Find the best action $a_{t+1} = \arg\max V(s, a; (\theta_T^s, \theta_T^a))$

There are two neural networks in DQN. The Current network

makes decisions, and the Target network gives updating targets. However, the Target network always chooses the action with maximum value, which inevitably leads to overestimation. Therefore, Van et al. [27] improved it and proposed the Double Q-learning algorithm, which does not select the next optimal action by the Target network, but replaces it with the Current network as Algorithm 2. The separation of action selection and value estimation ensures the independence of updating targets. The two networks provide the action and the estimated value respectively, which further ensure the stability of learning. Moreover, there is another improvement in network structure, Dueling DQN [28]. It outputs two branches, respectively estimating the next state and action values as Algorithm 3. The separation of the action value and the state value estimations accurately reflects the influence of action selection. This independent network structure further improves the accuracy of value estimation and speeds up the training. The Algorithm 4 Double Dueling DQN combines these two improvements. For one thing, the next action is determined by the Current network; for another thing, both the Current network and the Target network are split into two branches to estimate the state value and action value, respectively. Therefore, Double Dueling DQN inherits the advantages of both at the same time.

Algorithm 4 Double Dueling DQN based Algorithm

Initialize the Experience Replay $\mathcal{D}$ with the capacity N.
Initialize the Current network $N_C(s,a;(\theta_C^s,\theta_C^a))$, and the Target network $N_T(s,a;(\theta_T^s,\theta_T^a))$, $\theta_T^s=\theta_C^s, \theta_T^a=\theta_C^a$.
for *round* $= 1, M$ **do**
  $done = None$
  while NOT *done* **do**
    $a_t = \begin{cases} \text{random a}, & \delta \\ \arg\max N_{C,t}(s,a;(\theta_{C,t}^s,\theta_{C,t}^a)), & 1-\delta \end{cases}$
    Agent excutes a_t in the ocean environment.
    Environment gives r_t, s_{t+1}, and *done*.
    Store the transition (s_t, a_t, r_t, s_{t+1}) into $\mathcal{D}$.
  end while
  while *round* MOD η_C **do**
    Randomly sample minibatch transitions from $\mathcal{D}$.
    Estimate values $V(s_t;\theta_C^s)$, $V(s_t,a_t;\theta_C^a)$
    Calculate the value of the next state-action pair $V(s_t,a_t;(\theta_C^s,\theta_C^a)) = V(s_t;\theta_C^s) + V(s_t,a_t;\theta_C^a) - \frac{1}{|A|}\sum_{a\in A} V(s_t,a_t;\theta_C^a))$
    Find best action $a_{t+1} = \arg\max V(s_t,a_t;(\theta_C^s,\theta_C^a))$
    Target $y_t = r_t + \gamma N_T(s_{t+1},a_{t+1};\theta_T)$
    $(\theta_{C,t}^s,\theta_{C,t}^a) = (\theta_{C,t}^s,\theta_{C,t}^a) + \alpha[y_t - N_C\left(s_{t+1},a_{t+1};(\theta_{C,t}^s,\theta_{C,t}^a)\right)]\nabla N_C\left(s,a;(\theta_{C,t}^s,\theta_{C,t}^a)\right)$
  end while
  while *round* MOD η_T **do**
    $\theta_T^s=\theta_C^s, \theta_T^a=\theta_C^a$
  end while
end for

4. Experiments

4.1. Basic settings

The study sea area is $142°E-142.5°E$, $11°N-11.5°N$, -10000 m below the sea-surface, and the data sets are download from the National Marine Data Center. Based on the digital twins, we construct a virtual space of a 3D ocean environment to map the real sea area. We apply the grid method to divide the 3D ocean environment into a grid space of 40 × 40 × 40, in which the agent can explore and plan the path. The total number of rounds of training is 5000, and the maximum number of exploration steps per round is 1000. For action selection, the agent has the probability δ of random action, which encourages to explore more possibilities and avoids falling into local optimization. With the increase of the number of rounds, agent accumulates pieces of experience and learns some policies. There is a problem with exploration and exploitation in reinforcement learning. Exploration means that the agent takes a random action to find more unknown situations; exploitation refers to choosing the action based on the historical experience. More exploration will reduce the learning rate, but more exploitation will cause local optimization. Therefore, we design an exploration-decay strategy to balance this dilemma. Specifically, with the increase in experience, the probability of exploration decreases as

$$\delta_t = \delta_T + (\delta_0 - \delta_T)e^{\left(\frac{t}{\omega}\right)} \tag{10}$$

in which δ_0 is the maximum exploration, δ_T is the minimum exploration, and ω is the adjusted frequency. t is a global controller, which increases cumulatively throughout the training process.

We implement these four reinforcement learning path planning algorithms on Ubuntu 20.04 in python 3.7. They have the same default parameters. The capacity of Experience Replay is 2^{17}, reward discount factor $\gamma = 0.99$, learning rate $\alpha = 10^{-4}$.

We use the learning objective, reward of every round in (5), to measure the performance of the algorithms. We use the number of steps and the length of the path as evaluating indicators to compare paths. Because of the fixed time interval for step of every interaction, the number of steps reflects the time consumption of simulation. For a trajectory, the historical positions are P_0, P_1, P_2 … P_T, so the length of this path is calculated as

$$Length = \sum_{i=o}^{T-1} \|P_i - P_{i+1}\|_2 \tag{11}$$

4.2. Neural network parameters

We design a reinforcement learning based path planning algorithm, and use the deep neural network to realize the end-to-end training. The parameters of neural network play an important role in the performance of the algorithm. Appropriate parameters can maximize the effect of the algorithm. Therefore, it is necessary to determine the value of network parameters first. The details are as follows.

4.2.1. Network width

As mentioned above, the input of the neural network is a six-dimensional continuous vector, but the output is a discrete vector. In essence, deep reinforcement learning is to use a neural network to quickly fit the optimal strategy of the agent in the state transition of an unknown environment. We choose the fully connected network to characterize the mapping relationship between the input and the output. Deep layers of the neural network increase the training cost and time consumption. Usually, the neural network of 3–5 layers is suitable. We choose a three-layer fully connected neural network. The width of the network is the number of neurons of a single layer. The more neural nodes, the stronger the nonlinear ability and characterization ability of the features. However, more nodes will increase the complexity of the network and the difficulty of training. Appropriate network width can not only represent and transfer information completely, but also reduce the training complexity. We compare four different network widths. Fig. 4 records the training results. The small width can not completely summarize the effective input characteristics, so that the neural network can not converge as Fig. 4(a) and (b). When the network width increases to 64, the network has sufficient expression ability. With the increase of training rounds, the reward gradually converges and stabilizes at the maximum value. However, when the network width continues to increase, the complexity increases, resulting in large variance and unstable performance as Fig. 4(d). Therefore, the optimal width of the network is 64, which is better suitable for our task.

4.2.2. Minibatch size

In the process of training, the agent periodically samples a minibatch of experience from the Experience Replay to calculate the target values for gradient updating of neural network parameters. The number of samples affects the speed of training and convergence, so it is a critical

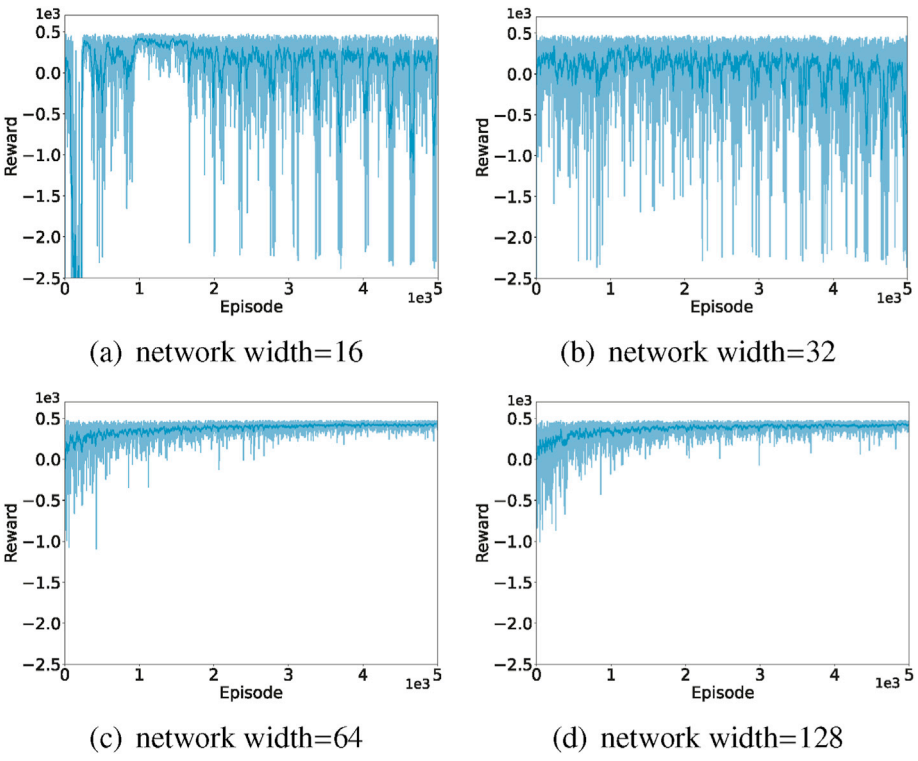

Fig. 4. The results of network width.

parameter for training speed. Small size leads to a decrease in the total number of training samples, which makes the neural network unable to extract enough useful information from limited samples. On the contrary, a large size increases the burden of the network, so that the neural network can not deal with many mappings at the same time, causing a large variance. Therefore, the size of minibatch should match the

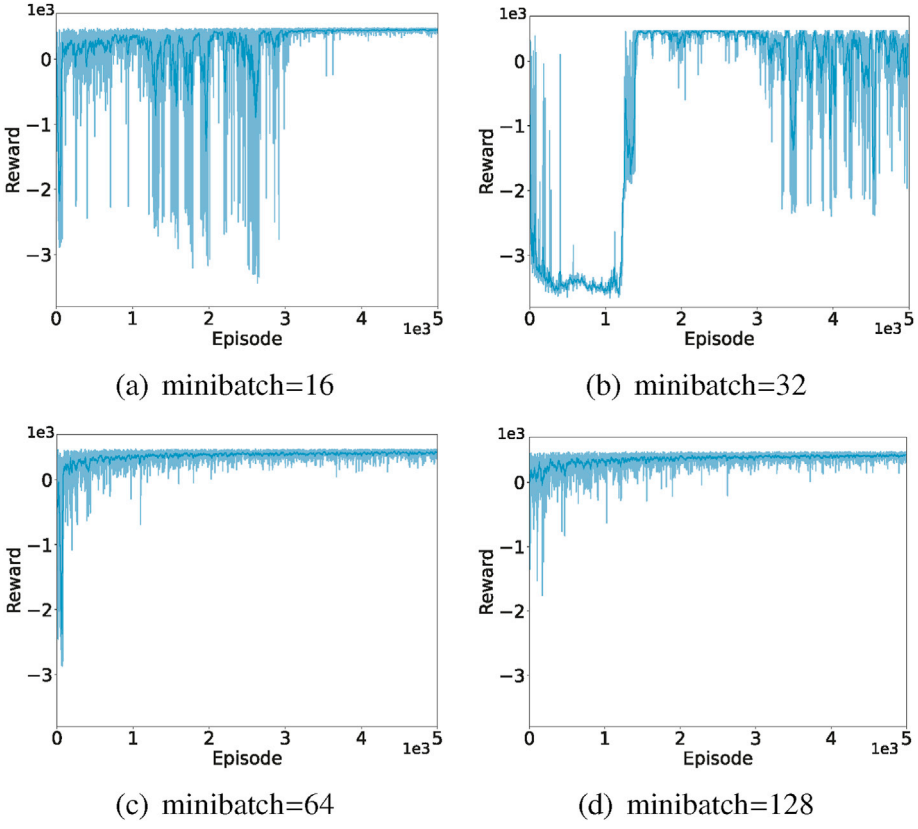

Fig. 5. The results of minibatch.

characterization ability of the network. We compare four different sizes in Fig. 5. There is randomness in the action exploration of the agent, and the probability of exploration is large in the early stage. Fig. 5(a) and (b) show two types of failure due to few training samples. In Fig. 5(a), the agent initially explores a better path randomly and obtaines a high reward. However, due to the small number of samples and the large gap between samples, the training result is unstable. On the contrary, large samples are incompatible with the characterization ability of the network, which will cause large variance as shown in Fig. 5(d). On balance, we choose the minibatch size of 64 that matches the representation ability of the neural network.

4.2.3. Update time

Update time represents the number of training for a fixed strategy. For each update, the agent will randomly collect a minibatch of experience for training. Before and after the update, two agents with different neural network parameters are used to collect samples. When the number of updates is not enough, the strategies of the two agents are similar, which will cause the generated trajectory samples to be highly repetitive, the early learning speed is slow, and even the appropriate path cannot be explored. When there are too many update times, the difference between the two agents increases, resulting in an increase in the distribution divergence of the trajectory, and it is impossible to learn beneficial knowledge from the previous samples. As a result, the learning target is unstable and the variance is too large. We compare four different update times in Fig. 6. Fig. 6(a) and (b) have the similar results. Due to the small difference between strategies, the network parameters are not stable in a good strategy, so there is a large jitter in the training process. Compared with Fig. 6(d) and (c), which has smaller variance and faster learning rate, so we choose 128 update times.

From these comparative experiments, we determine the optimal network parameters, which ensure the maximum performance of the neural network.

4.3. Effectiveness experiment

Based on the optimal parameters, we implement four algorithms to verify the effectiveness of the reinforcement learning path planning algorithm. For convenience, we uniformly use the grid coordinates in the virtual digital space. The starting point is (0, 0, 40) and the target point is (40,40,40). Fig. 7 shows the results of the reward during the training. Due to the mechanism of exploration and exploitation in reinforcement learning, random actions may achieve poor results, so the reward curve is accompanied by deviation. In the beginning, the four algorithms can randomly explore the path from the starting point to the target point, and obtain a larger cumulative reward. However, overestimation brings a bias to the learning of Deep Q-Learning, which results in drastic performance degradation. Although the Double Q-Learning eases the overestimation to a certain extent, the simultaneous estimation of the state and action leads to the inaccuracy and insensitivity of the value

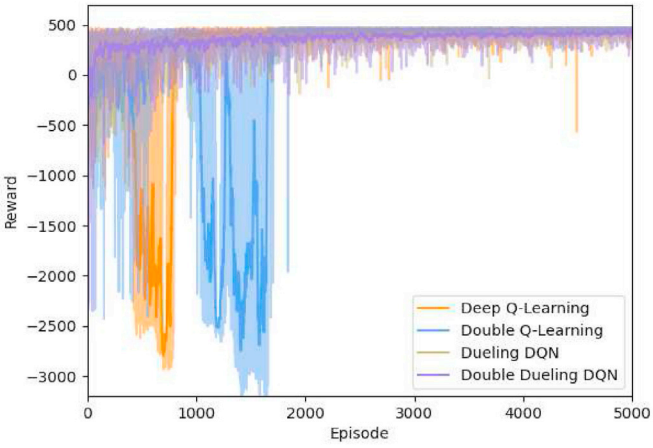

Fig. 7. The result of the training process.

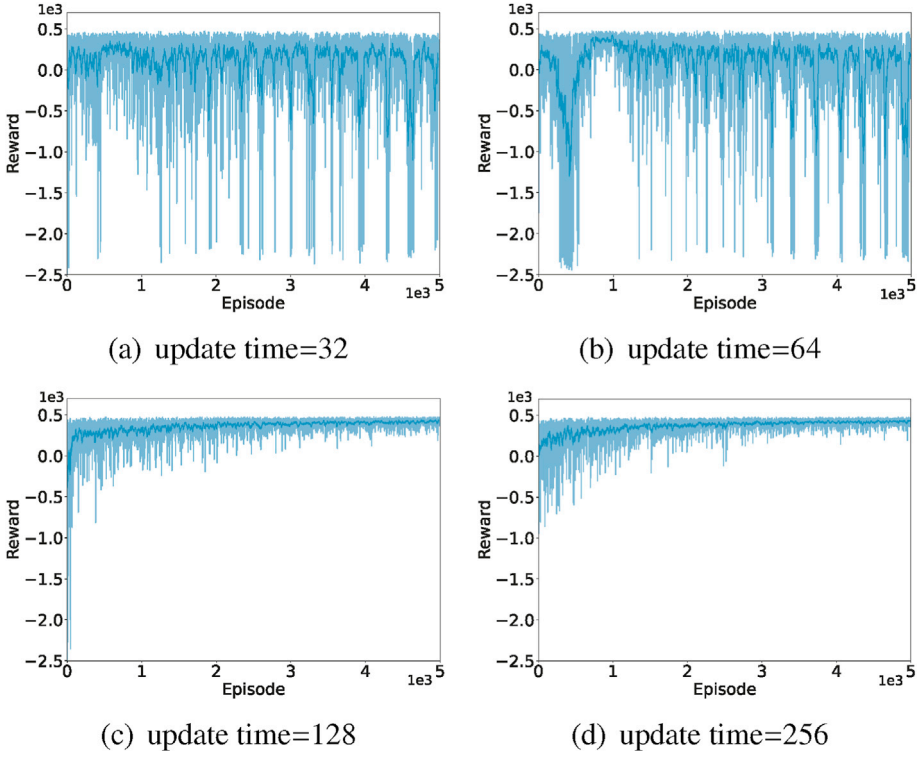

Fig. 6. The results of update times.

estimation, and performance degradation occurs in the later stage. Dueling DQN and Double Dueling DQN present good performance, and the reward values not only converge quickly, but also remain stable. Comparing Double Q-Learning and Double Dueling DQN, it is clearly confirmed that the separation of network structure can significantly improve the effect, with fast and stable convergence. Comparing Dueling DQN and Double Dueling DQN, the two training curves are similar, which shows that the effect of the double network is not obvious. The separation of network structure can also alleviate the problem of overestimation. The reason is that the separation of state value and action value effectively improves the accuracy of estimation. Fig. 8 visualizes the simulation path of the Double Dueling DQN in the virtual space. In this virtual grid space, the blue arrows represent currents; the green and red dots represent the starting and target points respectively. The blue dots denote the positions and form a zigzag path.

4.4. Different ocean environments

The remarkable characteristic of the ocean environment is uncertainty. The movement of AUG will be disturbed by many factors, among which the current has a great influence. The position of AUG is affected by the current intensity and its own strategy. Therefore, when studying the motion in different ocean environments, it is of great significance to analyze the influence of current intensity. We design five different intensities to compare the performance of the algorithms. The intensity changes from 0.7 to 1.2, which indicates the relative velocity of the current and itself.

Fig. 9 records the results of the step. With the increase of the current intensity, the single-step motion range of AUG increases, so the total number of steps to reach the target point should be reduced, showing a downward trend. Deep Q-Learning and Double Q-Learning are rising as a whole, indicating that they can not adapt to large intensity, resulting in a decline in performance in the later stage. Dueling DQN and Double Dueling DQN are relatively stable, and the latter can maintain a relatively stable effect under the condition of high intensity. Fig. 10 records the results of length. Although the increase in intensity expands the range of single-step motion and reduces the total number of steps, the relative total path length will not change significantly. Therefore, a good algorithm should maintain a stable path length. Deep Q-Learning and Double Q-Learning suffer a large margin, but Dueling DQN and Double Dueling DQN are more stable, and they are the better choices.

5. Related work

5.1. Digital twins

Recently, digital twins have attracted the attention of many institutions and scholars, and related researches have been carried out. For one thing, digital twins have gradually led to the reform of the mode of production. Tao et al. [29] comprehensively summarized the state-of-the-art of the main applications of digital twins in the industry, including its key components and the latest progress. Then, they proposed the framework of Digital Twin-driven Product Design (DTPD) [30]. To realize the share of simulation models, Hatledal et al. [31] proposed an open-source co-simulation framework based on Functional Mock-up Interface (FMI) and Remote Procedure Call (RPC) technologies, which can be implemented independently of language and platform. Mukherjee et al. [32] introduced digital twins in 3D printing manufacturing, which reduced attempts, diminished defects, and shortened time. For another thing, digital twins are also coming into our modern life. The smart city is one of the most important applications. For example, Kent et al. [33] applied digital twins to city planning, proposed a platform, City Blocks, based on virtual reality, and tested it with the public. Francisco et al. [34] discussed on the base of the digital twin–enabled urban energy management platforms to realize smarter energy management by smart meter data streams. In Ref. [35], authors

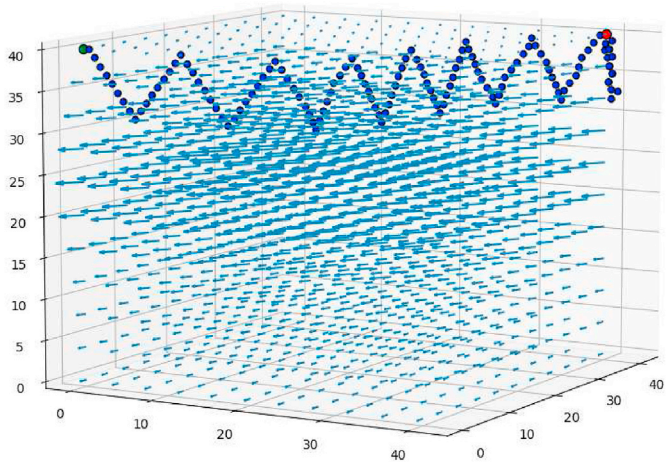

Fig. 8. The simulation path in digital virtual space.

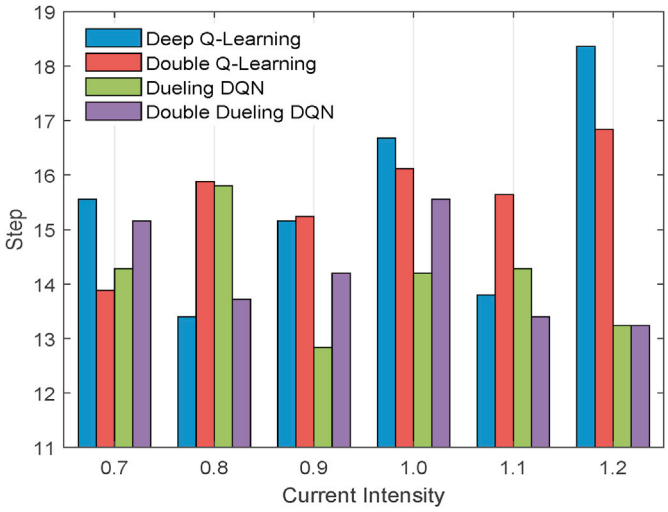

Fig. 9. The effect of intensity on step.

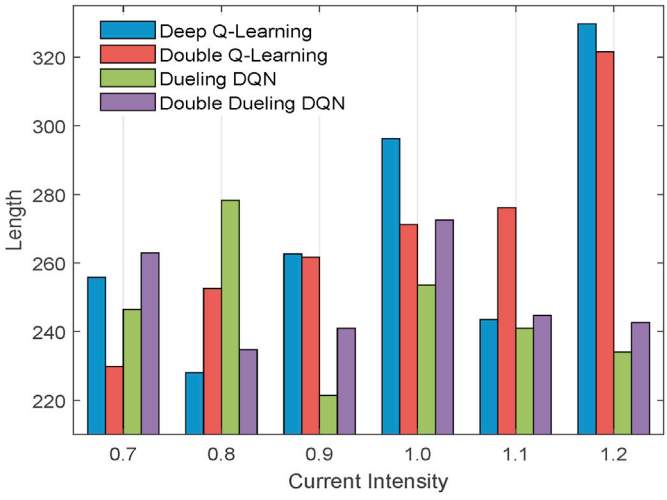

Fig. 10. The effect of intensity on length.

introduced a comprehensive information and communication technology platform to better evaluate and develop the design, construction, and performance of residential buildings. The application of digital twins is

constantly emerging around us and has promoted the process of industrial digitization. It narrows the gap between the real world and the virtual world, which is the focus of many industrial applications.

5.2. Edge computing

In the digital society, the development of information technology has brought exponential growth in data. Especially in the IoT and IoUT, millions of terminal nodes produce a large amount of data every second, which causes serious computing pressure on the central cloud and capacity demands on hardware equipment. As a new technical solution, edge computing migrates data computing and storage from the centralized cloud to the distributed terminal devices, which effectively alleviates the data dilemma [36]. Some key technical issues of edge computing are also being concerned. For example, Li et al. [37] investigated the problem of Content Caching (CC) and User Association (UA) for edge computing. Goudarzi et al. [38] studied the deployment of applications to save the running time and energy consumption under the condition of limited computing resources. Xia et al. [39] were concerned with the Mobile Edge Computing (MEC) and Energy Harvesting (EH) and proposed an online distributed optimization algorithm to enable efficient management of computing resources. Li et al. [40] studied the Edge Data Integrity (EDI) problem. They designed an approach to examine the cache data and locate the damaged data. Chen et al. [41] focused on the practical and challenging situations, and they proposed an asynchronous federated learning scheme that effectively solves the heterogeneous resources and unstable communications. Based on edge computing, our system completes autonomous path planning in the terminal AUG, which further reduces the computing pressure in the cloud and reduces the interference caused by unstable data transmission.

5.3. Reinforcement learning

Reinforcement learning is a new machine learning algorithm. At first, reinforcement learning is mainly used in games. For example, AlphaGo [42] based on reinforcement learning algorithm has been able to reach the level of human players through self-learning, or even overcome. In large-scale Real-Time Strategy (RTS) games, reinforcement learning can achieve good performance, such as StarCraft II [43], MOBA [44]. Then reinforcement learning expands to more fields. For example, Ni et al. [45] proposed a dynamic game solution in smart grid security area. Xiao et al. [46] used reinforcement learning in the recommendation system to improve the quality of recommendations and protect user privacy. In addition, there is a Multi-Agent Reinforcement Learning (MARL), of which the number of agents is more than one. Zhang et al. [47] proposed a cooperative MARL algorithm, Policy Gradient Potential (PGP), to find the optimal joint policy with maximum global reward. Chu et al. [48] found the solution for Adaptive Traffic Signal Control (ATSC) in complex urban traffic networks by MARL. Reinforcement learning does not need to provide a large number of prior data just through trial and error learning. Autonomous learning endows the agent with good flexibility. The neural network used in deep reinforcement learning avoids the cumbersome manual feature design by providing an end-to-end learning method. With the above significant advantages, reinforcement learning has been applied in many fields, so we exploit it to provide a stable and optimal path in our system.

6. Conclusion

In this paper, we propose a digital twins enabled path planning system for the underwater intelligent internet vehicle. First of all, it constructs a virtual digital space that maps the real 3D ocean environment through the real sensor data. The simulation environment based on the digital twins narrows the gap with practical industrial application and makes up for the lack of completeness, including all-around ocean information such as topography, currents, and waves. Then, a terminal closed-loop control model based on edge computing is integrated into each underwater vehicle to control the simulation process of path planning. This terminal response closed to the sensing data effectively avoids the interference of acoustic signal transmission and reduces the computing pressure on the cloud. Finally, we design and compare the value-based reinforcement learning algorithms for AUG path planning. The end-to-end learning method has good flexibility and can adapt to the complex and uncertain underwater environment. The well-designed state input and parameter architecture help to accelerate the neural network convergence and find the optimal path. This paper solves some difficulties in the underwater intelligent internet vehicle path planning, which lays a foundation for further industrial realization. In the future, we will exert ourselves to meet more challenges, such as the cooperation between multiple underwater intelligent internet vehicles for more complex underwater tasks.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was partially supported by the National Natural Science Foundation of China (No. 61871283).

References

[1] D. Deng, X. Li, V. Menon, M.J. Piran, H. Chen, M.A. Jan, Learning-based Joint UAV Trajectory and Power Allocation Optimization for Secure IoT Networks, Digital Communications and Networks 8 (4) (2022) 415–421.
[2] Q. Wang, H.-N. Dai, Q. Wang, M.K. Shukla, W. Zhang, C.G. Soares, On connectivity of UAV-assisted data acquisition for underwater internet of things, IEEE Internet Things J. 7 (6) (2020) 5371–5385.
[3] H. Lan, Y. Lv, J. Jin, J. Li, D. Sun, Z. Yang, Acoustical observation with multiple wave gliders for internet of underwater things, IEEE Internet Things J. 8 (4) (2021) 2814–2825.
[4] Y. Zhang, B. Kieft, B.W. Hobson, B.-Y. Raanan, S.S. Urmy, K.J. Pitz, C.M. Preston, B. Roman, K.J. Benoit-Bird, J.M. Birch, F.P. Chavez, C.A. Scholin, Persistent sampling of vertically migrating biological layers by an autonomous underwater vehicle within the beam of a seabed-mounted echosounder, IEEE J. Ocean. Eng. 46 (2) (2021) 497–508.
[5] L. Feng, Z. Lv, G. Guo, H. Song, Pheromone based alternative route planning, Digit. Commun. Netw. 2 (3) (2016) 151–158.
[6] O.A. Gbadamosi, D.R. Aremu, Design of a modified Dijkstra's algorithm for finding alternate routes for shortest-path problems with huge costs, in: 2020 International Conference in Mathematics, Computer Engineering and Computer Science (ICMCECS), 2020, pp. 1–6.
[7] T. Zheng, Y. Xu, D. Zheng, AGV path planning based on improved a-star algorithm, in: 2019 IEEE 3rd Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), 2019, pp. 1534–1538.
[8] H. Huang, P. Huang, S. Zhong, T. Long, S. Wang, E. Qiang, Y. Zhong, L. He, Dynamic path planning based on improved D* algorithms of gaode map, in: 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), 2019, pp. 1121–1124.
[9] C. Zhang, C. Hu, J. Feng, Z. Liu, Y. Zhou, Z. Zhang, A self-heuristic ant-based method for path planning of unmanned aerial vehicle in complex 3-D space with dense u-type obstacles, IEEE Access 7 (2019) 150775–150791.
[10] P. Wu, T. Li, G. Song, UCAV path planning based on improved chaotic particle swarm optimization, in: 2020 Chinese Automation Congress (CAC), 2020, pp. 1069–1073.
[11] V. Roberge, M. Tarbouchi, G. Labonté, Fast genetic algorithm path planner for fixed-wing military UAV using GPU, IEEE Trans. Aero. Electron. Syst. 54 (5) (2018) 2105–2117.
[12] J. Yang, M. Xi, B. Jiang, J. Man, Q. Meng, B. Li, FADN: fully connected attitude detection network based on industrial video, IEEE Trans. Ind. Inf. 17 (3) (2020) 2011–2020.
[13] Y. Wang, J. Wang, W. Zhang, Y. Zhan, S. Guo, Q. Zheng, X. Wang, A survey on deploying mobile deep learning applications: a systemic and technical perspective, Digit. Commun. Netw. 8 (1) (2022) 1–17.
[14] D. Jiang, Y. Wang, Z. Lv, S. Qi, S. Singh, Big data analysis based network behavior insight of cellular networks for industry 4.0 applications, IEEE Trans. Ind. Inf. 16 (2) (2019) 1310–1320.
[15] L. Qiao, S. Dang, B. Shihada, M.-S. Alouini, R. Nowak, Z. Lv, Can blockchain link the future? Digit. Commun. Netw. 8 (5) (2021) 2352–8648.

[16] X. Li, B. He, Y. Zhou, G. Li, Multisource model-driven digital twin system of robotic assembly, IEEE Syst. J. 15 (1) (2021) 114–123.
[17] C. Gehrmann, M. Gunnarsson, A digital twin based industrial automation and control system security architecture, IEEE Trans. Ind. Inf. 16 (1) (2020) 669–680.
[18] Z. Lv, B. Hu, H. Lv, Infrastructure monitoring and operation for smart cities based on IoT system, IEEE Trans. Ind. Inf. 16 (3) (2019) 1957–1962.
[19] W. Qin, S. Chen, M. Peng, Recent advances in industrial internet: insights and challenges, Digit. Commun. Netw. 6 (1) (2020) 1–13.
[20] X. Li, J. Wan, H.-N. Dai, M. Imran, M. Xia, A. Celesti, A hybrid computing solution and resource scheduling strategy for edge computing in smart manufacturing, IEEE Trans. Ind. Inf. 15 (7) (2019) 4225–4234.
[21] J. Chen, S. Chen, S. Luo, Q. Wang, B. Cao, X. Li, An intelligent task offloading algorithm (iTOA) for UAV edge computing network, Digit. Commun. Netw. 6 (4) (2020) 433–443.
[22] Z. Lv, W. Xiu, Interaction of edge-cloud computing based on SDN and NFV for next generation IoT, IEEE Internet Things J. 7 (7) (2019) 5706–5712.
[23] A. Ghadirzadeh, X. Chen, W. Yin, Z. Yi, M. Björkman, D. Kragic, Human-centered collaborative robots with deep reinforcement learning, IEEE Rob. Autom. Lett. 6 (2) (2021) 566–571.
[24] L. Huang, X. Feng, C. Zhang, L. Qian, Y. Wu, Deep reinforcement learning-based joint task offloading and bandwidth allocation for multi-user mobile edge computing, Digit. Commun. Netw. 5 (1) (2019) 10–17.
[25] R. Wang, X. Jiang, Y. Zhou, Z. Li, D. Wu, T. Tang, A. Fedotov, V. Badenko, c, Digital Communications and Networks 8 (1) (2022) 1–17.
[26] M. Volodymyr, K. Koray, S. David, A.A. Rusu, V. Joel, M.G. Bellemare, G. Alex, R. Martin, A.K. Fidjeland, O. Georg, Human-level control through deep reinforcement learning, Nature (518) (2015) 529–533.
[27] H. Van Hasselt, A. Guez, D. Silver, Deep reinforcement learning with double q-learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2016, pp. 1–12.
[28] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, N. Freitas, Dueling network architectures for deep reinforcement learning, in: International Conference on Machine Learning, PMLR, 2016, pp. 1995–2003.
[29] F. Tao, H. Zhang, A. Liu, A.Y. Nee, Digital twin in industry: state-of-the-art, IEEE Trans. Ind. Inf. 15 (4) (2018) 2405–2415.
[30] F. Tao, F. Sui, A. Liu, Q. Qi, M. Zhang, B. Song, Z. Guo, S.C.-Y. Lu, A.Y. Nee, Digital twin-driven product design framework, Int. J. Prod. Res. 57 (12) (2019) 3935–3953.
[31] L.I. Hatledal, A. Styve, G. Hovland, H. Zhang, A language and platform independent co-simulation framework based on the functional mock-up interface, IEEE Access 7 (2019) 109328–109339.
[32] T. Mukherjee, T. DebRoy, A digital twin for rapid qualification of 3D printed metallic components, Appl. Mater. Today 14 (2019) 59–65.
[33] L. Kent, C. Snider, B. Hicks, Early stage digital-physical twinning to engage citizens with city planning and design, in: 2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR), IEEE, 2019, pp. 1014–1015.
[34] A. Francisco, N. Mohammadi, J.E. Taylor, Smart city digital twin–enabled energy management: toward real-time urban building energy benchmarking, J. Manag. Eng. 36 (2) (2020) 04019045.
[35] N. Mohamed, J. Al-Jaroodi, S. Lazarova-Molnar, Leveraging the capabilities of industry 4.0 for improving energy efficiency in smart factories, IEEE Access 7 (2019) 18008–18020.
[36] M. De Donno, K. Tange, N. Dragoni, Foundations and evolution of modern computing paradigms: cloud, IoT, edge, and fog, IEEE Access 7 (2019) 150936–150948.
[37] Y. Li, H. Ma, L. Wang, S. Mao, G. Wang, Optimized content caching and user association for edge computing in densely deployed heterogeneous networks, IEEE Trans. Mobile Comput. 21 (6) (2022) 2130–2142.
[38] M. Goudarzi, H. Wu, M. Palaniswami, R. Buyya, An application placement technique for concurrent IoT applications in edge and fog computing environments, IEEE Trans. Mobile Comput. 20 (4) (2021) 1298–1311.
[39] S. Xia, Z. Yao, Y. Li, S. Mao, Online distributed offloading and computing resource management with energy harvesting for heterogeneous MEC-enabled IoT, IEEE Trans. Wireless Commun. 20 (10) (2021) 6743–6757.
[40] B. Li, Q. He, F. Chen, H. Jin, Y. Xiang, Y. Yang, Auditing cache data integrity in the edge computing environment, IEEE Trans. Parallel Distr. Syst. 32 (5) (2021) 1210–1223.
[41] Z. Chen, W. Liao, K. Hua, C. Lu, W. Yu, Towards asynchronous federated learning for heterogeneous edge-powered internet of things, Digital Communications and Networks 7 (3) (2021) 317–326.
[42] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, et al., A general reinforcement learning algorithm that masters chess, shogi, and go through self-play, Science 362 (6419) (2018) 1140–1144.
[43] O. Vinyals, I. Babuschkin, W.M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D.H. Choi, R. Powell, T. Ewalds, P. Georgiev, et al., Grandmaster level in StarCraft II using multi-agent reinforcement learning, Nature 575 (7782) (2019) 350–354.
[44] D. Ye, G. Chen, W. Zhang, S. Chen, B. Yuan, B. Liu, J. Chen, Z. Liu, F. Qiu, H. Yu, et al., Towards playing full moba games with deep reinforcement learning, Adv. Neural Inf. Process. Syst. 33 (2020) 621–632.
[45] Z. Ni, S. Paul, A multistage game in smart grid security: a reinforcement learning solution, IEEE Transact. Neural Networks Learn. Syst. 30 (9) (2019) 2684–2695.
[46] Y. Xiao, L. Xiao, X. Lu, H. Zhang, S. Yu, H.V. Poor, Deep-reinforcement-learning-based user profile perturbation for privacy-aware recommendation, IEEE Internet Things J. 8 (6) (2021) 4560–4568.
[47] Z. Zhang, Y.-S. Ong, D. Wang, B. Xue, A collaborative multiagent reinforcement learning method based on policy gradient potential, IEEE Trans. Cybern. 51 (2) (2021) 1015–1027.
[48] T. Chu, J. Wang, L. Codecà, Z. Li, Multi-agent deep reinforcement learning for large-scale traffic signal control, IEEE Trans. Intell. Transport. Syst. 21 (3) (2019) 1086–1095.