Knowledge-Based Systems 196 (2020) 105201

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

The autonomous navigation and obstacle avoidance for USVs with ANOA deep reinforcement learning method

Xing Wu [a,b,*], Haolei Chen [a], Changgu Chen [a], Mingyu Zhong [a], Shaorong Xie [a], Yike Guo [a], Hamido Fujita [c,d,e]

[a] School of Computer Engineering and Science, Shanghai University, Shanghai, China
[b] Shanghai Institute for Advanced Communication and Data Science, Shanghai University, Shanghai, China
[c] Faculty of Information Technology, Ho Chi Minh City University of Technology (HUTECH), Ho Chi Minh City, Viet Nam
[d] Andalusian Research Institute on Data Science and Computational Intelligence (DaSCI), University of Granada, Spain
[e] Faculty of Software and Information Science, Iwate Prefectural University (IPU), Iwate, Japan

ARTICLE INFO

Article history:
Received 15 March 2019
Received in revised form 4 November 2019
Accepted 5 November 2019
Available online 21 March 2020

Keywords:
Autonomous navigation
Obstacle avoidance
Reinforcement learning
Unmanned surface vehicle (USV)

ABSTRACT

The unmanned surface vehicle (USV) has been widely used to accomplish missions in the sea or dangerous marine areas for ships with sailors, which greatly expands protective capability and detection range. When USVs perform various missions in sophisticated marine environment, autonomous navigation and obstacle avoidance will be necessary and essential. However, there are few effective navigation methods with real-time path planning and obstacle avoidance in dynamic environment. With tailored design of state and action spaces and a dueling deep Q-network, a deep reinforcement learning method ANOA (Autonomous Navigation and Obstacle Avoidance) is proposed for the autonomous navigation and obstacle avoidance of USVs. Experimental results demonstrate that ANOA outperforms deep Q-network (DQN) and Deep Sarsa in the efficiency of exploration and the speed of convergence not only in static environment but also in dynamic environment. Furthermore, the ANOA is integrated with the real control model of a USV moving in surge, sway and yaw and it achieves a higher success rate than Recast navigation method in dynamic environment.

1. Introduction

Unmanned surface vehicle is a small surface ship with the ability of autonomous planning and navigation, which can accomplish missions such as the environmental perception and target detection under the autonomous mode or manual intervention. The unmanned aerial vehicle (UAV), unmanned ground vehicle (UGV), unmanned underwater vehicle (UUV) and USV are important parts of unmanned system, and their cooperative operations jointly construct holonomic unmanned marine system [1,2]. Once equipped with multiple sensors, communication devices and advanced control devices, USVs will be flexible and intelligent to carry out different missions such as marine detection, water quality measurement and so on.

Different missions will require USVs to be deployed in various marine areas, especially in the harsh and dangerous marine environment for big ships. Thus there is a high demand for autonomous navigation and obstacle avoidance for USVs, which is to find an optimal or approximately optimal route from the starting point to target under certain constraints. This will ensure that USVs could navigate through all obstacles without collisions.

With the theoretical and technical achievements, especially in reinforcement learning and deep learning, the development of unmanned systems [3–5] has been dramatically promoted. Traditional navigation and path planning techniques include graphic method, dynamic window method [6], artificial potential field method and so on. There have been some heuristic path planning algorithms, which include genetic algorithms [7,8] and swarm intelligence algorithms [9–12]. Each kind of approaches has strengths and weaknesses. Traditional methods are easy to fall into traps in complex environments and have lower probability to reach destination with a reasonable route compared with heuristic techniques. Some heuristic methods are slow in speed and unable to detect and avoid obstacles in real time in some cases. Reinforcement learning algorithms are based on rewards and punishments mechanism to improve performances completing the missions. The exploration and greedy policy of reinforcement learning algorithms are especially suitable for path planning in sophisticated environment. The higher the random exploration probability is, the better the obtained navigation routes will be.

Inspired by the theoretical achievements of reinforcement learning and deep learning, ANOA method is proposed for the

* Corresponding author.
E-mail address: xingwu@shu.edu.cn (X. Wu).

https://doi.org/10.1016/j.knosys.2019.105201
0950-7051/

autonomous navigation of USVs. The main contributions of this paper can be summarized as follows:

- With tailored design of state and action spaces and a dueling deep Q-network, a deep reinforcement learning method ANOA is proposed for the autonomous navigation and obstacle avoidance of USVs, which has better performance than deep Q-network (DQN) and Deep Sarsa not only in static environment but also in dynamic environment. Furthermore, a real control model of USVs moving in surge, sway and yaw is integrated with proposed ANOA and a frequently used heuristic approach, Recast navigation. In dynamic environment, ANOA achieves a higher success rate than Recast navigation after ANOA is fully trained.
- A dueling deep Q-network as the deep learning module is proposed to sense the sea environment of USVs for informative feature learning. This dueling deep Q-Network is trained with a variant of Q-learning, whose input is raw pixels and whose output is a value function to estimate future rewards, also known as Q values. The reinforcement learning part interacts with deep learning part by obtaining its Q value estimation to make decisions. The proposed deep learning part could work with a finite space of memory, but its performance is as good as a well optimized Q table does in traditional Q learning.
- For the autonomous navigation and obstacle avoidance of USVs, the pros and cons of different methods are discussed on a simulation platform constructed with open source tools. On the simulation platform, ANOA, DQN and Deep Sarsa are quantitatively evaluated with the reward, loss value and average Q-value not only in static environment but also in dynamic environment. Moreover, the comparison between ANOA and Recast Navigation are quantitatively evaluated with the success rate in dynamic environment.

The contribution on the sustainability and potential replication of the research could be expanded to all unmanned vehicles. That is to say, the proposed ANOA method and the simulation platform for USVs could also be used for UAVs, UGVs and UUVs conditionally.

2. Related work

In the changeable and complex marine areas, USVs have to explore the marine environment by self-navigation to approach destinations safely. To be intelligent and autonomous, USVs should be aware of surrounding obstacles with the help of different types of sensors. With real-time environmental awareness, USVs should be capable of making right decisions without human control or interventions aiming at predetermined destinations.

The path planning has been a popular research topic for years. On one hand, traditional global path planning and real time navigation algorithms [13–15] utilized divided environmental structure space for path exploration. The mode and precision of division directly affect the efficiency of a path exploration. On the other hand, researchers used swarm optimization algorithms [16] and genetic algorithms [17] to improve the effectiveness of path planning. Raboin et al. [18,19] presented a heuristic planning approach on guarding a valuable asset by a team of autonomous unmanned surface vehicles operating in a continuous state–action space. Some heuristic algorithms may require a large amount of iterations to get satisfactory results. Thus they are slow in speed and unable to detect and avoid obstacles in real time, which are not flexible enough for practical applications in dynamic environment. To be specific, slight movements of obstacles and the disturbance of environment information retrieval may lead to failure of these methods. As the complexity of maritime systems is increasing, traditional global path planning models would become unable to meet the challenge.

Recent developments [20–22] of reinforcement learning and deep learning have brought new ideas into addressing the problem of dynamic obstacles, which could provide feasible solutions of real-time control in changeable environments. For example, the Q-Learning [23] with Q-table and deep Q-network (DQN) [24] with deep neural network and other deep reinforcement learning methods [25,26] would be potential keys to these problems. Lowe et al. [27] explored deep reinforcement learning methods for multi-agent domains. They presented an adaptation of actor-critic methods which take action policies of other agents into consideration, which could successfully learn policies that require complex multi-agent coordination. Mnih V et al. [28] proposed the first deep learning model to successfully learn control policies directly from high-dimensional sensory input for reinforcement learning. The model is a convolutional neural network whose input is raw pixels and output are value function estimating future rewards. Wu et al. [29] presented a framework called CISTAR that integrates the widely used traffic simulator SUMO with a standard deep reinforcement learning library RLLab, which helps to learn locally optimal policies (with respect to the policy parameterization) for a variety of objectives such as matching a target velocity or minimizing fuel consumption. Peng et al. [30] took StarCraft combat game as a case study where the task was to coordinate multiple agents as a team to defeat their enemies. They introduced Multi-agent Bidirectionally Coordinated Network (BiCNet) with vectorized extension of actor-critic formulation. These reinforcement learning and deep learning methods have proved their potential in the control of agents in cooperative or competitive environments.

With the help of reinforcement learning and deep learning methods, there are some research findings for the problem of autonomous navigation and obstacle avoidance. Chen et al. [31] presented a decentralized multi-agent collision avoidance algorithm based on a novel application of deep reinforcement learning, which effectively offloads the online computation (for predicting interaction patterns) to an offline learning procedure. Long et al. [32] proposed a novel end-to-end framework to generate reactive collision avoidance policy for efficient distributed multi-agent navigation. Liu et al. [33] proposed an algorithm of dynamic multi-step reinforcement learning based on virtual potential field path planning. Romero-Marti et al. [34] studied the path planning of mobile robots based on reinforcement learning, aiming at the balance between exploration and exploitation. The robot was provided with a topological map of a building floor (environmental map), with which the robot learnt a path from one location to another by means of Q-learning. Huang et al. [35] proposed an approach to navigate the robot from the start location to the target location without collisions with static and dynamic obstacles. They have improved the original Q-learning algorithm in environment modeling, reward function, and the adapted policy to make the robot stay away from obstacles, reduced the probability of collisions, and reached the target as fast as possible. Cheng et al. [36] proposed a concise deep reinforcement learning obstacle avoidance (CDRLOA) algorithm with the powerful deep Q-networks architecture, in which a comprehensive reward function is specifically designed for obstacle avoidance, target approaching, speed modification, and attitude correction. This proposed architecture offered a potential answer to the problem of obstacle avoidance for the underactuated unmanned marine vessel with unknown environmental disturbance. Guo et al. [37] proposed a novel hierarchical path planning algorithm for mobile robots based on A* and reinforcement learning with a structure of two layers. The path obtained by the proposed algorithm is smooth and safe for executing. Buitrago

Martinez et al. [38] proposed a hierarchical reinforcement learning approach for motion planning in mobile robotics, which was validated with robot motion planning tasks in simulation and in an experimental environment.

Aiming at addressing the problem of autonomous navigation and obstacle avoidance for USVs, the pros and cons of previous research methods are analyzed and our own model is established accordingly. In accordance with the model, a deep reinforcement learning method ANOA is proposed to achieve real-time path planning with obstacle avoidance in sophisticated marine environment. Furthermore, the reward function of ANOA is designed to evaluate the navigation results. On the constructed simulation platform, a variety of reinforcement learning methods such as ANOA, DQN and Deep Sarsa [39] are testified to verify their performances both in static and in dynamic environments.

3. The navigation and obstacle avoidance for USVs

3.1. Problem formulation

For the autonomous navigation of USVs, the most important is to reach predetermined target without collision with obstacles in the environment. Success in the autonomous navigation relies heavily on efficient methods, which include following parts:

(1) Instant decision-making for movement strategies immediately after getting observations of different environments.

(2) Sequences of control actions for USVs without stop even in emergency.

(3) Each action of USVs complying with their designs and easy to implement.

A reinforcement learning method ANOA is proposed to address above problems of universal navigation, which will manipulate USVs to reach the predetermined destination without any collision. Through the navigation, USVs will observe the information of environment and record the state information while the proposed ANOA produce the corresponding control strategy. That is to say, the proposed ANOA is the brain of USVs whose autonomous navigation with obstacle avoidance is realized by a series of control actions. The formal definition of parameters is as follows: The USV observes a state s_t at each time step t, then an action $a_t(s_t)$ is selected to make a transition from current state s_t to next time state s_{t+1}. Meanwhile, the USV receives an immediate reward R_t for each transition. The control action set $a = \{a_1, a_2, a_3 \ldots a_n\}$ is predefined to update the location of the USV. Besides, T_p is the historical time step for observations. That is to say, ANOA takes the past T_p states as input to select current action.

3.2. Design of ANOA algorithm

The navigation of a USV could be defined as a Markov Decision Process (MDP) [40] under discrete time steps. The proposed ANOA reinforcement learning algorithm contains the following components: states, actions, transition function, policy and rewards. For a state s_t, an action a_t is determined by policy $\pi(s_t)$. Then the next state s_{t+1} is determined accordingly by the transition function, and an immediate reward is calculated. Considering consecutive actions with immediate rewards $\{R_1, R_2, R_3, \ldots, R_t\}$, the quality of a state could be measured by expected returns to some extent. The G_t is defined as follows to represent the expected return of the USV at a certain time:

$$G_t = R_t + \lambda * R_{t+1} + \lambda^2 * R_{t+2} + \cdots = \sum_{k=1}^{T} \lambda^{k-1} * R_{t+k-1} \tag{1}$$

where G_t is the sum of discounted rewards after time t, and λ is the discount factor. In another word, G_t is the expected cumulative future discounted reward. In Eq. (1), the value of G_t only depends on the current state s_t, which is in fact a property of MDP. By the law of large numbers, we could estimate the expectation of $G_t(s_t)$ by observing the navigation of the USV in a large number of times, formally as:

$$v_\pi(s) = E[G_t | S_t = s] \tag{2}$$

Substitute Eq. (1) into Eq. (2) and we have Bellman equation:

$$\begin{aligned} v(s) &= E[G_t|S_t = s] = E[R_t + \lambda * (R_{t+1} + \lambda * R_{t+2} + \cdots)|S_t = s] \\ &= E[R_t + \lambda * G_{t+1}|S_t = s] = E[R_t + \lambda * v(S_{t+1})|S_t = s] \end{aligned} \tag{3}$$

Eq. (3) tells us that the value of $v(s_t)$ is determined by R_t and $v(s_{t+1})$.

There are multiple actions that can be selected at each state, so it is better to use $q(s, a)$ instead of $v(s)$, where $q(s, a)$ is the expected reward of the next state of s given the action a. The more rewards, the better for reinforcement learning methods. Thus the proposed ANOA tries to find the strategy that maximizes $v(s)$ or its equivalently $q(s, a)$, which is defined in Eq. (4):

$$\pi^* = argmaxq_\pi(s, a) \tag{4}$$

There is no closed form of solutions to the optimization problem descripted in Eq. (4). However, there are several iterative solutions such as Q learning and Sarsa. There are a variety of actions that could be selected at each state for the USV and the next state is different. It is not enough to calculate the action–value function because the optimal strategy is needed. Therefore, based on the valuation iteration, the current Q value and rewards are used to update the historical Q value. The process of updating Q value is (i.e. Q-Learning):

$$\begin{aligned} Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha * (R_t + \lambda * max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) \\ - Q(s_t, a_t)) \\ s_t \leftarrow s_{t+1} \end{aligned} \tag{5}$$

In Eq. (5), we will choose a_t from s_t using policy derived from Q. When action a_t is taken, reward R_t is observed and state s_{t+1} is also observed. Instead of giving the estimated Q value directly to the new Q value, it is approached in a gradual way by α. From the perspective of decision-making, the principles of both Q-learning and Sarsa are based on the Q-table. In Q-learning, the control action with the largest Q value will be selected, and they will be applied to obtain the reward value in current environment. However, the updating mode of Sarsa is slightly different from Q-learning and is defined as follows:

$$\begin{aligned} Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha * (R_t + \lambda * Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)) \\ s_t \leftarrow s_{t+1}, a_t \leftarrow a_{t+1} \end{aligned} \tag{6}$$

In Sarsa, we take action a and observe R_t, s_{t+1}. Then we choose a_{t+1} from s_{t+1} using policy derived from Q. The above discussion demonstrates that Q-learning is a greedy algorithm based on $maxQ(s_{t+1})$, and it does not care about the "wrong" and so-called "dead" states of the USV such as collision with obstacles or beyond the range of map. While Sarsa is a conservative approach which updates $Q(s_t, a_t)$ based on $Q(s_{t+1}, a_{t+1})$. Thus it is more sensitive to irrational states, which will choose the control action far from "dangerous" states first, and it will consider to obtain the maximum reward value.

Since the space of the marine environment is continuous, there are infinite amount of states, which incurs infinite amount of state–action pairs. It is impossible for us to traverse all states and construct a Q-value table, which will need infinite time and memory space. DQN brings the solution to the dilemma. It is the value function approximation that is established, which transfers

the updating problem of Q value matrix into the function fitting problem as follows:

$$Q(s, a, \theta) \approx Q^{\pi}(s, a) \tag{7}$$

The function approximation is described as a parameterized function of states and actions for the USV. The network parameters θ is updated by making the Q function approximates the optimal Q value. Then the mean-square error is used to define the loss function:

$$L(\theta) = E[(R + \gamma * \underset{a'}{argmax}\, Q_t(s', a', \theta) - Q_t(s, a, \theta))^2] \tag{8}$$

where γ is the learning factor and its gradient about the parameter θ is:

$$\frac{\delta L(\theta)}{\delta \theta} = E[(R + \gamma * \underset{a'}{argmax}\, Q_t(s', a', \theta) - Q_t(s, a, \theta))^2] * \frac{\delta Q(s, a, \theta)}{\delta \theta} \tag{9}$$

Traditional DQN generally overestimates Q value of control actions for the USV [25], and the estimation error will accumulate with the increase of the number of control actions. Because the overestimation is not uniform, the Q value of a sub-optimal control action could exceed the Q value of an optimal control action. Under this circumstances, it would be possible that the optimal strategy would never be found. Thus the ANOA algorithm with dueling deep Q-network is proposed to address this problem. The deep network of ANOA algorithm is divided into two parts on the end of layer, including the state value function $V(s)$ which represents the reward value of state, while the action advantage function $A(a)$ means the extra reward value of choosing an action. The Q value is estimated under state V value and action A value. The extra term $A(s)$ here ensures that actions that with better potential reward would be selected. Therefore according to the characteristics of ANOA algorithm, Q value can be divided into two parts: the state–action value and action-advantage value. The action-advantage value is independent of state and environment noise, which is a relative action–value in each state relative to other unselected actions.

$$Q(s, a; \theta, \alpha, \beta) = V(s; \theta, \beta) + A(s, a; \theta, \alpha) \tag{10}$$

where θ denotes the parameters of convolutional layers and α and β are the parameters of two streams of fully-connected layers in dueling deep Q-network. However, when Q value is given, the value V and A are not unique. In other words, the various combinations of value V and A can get the same Q value, which will make the algorithm lack stabilities. Therefore the average value of advantage function is used to improve the stability of the proposed ANOA algorithm:

$$Q(s, a; \theta, \alpha, \beta) = V(s; \theta, \beta) + (A(s, a; \theta, \alpha) - \frac{1}{|a|}\sum_{a'} A(s, a'; \theta, \alpha)) \tag{11}$$

Due to the deviation between the target value and the real value, the loss function is established to update the network parameters continuously as follows:

$$L(\theta) = E_{s,a,R,s'}[(y - Q_t(s, a, \theta))^2] \quad where$$
$$y = R + \gamma * \underset{a'}{argmax}\, Q_t(s', a', \theta) \tag{12}$$

Thus we can get the optimal Q value by updating the gradient $\frac{\delta L(\theta)}{\delta \theta}$. Aiming at the autonomous navigation with obstacle avoidance for the USV in marine environment, ANOA algorithm will calculate the real-time control strategy according to the observation information and the USV's state information, which drives the USV gradually approach the target to complete the autonomous navigation mission. Through ANOA algorithm, discrete control actions are generated on the grid-based discrete map. The algorithm can be formulated as:

$$A_t = f_{ANOA}(s_t, s_{t-1}, s_{t-2}, \ldots, s_{t-T_p}) \tag{13}$$

where f_{ANOA} is the proposed algorithm and the input of this algorithm is the past T_p states. A_t is the control action at time t and it can be implemented with a real control model for USVs. This real control model is a refined mathematical model for underactuated ships or USVs moving in surge, sway and yaw as shown in Fig. 1, which is obtained from the motion equation of the ship moving in six degrees of freedom under disturbances induced by wave, wind and ocean current by neglecting motion in heave, pitch and roll [41]. It can be referred to [42] for detailed development of the original mathematical model for underactuated ships or USVs moving in six degrees of freedom. The control model can be described with following equations:

$$\begin{cases} \dot{x} = \mu\cos\psi - \nu\sin\psi \\ \dot{y} = \mu\sin\psi + \nu\cos\psi \\ \dot{\psi} = r \\ \dot{\mu} = \frac{m_{22}}{m_{11}}\nu r - \frac{d_{11}}{m_{11}}\mu + \frac{1}{m_{11}}\tau_{\mu} \\ \dot{\nu} = -\frac{m_{11}}{m_{22}}\mu r - \frac{d_{22}}{m_{22}}\nu \\ \dot{r} = -\frac{(m_{11}-m_{22})}{m_{33}}\mu\nu - \frac{d_{33}}{m_{33}}r + \frac{1}{m_{33}} \end{cases} \tag{14}$$

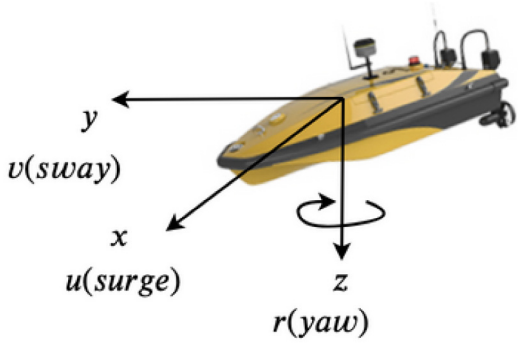


Fig. 1. Definition of surge, sway and yaw modes of motion.

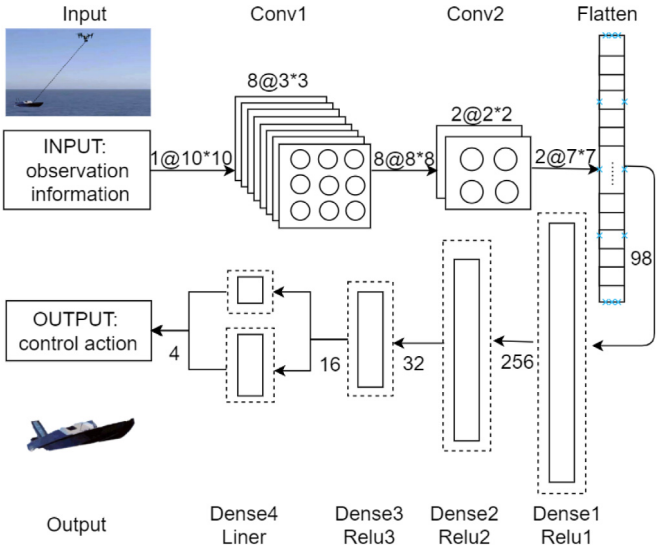


Fig. 2. The architecture of the dueling deep Q-network for the ANOA algorithm.

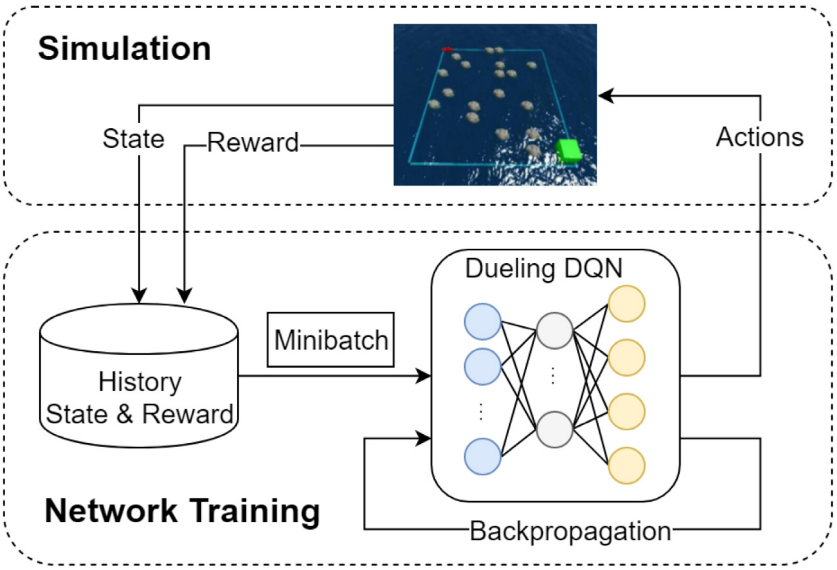


Fig. 3. The main components and data flow of the ANOA algorithm.

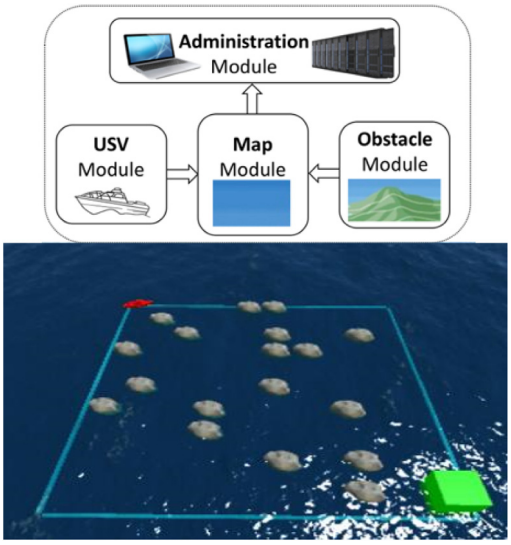


Fig. 4. The architecture of the simulation platform. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

where x, y and ψ are the surge displacement, sway displacement and yaw angle in the absolute coordinate system; μ, ν and r denote surge, sway and yaw velocities. Dots above these letters denote the differentiation. The terms m_{ii} are constants,in which $1 \leq i \leq 3$ denotes the USV inertia including added mass. The positive constant terms d_{jj}, $1 \leq j \leq 3$, represent the hydrodynamic damping in surge, sway and yaw. τ_u is the surge force and τ_r is the yaw moment. These parameters can be obtained by USVs with multiple sensors.

The architecture of the dueling deep Q-network for the ANOA algorithm is demonstrated in Fig. 2.

A dueling deep Q-network is designed to parameterize the Q-values. The input to the dueling deep Q-network is a one-channel 10×10 image produced by the simulation platform while the output is a multi-dimensional vector corresponding to the predicted Q-values of a possible action. The first layer is an 8 kernel 3×3 convolution layer and the second layer is a 2 kernel 2×2 convolution layer. A convolution layer has kernels with a certain size of receptive field which can be applied on the image to extract spatial features of the state. Hence, the USV can efficiently learn how to navigate through different environments with different spatial properties. Afterwards, the state is flattened into a 98-dimensional vector and subsequently fed into 4 fully connected layers with 256, 32, 16 and 4 units, namely Dense1 to Dense4. Noticeably, Dense3 has two streams to separately estimate state-value and the advantages for each action and they are combined by Dense4 following Eq. (11). With the output of Dense4 as estimated Q-values, control actions for the USV can be easily obtained.

A concise illustration of the main components and data flow of the ANOA algorithm is displayed in Fig. 3. The central part of the figure is the dueling deep Q-network. Simulation part on the upper side shows how reinforcement learning part uses the dueling deep Q-network to make decisions and obtains states and rewards. Lower side, namely network training part illustrates the forward and backward processes that optimize the network parameters for ANOA.

In the feedback part of ANOA algorithm, we set the corresponding reward value for different circumstances: rwd_{target} represents the reward value of reaching the destination, $rwd_{outside}$ is the reward value that the USV is beyond the range of predefined map, $rwd_{collision}$ is the reward value for the USV which has collision with obstacles, and $rwd_{interim-status}$ represents the reward value that the USV in interim status and is used for controlling the USV to reach the terminal as soon as possible in shortest distance. During the training process, the target $\hat{Q}-network$ with parameters θ^- is the same as the online network Q except that its parameters are copied every C steps from the target network, in which C is a constant. The steps of proposed ANOA algorithm are summarized as follows:

Algorithm 1 ANOA algorithm for the navigation of USVs.

Input: observation information $obs_t = [S_t, A_{t-1}]$, Q-network and its parameters θ, target $\hat{Q} - network$ and its parameters θ^-
Output: weights θ^* for ANOA networks
Initialize the obs_t
Initialize the experience replay repository D to capacity N, and Initial the historical
observations repository D_0 to capacity T_p
Initialize the $Q - network$ with random weights θ, and Initial the target $\hat{Q} - network$

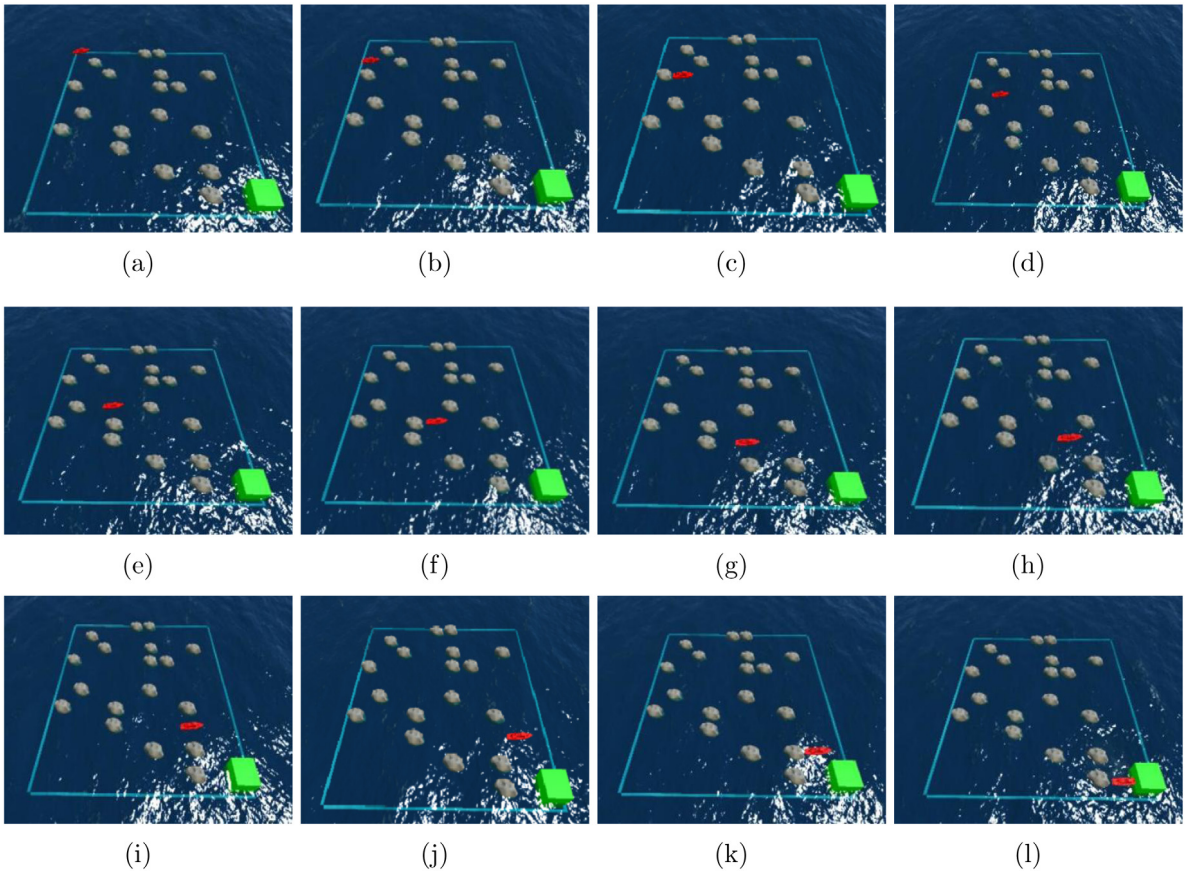


Fig. 5. The performance of ANOA in static environment.

with $\theta^- = \theta$
Initialize the parameters α, β of two streams of fully-connected layers in dueling deep Q-network, and
the Q-value of each action could be obtained by Q = V+A
for episode =1, M do
for t =1,T do
Fetch the observation from D_0, and form the input $S_t = [s_t, s_{t-1}, ..., s_{t-T_p}]$ according to the $[S_t, A_{t-1}]$
Select $A_t = argmax_{A_t' \in A} Q(S_t, A_t'; \theta)$, otherwise select a random action A_t with probability ε
Execute action A_t, observe reward R_t and calculate S_{t+1}
Store the transition (S_t, A_t, R_t, S_{t+1}) in D
Sample minibatch (S_t, A_t, R_t, S_{t+1}) randomly from D
Set $y(t) = R(t) + \gamma * argmax_{a'} Q_t(S_{t+1}, A_{t+1}; \theta^-)$ and train the network with loss function $L(\theta) = E[(y(t) - Q_t(S_t, A_t, \theta))^2]$
Reset the Q-network $\hat{Q} = Q$ every C steps
End for
End for
Return weights θ^* for Q-network

4. Experiments

4.1. The simulation platform

To carry out experiments of autonomous navigation and obstacle avoidance for USVs, a simulation platform is constructed as demonstrated in Fig. 4. The simulation platform is implemented with the Unity Machine Learning Agents Toolkit (ML-Agents) [43]. It is an open-source Unity plugin that enables games and simulations to serve as environments for training intelligent agents. Agents can be trained using reinforcement learning, imitation learning, neuroevolution, or other machine learning methods through a simple-to-use Python API. On the simulation platform, the movements of USVs in different marine environments are simulated.

We designed four modules for the simulation platform:

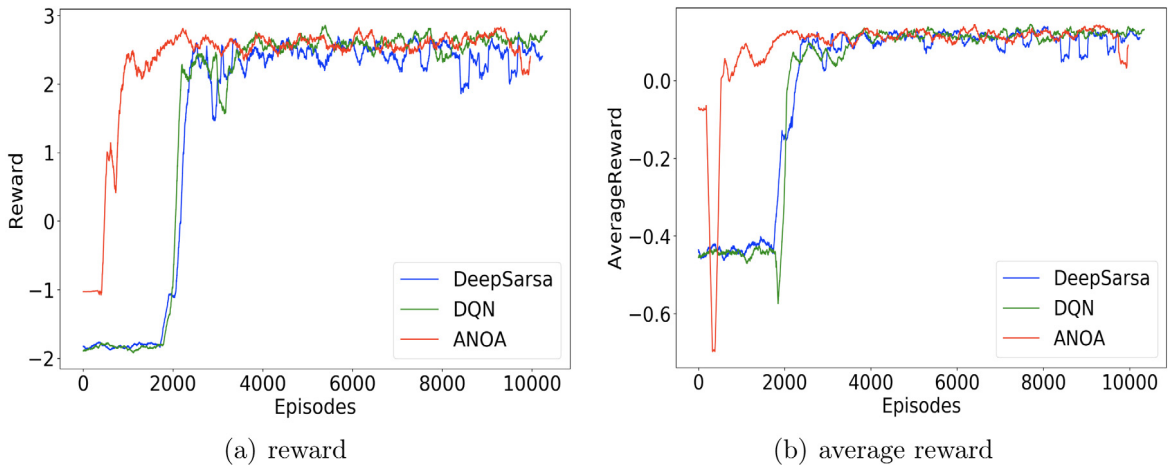
(1) The USV module is responsible for setting performance parameters, planning paths and making decisions in action choices for USVs.

(2) The map module has two responsibilities. One is setting the property of maps, which are grid-based discrete map or two-dimensional continuous map. The other is setting the basic map parameters such as the area, the starting point and the destination.

(3) The obstacle module is responsible for setting the motion mode and basic information of obstacles, such as the number and positions of obstacles.

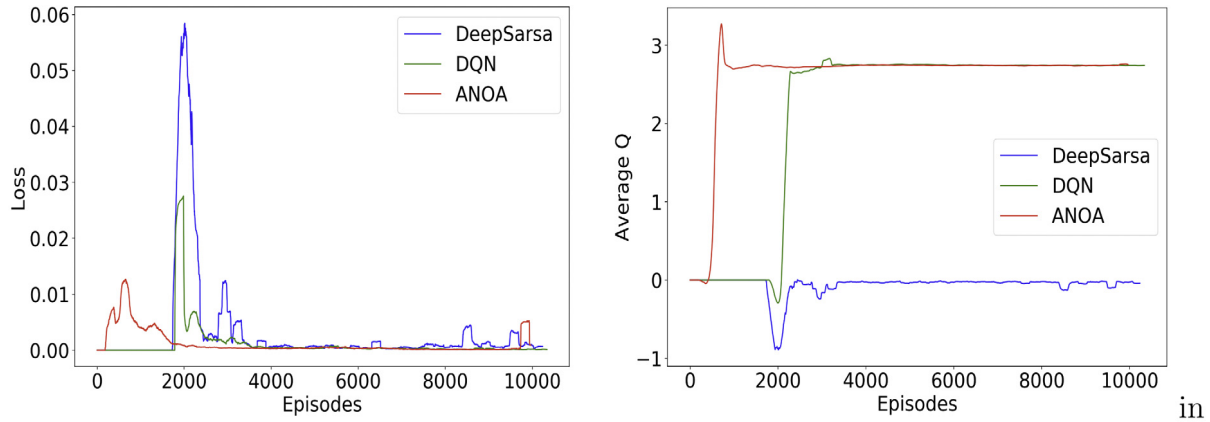
(4) The administration module is responsible for the coordination of above modules, which will make judgments about current status and navigation results, including the destination reached, collisions with obstacles or out of bounds. Furthermore, the administration module is responsible for the implementation of different navigation and obstacle avoidance algorithms.

The architecture of the simulation platform is shown in Fig. 4: the red boat is on behalf of the USV; the green cube represents the destination and the gray reefs represent obstacles. The wind velocity and direction and ocean currents and dynamics are not taken into consideration in this simulation platform.



(a) reward (b) average reward

Fig. 6. The change of reward at each episode in static environment.



(a) loss value (b) average Q-value

Fig. 7. The change of loss and Q-value at each episode in static environment.

The combination of proposed modules will provide an intuitive interface for the observation on the process and results of different algorithms. In real marine environments, the USV module could be modified and installed in USVs and the map module, the obstacle module and the administration module could be installed in an Unmanned Aerial Vehicle (UAV). Due to its' aerial view of marine environments, the UAV could update the map module and obstacle module. What is more, the administration module in the UAV will have optimal control strategy according to obtained global information.

For the experiment of proposed ANOA algorithm, a 10×10 grid-based discrete map is adopted. Thus observation information including the USV and obstacles is expressed in a matrix form, which is the input for the proposed dueling deep Q-network as shown in Fig. 2. This matrix is initialized with 0 in each cell, which also serves as the feasible path point. In this matrix, 1 is the obstacle location, 2 is the USV location and 3 is the destination. According to the input of dueling deep Q-network and the matrix, the output will be generated after computations. The output is the movement of the USV in four cardinal directions: north, east, south, and west. The four cardinal directions are represented by integer $1 - 4$ and the step length of the movements is 1. In addition, the reward function is designed to evaluate the returns of the USV in a certain state, which is defined as follows:

(1) The reward value is set to -1 when the USV is beyond the range of defined map;

(2) The reward value is set to -2 when the USV has collisions with obstacles;

(3) The reward value is set to 3 when the USV reaches the defined destination;

(4) The reward value is set to -0.001 when the USV is in interim states, which enable the USV to be on a possible path to the destination.

According to above-mentioned parameter settings, experiments are carried out to test the effectiveness and performances of proposed ANOA algorithm. To avoid any implementation related issue, we used two open source and well-reviewed package to build the simulation platform and carry out the experiment. The platform is built with The Unity Machine Learning Agents Toolkit [43] and the ANOA algorithm is implemented with Keras-RL [44].

4.2. Experimental results of reinforcement learning methods in static environment

On the predefined 10×10 grid-based discrete map, there are stationary obstacles that are shown in Fig. 5. The step length of the USV's movements is set to 1. The starting point of the USV is set to the top left cell of the map and the destination is set to the bottom right. The movements of the USV with proposed ANOA algorithm are demonstrated in the consecutive images of Fig. 5. Not all frames are shown in Fig. 5 due to the limited space. From Fig. 5, it is clearly demonstrated that the USV achieves smooth

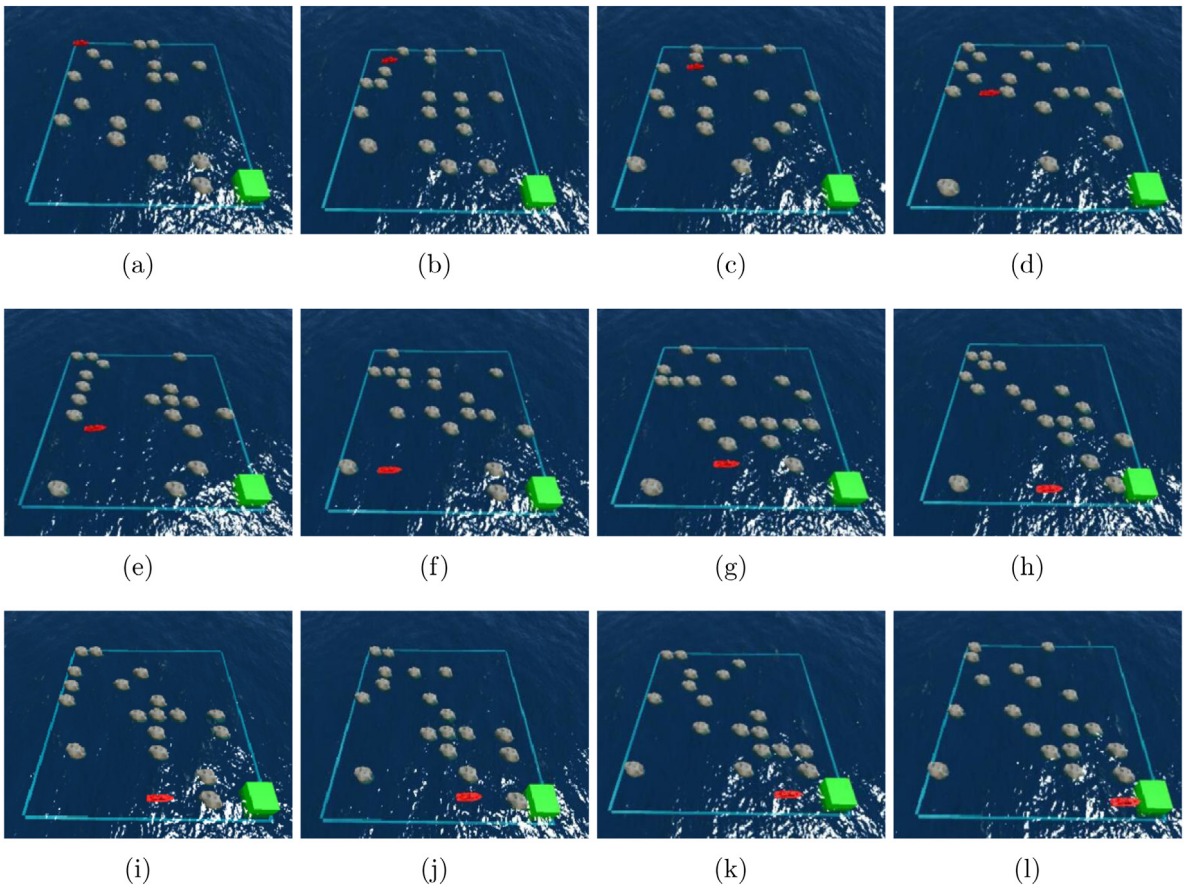


Fig. 8. The performance of ANOA in dynamic environment.

navigation following the shortest path without collisions against any obstacles, which proves the effectiveness of ANOA in static environment.

To further analyze the performance of ANOA algorithm, the reward, average reward, loss value and average Q-value are demonstrated in Figs. 6 and 7. In Fig. 6, the curves of reward value and average reward value from DQN, Deep Sarsa and ANOA are demonstrated, which have similar trends but different details. The ANOA algorithm is the first to explore an effective strategy and its initial reward value is higher than DQN and Deep Sarsa. When the maximum reward value is found by DQN, Deep Sarsa and ANOA in the late-stage of exploration, the ANOA algorithm has a relatively stable fluctuation, while DQN and Deep Sarsa have stronger fluctuations than ANOA.

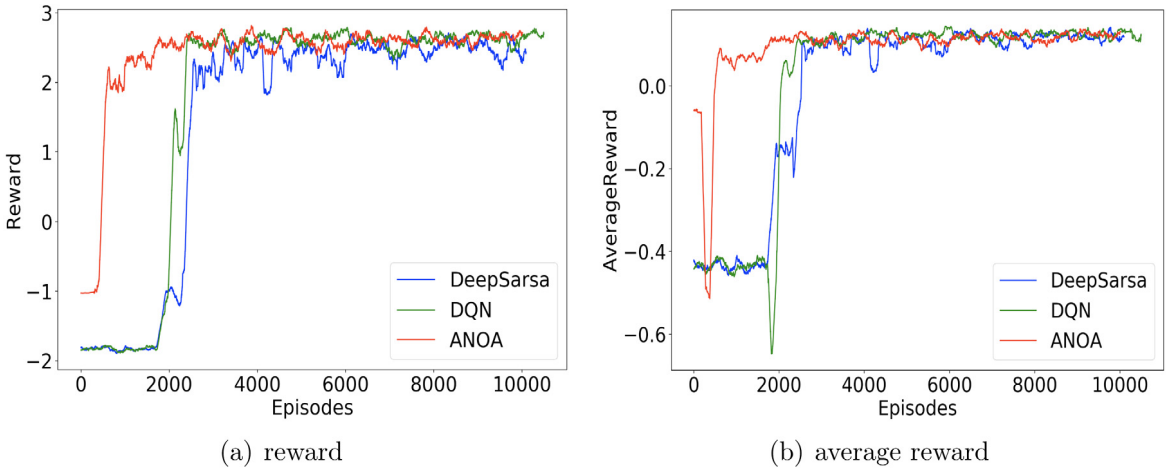
The loss value of DQN, Deep Sarsa and ANOA at each episode are shown in Fig. 7(a). As demonstrated, the proposed ANOA is the fastest one to find effective strategies. In the early stage, the loss value of proposed ANOA algorithm is much lower than that of DQN and Deep Sarsa. Although these three algorithms eventually converge, the overall volatility of Deep Sarsa is significantly greater than DQN and ANOA. The average Q-value of DQN, Deep Sarsa and ANOA at each episode are shown in Fig. 7(b). Because the calculation of the Q value of Deep Sarsa is different from that of DQN and ANOA as discussed in Section 3.1, there are big differences among these three algorithms.

From Figs. 6 and 7, we could safely draw a conclusion that ANOA is faster than DQN and Deep Sarsa in the effective exploration. Furthermore, in terms of convergence speed, ANOA is the highest among three algorithms. ANOA successfully converges within 2000 episodes, while DQN and Deep Sarsa converge after around 3000 episodes. The average Q value is an important evaluation index for reinforcement learning algorithms. The learning process of ANOA converges at around 1000 episodes, while DQN and Deep Sarsa converge after 2000 episodes. Compared with DQN and Deep Sarsa, ANOA has higher efficiency in the exploration, better performances in finding effective strategies and better stability in the late-stage of convergence.

4.3. Experimental results of reinforcement learning methods in dynamic environment

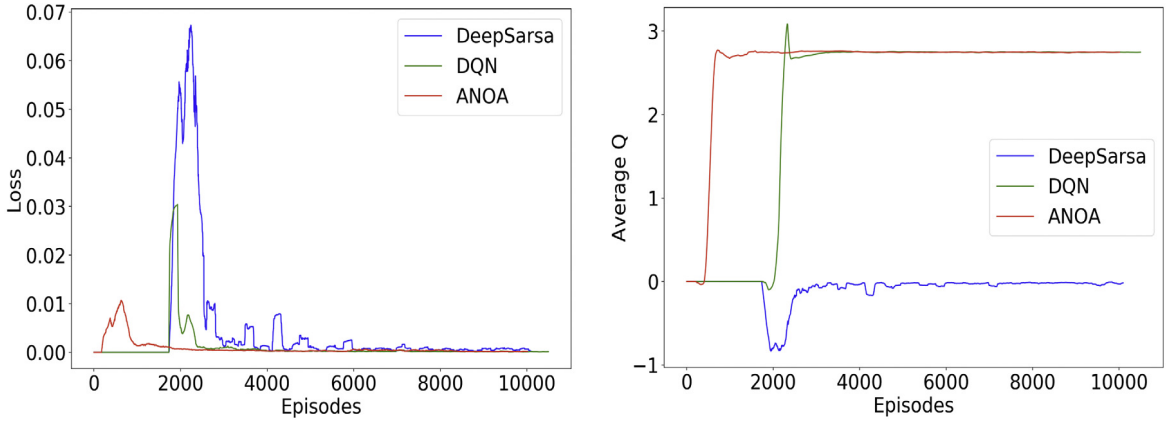
In order to prove the robustness of ANOA algorithm, the dynamic environment is introduced with moving obstacles. The random appearances of obstacles are introduced into the obstacle module of the simulation platform. Then a large amount of experiments are conducted to check whether the USV can avoid obstacles and follow the shortest path to reach destination by autonomous navigation with ANOA. The movements of the USV in dynamic environment are demonstrated in Fig. 8 and the performances of ANOA, DQN and Deep Sarsa are shown in Fig. 9. From Fig. 8, it is clearly demonstrated that the USV achieves smooth navigation following the shortest path without collisions against any obstacles even obstacles could randomly move in the whole navigation process.

It has been proved that ANOA outperforms DQN and Deep Sarsa in static environment. Furthermore, the performances of these three algorithms in dynamic environment should be evaluated. The experimental setup is consistent within last section. The trends of reward value are demonstrated in Fig. 9(a). The ANOA is the first to explore an effective strategy and the initial reward value is higher than DQN and Deep Sarsa. When the maximum reward value is found by DQN, Deep Sarsa and ANOA in the late-stage of exploration, the ANOA and DQN have a relatively



(a) reward (b) average reward

Fig. 9. The change of reward at each episode in dynamic environment.



(a) loss value (b) average Q-value

Fig. 10. The change of loss and Q-value at each episode in dynamic environment.

stable fluctuation, while Deep Sarsa has violent fluctuations. Accordingly, the change of average reward value at each episode is demonstrated in Fig. 9(b). The trend in this figure is similar to that of Fig. 9(a).

The trends of loss value at each episode are shown in Fig. 10(a). As expected, the ANOA algorithm was fast and efficient in finding the effective strategies and the loss value of the ANOA algorithm was much lower than that of DQN and Deep Sarsa in the early stage. The peak of ANOA loss is 0.01, while the peak of DQN and Deep Sarsa loss are as high as 0.03 and 0.068 respectively. Although these three algorithms eventually converge, the overall volatility of Deep Sarsa was significantly greater than the other two algorithms. The trends of average Q-value are shown in Fig. 10(b). Because the calculation of Q value of Deep Sarsa is different from that of the other two algorithms. Nonetheless, ANOA successfully converges within 1000 episodes, while DQN and Deep Sarsa converge after 2000 episodes. Compared with DQN and Deep Sarsa, ANOA starts the effective exploration earlier than the other two algorithms and its convergence speed is the highest and the convergence curve is the smoothest. This is because ANOA utilizes the average value of advantage function when calculating the Q value, which improved the stability of the algorithm.

The above experimental results prove that ANOA is effective and highly efficient not only in static environment but also in dynamic environment. Compared with DQN and Deep Sarsa, ANOA has higher exploration efficiency and convergence speed.

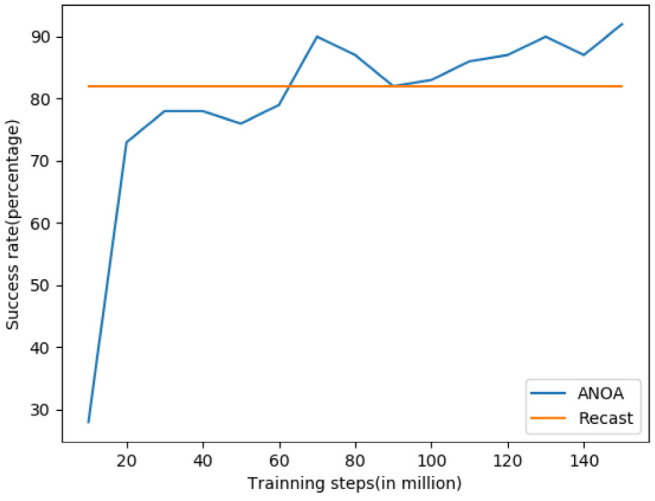


Fig. 11. The success rate of ANOA and Recast navigation.

Therefore, under the guidance of ANOA algorithm, the USV can navigate towards the destination following the shortest path without collisions against any obstacles.

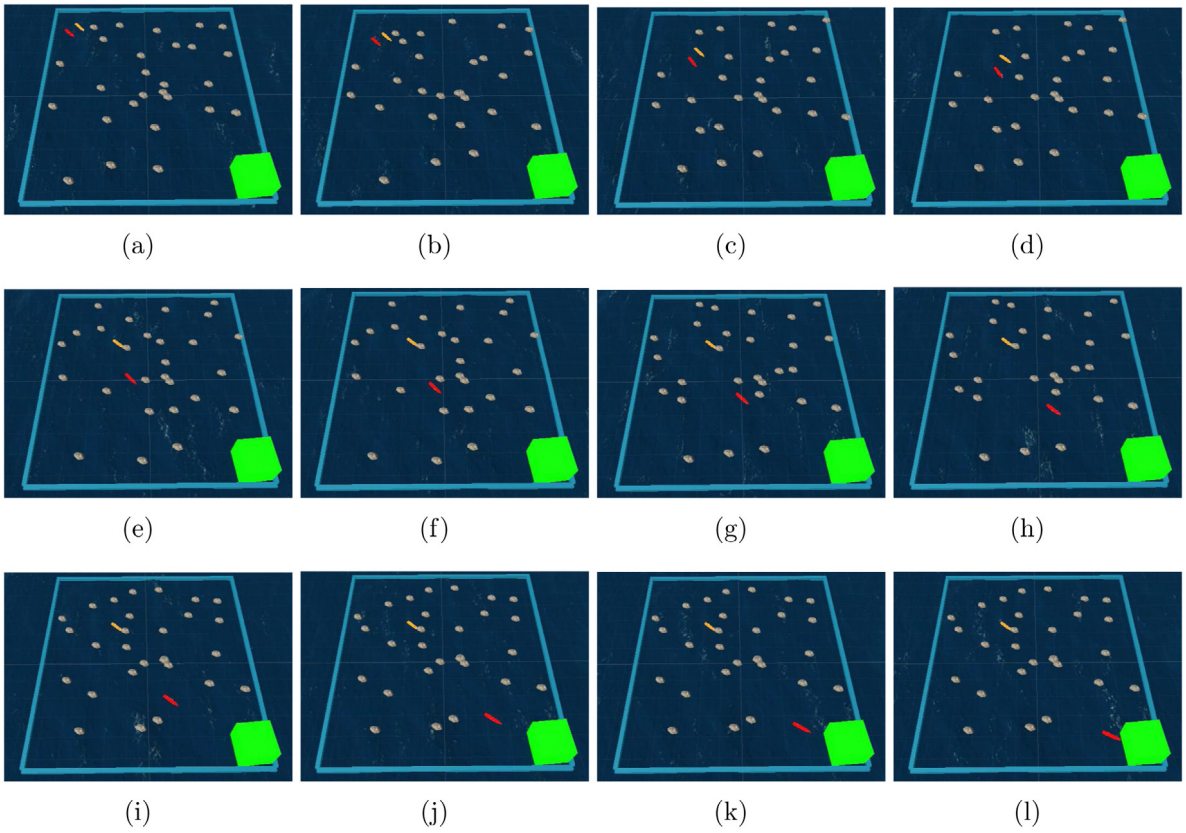


Fig. 12. The performance of ANOA and Recast navigation in dynamic environment I. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

4.4. Experimental results of ANOA and Recast navigation in dynamic environment

To evaluate ANOA method in real data, the real control model for USVs described in Eq. (14) is integrated. Furthermore, a non-neural network baseline method, Recast navigation [45], is implemented. To compare the performance between ANOA and Recast navigation in dynamic environment, Recast navigation is also integrated with the real control model.

Recast navigation is a state-of-the-art non-neural network based navigation method, which is widely used in practice. The success rate of ANOA and Recast navigation with respect to the training steps are drawn in Fig. 11. Since the baseline Recast navigation is not a learning method but a heuristic method, the success rate of Recast is a constant value with respect to training steps.

From Fig. 11, it can be observed that Recast navigation is a competitive baseline method and achieves better performance than ANOA when it is trained with less than 70 million steps. After 70 million steps of training, the ANOA outperforms the Recast navigation method. With around 90 million training steps, the ANOA falls into a local optimum and has the same success rate of the baseline method. However, the ANOA efficiently explores the possible solutions and achieves better performance with more than 100 million training steps. After 150 million training steps, ANOA is compared with Recast navigation in dynamic environment. Both of them are integrated with real control model described in Eq. (14).

The red boat is on behalf of the USV controlled by ANOA algorithm and the yellow boat is on behalf of the USV controlled by Recast navigation method. Both the red USV and the yellow USV start from the starting point and head for the destination. Although obstacles appear randomly, both boats can reach predetermined target without any collision with obstacles in most cases. In some cases, the yellow USV controlled by Recast navigation method may hit obstacles in the early stage of navigation as shown in Fig. 12 or in the late stage of navigation as shown in Fig. 13. The failures of Recast navigation are probably because the Recast navigation method needs a large amount of iterations to get satisfactory results and unable to make instant decisions to avoid obstacles.

Heuristic methods generally have long iteration time, whereas reinforcement learning methods have long training time. Both methods have strengths and weaknesses and they have different applicable scenarios. In the autonomous navigation and obstacle avoidance for USVs, ANOA can achieve a higher success rate than Recast navigation after ANOA is fully trained.

5. Discussions

The autonomous navigation and obstacle avoidance for USVs is of scientific significance and practical value since USVs could get to marine areas dangerous for ships with sailors. With the help of different types of sensors, USVs should be aware of surrounding obstacles during the autonomous navigation. A qualified autonomous navigation algorithm should be able to ensure instant decision-making for movement strategies immediately after getting observation of environments, sequences of control actions for USVs without stop even in emergency and each action of USVs complying with their designs and control model.

Path planning methods with global environment information are easy to fall into traps in complex environments. Heuristic methods focus on finding good heuristic evaluation for the problem, which may lead to unsatisfactory performance in dynamic environment. In dynamic environment, heuristic methods

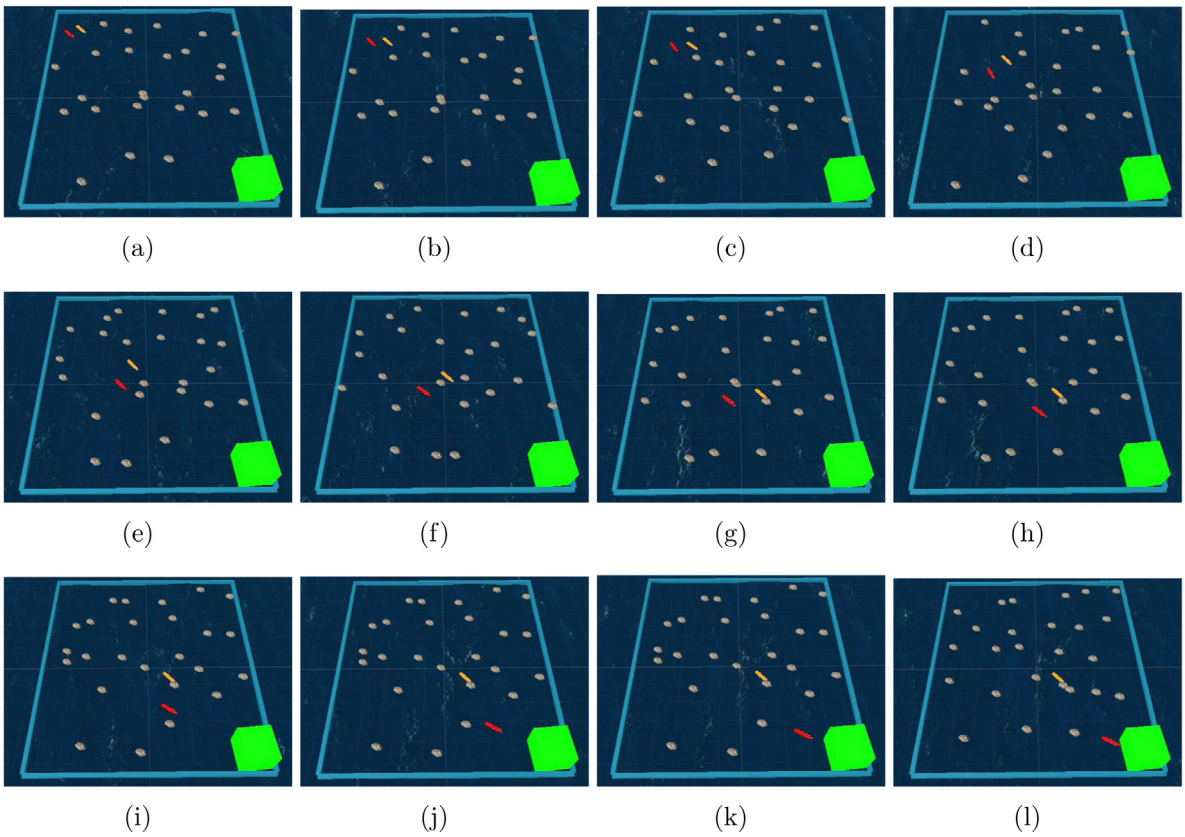


Fig. 13. The performance of ANOA and Recast navigation in dynamic environment II. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

are slow in speed and unable to detect and avoid obstacles in real time in some cases. However, the proposed ANOA is specially designed for guidance problem in dynamic environment and does not need any manually designed heuristic policy. What is more, reinforcement learning methods provide a direct adaptive optimal control of nonlinear systems, which is quite in accord with autonomous navigation and obstacle avoidance even in dynamic environments. Heuristic methods and reinforcement learning methods have strengths and weaknesses and they have different applicable scenarios. Putting deep reinforcement learning methods into navigation scenarios is not easy because it is difficult to design state and action spaces in dynamic environments. The proposed ANOA algorithm with a dueling deep Q-network could meet the challenge, whose exploration and greedy policy are especially suitable for path planning in sophisticated environment. The higher the random exploration probability is, the better the obtained navigation routes will be.

For the problem of autonomous navigation and obstacle avoidance, there are a large number of actions could be chosen by the USV. The performance advantage of ANOA over DQN and Deep Sarsa lies partly in its ability to learn the state-value function efficiently. Thus ANOA with the dueling network over DQN and Deep Sarsa is especially prominent when the number of actions is large. To generalize learning across actions, ANOA has a better policy evaluation than DQN and Deep Sarsa in the presence of many similar-valued actions in autonomous navigation and obstacle avoidance. Furthermore, the ANOA tries to avoid being overoptimistic. Accordingly, ANOA outperforms DQN and Deep Sarsa in exploration efficiency and convergence speed.

6. Conclusions

To enhance the intelligence of USVs in the sophisticated marine environment, the ANOA algorithm is proposed for real-time path planning with obstacle avoidance. On the constructed simulation platform, multiple experiments are carried out to verify the effectiveness and efficiency of ANOA algorithm. According to the experimental results, ANOA outperforms deep Q-network (DQN) and Deep Sarsa in the efficiency of exploration and the speed of convergence not only in static environment but also in dynamic environment. Moreover, a real control model of USVs moving in surge, sway and yaw is integrated with proposed ANOA and a frequently used heuristic approach, Recast navigation. In dynamic environment, ANOA achieves a higher success rate than Recast navigation after ANOA is fully trained. However, the wind velocity and direction are not taken into consideration in the simulation platform. Our future work will focus on two aspects: one is the real sea environments with the impact of the wind power to improve ANOA algorithm; the other one is the collaborative navigation of multiple USVs will be studied.

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.

Acknowledgments

This paper is supported by the National Natural Science Foundation of China (Grant No. 61625304) and by the State Key Program of National Nature Science Foundation of China (Grant No. 61936001).

References

[1] H. Kandath, T. Bera, R. Bardhan, S. Sundaram, Autonomous navigation and sensorless obstacle avoidance for ugv with environment information from uav, in: IEEE International Conference on Robotic Computing, 2018, pp. 266–269.

[2] B. Arbanas, A. Ivanovic, M. Car, M. Orsag, T. Petrovic, S. Bogdan, Decentralized planning and control for uav–ugv cooperative teams, Auton. Robots (2018) 1–18.

[3] I. Adamski, R. Adamski, T. Grel, A. Jędrych, K. Kaczmarek, H. Michalewski, Distributed deep reinforcement learning: Learn how to play atari games in 21 minutes, in: International Conference on High Performance Computing, 2018, pp. 370–388.

[4] A.E. Sallab, M. Abdou, E. Perot, S. Yogamani, Deep reinforcement learning framework for autonomous driving, Electron. Imaging 2017 (19) (2017) 70–76.

[5] A.E. Sallab, M. Abdou, E. Perot, S. Yogamani, End-to-end deep reinforcement learning for lane keeping assist, 2016, arXiv preprint arXiv:1612.04340.

[6] H. Sharma, T. Sebastian, P. Balamuralidhar, An efficient backtracking-based approach to turn-constrained path planning for aerial mobile robots, in: European Conference on Mobile Robots, 2017, pp. 1–8.

[7] Q. Guo, Z. Zhang, Y. Xu, Q. Guo, Z. Zhang, Y. Xu, Path-planning of automated guided vehicle based on improved dijkstra algorithm, in: Control and Decision Conference, 2017, pp. 7138–7143.

[8] H.E. Pan-Bo, W.U. Chun-Xue, Improved genetic algorithm route planning based on multiple constraints, 2018, Pan-Bo, H.E. and Chun-Xue, W.U.

[9] S. Perezcarabaza, J. Bermudezortega, E. Besadaportas, J.A. Lopezorozco, J.M. De, la Cruz, A multi-uav minimum time search planner based on acor, in: Genetic and Evolutionary Computation Conference, 2017, pp. 35–42.

[10] M.D. Phung, H.Q. Cong, T.H. Dinh, Q. Ha, Enhanced discrete particle swarm optimization path planning for uav vision-based surface inspection, Autom. Constr. 81 (2017) 25–33.

[11] C.C. Fang, P.M. Sun, Path planning of robot based on improved ant colony algorithm, Meas. Control Technol. (2018).

[12] Y. Wei, K.X. Zhao, D.S. Wang, Application of improved particle swarm optimization algorithm in path planning of mobile robot, Fire Control Command Control (2018).

[13] S.Y. Zhang, Y.K. Shen, W.S. Cui, Path planning of mobile robot based on improved artificial potential field method, Appl. Mech. Mater. 644–650 (2014) 154–157.

[14] Y. Sazaki, H. Satria, M. Syahroyni, Comparison of a and dynamic pathfinding algorithm with dynamic pathfinding algorithm for npc on car racing game, in: International Conference on Telecommunication Systems Services and Applications, 2018, pp. 1–6.

[15] S. Guo, H. Sun, Z. Chen, Path planning method for mobile robot based on improved genetic algorithm, Electron. World (2017).

[16] W. Li, W. jun Wu, H. min Wang, X. qi Cheng, H. jun Chen, Z. hua Zhou, R. Ding, Crowd intelligence in ai 2.0 era, Front. Inf. Technol. Electron. Eng. 18 (1) (2017) 15–43.

[17] C. Lamini, S. Benhlima, A. Elbekri, Genetic algorithm based approach for autonomous mobile robot path planning, Procedia Comput. Sci. 127 (2018) 180–189.

[18] E. Raboin, P. Svec, D. Nau, S.K. Gupta, Model-predictive target defense by team of unmanned surface vehicles operating in uncertain environments, in: IEEE International Conference on Robotics and Automation, 2013, pp. 3517–3522.

[19] E. Raboin, P. Švec, D.S. Nau, S.K. Gupta, Model-predictive asset guarding by team of autonomous surface vehicles in environment with civilian boats, Auton. Robots 38 (3) (2015) 261–282.

[20] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, F.E. Alsaadi, A survey of deep neural network architectures and their applications, Neurocomputing 234 (2017) 11–26.

[21] A.V. Mityakov, V.K. Varankin, Y.S. Tatarinov, Application of modern architectures of deep neural networks for solving practical problems, in: Xx IEEE International Conference on Soft Computing and Measurements, 2017, pp. 389–390.

[22] T. Dierks, S. Jagannathan, Output feedback control of a quadrotor uav using neural networks, IEEE Trans. Neural Netw. 21 (1) (2010) 50.

[23] R.S. Sutton, A.G. Barto, Reinforcement learning: An introduction, IEEE Trans. Neural Netw. 9 (5) (1998) 1054.

[24] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529.

[25] H.V. Hasselt, A. Guez, D. Silver, Deep reinforcement learning with double q-learning, Comput. Sci. (2015).

[26] Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, N. De Freitas, Dueling network architectures for deep reinforcement learning, 2015, arXiv preprint arXiv:1511.06581.

[27] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, I. Mordatch, Multi-agent actor-critic for mixed cooperative-competitive environments, in: Advances in Neural Information Processing Systems, 2017, pp. 6379–6390.

[28] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing atari with deep reinforcement learning, Comput. Sci. (2013).

[29] C. Wu, K. Parvate, N. Kheterpal, L. Dickstein, A. Mehta, E. Vinitsky, A.M. Bayen, Framework for control and deep reinforcement learning in traffic, in: IEEE International Conference on Intelligent Transportation Systems, 2018, pp. 1–8.

[30] P. Peng, Y. Wen, Y. Yang, Q. Yuan, Z. Tang, H. Long, J. Wang, Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play starcraft combat games, 2017, arXiv preprint arXiv:1703.10069.

[31] Y.F. Chen, M. Liu, M. Everett, J.P. How, Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning, in: Robotics and Automation (ICRA), 2017 IEEE International Conference on, 2017, pp. 285–292.

[32] P. Long, W. Liu, J. Pan, Deep-learned collision avoidance policy for distributed multiagent navigation, IEEE Robot. Autom. Lett. 2 (2) (2017) 656–663.

[33] J. Liu, W. Qi, X. Lu, Multi-step reinforcement learning algorithm of mobile robot path planning based on virtual potential field, in: International Conference of Pioneering Computer Scientists, Engineers and Educators, 2017, pp. 528–538.

[34] D.P. Romero-Martí, J.I. Núñez-Varela, C. Soubervielle-Montalvo, A. Orozco-De-La-Paz, Navigation and path planning using reinforcement learning for a roomba robot, in: Robotica, 2016 XVIII Congreso Mexicano de, 2017, pp. 1–5.

[35] L. Huang, H. Qu, M. Fu, W. Deng, Reinforcement learning for mobile robot obstacle avoidance under dynamic environments, in: Pacific Rim International Conference on Artificial Intelligence, 2018, pp. 441–453.

[36] Y. Cheng, W. Zhang, Concise deep reinforcement learning obstacle avoidance for underactuated unmanned marine vessels, Neurocomputing 272 (2018).

[37] Q. Guo, L. Zuo, R. Zheng, X. Xu, A hierarchical path planning approach based on reinforcement learning for mobile robots, in: International Conference on Intelligent Science and Big Data Engineering, 2013, pp. 393–400.

[38] A. Buitragomartinez, R. Fernando, De La Rosa, F. Lozanomartinez, Hierarchical reinforcement learning approach for motion planning in mobile robotics, in: Robotics Symposium and Competition, 2014, pp. 83–88.

[39] D. Zhao, H. Wang, K. Shao, Y. Zhu, Deep reinforcement learning with experience replay based on sarsa, in: 2016 IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, 2016, pp. 1–6.

[40] M.L. Littman, Markov games as a framework for multi-agent reinforcement learning, Mach. Learn. Proc. (1994) 157–163.

[41] K.D. Do, Z.-P. Jiang, J. Pan, Robust adaptive path following of underactuated ships, Automatica 40 (6) (2004) 929–944.

[42] T.I. Fossen, Marine control systems: guidance, navigation and control of ships, in: Rigs and Underwater Vehicles, Marine Cybernetics AS, 2002.

[43] A. Juliani, V.-P. Berges, E. Vckay, Y. Gao, H. Henry, M. Mattar, D. Lange, Unity: A general platform for intelligent agents, 2018, arXiv preprint arXiv:1809.02627.

[44] M. Plappert, Keras-rl, 2016, https://github.com/keras-rl/keras-rl.

[45] M. Mononen, Recast navigation, 2011, 2018, Google Project: http://code.google.com/p/recastnavigation [Accessed on Jauary 25, 2012].