ARTICLE IN PRESS
JID: NEUCOM [m5G;July 6, 2017;21:54]

Neurocomputing 000 (2017) 1–11

Contents lists available at ScienceDirect

Neurocomputing

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

Concise deep reinforcement learning obstacle avoidance for underactuated unmanned marine vessels

Yin Cheng, Weidong Zhang*

Department of Automation, Shanghai Jiaotong University, Shanghai 200240, PR China

ARTICLE INFO

Article history:
Received 8 April 2017
Revised 19 June 2017
Accepted 26 June 2017
Available online xxx

Communicated by Hongli Dong

Keywords:
Deep learning
Reinforcement learning
Obstacle avoidance
Nonlinear dynamics
Underactuated unmanned marine vessel

ABSTRACT

This research is concerned with the problem of obstacle avoidance for the underactuated unmanned marine vessel under unknown environmental disturbance. A concise deep reinforcement learning obstacle avoidance (CDRLOA) algorithm is proposed with the powerful deep Q-networks architecture to overcome the usability issue caused by the complicated control law in the traditional analytic approach. Furthermore, a comprehensive reward function is specifically designed for obstacle avoidance, target approaching, speed modification, and attitude correction. Compared to the analytic methods, the proposed algorithm based on reinforcement learning shows notable advantages in utility and extendibility. With the same CDRLOA system, the targets and the constraints are highly customizable for various of special requirements. Extensive experiments conducted have demonstrated the effectiveness and conciseness of the proposed algorithm.

© 2017 Published by Elsevier B.V.

1. Introduction

The application of the autonomous maritime system is becoming more and more prevalent due to its flexibility and versatility both in the civil and military field. For all kinds of application scenarios, it is of extreme importance to avoid obstacles such as rocks, floaters, debris and other ships. For the autonomous maritime vessels, the obstacle detection, information fusion, avoidance algorithm and the control strategy must be located onboard vessels. Consequently, the major challenge is the realization of real-time obstacle avoidance control strategy. Therefore, the applicable decision-making operator has an essential role in autonomous navigation and obstacle avoidance. Many positive results on this topic have been reported in the literature and readers are referred to the papers [1,2]. Lisowski and Smierzchalski [3] first applied the mathematical algorithm on the ship's dynamic mathematical model (static, kinetic, dynamic and matrix models) by generating a sequence of maneuvers.

However, mathematical algorithms have their particular limitations, and the avoidance performance intrinsically depends on the fine-grained models of obstacles and the dynamics of vessels. The slight changes of obstacles and the disturbance of the environment may lead to model's failure. Moreover, as the maritime system complexity increase, the mathematical algorithms become harder to design and deploy. In all the study as mentioned earlier of obstacles avoidance, there exist three main issues to be resolved: (1) The algorithms' ability to deal with the complex dynamic system is limited. Traditional mathematical algorithms are feeble to the changes and uncertainty of systems, while the weak representation capacity is the major flaw of traditional reinforcement learning approaches. (2) The control law obtained by most mathematical algorithms is formed as complicated formulas, while they are often too complicated to be deployed in practical applications. (3) The previous architectures are designed to accommodate some specified situations. Consequently, these approaches are not interoperable and portable for diverse and complex navigation requirements.

On the other side, the recent development of artificial intelligence area [4,5] has profound effects on the industrial world, which brings researchers powerful algorithms to characterize and control the extremely complex system under the changing environment. The ancient game of Go has long been viewed as the most difficult and challenging classic game, while the strategy of Go players can also be considered as the output of a controlled system with high complexity. David Silver and Aja Huang [6] managed to design a group of deep neural networks known as AlphaGo that are trained by the deep reinforcement learning (DRL) from games of self-play to beat human Go champions. Comparing with prior knowledge based traditional algorithms, DRL is with greater capacity to adapt complex system environment while it is capable of self-learning. Positive results in [7] have demonstrated that the successful control policies can be learned directly from DRL on

* Corresponding author.
E-mail addresses: wdzhang@sjtu.edu.cn, ycheng_mit@sjtu.edu.cn (W. Zhang).

http://dx.doi.org/10.1016/j.neucom.2017.06.066
0925-2312/© 2017 Published by Elsevier B.V.

Please cite this article as: Y. Cheng, W. Zhang, Concise deep reinforcement learning obstacle avoidance for underactuated unmanned marine vessels, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.06.066

challenging classic games under various scenarios. Besides, traditional reinforcement learning approaches have been applied in the autonomous movement of the four-wheeled robot [8]. After lots of self-learning processes, the robot car had succeeded in navigating in the environment with multiple obstacles. Fathinezhad and Derhami [9] proposed supervised fuzzy sarsa method for robot navigation by utilizing the advantages of both supervised and reinforcement learning algorithms.

Motivated by all of these theories and realistic reasons, this paper focuses on the field of advanced artificial intelligence approaches, e.g. deep learning architectures, reinforcement learning algorithms. A concise autonomous obstacles avoidance system, which is implemented by avoidance reward algorithm and deep reinforcement learning approach, is proposed for the complex unmanned marine vessels with unknown dynamics, taking autonomous surface vessels (ASVs) as the cases. The main contributions of this work can be summarized as follows:

(1) A concise deep reinforcement learning obstacles avoidance (CDRLOA) system is developed to deal with the complex navigation situations and the unknown dynamics of the environment. Combining the proposed avoidance reward algorithm and deep reinforcement learning (DRL) approach, CDRLOA system improves its controller with the self-play process. By applying the concise control strategies learned from the system, vessels are able to reach the target through highly simplified control instructions while avoiding the collision.

(2) Corroborating the effectiveness of the proposed algorithm with the obstacles avoidance navigation tasks, which consist of obstacles scattered within the domain, the destination to be reached, and the standard ASV with unknown environmental dynamics. The experiments have achieved satisfying results and shown the conciseness of the control strategies learned from CDRLOA system.

2. System description and preliminaries

2.1. Preliminaries

Throughout the paper, $|\cdot|$ denotes the absolute value for a scalar variable or the members of a specific set. $\mathbb{E}$ denotes the expectation operator, and $\mathbb{V}$ denotes the variation operator. $\hat{(\cdot)}$ is the estimation of $(\cdot)$ and $\delta(\cdot)$ is the Dirac function [10]. The case study of obstacles avoidance takes ASVs as the example. The horizontal motion of a surface vessel is unusually described by the motion components in surge, sway, and yaw [11]. Based on this, $\boldsymbol{\upsilon} = [u, v, r]^T \in \mathbb{R}^3$ and $\boldsymbol{\eta} = [x, y, \psi]^T \in \mathbb{R}^3$ are chosen as the velocity vector and position vector. Among these, (ψ) is the heading of the vessel and $(x,\ y)$ is the position in the earth-fixed inertial frame. The linear velocities $\boldsymbol{\upsilon} = [u, v, r]^T$ correspond to surge and sway, and for yaw in the body-fixed frame of vessel. r_0 is the obstacle detection radius. Fig. 1 illustrates the major concepts of the movement process in this case. The nonlinear dynamic equations of motion [12] can be expressed as:

$$\begin{aligned} &\dot{\boldsymbol{\eta}} = \boldsymbol{R}(\psi)\boldsymbol{\upsilon} \\ &\boldsymbol{M}\dot{\boldsymbol{\upsilon}} = \boldsymbol{\tau} - \boldsymbol{C}(\boldsymbol{\upsilon})\boldsymbol{\upsilon} - \boldsymbol{D}(\boldsymbol{\upsilon})\boldsymbol{\upsilon} - \boldsymbol{g}(\boldsymbol{\upsilon}) + \boldsymbol{\tau_w} \end{aligned} \tag{1}$$

where $\boldsymbol{R}(\cdot)$ is the 3 DOF rotation matrix for the horizontal motion of ASV. This matrix has the properties that $\boldsymbol{R}(\psi)^T\boldsymbol{R}(\psi) = \boldsymbol{I}$ and $\|\boldsymbol{R}(\psi)\| = 1$ for all ψ. Generally, $\frac{d}{dt}\{\boldsymbol{R}(\psi)\} = \dot{\psi}\boldsymbol{R}(\psi)\boldsymbol{S}$, where

$$\boldsymbol{J}(\eta) \stackrel{\text{3DOF}}{=} \boldsymbol{R}(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}, \boldsymbol{S} \stackrel{\text{3DOF}}{=} \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{2}$$

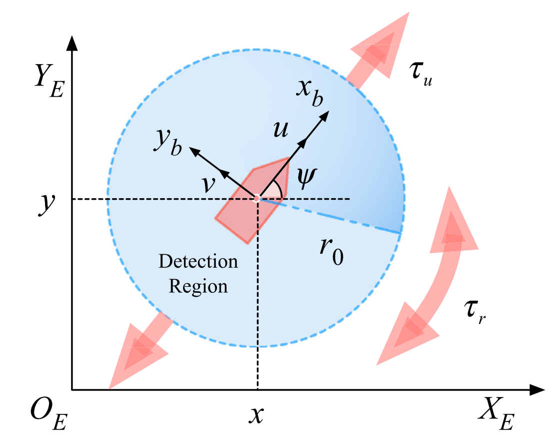


Fig. 1. Major components of the autonomous underactuated vessel.

The system inertia matrix $M = M^T = M_A + M_{RB} > 0$ is the combination of added mass matrix M_A and rigid-body matrix M_{RB}:

$$M_A = \begin{bmatrix} -X_{\dot{u}} & 0 & 0 \\ 0 & -Y_{\dot{v}} & -Y_{\dot{r}} \\ 0 & -Y_{\dot{r}} & -N_{\dot{r}} \end{bmatrix}, M_{RB} = \begin{bmatrix} m & 0 & 0 \\ 0 & m & mx_g \\ 0 & mx_g & I_z \end{bmatrix} \tag{3}$$

Similarly, the skew-symmetric matrix $C(\upsilon) = -C(\upsilon)^T$ of Coriolis and centripetal terms are also consisted of two parts:

$$C(\upsilon) = \begin{bmatrix} 0 & 0 & c_{13}(\upsilon) \\ 0 & 0 & c_{23}(\upsilon) \\ -c_{13}(\upsilon) & -c_{23}(\upsilon) & 0 \end{bmatrix} = C_A(\upsilon) + C_{RB}(\upsilon) \tag{4}$$

$$C_A(\upsilon) = \begin{bmatrix} 0 & 0 & Y_{\dot{v}}v + Y_{\dot{r}}r \\ 0 & 0 & -X_{\dot{u}}u \\ -Y_{\dot{v}}v & X_{\dot{u}}u & 0 \end{bmatrix} \tag{5}$$

$$C_{RB}(\upsilon) = \begin{bmatrix} 0 & 0 & -m(x_g r + v) \\ 0 & 0 & mu \\ m(x_g r + v) & -mu & 0 \end{bmatrix} \tag{6}$$

$D(\upsilon)$ is the nonlinear damping matrix for the system inertia

$$D(\upsilon) = \begin{bmatrix} d_{11}(\upsilon) & 0 & 0 \\ 0 & d_{22}(\upsilon) & d_{23}(\upsilon) \\ 0 & d_{32}(\upsilon) & d_{33}(\upsilon) \end{bmatrix} \tag{7}$$

with

$$\begin{aligned} d_{11}(\upsilon) &= -X_u - X_{|u|u}|u| - X_{uuu}u^2 \\ d_{22}(\upsilon) &= -Y_v - Y_{|v|v}|v| - Y_{|r|v}|r| \\ d_{23}(\upsilon) &= -Y_r - Y_{|v|r}|v| - Y_{|r|r}|r| \\ d_{32}(\upsilon) &= -N_v - N_{|v|v}|v| - N_{|r|v}|r| \\ d_{33}(\upsilon) &= -N_r - N_{|v|r}|v| - N_{|r|r}|r| \end{aligned} \tag{8}$$

The coefficients $[\{X_{(\cdot)}, Y_{(\cdot)}, N_{(\cdot)}\}$ are the so-called hydrodynamic derivatives that represent the hydrodynamic forces and moments acting on the vessel. $g(\upsilon) = [g_u, g_v, g_r]^T \in \mathbb{R}^3$ indicates the unmodeled dynamics.

The control input vector τ denotes the propulsion surge force and the yaw moment, which is given by

$$\tau = \begin{bmatrix} \tau_u & 0 & \tau_r \end{bmatrix}^T \in \mathbb{R}^3 \tag{9}$$

To further complicate the requirements, an underactuated vessel is under consideration. As can be observed in the propulsion force and moment vector τ, the independent actuators for the sway control are unnecessary. Accordingly, this vessel has more extensive applicability and fewer requirements for hardware. The disturbance from the environment can be represented by the vector

Please cite this article as: Y. Cheng, W. Zhang, Concise deep reinforcement learning obstacle avoidance for underactuated unmanned marine vessels, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.06.066

τ_w, which is given by

$$\tau_w = \begin{bmatrix} \tau_{wu} & \tau_{wv} & \tau_{wr} \end{bmatrix}^T \in \mathbb{R}^3 \tag{10}$$

where τ_{wu}, τ_{wv} are the disturbance force on the surge and sway, respectively, while τ_{wr} is the disturbance moment on the yaw. Unknown external disturbances are defined follow the reference [27], in which the cooperative control is conducted with these unknown dynamics.

$$\begin{aligned} \tau_{wu} &= -2\cos(0.5\,t)\cos(t) + 0.3\cos(0.5\,t)\sin(0.5\,t) - 3 \\ \tau_{wv} &= 0.01\sin(0.1\,t) \\ \tau_{wr} &= 0.6\sin(1.1\,t)\cos(0.3\,t) \end{aligned} \tag{11}$$

2.2. Problem formulation

In the autonomous obstacles avoidance process, the main task is to manipulate the vessel to achieve the given destination and prevent the collision with obstacles. Success in this endeavor depends on the development of an efficient real-time intelligence algorithm, such that: (1) The computation for control strategies are conducted simultaneously with the observation process for the states and obstacles. (2) Sequences of control strategies manipulate the vessel to avoid the obstacles and reach its destination. (3) Control strategies in each sequence are convenient for the engineering implementation.

The obstacles avoidance functionality is realized by a finite sequence of control strategies. The process of implementation is to design a discrete control law τ for the vessel using its current observations $obs_t \in \mathbb{R}^{N_{obs}}$ and historical observations $obs_{t-1},\ i = t - T_p, \dots, t-1$. Where N_{obs} is the number of parameters in each observation, and T_p is the historical time steps for observations to be considered in the proposed algorithm, such that

$$\begin{aligned} \tau_t &= f_{\mathrm{CDRLOA}}\left(obs_t, obs_{t-1}, \dots, obs_{t-T_p}\right) \in \mathbb{R}^2 \\ obs_t &= [\sigma_t, \upsilon_t, \eta_t, \tau_{t-1}] \in \mathbb{R}^9 \end{aligned} \tag{12}$$

where f_{CDRLOA} is the proposed AI controller, and $\sigma_t \in \mathbb{R}^1$ is a binary value which represents the searching results for obstacles with sonar at time t. The general framework of obstacle avoidance for the underactuated autonomous vessel is demonstrated in Fig. 1. In addition, the vessel model is derived under the following assumption [13].

Assumption 1. The vessel model under the obstacles avoidance tasks is horizontal and with three DOF.

- The dynamics of vessel associate with the motion in heave, roll, and pitch are ignored within the obstacles avoidance task.
- The mass distribution is homogeneous and *xz*-plane of symmetry.
- The center of gravity and buoyancy are located vertically on the z-axis.

Assumption 2. The vessel fulfills its tasks by applying a finite number of control behaviors.

- The length of control sequence T_c is a fixed constant.
- The vessel is able to update its information at regular intervals.

3. Design of the CDRLOA algorithm

Given the operation states of the vessel and the observed information in real time, the obstacles avoidance problem is achieved by applying a series of control behaviors with a fixed length. Accordingly, the data acquisition module is necessary to collect the current states of vessel, including position $(x,\ y)$, heading (ψ), surge (u) ,sway (v), and the angular rate (r). Besides, to avoid obstacles in real time, the detecting equipment such as sonar checks for the existence of any obstacles denoted by η_t within a fixed radius at the moment t. Together with states of vessel and the control behaviors τ_{t-1} in the last moment, the complete observation information obs_t is constituted as $[\sigma_t, \upsilon_t, \eta_t, \tau_{t-1}] \in \mathbb{R}^9$. The data fusion module integrates the heterogeneous data and stores the historical observation vectors $obs_{t-1},\ i = t - T_p, \dots, t-1$ to improve the performance of CDRLOA system. The decision-making module learns the sequence of control behaviors through deep reinforcement learning approach founded on the fusion data and historical observations provided by the data fusion module.

In this section, the main components of CDRLOA (see Fig. 2) and their interactions are described. Furthermore, the specific design details and features are also covered.

3.1. Data fusion module with convolutional neural networks

The data fusion module takes the observations generated by the data acquisition module as the raw input. Each observations vector obs_t consists of 3 parts, including the kinestate (υ_t, η_t) of vessel, the existence state σ_t of obstacles at moment t, and the previous control behavior τ_{t-1}. Among these, the existence states σ_t is a logical variable which represents if there exist any obstacles within a given radius r_d. Unlike the traditional detection method, the precise locations of obstacles are not required for the proposed algorithm. The existence of obstacles σ_t provides enough information for obstacles avoidance system to make decisions. Due to the sections of the observation vector obs_t are qualitatively different, deep convolutional neural networks (CNN) are applied to the data fusion module to extract the joint information across the different types of variables.

CNN is a special kind of artificial neural network that is biologically-inspired variants of multilayer perceptrons [14]. There contain complex tissue and structures of cells in animal's visual cortex, and these cells are only sensitive to the small range of the entire visual field, known as the receptive field [15]. These cells in the visual cortex act as a local filter sweeps through the entire visual field to exploit the relationship and interactions within the local receptive field. The design of CNN is enlightened by the powerful visual processing system of animals.

Sparse connectivity and shared weights are the two key points of CNN architecture. Sparse connectivity means convolutional filter kernels are connected sparsely with decreased cells along the layers. In this way, the filter kernels are forced to make the strongest response for the spatially local input patterns with the limited hidden cells. Furthermore, these filters are replicated across the entire visual fields including their connectivity styles and internal parameters, which are the characteristic of shared weights [16]. By this means, the filters are able to learn the commonalities and relationships across the different parts of the entire data blocks.

If we denote the x as the input data and h^k as the kth feature map, while the filters are determined by the weights W^k with given bias b_k. The action function is a nonlinear mapping, while the function used in this note is known as Rectified Linear Units (ReLU) [17] which is proved to be more biologically supportive. The computational process can be expressed as

$$h^k_{ij} = \mathrm{activation}\left(\left(W^k * x\right)_{ij} + b_k\right) \tag{13}$$

where the asterisk $*$ denotes the convolutional operator

$$o[m,n] = f[m,n] * g[m,n] = \sum_{u=-\infty}^{\infty} \sum_{v=-\infty}^{\infty} f[u,v]\, g[m-u, n-v] \tag{14}$$

The size of output feature map is decided as

$$\mathrm{Output}_{\mathrm{size}} = \frac{\mathrm{Input}_{\mathrm{size}} - \mathrm{Kernel}_{\mathrm{size}} + 2 \times \mathrm{Padding}_{\mathrm{size}}}{\mathrm{Stride}} + 1 \tag{15}$$

ARTICLE IN PRESS
JID: NEUCOM [m5G;July 6, 2017;21:54]

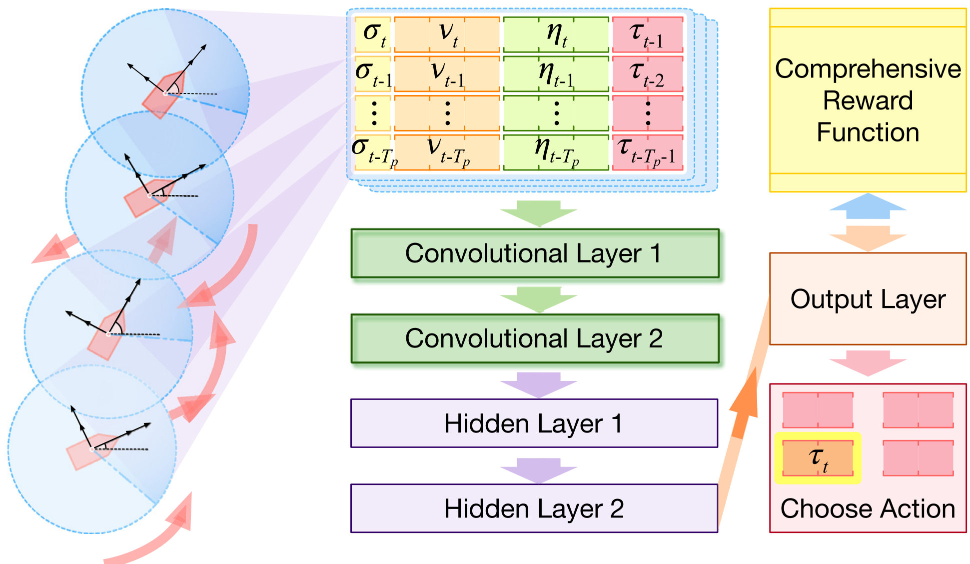
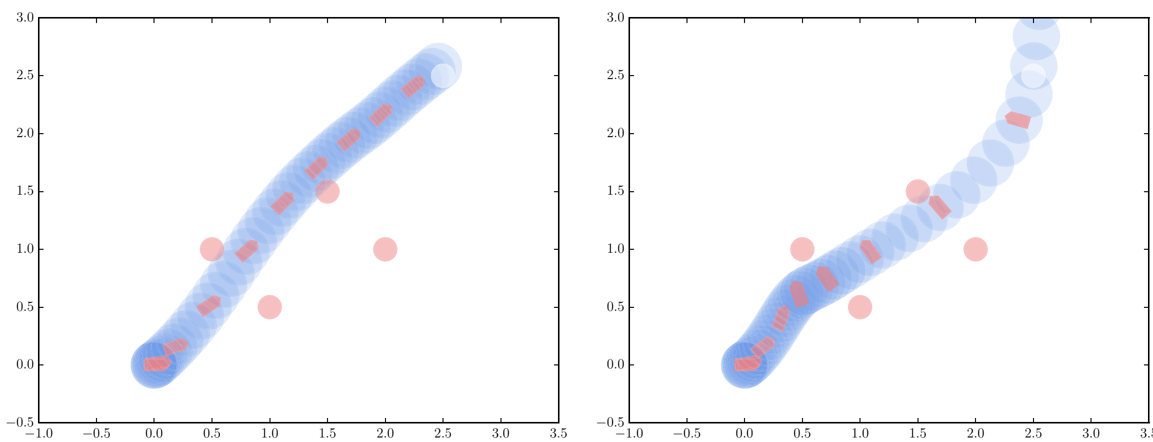


Fig. 2. Framework of CDRLOA system for obstacle avoidance.



(a) Training without disturbance (b) Training with disturbance

Fig. 3. Testing results for disturbance within 300 epochs.

Exploiting more than one single observation vector is an effective way to enhance the quality of the encoded outputs. CDRLOA system not only takes the current obs_t but also T_p historical observed vectors ($obs_{t-i}, i \in 1, \ldots, T_p$). Therefore, the input for CNN module at the moment t can be expressed as

$$\boldsymbol{X}_{\text{CNN}}(t) = \left[obs_t \quad obs_{t-1} \quad \cdots \quad obs_{t-T_p}\right]^T = \begin{bmatrix} \sigma_t & \upsilon_t & \eta_t & \tau_{t-1} \\ \sigma_{t-1} & \upsilon_{t-1} & \eta_{t-1} & \tau_{t-2} \\ \vdots & \vdots & \vdots & \vdots \\ \sigma_{t-T_p} & \upsilon_{t-T_p} & \eta_{t-T_p} & \tau_{t-T_p-1} \end{bmatrix} \tag{16}$$

After the convolutional process layer upon layer, each input observation matrix $X_{\text{CNN}}(t)$ will be mapped into a comprehensive state vector StateID_t which characterizes the operation states of the vessel, while the historical information on control behaviors and the existence of obstacles are also taken into consideration.

3.2. Decision making module with deep reinforcement learning

By building on the bases of data acquisition module and data fusion module, the autonomous control sequences are learned in the decision-making module. The decision-making module takes the StateID_t encoded by the data fusion module as the inputs, exploring and exploiting the latent optimal control behaviors by deep reinforcement learning approach which is aimed at controlling intelligent agents so that the given target tasks can be achieved even in the unknown environments. A reinforcement learning problem contains various integral components such as states, actions, transitions, rewards, policies and values [18]. In this note, we consider the autonomous vessel as the agent to be controlled, and the encoded StateID_t, control behaviors τ_t, the performance of obstacles avoidance are the states, actions, and rewards in the algorithm framework, respectively.

The mathematical nature of the reinforcement learning problem can be viewed as the Markov Decision Process (MDP) under the discrete time. The agent is the major component that interacts with the environment which represents the unknown dynamics and obstacles to be avoided in this note. At each time step t, the agent observes a state $s_t \in S$ which corresponds to the encoded StateID_t of vessel. Then an action $a_t \in A$ is selected to make a transition from current state s_t to new state s_{t+1}. With every transition, the agent receives a immediate reward $r_t = r(s_t, a_t, s_{t+1}) \in \mathbb{R}$. This process corresponds to the control behaviors and

Please cite this article as: Y. Cheng, W. Zhang, Concise deep reinforcement learning obstacle avoidance for underactuated unmanned marine vessels, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.06.066

performance for the vessel to avoid the obstacles and reach the destination. In this case, not all the control actions are feasible, thus the practical action set for vessel can be expressed as $A(s) \subseteq A = \left\{a^1, a^2, \ldots, a^K\right\}$. Transitions T for states satisfy the Markovian property

$$P\left(s_{t+1}|s_t, a_t, s_{t-1}, a_{t-1}, \ldots\right) = P(s_{t+1}|s_t, a_t) = T(s_t, a_t, s_{t+1}) \tag{17}$$

In MDP, the discounted sum of immediate rewards is defined as return $R(h)$

$$R(h) = \sum_{t=1}^{T} \gamma^{t-1} r(s_t, a_t, s_{t+1}) \tag{18}$$

where agent's policy π can be viewed as the controller of vessel and the selected actions are determined by the policy. $\gamma \in [0, 1)$ is the discounted factor and h stands for the historical trajectory of agent. By selecting the optimal policy π^* through probability density $p^\pi(h)$, the maximal return can be achieved and the given tasks are completed at the same time

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, \mathbb{E}_{p^\pi(h)}[R(h)] \tag{19}$$

The control policies learnt from CDRLOA system is determined by the state-action value function $Q^\pi(s, a) \in \mathbb{R}$ which reflects the prediction of future rewards.

$$Q^\pi(s, a) = \mathbb{E}_{p^\pi(h)}[R(h)|s_1 = s, a_1 = a] \tag{20}$$

When a specific action a_t is applied on the state s_t, the agent will get an expected immediate reward $r(s_t, a_t)$

$$r(s_t, a_t) = \mathbb{E}_{p(s_{t+1}|s_t, a_t)}[r(s_t, a_t, s_{t+1})] \tag{21}$$

By recursion, $Q^\pi(s, a)$ can be expressed as

$$Q^\pi(s_t, a_t) = r(s_t, a_t) + \gamma \mathbb{E}_{\pi(a_{t+1}|s_{t+1})p(s_{t+1}|s_t, a_t)}[Q^\pi(s_{t+1}, a_{t+1})] \tag{22}$$

Diffident policy corresponds to diffident reward in the future, the optimal state-action value function $Q^*(s, a)$ at stat s for action a gets the maximal prediction rewards

$$Q^*(s, a) = \max_\pi Q^\pi(s, a) \tag{23}$$

Based on the optimal state-action value function $Q^*(s, a)$, the optimal policy $\pi^*(a|s.)$ is naturally expressed as

$$\pi^*(a_t|s_t) = \delta\left(a_t - \underset{a'}{\operatorname{argmax}}\, Q^*\left(s_t, a'\right)\right) \tag{24}$$

where the $\arg\max_{a'}$ notion denotes the action a' maximizes the optimal value function, while it looks best just after one step of lookahead based on the value function $Q^*(s_t, a')$. This kind of action focuses on the short term. Thus it is considered to be "greedy". In practice, to scale with the number of states and even the continuous problem with infinite states, an elaborate exploration mechanism is preferable. ϵ-greedy policies [19,20] are applied as follows:

$$\pi(a|s) \leftarrow \begin{cases} 1 - \varepsilon + \varepsilon/|A| & \text{if } a = \operatorname{argmax}_{a' \in A} Q^\pi\left(s, a'\right), \\ \varepsilon/|A| & \text{otherwise}, \end{cases} \tag{25}$$

where $\varepsilon \in (0, 1]$ is a tuning parameter that denotes the randomness.

In the condition of small action space and state space, the recursion equations can be solved iteratively. Thus, the state-action value function can be well estimated. However, as the scale of action space or state space gets large, the explosion of computation costs makes it infeasible to solve the equations directly especially when the state space is continuous. Accordingly, an approximator [21] for the value function is essential in the practical. In this note, the deep neural network approximator [22] for value function is applied to minimize the loss function as follows:

$$\begin{aligned} L_i(\theta_i) &= \mathbb{E}_{s,a,r}\left[\left(\mathbb{E}_{s'}[y|s, a] - Q(s, a; \theta_i)\right)^2\right] \\ &= \mathbb{E}_{s,a,r,s'}\left[y - Q(s, a; \theta_i)^2\right] + \mathbb{E}_{s,a,r}[\mathbb{V}_{s'}[y]] \end{aligned} \tag{26}$$

The basic approximator for value function is tending to lose the memories that encountered in the past episodes. A crucial technique known as experience replay [23,24] maintains a buffer of the previous experiences and breaks up the correlations between the neighboring sequential samples. In this memory pool, a quadruple $e_t = (s_t, a_t, r_t, s_{t+1})$ is added as one experience record at each timestep. Moreover, batches of experiences are randomly sampled from the memory pool $D_t = \{e_1, e_2, \ldots, e_t\}$ during the training phase. By this means, the greater exploration efficiency can be achieved and the non-stationary aspect of training data is reasonably avoided.

3.3. Avoidance reward function

The avoidance reward function is the key component of CDRLOA, and it implicitly specifies the goal of tasks to be solved. This function acts as the only feedback system that evaluates the performance of the control behaviors with one scalar signal. Compared to the supervised learning algorithm, this signal is more evaluative than being instructive [25], making it more difficult to design the proper reward function. The proposed avoidance reward function specifies the immediate reward obtained for being a state. Unlike simple games with the definite situations of winning, losing and draw, the reward function of obstacles avoidance problem demands specifically assigned rewards for some particular states and action. Completion of the obstacles avoidance tasks requires the extended control sequence. Therefore the objectives and constraints are well designed in the intermediate subgoals of the reward function as follows:

To begin with, the distance between the vessel and its destination determines how close the target is approached. Thus the distance term assigns the negative values for all the position (x, y) of the vessel. The closer to the destination, the lesser immediate punishments are imposed on the agent.

$$R_{\text{distance}} = -\lambda_{\text{distance}} \sqrt{\left(x - x_{\text{goal}}\right)^2 + \left(y - y_{\text{goal}}\right)^2} \tag{27}$$

Suppose r_0 denotes the radius of the circular obstacles while the safe distance between the vessel and the center of the obstacle is twice as much. The danger of collisions can be expressed as

$$R_{\text{collisions}} = -\lambda_{\text{collisions}} \bigvee_{i=1}^{N_{\text{obs}}} \left(\sqrt{\left(x - x_{\text{obs}_i}\right)^2 + \left(y - y_{\text{obs}_i}\right)^2} < 2r_0\right) \tag{28}$$

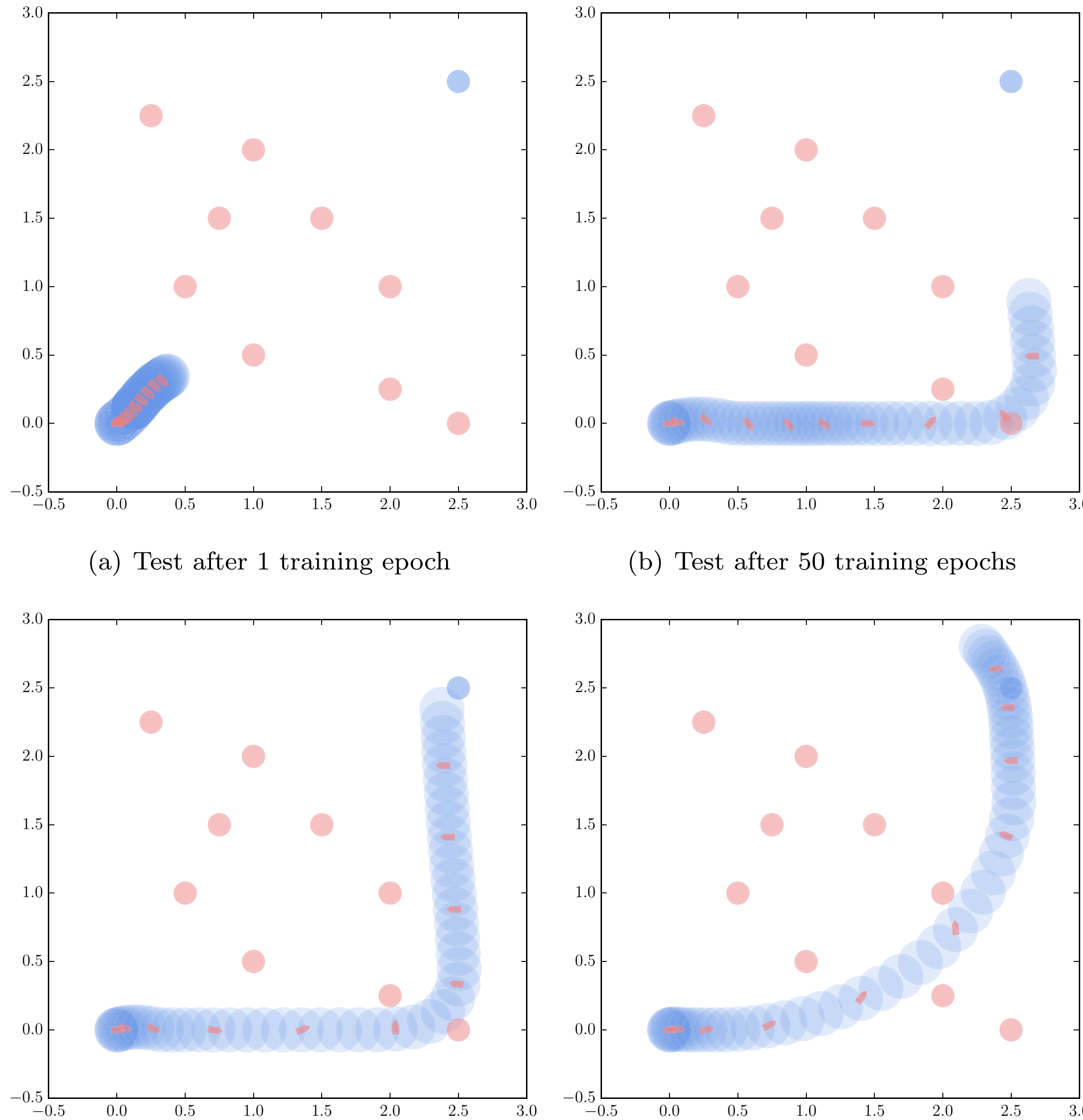
where N_{obs} is the number of obstacles to be avoided, and $\vee$ is the logical OR symbol. In practical, the existence of nearby obstacles can be detected through sonar equipment conveniently.

The excessive speed for vessel nearby the destination is unsafe and needless, thus the end speed term is defined as

$$R_{\text{end}} = \begin{cases} \lambda_{\text{end}}/(|u| + |v|) & \sqrt{\left(x - x_{\text{goal}}\right)^2 + \left(y - y_{\text{goal}}\right)^2} < 2r_0 \\ 0 & \text{else} \end{cases} \tag{29}$$

In some cases, if the linear velocity of sway v greatly exceeds surge u, the vessel may encounter the drift phenomenon. As a precautionary measure, the drift reward term is given by

$$R_{\text{drift}} = \begin{cases} -\lambda_{\text{drift}} & |u| < |v| \\ 0 & \text{else} \end{cases} \tag{30}$$



(a) Test after 1 training epoch (b) Test after 50 training epochs

(c) Test after 100 training epochs (d) Test after 2500 training epochs

Fig. 4. Testing results for obstacle avoidance with basic reward function.

The influence of different types of reward terms is weighted by a constant vector $\boldsymbol{\lambda}$ and the relative sizes of its element are important. Suppose that the agent may encounter a large negative reward before it can reach the target and receive a minor positive reward at last. In this note, the various reward terms are combined into a heterogeneous avoidance reward function as follows

$$R(obs_t) = \boldsymbol{\lambda}^T \boldsymbol{R} = \begin{bmatrix} \lambda_{\text{distance}} \\ \lambda_{\text{collision}} \\ \lambda_{\text{end}} \\ \lambda_{\text{drift}} \end{bmatrix}^T \begin{bmatrix} R_{\text{distance}}(x_t, y_t) \\ R_{\text{collision}}(x_t, y_t) \\ R_{\text{end}}(x_t, y_t, u_t, v_t) \\ R_{\text{drift}}(u_t, v_t) \end{bmatrix} \tag{31}$$

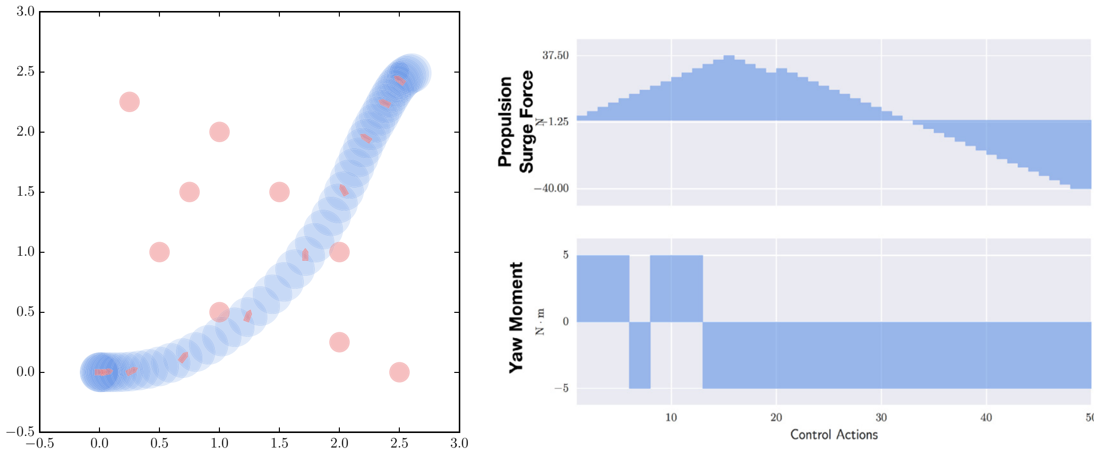
4. CDRLOA for the underactuated vessel with unknown dynamics

In this section, CDRLOA system is proposed as the combination of data as mentioned earlier acquisition module, data fusion module, and decision-making module. The system is able to collect the current operational state of the vessel and evaluate whether the obstacles are at a safe distance. Based on the data contained in the memory pool, lots of self-play trials are conducted to find out the elegant control strategies under various circumstances. Once the training process is completed, the vessel is able to automatically navigate the obstacles and achieve the destination under the commands of CDRLOA system.

The vessel is an underactuated system with three DOF. Therefore the control vector can be expressed as $\boldsymbol{\tau}(t) = \left[\tau_u(t) 0 \tau_r(t)\right]^T$. By virtue of the proposed system, complex tasks under unknown disturbance from the environment can be realized by applying a group of fairly concise control behaviors. Specifically, there are only two control behaviors both for propeller and rudder respectively thus they are advantageous for the engineering implementation.

$$\tau_u(t) = \begin{cases} \tau_u(t-1) + \Delta F_u & \text{Increase the thrust} \\ \tau_u(t-1) - \Delta F_u & \text{Decrease the thrust} \end{cases} \tag{32}$$

Please cite this article as: Y. Cheng, W. Zhang, Concise deep reinforcement learning obstacle avoidance for underactuated unmanned marine vessels, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.06.066



(a) Test after 2500 training epochs (b) The corresponding control sequence

Fig. 5. Testing results and control sequence learned with end-enhancement reward function.

where $\tau_u(0) = 0$, and ΔF_u is the base increment for thrust. Similarly, the $\tau_r(0) = 0$, F_r is the base thrust for the rudder and control behaviors of rudder are even simpler than propeller's

$$\tau_r(t) = \begin{cases} F_r & \text{Apply force to the rudder from one side} \\ -F_r & \text{Apply force to the rudder from other side} \end{cases} \quad (33)$$

In CDRLOA system, the decision making module is designed with the Q value function approach, therefore the networks to be trained in this system is noted as Q-networks and the intermediate target is noted as $\hat{Q}$-networks. In the first place the linear velocities $\boldsymbol{\upsilon}_t$, the position $\boldsymbol{\eta}_t$ of the vessel and the existence of nearby obstacles $\boldsymbol{\sigma}_t$ should be initialized as zeros. Note that the previous control behaviors $\tau_u(t-1)$ and $\tau_r(t-1)$ are the components of the observation vector obs_t, the $\tau_u(0)$ and $\tau_r(0)$ are likewise initialized as zeros. Two different repositories $\boldsymbol{D}_o$, $\boldsymbol{D}_e$ are established to store the T_p historical observation vectors and N_e transition quadruples $(\boldsymbol{X}_{\text{CNN}}(t), \boldsymbol{\tau}_t, R_t, \boldsymbol{X}_{\text{CNN}}(t+1))$ known as experiences respectively. In each navigation episode, T_c time steps are considered to be the length of a control sequence, while the corresponding control behavior is applied in order at every time step. In order to compose the observation vector obs_t at time step t, the previous control behaviors are read from the repository $\boldsymbol{D}_o$. With T_p historical observations, the current comprehensive state for Q-networks at time step t is expressed as $\boldsymbol{X}_{\text{CNN}}(t) = \left[obs_t obs_{t-1} \cdots obs_{t-T_p}\right]^T$. To explore the better control strategies, the control behavior $\boldsymbol{\tau}_t$ is chosen randomly with the probability of ϵ on one hand, while on the other hand $\boldsymbol{\tau}_t$ is chosen through Q-networks as $\text{argmax}_{\boldsymbol{\tau}' \in A} Q^{\pi}\left(\boldsymbol{X}_{\text{CNN}}(t), \boldsymbol{\tau}'; \theta\right)$. Once the $\boldsymbol{\tau}_t$ is applied, the new velocities $\boldsymbol{\upsilon}_{t+1}$ and positions $\boldsymbol{\eta}_{t+1}$ can be obtained with the kinetics of vessel and the dynamics from the environment. Each time when an agent transforms from one state to another, it receives an immediate reward R_t from the environment to show the performance of this transition event. New observation obs_{t+1} and encountered experience are stored in the repository $\boldsymbol{D}_o$ and $\boldsymbol{D}_e$ respectively. To break the strong correlations between the sequential items, the minibatch of experiences $\left(\boldsymbol{X}_{\text{CNN}}(j), \boldsymbol{\tau}_j, R_j, \boldsymbol{X}_{\text{CNN}}(j+1)\right)$ are sampled randomly from $\boldsymbol{D}_e$. The intermediate target $R_j + \gamma \max_{\boldsymbol{\tau}'} \hat{Q}\left(\boldsymbol{X}_{\text{CNN}}(j+1), \boldsymbol{\tau}'; \theta^-\right)$ is denoted as y_j, and the Q-networks are trained with the loss function

$$\left(y_j - Q\left(\boldsymbol{X}_{\text{CNN}}(j), \boldsymbol{\tau}_j; \theta\right)\right)^2 \quad (34)$$

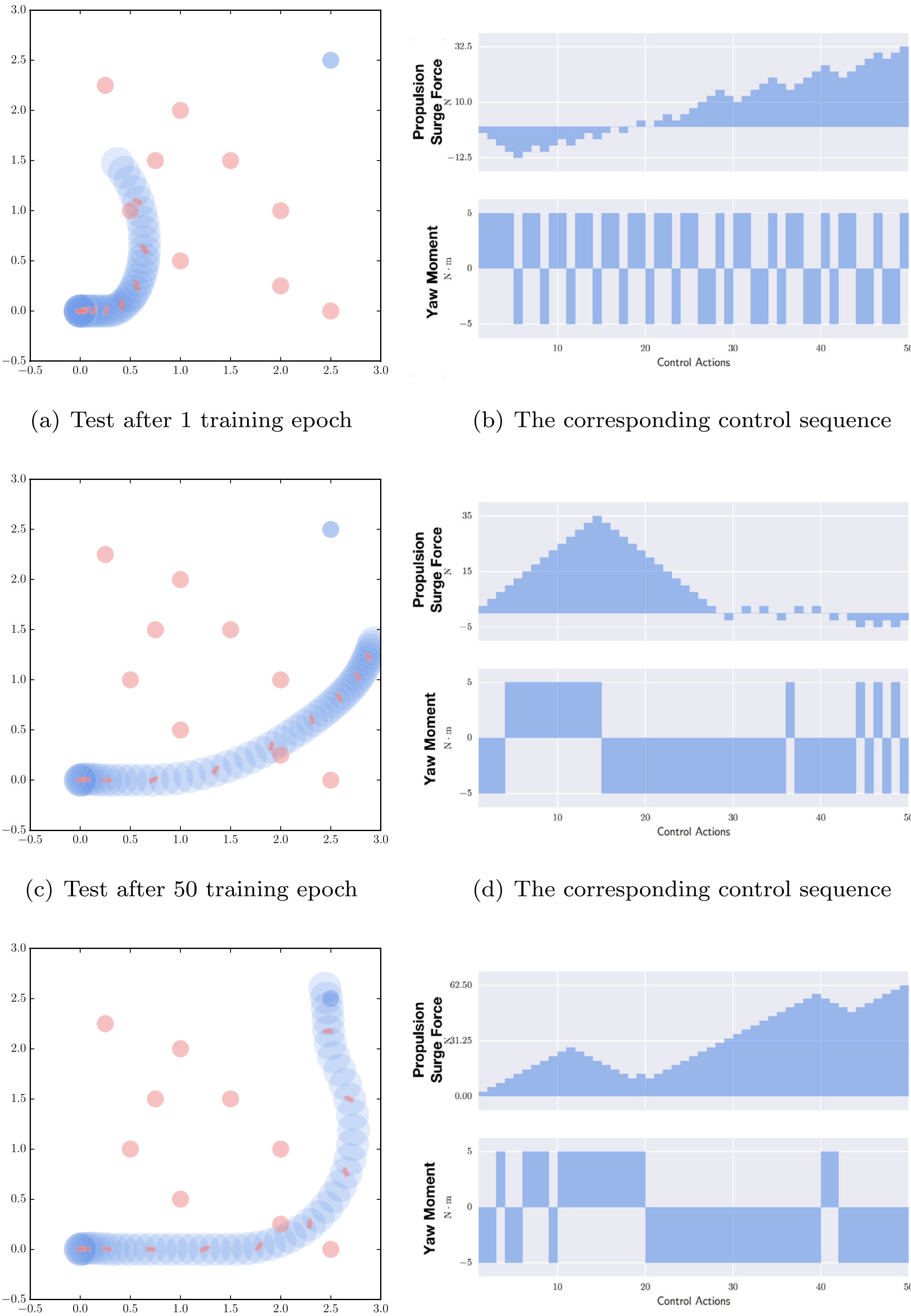
where the regular updates for the target are carried out for every C steps. Once the training process is completed, the weights parameters of networks are fixed, and CDRLOA system is ready for the obstacles avoidance tasks in real time.

5. Illustrative examples

In this section, the illustrative examples are presented to collaborate the effectiveness of proposed CDRLOA system. This system is developed on the Theano [26] deep learning framework and the Intel Math Kernel Library. In this note, these experiments consider a wide range of cases, from basic reward functions to the comprehensive reward function. In all experiments, the same parameters of kinetics model are used, the vessel has no knowledge of the disturbance of the environment. The experiments results are illustrated in figures demonstrating the full moving trails, heading, the distribution of obstacles and the target to achieve. The model parameters are implemented following the [27] in all cases. Specifically, the memory length T_p for historical observations is set as 3, and the length T_c for control sequence is set as 50. In practical, once the training process is completed, the arbitrary length of the control sequence can be generated by the application of controller learned. The safe distance r_0 is set as 0.8 m. The base increment for thrust ΔF_u is 5(N), and the base moment of force F_r is 5(N·m).

The training process and algorithms are the same with and without the disturbances. However, the disturbances make it much more difficult for an agent to train the controller and find out the appropriate control strategy. Under the same conditions except for disturbances, the difficulties in the training process are different. In this case, without the disturbance, the agent can find out the appropriate solutions within 300 training epochs (Fig. a), while the performance (Fig. b) with disturbance under 300 training epochs is not acceptable.

The symbols and color codes are applied as follows: In the position plots, the red circles are the obstacles to be avoided. The dark blue circle at the top right is the destination to be achieved. A trial of big light blue circles demonstrates the sonar detection range for the obstacles. The pentagons placed inside the light blue circles are the sketch maps of the vessel every five-time step, and the acute angle in each pentagon represents the instantaneous heading of the vessel.



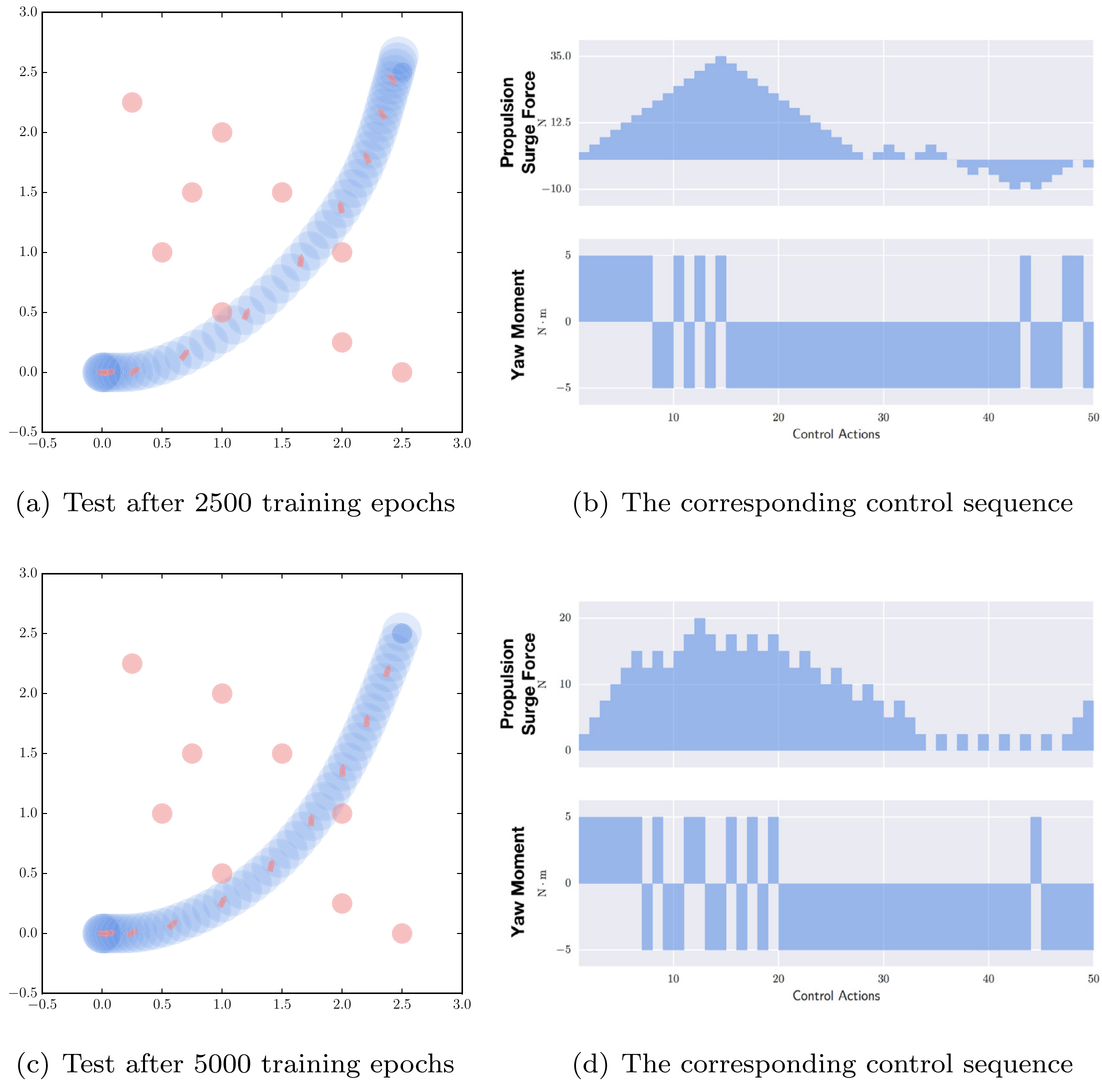
(a) Test after 1 training epoch (b) The corresponding control sequence

(c) Test after 50 training epoch (d) The corresponding control sequence

(e) Test after 100 training epoch (f) The corresponding control sequence

Fig. 6. The experiments with comprehensive reward function in the early stage.

Please cite this article as: Y. Cheng, W. Zhang, Concise deep reinforcement learning obstacle avoidance for underactuated unmanned marine vessels, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.06.066



(a) Test after 2500 training epochs
(b) The corresponding control sequence
(c) Test after 5000 training epochs
(d) The corresponding control sequence

Fig. 7. Final performance and fine-tuning improvement with comprehensive reward function.

5.1. *Obstacle avoidance with the basic reward function*

For comparison, the performance of basic avoidance reward function is demonstrated, while only the distance reward term R_{distance} and the obstacles reward term $R_{\text{obstacles}}$ are considered under the circumstance. The basic reward function can be expressed as

$$R_{\text{basic}}(obs_t) = \lambda_{\text{distance}} R_{\text{distance}}(x_t, y_t) + \lambda_{\text{obstacles}} R_{\text{obstacles}}(x_t, y_t) \tag{35}$$

where the reward factors $\lambda_{\text{distance}}$ and $\lambda_{\text{obstacles}}$ are set as -1.5 and -30, respectively. The detailed definition of R_{distance} and $R_{\text{obstacles}}$ are described in the Section 3.

Experiments for obstacles avoidance with the basic reward function are given in Fig. 4. It can be observed that the controller does not know how to maneuver the vessel in the first place, therefore the vessel swings around at the starting point. With the learning process goes on, the vessel finds it benefits sailing in the EW direction (the distance to the destination are decreased at the same time). When the vessel has already passed the destination in the EW direction, the distance reward term R_{distance} will no further increase unless the vessel approaches the destination in the NS direction as well. Eventually, the vessel comes up with the appropriate control sequence to avoid the obstacles and achieve the destination in the distance. However, the vessel has already overshot the destination a lot in the last few control behaviors, in this case, the further improvement on the basic reward function is necessary accordingly.

5.2. *Obstacle avoidance with the end-enhancement reward function*

The problem in overshooting the destination is that the reward does not change a lot near the destination. Moreover, the vessel does not get the formation of the destination within sight whereas the existence of obstacles is detected all along. Accordingly, the vessel has little time to reduce the terminal speed in the last few control behaviors. Based on the requirement for the terminal speed deduction, the end-enhancement reward term R_{end} is added to the basic avoidance reward function as follows:

$$R_{\text{end-enhancement}}(obs_t) = \begin{bmatrix} \lambda_{\text{distance}} \\ \lambda_{\text{obstacles}} \\ \lambda_{\text{end}} \end{bmatrix}^T \begin{bmatrix} R_{\text{distance}}(x_t, y_t) \\ R_{\text{obstacles}}(x_t, y_t) \\ R_{\text{end}}(x_t, y_t, u_t, v_t) \end{bmatrix} \tag{36}$$

where the λ_{end} is set as 30, and the detailed definition of R_{end} is illustrated in the Section 3 as well. With the R_{end}, the last few velocities u_t and v_t are limited, and the slower vessel is near

ARTICLE IN PRESS

Algorithm 1 CDRLOA algorithm for the underactuated vessels.

Input: the linear velocities $\boldsymbol{\upsilon}_t = [u_t, v_t, r_t]^T$, the position vector $\boldsymbol{\eta}_t = [x_t, y_t, \psi_t]^T$, and the detection results σ_t for obstacles nearby.
Output: weights parameter θ^* for CDRLOA networks

```
Initialize the υ_1 = η_1 = [000]^T, τ_u(0) = τ_r(0) = σ_1 = 0
Initialize the experience replay repository D_e to capacity N_e
Initialize the historical observations repository D_o to capacity T_p
Initialize the Q-networks with random weights θ
Initialize the target Q̂-networks with weights θ^- = θ
for episode = 1, M do
    for t = 1, T_c do
        Fetch the historical observations from D_o
        Consolidate the current observations as obs_t = [σ_t, υ_t, η_t, τ_{t-1}]^T
        if t ≤ T_p then
            Set obs_k = 0 ∈ ℝ^9, k = t − T_p, ..., 0
        end if
        Form the state as X_CNN(t) = [obs_t obs_{t-1} ··· obs_{t-T_p}]^T
        Selecting a random control behavior τ_t with probability ε
        otherwise select control behavior τ_t = argmax_{τ'∈A} Q^π(X_CNN(t), τ'; θ)
        Execute τ_t = [τ_u(t) 0 τ_r(t)]^T and calculate υ_{t+1}, η_{t+1}
        Detect the existence of obstacles σ_{t+1} and calculate X_CNN(t+1)
        Calculate the comprehensive avoidance reward R_t
        Store the new observation obs_{t+1} in repository D_o
        Store the experience (X_CNN(t), τ_t, R_t, X_CNN(t+1)) in D_e
        Sample mimibatch (X_CNN(j), τ_j, R_j, X_CNN(j+1)) randomly in D_e
        Set y_j = R_j + γ max_{τ'} Q̂(X_CNN(j+1), τ'; θ^-)
        Train the networks with (y_j − Q(X_CNN(j), τ_j; θ))^2 as loss function
        Reset the Q-Networks Q̂ = Q every 50 steps
    end for
end for
return weights parameter θ* for Q-networks
```

the destination, the greater reward the vessel receives. The end-enhancement term takes effect only near the target; the performance is the same with basic reward function at the early stage. The final control sequence with end-enhancement reward function and the learned control actions are shown in Fig. 5.

5.3. Obstacle avoidance with the comprehensive reward function

The end-enhancement reward function can solve the over-shooting problem within destination's neighboring areas. The vessel slows down as soon as it has passed the target, approaching the goal under the unknown dynamics. However, without the limitation of sway (v), sometimes the vessel may drift laterally around the target to prevent the overshooting problem. As formulated above, the vessel is underactuated, and thus the sway can not be controlled with an individual propeller. If the sway (v) is much faster than the surge (u), the vessel could be at substantial risks under certain conditions. To overcome this problem, a drift-limitation reward term R_{drift} is further added on the end-enhancement reward function. In this way, the comprehensive reward function consisting of four reward terms is proposed in Eq. (27), where λ_{drift} is set as 3. By punishing the motion states that the sway is larger than the surge, the vessel tries to limit the surge during the tasks as much as possible.

It can be observed that, unlike the end-enhancement term, the drift-limitation term acts on the whole course rather than the last few control behaviors. The vessel receives an immediate punishment whenever it violates the restriction of drift. And because of this, the motion states will look well distinct from those with basic reward function and end-enhancement improvement. Interestingly enough, in the previous experiments, the vessel is almost at a standstill from the start of training processes, whereas the vessel trained with the comprehensive reward function is forced to venture out from the very beginning. With the CDRLOA training procedure goes on, the vessel approaches the target as a quick learner. During the same 2500 exploration epochs, the vessel can find the shorter path than the previous reward functions (Fig. 6 and 7). Meanwhile, due to the restriction for sway, the headings of the vessel are placed more reasonable. Moreover, if we continue to train another 2500 epochs, a control sequence with almost wonderful headings and performance on obstacles avoidance can be found. It should be noted that this performance cannot be achieved through basic reward function and the end-enhancement improvement.

6. Conclusions

This note has proposed a deep reinforcement learning obstacle avoidance algorithm for the underactuated unmanned marine vessel under the unknown dynamics of the environment. The main advantage of the proposed algorithm is its conciseness and extendibility; the analytic control law is not required to maneuver the vessel. With the application of the proposed CDRLOA system, the vessel takes actions on the basis of the current observations, including the detection results for obstacles, and the basic measurements of the vessel's operational states. Various experiments have been conducted to corroborate the effectiveness of the algorithm. In addition, it is convenient to extend this architecture to other complex tasks to meet various kinds of customized requirements.

Acknowledgments

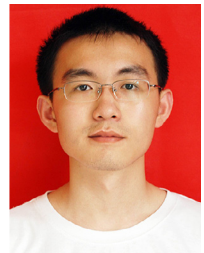
This paper is partly supported by the National Science Foundation of China (61473183, U1509211).

References

[1] T.A. Johansen, T. Perez, A. Cristofaro, Ship collision avoidance and COLREGS compliance using simulation-based control behavior selection with predictive hazard assessment, IEEE Trans. Intell. Transport. Syst. 17 (12) (2016) 3407–3422.
[2] P. Wu, S. Xie, H. Liu, M. Li, H. Li, y. Peng, X. Li, J. Luo, Autonomous obstacle avoidance of an unmanned surface vehicle based on cooperative manoeuvring, Industr. Robot: Int. J. 44 (1) (2017).
[3] J. Lisowski, R. Smierzchalski, Methods to assign the safe manoeuvre and trajectory avoiding collision at sea, WIT Trans. Built Environ. 12 (1970).
[4] 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.
[5] H.-G. Han, S. Zhang, J.-F. Qiao, An adaptive growing and pruning algorithm for designing recurrent neural network, Neurocomputing 242 (2017) 51–62.
[6] D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al., Mastering the game of go with deep neural networks and tree search, Nature 529 (7587) (2016) 484–489.
[7] 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) 529–533.
[8] M. Duguleana, G. Mogan, Neural networks based reinforcement learning for mobile robots obstacle avoidance, Expert Syst. Appl. 62 (2016) 104–115.
[9] F. Fathinezhad, V. Derhami, M. Rezaeian, Supervised fuzzy reinforcement learning for robot navigation, Appl. Soft Comput. 40 (2016) 33–41.
[10] B.P. Lathi, et al., Linear systems and signals, vol. 2, Oxford University Press, New York:, 2005.

Please cite this article as: Y. Cheng, W. Zhang, Concise deep reinforcement learning obstacle avoidance for underactuated unmanned marine vessels, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.06.066

[11] C.A. Woolsey, Review of marine control systems: guidance, navigation, and control of ships, rigs and underwater vehicles, J. Guid. Control Dyn. 28 (3) (2005) 574–575.
[12] T.I. Fossen, Handbook of Marine Craft Hydrodynamics and Motion Control, John Wiley & Sons, 2011.
[13] T. Perez, Ship Motion Control: Course Keeping and Roll stabzilisation Using Rudder and Fins, Springer Science & Business Media, 2006.
[14] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, MIT Press, 2012, pp. 1097–1105.
[15] Y. LeCun, M. Ranzato, Deep learning tutorial, in: Proceedings of Tutorials in International Conference on Machine Learning (ICML13), Citeseer, 2013.
[16] J. Xu, X. Luo, G. Wang, H. Gilmore, A. Madabhushi, A deep convolutional neural network for segmenting and classifying epithelial and stromal regions in histopathological images, Neurocomputing 191 (2016) 214–223.
[17] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, Computer Vision and Pattern Recognition, arXiv preprint arXiv:1409.1556.
[18] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, vol. 1, MIT press Cambridge, 1998.
[19] L. Baird, A.W. Moore, Gradient descent for general reinforcement learning, Adv. Neural Inf. Process. Syst. (1999) 968–974.
[20] R.S. Sutton, D.A. McAllester, S.P. Singh, Y. Mansour, et al., Policy gradient methods for reinforcement learning with function approximation., in: Proceedings of International Conference on Neural Information Processing Systems (NIPS), vol. 99, 1999, pp. 1057–1063.
[21] C.J. Watkins, P. Dayan, Q-learning, Mach. Learn. 8 (3–4) (1992) 279–292.
[22] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing Atari with Deep Reinforcement Learning, arXiv preprint arXiv:1312.5602.
[23] H. Modares, F.L. Lewis, M.-B. Naghibi-Sistani, Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems, Automatica 50 (1) (2014) 193–202.
[24] S. Adam, L. Busoniu, R. Babuska, Experience replay for real-time reinforcement learning control, IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.) 42 (2) (2012) 201–212.
[25] R.S. Sutton, Introduction: the challenge of reinforcement learning, Mach. Learn. 8 (3) (1992) 225–227.
[26] R. Al-Rfou, G. Alain, A. Almahairi, C. Angermueller, D. Bahdanau, N. Ballas, et al., Theano: a Python framework for fast computation of mathematical expressions, Symbolic Computation, arXiv preprint arXiv:1605.02688.
[27] Z. Peng, D. Wang, T. Li, Z. Wu, Leaderless and leader-follower cooperative control of multiple marine surface vehicles with unknown dynamics, Nonlinear Dyn. 74 (1–2) (2013) 95–106.



Yin Cheng received the B.Eng. degrees from the Institute of Electrical Engineering, Hohai University, Nanjing, China, in 2013. He is currently working toward the Ph.D. degree at the School of Electronic and Electric Engineering, Shanghai Jiao Tong University, Shanghai, China. His research interests include deep reinforcement learning, automated driving, and smart power system.

Weidong Zhang received the B.S., M.S., and Ph.D. degrees from Zhejiang University, Hangzhou, China, in 1990, 1993, and 1996, respectively. He then worked as a Postdoctoral Fellow at Shanghai Jiao Tong University. He joined Shanghai Jiao Tong University in 1998 as an Associate Professor and has been a Full Professor since 1999. From 2003 to 2004, he was at the University of Stuttgart, Stuttgart, Germany, as an Alexander von Humboldt Fellow. He has received National Science Fund for Distinguished Young Scholars of China. In 2011, he became a Chair Professor at Shanghai Jiao Tong University. He is currently the Director of the Engineering Research Center of Marine Automation, Shanghai Municipal Education Commission, and the Deputy Dean of the Department of Automation, Shanghai Jiao Tong University. He is the author of more than 300 refereed papers and one book and holds 31 patents. His research interests include control theory and its applications in various fields.

Please cite this article as: Y. Cheng, W. Zhang, Concise deep reinforcement learning obstacle avoidance for underactuated unmanned marine vessels, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.06.066