Development Economics of China
2026, VOL. 10, NO. 1, 61-66
DOI: 10.47297/wspdecWSP2515-797313.20261001

Deep Reinforcement Learning-based Obstacle Avoidance for Robot Movement in Warehouse Environments

Mincong Kong, Chaoqun Xiang, Yaxuan Cao, Yilu Guo

Qingdao City University, Qingdao, Shandong, 266106, China

ABSTRACT
At present, in most warehouse environments, the accumulation of goods is complex, and the management personnel in the control of goods at the same time with the warehouse mobile robot trajectory interaction, the traditional mobile robot can not be very good on the goods and pedestrians to feed back the correct obstacle avoidance strategy, in order to control the mobile robot in the warehouse environment efficiently and friendly to complete the obstacle avoidance task, this paper proposes a deep reinforcement learning based on the warehouse environment, the mobile robot obstacle avoidance Algorithm.

KEYWORDS
Warehouse robot; Obstacle avoidance strategy; Deep reinforcement learning; Complex environment

1 Introduction

Warehouse robots work in complex and variable scenarios covering complex stacked goods and warehouse personnel. The traditional obstacle avoidance algorithm uses the information of the surrounding obstacles at the current moment and relies on methods such as geometric configuration or sampling to obtain the final obstacle avoidance strategy[1-2], which does not take into account the changes of the future state of the surrounding pedestrians, and in the complex environment, the algorithm is prone to unnatural behaviours, such as oscillation, due to the randomness of the pedestrian's movement, which makes the robot's control signals jump frequently, and fails to satisfy the requirements of safety and comfort in warehouse environments. . Therefore, in order to efficiently complete the obstacle avoidance task in the cargo-intensive warehouse environment, the obstacle avoidance algorithm of the mobile robot needs to analyse the pedestrian's behaviour. At the same time, with the development of human-computer interaction concept, the robot's obstacle avoidance task not only needs to meet the high efficiency and real-time performance, but also needs to take into account the comfort requirements of pedestrians in the process of interaction with people.

2 Obstacle Avoidance Algorithm based on Deep Reinforcement Learning

The goal of using reinforcement learning algorithm to solve the obstacle avoidance problem is to obtain the optimal strategy π^* from the joint state s^{jn}_t of the robot to the control output at, as shown in Fig. 1. In the obstacle avoidance algorithm based on reinforcement learning, the intelligent body obtains the state s^{jn}_t of the robot and all the pedestrians in the surrounding area at the current moment by interacting with the environment, and then discretises the robot's control signal into a certain size of action space by one-step prediction.

Then the control signals of the robot are discretised into an action space of a certain size, and the state after the execution of each control signal in the action space is predicted by a one-step prediction model, and the predicted state $s^{jn}_{t+\Delta t}$ is obtained; then the predicted states$^{jn}_{t+\Delta t}$ is inputted into the value function network, and the value of the state is obtained by combining the reward function, and the optimal action in the action space corresponding to the maximal value is selected as the final output action of the robot, which is a_t, as shown in Equation (1). During the interaction between the intelligent body and the environment, the value function network V* is continuously optimised until it converges.

$$a_t = \arg\max R(S^{jn}_t, a_t) + \gamma^{\Delta t \bullet v} pref V^*(S^{jn}_{t+\Delta t}) \tag{1}$$

where $s^{jn}_{t+\Delta t}$ = prop(s^{jn}_t , Δt , a_t), denotes that the prediction model approximates the joint state of the robot after the Δt

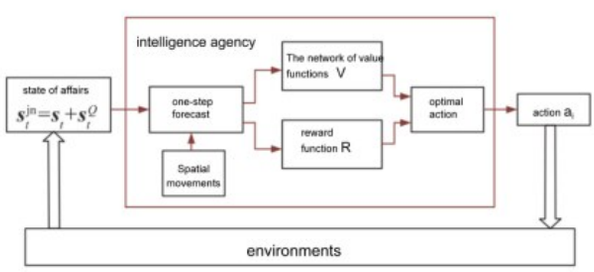


Figure 1 Obstacle avoidance algorithm based on deep reinforcement learning

moment by predicting the motion of the robot and its surrounding pedestrians after the Δt moment through a simple linear model. The following is the specific selection method for each element of reinforcement learning:

State Space: Considering that both the robot's own state and the state of other pedestrians in the environment will affect the final decision, the joint state $s^{jn}_t = [s_t, s^o_t]$ is used as the input state for reinforcement learning at time t. The input state is the robot's own state and the state of other pedestrians in the environment. Where, $s_t = [p_x, p_y, v_x, v_y, r, g_x, g_y, v_{pref}, \varphi]$ is the state of the robot, which includes the robot's position, movement speed, size, target point, optimal running speed and steering angle; sot = $[s^{1o}_t, s^{2o}_t, ..., s^{no}_t]$ is the state of all the pedestrians in the environment, which is different from st, and sot = $[s^{1o}_t, s^{2o}_t, ..., snot]$ is the state of all pedestrians in the environment. Since in a crowded environment, the robot cannot obtain the information of pedestrians' target point, optimal running speed and steering angle, the state siot of the ith pedestrian includes only the pedestrian's position, movement speed, size and the coded information of the surrounding pedestrians, $s^{io}_t = [p_x, p_y, v_x, v_y, r, r_i]$.

Action space: The action space is taken as a set of feasible robot velocity control signals, the robot's action at time t a_t = [v, ω], where v, ω represent the linear and angular velocities of the robot, respectively.

Value Functions: The goal of the reinforcement learning algorithm is to obtain an optimal value function V^*, and the optimal policyπ^*can be obtained through a network of value functions, so that the robot receives the optimal desired reward in its interaction with the environment.

$$V^*(S^{jn}_t) = \sum_{t'=t}^{T} \gamma^{t' \bullet v_{pref}} R_{t'}(s^{jn}_{t'}, \Pi^*(s^{jn}_{t'}))$$

$$\pi^*(s^{jn}_t) = \arg\max R(s^{jn}_t, a_t) + \gamma^{\Delta t \bullet V_{pref}} \int_{s^{jn}_{t+\Delta t}} p(s^{jn}_t, a_t, s^{jn}_{t+\Delta t}) V^*(s^{jn}_{t+\Delta t}) ds^{jn}_{t+\Delta t} \quad (2)$$

where R(s^{jn}_t , a_t) is the reward value at time t; γ ∈ (0, 1) is the decay coefficient; P(s^{jn}_t , a_t , $s^{jn}_{t+\Delta t}$) is the state transfer probability; and vpref denotes the speed of the robot running when there are no obstacles, which serves as a normalisation term in the decay coefficient afterwards.

The value function can learn certain information about the environment through the continuous interaction between the robot and the environment, but the complex pedestrian interaction rules cannot be learnt well using only a shallow multilayer perceptron, in this paper, the value function is modified in order to improve the learning ability of the value function in crowded and complex environments, and further modifications are made to the reward function to address the requirements of pedestrian comfort. In addition, this paper uses an LSTM network for the case of variable number of pedestrians.

3 Specific Algorithms

In this section, by introducing pedestrian interaction information in the value function network and modifying the reward function for reinforcement learning, obstacle avoidance algorithms that meet the requirements of human-computer interaction are obtained based on deep reinforcement learning.

3.1 Improvement of Value Function Network based on Pedestrian Interaction

As shown in Figure 2, the improved value function network μ is composed of three parts: the pedestrian interaction information module (crowd interaction module), the L_{STM} network $\Phi_{LSTM}(\cdot)$, and the multilayer perceptron (MLP). The pedestrian interaction module first extracts pedestrian features from the original agent state sjnt. Then, through the L_{STM} network $\Phi_{LSTM}(\cdot)$, the pedestrian features of an indefinite number are combined to obtain the joint hidden state ho of all pedestrians. Finally, ho and the robot's own state are jointly input into the multilayer perceptron network $\psi_M(\cdot)$ to obtain the corresponding value. This section mainly introduces the specific method for analyzing pedestrian interaction behavior in the improved value function network.

The decisions between pedestrians influence each other. If the relationship between each pair of pedestrians is accurately described, it will lead to a time complexity of $O(N^2)$. As the number of pedestrians in the environment increases, the computational burden will become extremely heavy. In addition, pedestrians who are far apart will hardly have an impact on each other's movements. Describing the relationship between the two will not only increase the computational complexity but also increase the learning burden of the network. Accordingly, this paper adopts a special hybrid grid - the angular pedestrian grid (APG)to encode the local environment of pedestrians to eliminate information that is useless for obstacle avoidance strategies. In order to better capture the dynamic characteristics of pedestrians, the angular pedestrian grid (APG) introduces a high-resolution grid, increasing the extensiveness of information. Figure 3 shows the encoding process for the i-th pedestrian P^i_H. Let the total number of pedestrians be N. The angular pedestrian grid (APG) only considers pedestrians within a circle centered on pedestrian P^i_H and with a radius of rmax. The circle is equally divided into K sectoral areas, and the nearest distance to pedestrian P^i_Hin each sectoral area is selected as the encoding value of that sectoral area. After being encoded by the angular pedestrian grid (APG), pedestrian P^i_H is output as a one-dimensional vector $r_i = [r_i,1,...,r_{i,k},...,r_{i,}K]$, where the mathematical representation method of ri,k (the encoding value of the k-th sector) is shown in Equation (3).

$$r_{i,k} := \min(r_{\max}. \min(\{\rho_{i,j}, j \in N_{i,k}\}))$$

$$N_{i,k:}=\{j\in\{1,...,N\}\backslash\{i\},\boxtimes_{i,j}\in[\gamma_k,\gamma_{k+1}]\} \quad (3)$$

Where k ∈ {1,..., K}, and the sector angle γk corresponding to each grid is γk := (k − 1)2k/K. $(\rho_{i,j}, \varphi_{i,j})$ represents the coordinates of pedestrian P^j_H in the polar coordinate system centered on pedestrian P^i_H. Therefore, r_i is only related to the pedestrian positions.

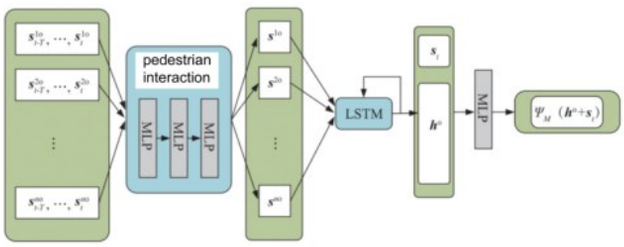


Figure 2 Improved network structure of value function

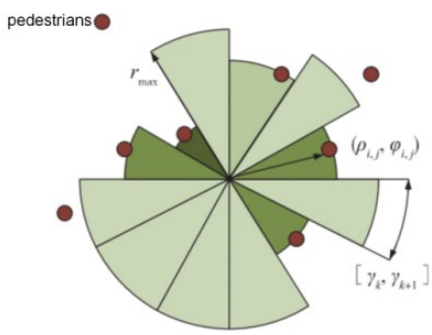


Figure 3 Angular pedestrian grid

Compared with the standard 2D grid, APG only linearly affects the dimension of the input, but it can still capture changes in radial distance with continuous resolution rather than discrete grid cells. In addition, the closer pedestrian P^j_H is to pedestrian P^j_H, the more accurate the observability of the angular position change of pedestrian P^j_H becomes. For an APG grid with given parameters, the state soit = $[p^i_x, p^i_y, v^i_x, v^i_y, r, r_i]$ of pedestrian P^i_H at time t can be obtained. This state variable will be used as the input of the improved value function network.

The movement of the robot is not only influenced by the pedestrian information at the current moment, but also has a great relationship with the pedestrian information at previous moments. Moreover, by relying on the joint influence of multiple moments, the short-sightedness of the obstacle avoidance strategy can be overcome, and the problem of control signal jumps caused by relying only on the immediate environmental information at the current moment can be reduced. Historical data can be used for obstacle avoidance to achieve better obstacle avoidance effects. In time series prediction, the attention mechanism has achieved good application results. In this paper, the self-attention mechanism is used to extract the time series features of a single pedestrian, and learn the relative importance of T moments of data including the current moment and the joint influence on the robot's obstacle avoidance strategy.

The continuous trajectory sequence $[s^{oi}_{t-T}, ..., s^{oi}_{t-k}, ..., s^{oi}_t]$ of pedestrian P^i_H within time T is known. As shown in Figure 4, first, the pedestrian features at each moment are encoded, and a fixed-length output e_m is obtained through the multilayer perceptron $\psi_e(\cdot)$ (MLP).

$$e_m=\varphi_e(s^{io}_{t-m},W_e) \quad (4)$$

Then, the vector e_m is input into the perceptron $\psi_h(\cdot)$, and the interaction characteristic vector h_m between the states of

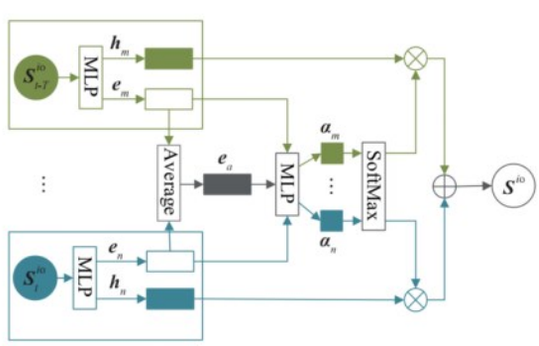


Figure 4 Attention mechanism model

pedestrian P^i_H at different times is extracted from it.

$$e_k = \psi_h(e_m, W_h) \tag{5}$$

Next, through the multilayer perceptron $\psi_\alpha(\cdot)$, the relative importance of pedestrian states at different times is calculated, and the attention score α_m of each interaction feature vector is obtained.

$$e_a = \frac{1}{T}\sum_{k=1}^{T} e_k \tag{6}$$

$$a_m = \psi_a(e_m, e_a, W_a) \tag{7}$$

e_a is the average pooling vector of all moments. $\psi_e(\cdot)$, $\psi_h(\cdot)$, and $\psi_\alpha(\cdot)$ are fully connected networks. Their nonlinear activation functions are all Re-LU. W_e, W_h, and W_α are the weight values of the fully connected networks respectively. The specific network parameters will be given in the experimental section. Using the interaction feature vector e_m and the corresponding attention score α_m at each moment, the final state output e^o of pedestrian P^j_H can be obtained by linearly combining the interaction feature vectors of all moments.

$$e^o = \sum_{k=1}^{T} soft\max(a_k) e_m \tag{8}$$

Finally, the LSTM network processes an indefinite number of obstacles in the environment and finally obtains a fixed-length output vector h^o.

3.2 Comfort Reward Function

In the scenario where the robot interacts with people, if pedestrians do not initiate service requests to the robot, in order to ensure the comfort requirements of pedestrians, the robot should try to avoid appearing within the comfort range of pedestrians as much as possible, that is, the robot should not enter the personal distance of pedestrians. In addition, according to reference, the lateral comfortable distance for the robot to overtake pedestrians from behind to the front is 0.7 m. If it is less than this distance, it will also cause discomfort to pedestrians. Therefore, based on the above requirements for pedestrian comfort, this paper modifies the reward function and penalizes the state where the robot enters the comfort range of pedestrians.

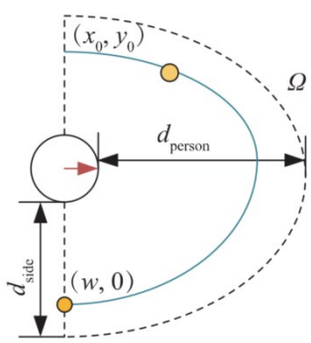


Figure 5 Personal distance zones for people

According to the knowledge of pedestrian comfort, the pedestrian comfort zone Ω is defined: taking the pedestrian as the coordinate origin and the forward direction as the x-axis, a pedestrian coordinate system is established. The pedestrian comfort zone is represented by a semi-ellipse Ω with the pedestrian as the origin, the maximum personal distance d_{person} as the major axis, and the side comfort distance d_{side} as the minor axis (only considering the situation that appears within the pedestrian's field of vision. d_{side} is 0.7 m and d_{person} is 1.2 m).Suppose the coordinates of the robot in the pedestrian coordinate system are (x_0, y_0). The indicator function $I(x_0, y_0)$ can be used to determine whether the robot has driven into the pedestrian comfort zone Ω. $I(x_0, y_0)$ being 1 means the robot has entered the pedestrian comfort zone, and $I(x_0, y_0)$ being 0 means the robot has not entered the pedestrian comfort zone.

$$I(x_0, y_0) = 0, \frac{x_0^2}{d_{person}^2} + \frac{y_0^2}{d_{side}^2} \geq 1$$

$$I(x_0, y_0) = 1, \frac{x_0^2}{d_{person}^2} + \frac{y_0^2}{d_{side}^2} \leq 1 \tag{9}$$

If the robot reaches the pedestrian comfort zone of pedestrians, that is, $I(x_0, y_0) = 1$, it will cause discomfort to pedestrians, and the robot needs to be given corresponding penalties. The magnitude of the penalty depends on the position of the robot in the pedestrian comfort zone. The closer the robot is to the pedestrian, the greater the penalty. As shown in Figure 5, the robot is marked in yellow. If the robot (yellow circle) reaches the pedestrian comfort zone, it can be assumed that there is the same penalty on the ellipse with the same ratio of major and minor axes as the pedestrian comfort zone at the robot's coordinate in the pedestrian comfort zone. That is, all points on the blue ellipse shown in

Figure 5 have the same penalty term, so (x_0, y_0) and (ω, 0) have the same penalty. Since being closer to people is more likely to cause people's discomfort, and the discomfort will become greater and greater. Therefore, an exponential function is used to characterize this penalty value. The maximum value of the penalty is r_p. So the obtained penalty function related to ω is as follows:

$$R_t^{social}(\omega) = -r_p(e^{-(\omega - d_{side})} - 1) \tag{10}$$

where w is the short axis of the blue ellipse.

$$w = \sqrt{\frac{(x_0^2 \bullet d_{side}^2)}{d_{person}^2 + y_0^2}} \tag{11}$$

The original reward function $R^o_t(s^{jn}_t, a_t)$ is the reward function used in reference, in which the penalty term when the robot is close to an obstacle is removed.

$$\begin{aligned} R_t^o(s_t^{jn}, a_t) &= -0.25, if, d_t \le 0 \\ R_t^o(s_t^{jn}, a_t) &= 2, elseif, p_t = p_g \\ R_t^o(s_t^{jn}, a_t) &= 0, otherwise \end{aligned} \tag{12}$$

At the same time, in order to avoid excessive changes in the angular velocity of the robot, a penalty is imposed on cases where the angular velocity changes too much.

$$\begin{aligned} R_t^A(s_t^{jn}, a_t) &= -r_{Angle}, if, \phi_t - \phi_{t-1} \ge \pi \\ R_t^A(s_t^{jn}, a_t) &= 0, otherwise \end{aligned} \tag{13}$$

The final reward function is

$$R_t(s_t^{jn}, a_t) = R_T^0(s_t^{jn}, a_t) + R_t^{social}(s_t^{jn}, a_t) + R_t^{jn}(s_t^{jn}, a_t) \tag{14}$$

Through this reward function, the obstacle avoidance strategy of the robot can be more in line with human comfort criteria.

3.3 Network Training

The training method of the entire reinforcement learning algorithm adopts the temporal difference method. To increase the utilization rate of samples, the information of each interaction between reinforcement learning and the environment is stored in the replay buffer. During network training, training data is obtained by sampling from the replay buffer through the method of uniform sampling. The fixed target network method is used to accelerate the convergence of the algorithm and reduce the variance of the algorithm. At the same time, to further accelerate convergence, before training, the dataset generated by the traditional method ORCA is used for network pre-training.

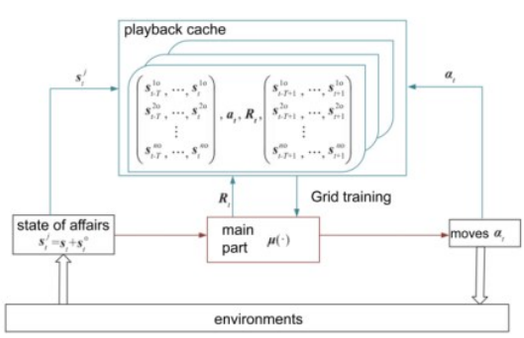


Figure 6 Network training

4 Experimental Validation

This chapter will verify the effectiveness and generalization of the improved value function network and the feasibility of the comfort reward function by comparing the performance of obstacle avoidance algorithms with different encoding methods and the performance of obstacle avoidance algorithms before and after modifying the reward function.

Hardware platform: The hardware platform for the experiment is a desktop computer with Intel Core i7-7700 3.60 GHz and NVIDIA Geforce GTX 1050 Ti.

Simulation environment: All experiments are carried out in a simulation environment built based on Gym. In the simulation environment, a circle with a radius of 0.3 m is used to replace pedestrians and robots. The task of pedestrians and robots is to run from the starting point to the target point. Among them, the movement of pedestrians is controlled by the traditional obstacle avoidance algorithm ORCA strategy. The selection of starting points and target points adopts a cross-encounter method. That is, the initial positions of pedestrians and robots are randomly located on the same circle with a radius of 4 m, and the target point is the point that is centrally symmetric to its initial position with respect to the center of the circle.

Robot action space setting: The angular velocity and linear velocity of the robot in the simulation. In the experiment,

the discrete action space of the robot is selected as the combination of the robot's linear velocity space and angular velocity space, with a size of 80. Among them, the linear velocity space is uniformly sampled in [0, v_{pref}], and the angular velocity space is uniformly sampled in [0, 2π], with sampling sizes of 5 and 16 respectively.

4.1 Experimental Validation of the Improved Value Function Network

This subsection verifies the effectiveness of the improved value function network in this paper by comparing it with ORCA , CADRL , and obstacle avoidance algorithms with two different encoding methods based on social force and local map. In order to avoid the influence of the comfort reward function on this module, the reward function here adopts the reward function in paper.Therefore, the difference between the obstacle avoidance algorithms based on reinforcement learning lies only in the processing method of pedestrian interaction information. For convenience of expression, the algorithm based on social force encoding is called SF_RL, the method based on local map encoding is called LM_RL, and the algorithm based on APG encoding proposed in this paper is called APG_RL.

Figure 7 shows the change curves of the cumulative return function after 10, 000 rounds of training of four reinforcement learning-based obstacle avoidance algorithms, CADRL, SF_RL, LM_RL, and APG_RL, in a simulation environment with 6 pedestrians. Among them, the convergence performance of CADRL, LM_RL, and APG_RL is significantly higher than that of SF_RL. This is because there may be errors in the description of the interaction information between pedestrians using the social force model, thus reducing the convergence of the network.

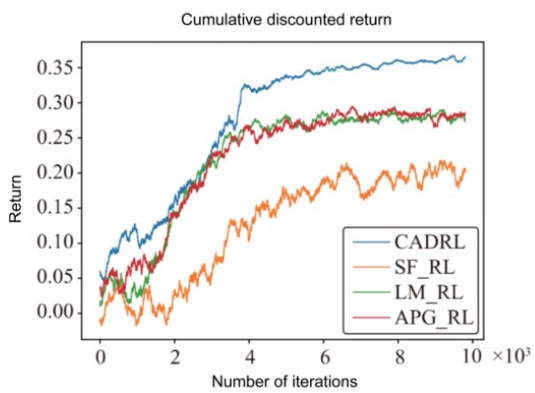


Figure 7 Learning return curve of different algorithms

4.2 Experimental Verification of the Pedestrian Comfort Reward Function

This experiment experimentally verified the feasibility of the reward function proposed in this paper. The values of r_p and r_{Angle} are both 0.02. Without changing the APG_RL obstacle avoidance algorithm, training is carried out according to the comfort reward function designed by formula (13), and the comfortable obstacle avoidance strategies APG_CRL and APG_CARL are obtained respectively. The reward function of APG_CRL only adds the $R^{social}_t(w)$ term, and the reward function of APG_CARL adds both Rsocialt(w) and RA t(s^{jn}_t, a_t).

5 Conclusion

The comfortable obstacle avoidance algorithm for mobile robots proposed in this paper first addresses the deficiency that shallow value function networks are difficult to fit complex pedestrian environments. It improves the value function and designs a pedestrian interaction information module. The interaction information between pedestrians is extracted through the angular pedestrian grid, and the temporal features of pedestrian walking trajectories are extracted by using the attention mechanism. Secondly, by modifying the reward function, the pedestrian comfort requirements are introduced into the obstacle avoidance strategy, making the robot's obstacle avoidance strategy more in line with the needs of human-computer interaction. Through comparative analysis of simulation experiments, the algorithm proposed in this paper not only has a good obstacle avoidance success rate and adaptability in complex dynamic environments with dense crowds, but also can meet the requirements of pedestrian comfort.

References

[1] Fiorini P,Shiller Z. Motion planning in dynamic environments using velocity obstacles[J]. The International Journal of Robotics Research,1998, 17(7): 760-772.

[2] Mnih V,Kavukcuoglu K,Silver D,et al. Human-level control through deep reinforcement learning[J]. Nature,2015,518 (7540) : 529-533.

[3] Chiang H T L,Faust A,Fiser M,et al. Learning navigation behaviors end-to-end with AutoRL[J]. IEEE Robotics and Automation Letters,2019,4(2) : 2007-2014.