

Novel Deep Deterministic Policy Gradient Path-planning Algorithm for Robots

Fengying Wang,^{1,2} Ying Chen,² and Liming Du^{1*}

¹School of Information Engineering, Suqian University,
No. 399, South Huanghe Road, Suqian City, Jiangsu Province 223800, P.R. China

²School of Computer Science and Engineering, Shenyang Jianzhu University,
No. 25, Hunnan Middle Road, Hunnan District, Shenyang, Liaoning 110168, P.R. China

(Received February 9, 2026; accepted September 17, 2026)

Keywords: robot path planning, intelligent perception, self-attention mechanism, deep reinforcement learning

Visual perception is the primary modality for robots to perceive environments and plan collision-free motion. However, coupling conventional visual-sensor processing with the deep deterministic policy gradient (DDPG) algorithm in complex unstructured environments causes three deficiencies—inefficient use of visual features, weak perception of unknown regions, and sparse feedback—which limit the environmental adaptability of vision-based robot navigation. To this end, in this paper, we propose an improved DDPG algorithm integrating a self-attention (SA) mechanism, hindsight experience replay (HER), and an intrinsic curiosity module (ICM), called ICM-DDPG-HER. The SA module in the front-end visual branch uses scaled dot-product attention to enhance task-relevant features while suppressing redundant responses. HER, combined with a progressive learning strategy, relabels failed trajectories with hypothetical goals, converting them into valid samples and alleviating reward sparsity. A dual reward mechanism fusing curiosity-driven intrinsic and environment-driven extrinsic rewards compensates for sparse informative feedback and promotes active exploration. In Gazebo simulations, against the deep Q-network (DQN), DDPG, DDPG-HER, and SA-DDPG-HER, the proposed algorithm achieves the highest episodic reward in static (8.00 vs 5.45/4.10/1.01/–3.14) and dynamic (26.61 vs 24.50/18.57/15.81/7.30) environments and shortens the planned path by 4–40% in dynamic tests. The results indicate faster convergence, greater robustness, and better environmental-perception performance, enabling efficient obstacle avoidance and rapid target reaching in unknown complex environments.

1. Introduction

Autonomous navigation in unstructured environments requires a mobile robot to perceive its surroundings, localize itself, and plan collision-free motion toward a goal; in all three tasks, vision sensors play a central role.⁽¹⁾ Cameras and depth sensors provide rich geometric, textural, and semantic cues at low cost and have therefore become the dominant sensing modality on

*Corresponding author: e-mail: duliming@squ.edu.cn
<https://doi.org/10.18494/SAM6283>

mobile robots, humanoid robots, and intelligent service platforms. Driven by advances in deep learning and multimodal sensing, robot perception has shifted from hand-crafted feature extraction toward data-driven representation learning, making it possible to couple perception and decision-making in a single end-to-end pipeline. Nevertheless, when such pipelines are deployed in real-world unstructured scenes, three coupled bottlenecks remain largely unsolved: a passive perception mechanism that cannot actively acquire missing information, a pronounced degradation of perception quality under complex conditions, and low utilization efficiency of the interaction samples generated during training. Together, these bottlenecks prevent vision-based path-planning systems from transferring out of simulation. In this work, we target exactly these three bottlenecks.

Conventional visual perception follows a passive paradigm: sensors sample the environment at a fixed frequency from the current viewpoint, without any task-driven awareness of which information is still missing.⁽²⁾ When occlusion, specular reflection, or a viewing blind spot occurs, the system cannot adjust its observation pose to recover the missing cues and must instead rely on probabilistic inference from incomplete inputs; this frequently leads to perception failure. Research on embodied intelligence has begun to close this gap with active perception paradigms that establish an action-observation loop: the robot adjusts its viewpoint according to perceptual uncertainty and aggregates multiview observations to eliminate occluded blind spots. For example, See2Act, a diffusion-based imitation-learning framework for active robot perception, couples action denoising with viewpoint refinement, autonomously recovers informative viewpoints under severe occlusion, and improves task performance by up to 34% over prior methods.⁽³⁾ Related work has also begun on integrating active information acquisition into simultaneous localization and mapping (SLAM) and navigation pipelines.⁽⁴⁾ However, existing active perception strategies are computationally expensive, and their viewpoint selection usually depends on pretrained models, which hinders deployment on resource-constrained robotic platforms; the real-time performance of the perception-action loop still needs improvement.

Perception quality also degrades sharply in complex scenes. Models that perform well on curated benchmarks frequently generalize poorly to unstructured settings, where variable illumination, dynamic disturbances, object occlusion, and long-tail object distributions cause frequent missed detections and false positives, particularly for transparent and heavily cluttered targets.⁽²⁾ A common remedy is multisensor fusion: light detection and ranging (LiDAR) geometry is used to compensate for the limited depth and scale information available from vision alone, which improves robustness to a certain extent.⁽⁵⁾

Low sample-utilization efficiency further restricts the iterative optimization of perception and planning policies. High-quality annotated real-world data are expensive and scarce, yet the massive stream of raw data generated during robot operation remains largely unexploited: training still relies heavily on manual labels, while unlabeled and failure-case samples are discarded. Failure-driven interactive learning frameworks such as iTeach collect the failure episodes encountered during deployment and fine-tune the perception model online with only limited annotation, thereby achieving incremental adaptation in real environments.⁽⁶⁾ In other studies, efficiency was improved from the data-scheduling side by optimizing keyframe

sampling in imitation learning, filtering high-value samples and removing redundant ones to alleviate training imbalance.⁽⁷⁾ Despite these advances, existing pipelines remain weak at mining unlabeled and failed samples: large-scale manual annotation is still indispensable for stable performance, and the sample-efficiency bottleneck has not been fundamentally resolved.

Deep reinforcement learning (DRL) provides an end-to-end paradigm that unifies perception and decision-making, and has therefore become a mainstream solution for vision-based robot path planning. Mnih *et al.* integrated convolutional neural networks with Q-learning and proposed the deep Q-network (DQN), a milestone architecture that combined perceptual deep learning with reinforcement learning.⁽⁸⁾ Because the max operator in the DQN target systematically overestimates action values, van Hasselt *et al.* proposed double DQN (DDQN), which decouples action selection from action evaluation and effectively suppresses this bias.⁽⁹⁾ The actor-critic (AC) framework, in which an actor selects actions and a critic evaluates them, subsequently became the dominant architecture for continuous control.⁽¹⁰⁾ Building on AC, Lillicrap *et al.* proposed the deep deterministic policy gradient (DDPG) algorithm, which extends deterministic policy gradients to high-dimensional continuous action spaces and is therefore well matched to the continuous velocity commands required by mobile robots.⁽¹¹⁾

In conventional DRL, minibatches are drawn uniformly at random from the experience replay buffer, even though transitions collected at different training stages contribute unequally to policy improvement; this mismatch is a major source of sample inefficiency. A series of prioritized and structured replay schemes has therefore been proposed. Zhang and Qiu combined prioritized experience replay with DDQN and reported faster convergence.⁽¹²⁾ Huo and Tang introduced a multi-indicator experience replay method that accelerates learning in multi-objective settings.⁽¹³⁾ Li *et al.* proposed clustering experience replay, which groups transitions with k-means and guarantees that all transition categories are revisited sufficiently often, thereby mining the experience hidden in the current training round.⁽¹⁴⁾ Wang *et al.* developed a dynamic partitioned experience replay strategy that improves model generalization in unknown environments.⁽¹⁵⁾ The results of these studies confirmed that redesigning the replay mechanism is an effective lever for improving sample efficiency. Nevertheless, they reschedule existing transitions rather than increase the information content of each transition, so their benefit saturates when observations are high-dimensional and noisy.

Because the reward signal determines which behavior is reinforced, reward shaping and exploration bonuses constitute a second major research direction. To quantify how uncertain an agent is about its environment, Bellemare *et al.* proposed density-model-based pseudo-counts that unify count-based exploration with intrinsic motivation.⁽¹⁶⁾ Pathak *et al.* built a visual feature space and trained a self-supervised inverse-dynamics model, using the prediction error of a forward model as an intrinsic curiosity reward.⁽¹⁷⁾ Zhang *et al.* designed a reward that responds to the agent's behavioral trend across consecutive time steps, which improves stability and accelerates convergence,⁽¹⁸⁾ while Wang *et al.* proposed a confusion-matrix-based dynamic reward that adjusts reward values in real time as the environment changes.⁽¹⁹⁾ For the sparse-reward problem, which is central to goal-conditioned navigation, Li *et al.* embedded an artificial potential field into the reward function so that potential-energy variations continuously guide the agent toward the goal.⁽²⁰⁾ These works showed that a well-shaped auxiliary reward can substitute, to some extent, for dense external supervision.

In complex robot path planning, the robot must correlate visual observations across adjacent time steps and select the cues that are relevant to the current decision. Attention mechanisms provide a natural tool for this selection. Thomas *et al.* augmented DRL with self-attention models so that the policy can reason over heterogeneous input signals while remaining interpretable under varying conditions.⁽²¹⁾ Pan *et al.* systematically investigated the integration of self-attention and convolution and proposed a hybrid block that retains the strengths of both with minimal computational overhead.⁽²²⁾

More recently, these ideas have been extended to end-to-end navigation. Zhou and Zhang combined DRL with a spatiotemporal Transformer architecture and proposed an end-to-end intelligent obstacle avoidance method that jointly models the spatial distribution and the motion trajectories of obstacles, achieving a higher obstacle avoidance success rate and faster policy convergence than mainstream DRL methods.⁽²³⁾ Zhao *et al.* addressed the low coordination efficiency and poor dynamic adaptability of conventional DRL in multi-automated guided vehicle (AGV) systems by proposing improved AC-DDPG, which dynamically generates task sequences to reduce conflicts among AGVs and outperforms conventional DDPG in convergence speed, obstacle avoidance, and energy efficiency.⁽²⁴⁾ The results of these studies confirmed that attention-based multimodal fusion and auxiliary reward shaping are both effective, but they are generally investigated in isolation.

The above review shows that improving DRL through reward design and experience replay optimization is both theoretically sound and practically effective. However, most existing studies focus on the iterative optimization of network structures and training schedules, while paying insufficient attention to the deep exploitation of perceptual input and to the coupling between perception and exploration. As a result, passive perception, limited environmental representation, and low sample utilization persist in complex dynamic scenes, which restricts the generalization and stability of path-planning models. To strengthen the robot's capability to perceive, correlate, and efficiently exploit environmental information, intrinsic curiosity module (ICM)-DDPG-hindsight experience replay (HER), a multimodal perception-fusion DRL path-planning algorithm built upon DDPG, is proposed in this paper. This improved algorithm is embedded with a self-attention (SA) mechanism, HER, and an ICM. The algorithm improves perception and planning performance along three complementary dimensions, namely, feature representation, sample exploitation, and environmental exploration. The main contributions are as follows.

- (1) An SA-based multimodal perception fusion module is introduced to overcome the limited feature representation and weak robustness of single-modality visual perception. The module jointly encodes camera images, robot proprioceptive states, and LiDAR-derived environmental measurements, and uses scaled dot-product attention to adaptively model the intrinsic correlations among these heterogeneous sources. Unlike direct concatenation, the attention weights are computed from the data itself, so task-relevant features are amplified while redundant responses are suppressed. This not only improves perception accuracy and stability under illumination variation and clutter, but also extracts more usable information from every existing sample, which relieves the sample-efficiency bottleneck and provides a more reliable state representation for subsequent decision-making.

- (2) HER is combined with a progressive learning strategy to address reward sparsity and the resulting waste of interaction samples. In goal-conditioned navigation, only a small fraction of actions triggers a positive reward, so most trajectories collected in complex scenes carry no effective supervision signal. By relabeling failed trajectories with hypothetical goals, HER converts these invalid trajectories into valid training samples and stores each transition together with both the original and the relabeled goal. This enlarges the size and diversity of the replay buffer, turns wasted interaction into usable supervision, accelerates network convergence, and improves the adaptability of the learned policy to complex unstructured scenes.
- (3) An ICM is integrated into DDPG to construct a dual reward mechanism that balances exploitation and exploration. Conventional robot perception is passive: the robot only collects data along task-specified trajectories and therefore accumulates few samples in unknown regions, which leads to poor model generalization. The proposed intrinsic reward is defined as the discrepancy between the state predicted by a learned forward model and the actual next state; states that the robot cannot yet predict are treated as novel and rewarded. This drives the robot to actively visit and perceive unknown regions, breaks the limitation of passive perception, and enriches the training distribution with complex-scene samples, thereby improving perceptual robustness in dynamic unknown environments.

The remainder of this paper is organized as follows. In Sect. 2, the overall framework of ICM-DDPG-HER, including multimodal sensor fusion, the HER-enhanced DDPG backbone, and the curiosity-driven dual reward mechanism, is presented. Section 3 includes a description of the simulation setup and a quantitative comparison against four baseline algorithms in static and dynamic-obstacle environments, followed by a discussion of the results. Section 4 concludes the paper, and directions for future work are outlined.

2. ICM-DDPG-HER Intelligent Perception Path-planning Framework

2.1 Multimodal sensor information fusion

During path planning, a mobile robot must exploit complementary information from multiple sensors in order to perceive its surroundings accurately. The robot used in this study is equipped with a camera and a LiDAR. The camera captures the image in front of the robot, whereas the LiDAR perceives the surrounding environment, supports localization, and provides the distance and bearing of the target point relative to the robot. If such heterogeneous information is only partially exploited or is simply concatenated, the system requires considerably more interactions to learn and update, which may lead to excessively long training cycles or even unstable training outcomes. To address this issue, in this paper, we design a multisensor information fusion method based on the SA mechanism.

2.1.1 SA mechanism

SA is a mechanism that allows each element of a sequence to aggregate information from all other elements in accordance with learned relevance weights, and it has become a standard tool

for modeling long-range dependences in images. In this work, SA is applied to the visual branch so that the network can selectively enhance the image regions that are informative for navigation. Formally, given an input sequence x , SA produces an output sequence y , in which every output vector y is computed by taking into account the influence of all the input vectors x on x_1 . Taking the computation of y_1 as an example, as shown in Fig. 1, the relevance between each pair of input vectors is first measured by the dot-product method.

Consider two input vectors x_1 and x_2 . Two projection matrices, W_q and W_k , are learned during training: x_1 is multiplied by W_q to produce the query vector q , as defined in Eq. (1), whereas x_2 is multiplied by W_k to produce the key vector k , as defined in Eq. (2). The dot product of the query q and the key k gives the correlation α between the two input vectors, as shown in Eq. (3).

$$q = W_q x \quad (1)$$

$$k = W_k x \quad (2)$$

$$\alpha = q \cdot k \quad (3)$$

The dot-product method is applied between x_1 and every other input vector, and each resulting correlation α is normalized by a softmax function, as given in Eq. (4), so that the vectors most closely related to x_1 can be identified. Information is then aggregated in accordance with the normalized weight α' assigned to each vector. For this purpose, a value vector v is first computed; its computation is analogous to that of v and q , as shown in Eq. (5). In fact, the query q , the key k , and the value v are all obtained from the same input sequence through different learnable linear transformations. The value vectors v are then weighted by their corresponding coefficients α' and summed to yield the output y_1 , as given in Eq. (6). Consequently, if the correlation between x_1 and x_2 is high, the coefficient $\alpha_{1,2}$ is large and the output y_1 is therefore close to v_2 .

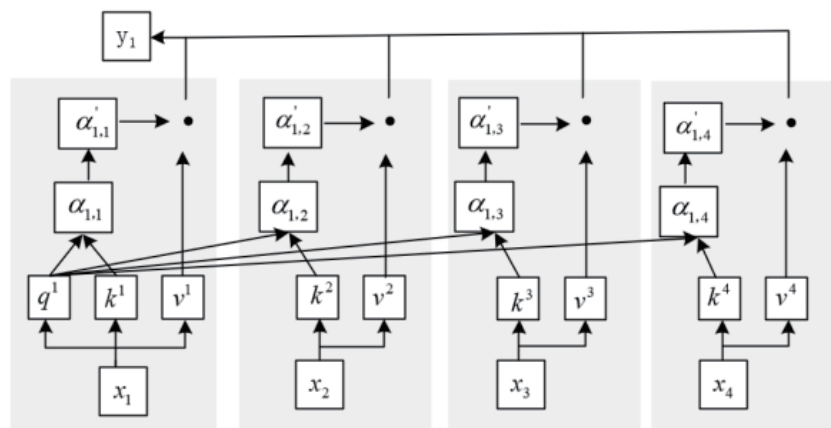


Fig. 1. Calculation of y_1 .

$$a'_{1,i} = \frac{\exp(a_{1,i})}{\sum_j \exp(a_{1,j})} \quad (4)$$

$$v = W_v x \quad (5)$$

$$y_1 = \sum_i a'_{1,i} v_i \quad (6)$$

2.1.2 Multimodal data fusion

The image data acquired by the camera are processed by the SA mechanism as follows. The feature map of an input image is first transformed by convolutional linear projections into the query matrix Q , the key matrix K , and the value matrix V . The matrices Q and K are then matched by a scaled dot product to obtain the correlation between image regions, and the resulting attention weights are applied to V . Formally, the SA operation is expressed as

$$\text{attention}(Q, K, V) = v \cdot \text{softmax}\left(\frac{Q \cdot K^T}{\sqrt{d}}\right), \quad (7)$$

where d denotes the dimension of the query and key vectors; dividing the dot product by this factor prevents the softmax from saturating and thus stabilizes training.

The LiDAR emits pulsed laser beams and measures key features of surrounding objects, including their position and velocity. From these measurements, the robot obtains essential environmental information, such as the distance and bearing of the target relative to itself. To exploit this information fully and improve learning efficiency, the data returned by the two LiDAR sensors are fused along the feature dimension. The fused vector is then used as part of the input to the network model. The whole process is illustrated in Fig. 2.

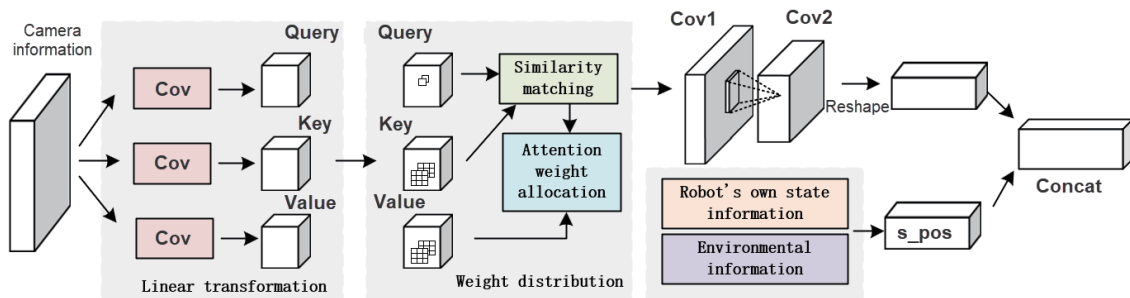


Fig. 2. (Color online) Multidata fusion process.

2.2 DDPG optimized with HER

Figure 3 illustrates the overall architecture of the proposed ICM-DDPG-HER path-planning model. It consists of four components: a sensor data fusion module, a DDPG AC module, an experience replay buffer optimized by HER, and an ICM. The image captured by the camera is first processed by the SA module; the resulting visual features are then fused with the obstacle information provided by the LiDAR and with the robot's own state. The fused multimodal representation serves as the input to the AC networks for motion decision-making, while the ICM derives an additional intrinsic reward from the same representation, and the HER module relabels stored transitions so as to densify the supervision signal.

2.2.1 DDPG algorithm

DDPG is an off-policy AC algorithm built on two stabilizing mechanisms, namely, experience replay and target networks. Experiences are stored in the replay buffer as tuples $\langle S_t, a_t, r_t, S_{t+1} \rangle$. S_t denotes the current state, a_t the action selected by the policy, r_t the reward returned by the environment, and S_{t+1} the next state. Sampling minibatches from the buffer following a given rule disrupts the temporal correlation between consecutive transitions and thereby improves generalization.

DDPG maintains four neural networks: the actor, the target actor, the critic, and the target critic. Updating the actor and the critic requires minibatches sampled from the replay buffer. In a sampled transition $\langle S_t, a_t, r_t, S_{t+1} \rangle$, the critic is updated by gradient descent so as to minimize the

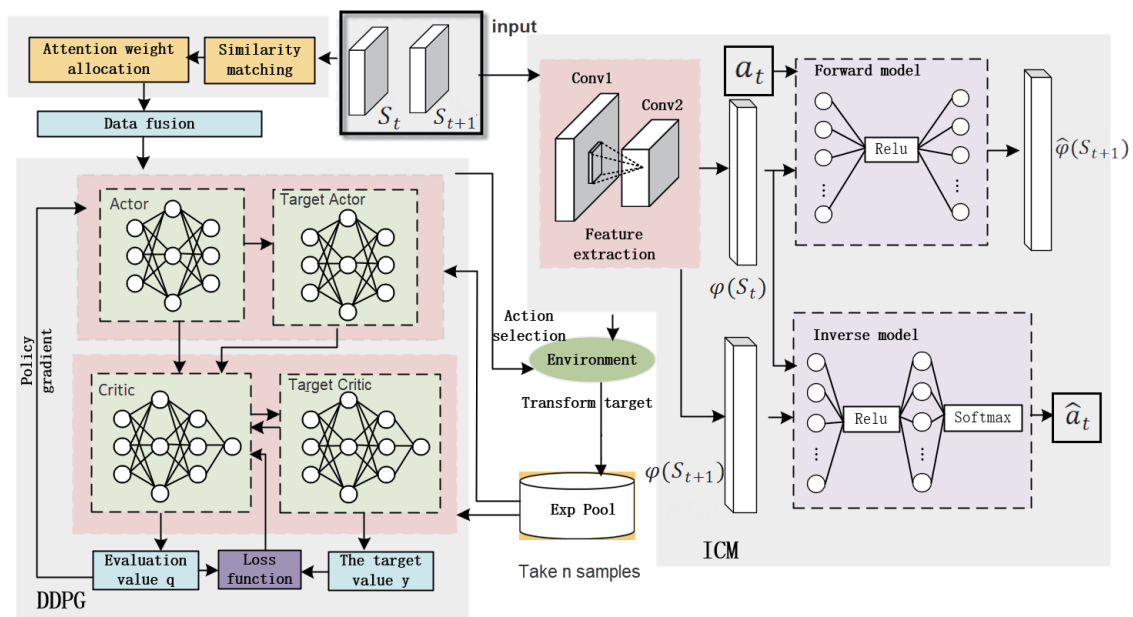


Fig. 3. (Color online) Proposed network model structure.

discrepancy between its evaluation and the target value, and the target value is produced by the target critic as

$$y_t = \begin{cases} r_t, & \text{if } is_end_t \text{ is true} \\ r_t + \gamma Q'(S_{t+1}, a_{t+1}, \omega'), & \text{if } is_end_t \text{ is false} \end{cases} \quad (8)$$

where Q' denotes the target critic, γ the discount factor, ω the parameters of the critic, and ω' the parameters of the target critic; the action at the next state is selected by the target actor.

2.2.2 DDPG-HER algorithm

Andrychowicz *et al.* proposed HER, a replay strategy designed for sparse-reward, goal-conditioned tasks.⁽²⁵⁾ In robot path planning, only a limited subset of actions triggers a positive reward, and a prolonged absence of reward severely degrades learning efficiency. HER adopts a progressive learning strategy that enables the robot to learn not only from successful experiences but also from failed ones: once an episode terminates, the state actually reached is treated as a hypothetical goal and the trajectory is replayed with this substituted goal, so that an episode that originally failed still yields at least one successful experience. In this way, HER alleviates the sparse-reward problem in DRL to a considerable extent.

Conventional DRL algorithms store transitions in the replay buffer in the form of $\langle S_t, a_t, r_t, S_{t+1} \rangle$. HER additionally stores the goal, which makes the data structure different from that of a classical DRL replay buffer. Because HER transforms the goal, every state is associated with the original goal g and with a substituted new goal g' ; consequently, the data must be stored in the replay buffer both as $\langle \bar{S}_t | g, a_t, r_t, S_{t+1} | g \rangle$ and as $\langle S_t | g', a_t, r_t, S_{t+1} | g' \rangle$.

2.3 Curiosity-driven intrinsic reward mechanism

The reward value is the primary supervisory signal in DRL: it determines which actions are reinforced and therefore shapes the learned policy. In robot path planning, a hand-crafted extrinsic reward is most commonly used, but such a reward is sparse and inherently uncertain, which adversely affects the learning performance of robot path optimization. To remedy this, in the present study, we construct an intrinsic reward by introducing a curiosity mechanism and integrate it with the extrinsic reward to form a dual reward mechanism, thereby improving the robot's capability to explore unknown environments. The curiosity mechanism operates as follows. First, the model predicts the future state from the current state and the current action. Then, the error between the predicted state and the actual state is fed back to the robot to measure how familiar the robot is with its current situation. On this basis, a curiosity-driven intrinsic reward model, namely, the ICM, is constructed; its structure is illustrated in Fig. 4.

2.3.1 ICM

The ICM consists of two sub-networks, an inverse model and a forward model. Both operate on a learned feature representation of the state rather than on raw observations, so that the

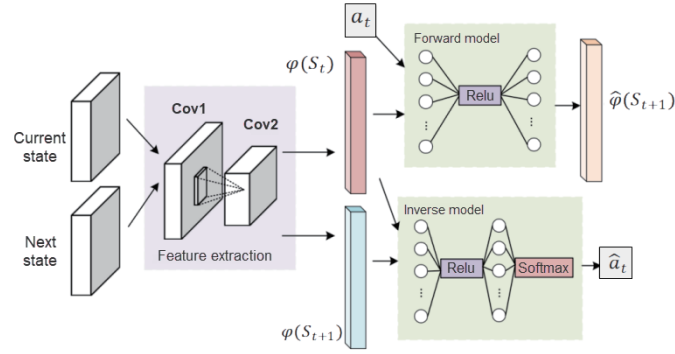


Fig. 4. (Color online) Basic structure diagram of the curiosity mechanism.

intrinsic reward reflects only those aspects of the environment that the robot can influence. In the inverse model, the current state S_t and the next state S_{t+1} are encoded into their corresponding feature vectors and fed to the network, which predicts the action $\hat{a}_t$ that caused the transition:

$$\hat{a}_t = g(S_t, S_{t+1}; \theta_I), \quad (9)$$

where g denotes the inverse model and θ_I its learnable parameters.

The forward model predicts the feature representation of the next state, $\hat{\phi}(S_{t+1})$, from the feature representation of the current state $\phi(S_t)$ and the action a_t :

$$\hat{\phi}(S_{t+1}) = f(\phi(S_t), a_t; \theta_F), \quad (10)$$

where f denotes the forward model and θ_F its learnable parameters.

2.3.2 Dual reward function

With the curiosity mechanism, the robot receives two reward components at each time step: the intrinsic reward r_t^i computed by the ICM and the extrinsic reward r_t^e provided by the environment. The intrinsic reward is defined as the prediction error of the forward model, so that states the robot cannot yet predict are regarded as novel and are rewarded, which encourages the robot to visit and perceive unknown regions. The agent therefore optimizes a composite reward that combines both components. The extrinsic reward is designed as follows.

$$r_t^e = \begin{cases} +10, & d_g < d_{gs} \\ -10, & d_o < d_{os} \\ 10(d_{g-1} - d_g), & d_{g-1} > d_g \\ 10(d_g - d_{g-1}), & d_g > d_{g-1} \end{cases} \quad (11)$$

- (1) The robot reaches the target. d_g denotes the distance between the robot and the target position, and the safety threshold d_{gs} is set to 0.2 m. If d_g is smaller than d_{gs} , the robot is considered to have reached the target and receives a reward of +10.
- (2) The robot collides with an obstacle. d_o denotes the distance between the robot and the obstacle, and a collision threshold d_{os} is defined. If d_o is smaller than d_{os} , the robot is considered to have collided with the obstacle and receives a reward of -10.
- (3) The robot approaches the target. Let d_{g-1} denote the distance between the robot and the target position at the previous time step. If the current distance d_g is smaller than d_{g-1} , the robot is considered to be approaching the target and receives a positive reward $10(d_{g-1} - d_g)$.
- (4) The robot moves away from the target. If d_g is larger than d_{g-1} , the robot is considered to be moving away from the target and receives a negative reward $10(d_g - d_{g-1})$.

2.4 Overall training procedure

The training procedure of ICM-DDPG-HER is summarized as follows. At each time step, the robot observes a camera image and LiDAR measurements, which are encoded and fused by the SA module into a state representation. The actor maps this representation to a continuous velocity command, which is executed in the Gazebo environment together with exploration noise. The environment returns the next state and the extrinsic reward defined in Sect. 2.3.2, while the ICM computes the intrinsic reward from the prediction error of the forward model. The resulting transition is stored in the replay buffer together with the current goal; when an episode terminates without reaching the goal, HER additionally relabels a copy of the transition with the achieved state as a hypothetical goal. Minibatches are then sampled from the buffer to update the critic, the actor, the ICM, and the corresponding target networks. This loop is repeated until the maximum number of episodes is reached.

3. Simulation Results and Analysis

All experiments were conducted on a workstation equipped with an NVIDIA GeForce GTX 1660 Ti GPU. The software stack comprises the robot operating system running on Ubuntu 18.04, the Gazebo simulator for constructing realistic robot operating scenarios, and Rviz for three-dimensional visualization. All algorithms were implemented in Python.

3.1 Simulation setup

3.1.1 Simulation environment

All experiments are conducted in a closed environment. Cubes, spheres, and cylinders are placed as static obstacles, and pedestrians moving near the robot act as dynamic obstacles. The static-obstacle and dynamic-obstacle training environments are shown in Figs. 5(a) and 5(b), respectively, and the corresponding testing environments are shown in Figs. 5(c) and 5(d).

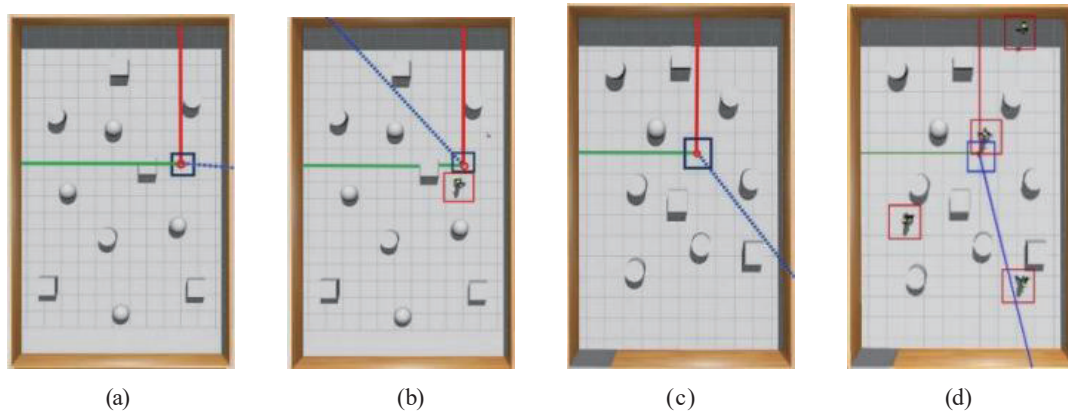


Fig. 5. (Color online) (a) Static-obstacle training environment. (b) Dynamic-obstacle training environment. (c) Static-obstacle testing environment. (d) Dynamic-obstacle testing environment. Green represents the x -axis, red the y -axis, and blue the z -axis.

In these experiments the robot uses a camera sensor and a LiDAR sensor. The camera obtains the image information in front of the robot, while the LiDAR senses the environment, performs positioning, and provides the bearing and distance of the target point. At the same time, the LiDAR helps the robot judge whether it encounters obstacles during training, which speeds up model convergence.

3.1.2 Parameter settings

The objective of robot training is to avoid obstacles effectively and to reach the target point along a short path, thereby continuously optimizing the network model proposed in this paper. The model is trained for 500 episodes, and the maximum number of steps per episode (Max_steps) is 400. If the robot has not reached the target point within 400 steps, the episode is regarded as a failure and the robot is reset before the next episode begins. A safety threshold is also defined for the target point: when the distance between the robot and the target falls below this threshold, the target is considered reached and the next episode can start. If the robot collides with an obstacle during training, the episode likewise fails and the robot is reset. The main parameters of the network model are listed in Table 1.

3.1.3 State space configuration

When DRL techniques are applied to robot path planning, the state space must be designed carefully. A well-designed state not only markedly improves training efficiency and shortens learning time, but also improves the quality of the planned path, enabling the robot to generate more accurate and efficient trajectories.

The DDPG algorithm can be applied in continuous state-action spaces, and this study is developed around it. Such continuous spaces, however, can make model convergence difficult. Therefore, the state is designed mainly from the obstacle information acquired by the camera and the environmental information explored by the LiDAR, namely, the image in front of the

Table 1
Parameter settings.

Parameter name	Description	Parameter value
γ	Exploration rate	0.9
α_A	Learning rate of actor	0.001
α_C	Learning rate of critic	0.001
$M_{capacity}$	Experience replay buffer size	10000
Num_{batch}	Sample size	64
τ	Soft update coefficient	0.01
$Learning_{start}$	Number of episodes before learning starts	70
$Episode$	Number of training episodes	500
Max_{steps}	Maximum number of steps per episode	400
λ	Loss weight	1.0
β	Weight balancing the forward and inverse model losses	0.8

robot and the location of the target point. These quantities constitute the perceptual input used to train the DRL model, and the state is defined as

$$state = (S_c, S_d, S_a), \quad (12)$$

where S_c denotes the camera information, S_d the distance from the robot to the target point, and S_a the bearing of the target point relative to the robot.

The environmental information captured by the robot's camera is shown in Fig. 6. Its detection range covers 180 degrees directly in front of the camera, providing a broad and clear field of view. The perception field of the robot LiDAR sensor is shown in Fig. 7: the scanning angle ranges from -3 to 3 degrees and the effective detection distance from 0.1 to 10 m. By scanning precisely with the LiDAR, S_d and S_a can be obtained, which provide detailed information about the robot's surrounding environment.

3.1.4 Action space configuration

The motion of the robot consists of translation, which produces forward and backward movement, and rotation, which produces left and right turns; these are represented by the linear velocity and the angular velocity, respectively. Because the action space of the robot is continuous, the linear velocity and the angular velocity output by the DRL network model are also continuous. In this study, the action space is designed as

$$a = (v_l, v_a), \quad (13)$$

where v_l is the linear velocity, limited to the range $(0, 1)$, and v_a is the angular velocity, limited to the range $(-0.5, 0.5)$. A positive angular velocity indicates that the robot turns left, whereas a negative angular velocity indicates that it turns right.



Fig. 6. (Color online) Environmental information captured by the camera.

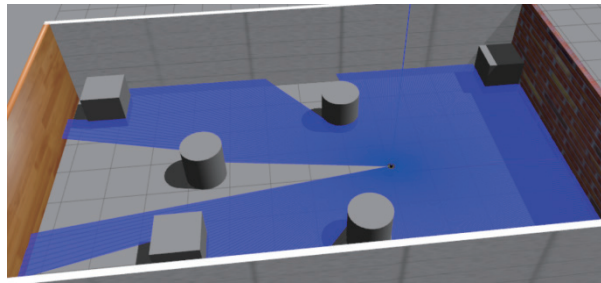


Fig. 7. (Color online) Perception field of the robot LiDAR sensor.

3.2 Training results and analysis

To verify the feasibility and superiority of the proposed algorithm, five algorithms are compared and analyzed: DQN, DDPG, DDPG-HER, the DDPG-HER algorithm based on the SA mechanism (SA-DDPG-HER), and the proposed ICM-DDPG-HER based on the SA mechanism. In the training stage, the reward value obtained by each algorithm is compared; in the testing stage, the length of the path planned by each of the five algorithms is used as the comparative metric.

3.2.1 Static-obstacle environment

To accelerate model convergence, the five algorithms were first trained for 500 episodes in an obstacle-free environment, and were then trained for a further 500 episodes in a simulated environment containing static obstacles.

Table 2 reports the average reward obtained by DQN, DDPG, DDPG-HER, SA-DDPG-HER, and the proposed ICM-DDPG-HER over every 100 episodes of training in the static-obstacle environment. A closer examination shows that the reward to DQN rises during the first 300 episodes but then plateaus during the last 200, with the average reward remaining below zero throughout. The average reward of DDPG increases noticeably in the last 100 episodes compared

Table 2

Average reward per 100 episodes during training in the static-obstacle environment.

Episodes	100	200	300	400	500	Avg
DQN	−7.69	−4.09	−1.28	−0.95	−1.68	−3.14
DDPG	−6.44	−1.24	0.36	2.29	10.70	1.01
DDPG-HER	−5.38	−3.56	3.47	8.30	17.39	4.10
SA-DDPG-HER	−6.58	0.96	5.54	11.55	15.78	5.45
Proposed method	−2.58	1.06	5.02	14.99	21.52	8.00

with the first 400, and its overall growth rate is higher than that of DQN. DDPG-HER remains stable during the first 200 episodes and then rises faster than DDPG between episodes 200 and 500. SA-DDPG-HER obtains a higher reward than all three baselines. The reward obtained by the proposed algorithm is almost the same as that of SA-DDPG-HER in the third block of 100 episodes and is better than that of the other four algorithms in the remaining four blocks.

Averaged over all 500 episodes, the proposed algorithm attains the highest average reward of 8.00, compared with 5.45 for SA-DDPG-HER, 4.10 for DDPG-HER, 1.01 for DDPG, and −3.14 for DQN; the corresponding absolute gains over DQN are 11.14, 8.59, 7.24, and 4.15, respectively. A higher reward indicates that the robot stays closer to the target and therefore has a greater probability of reaching it, and a reward that stabilizes at a high level at an earlier stage indicates a better network model. These results show that the proposed algorithm converges faster and to a better policy than all the baselines.

3.2.2 Dynamic-obstacle environment

In the same dynamic-obstacle environment, the five algorithms were trained for a further 500 episodes starting from the models trained with static obstacles. The reward obtained every 100 episodes is shown in Table 3. The reward obtained by DQN is the lowest among the five algorithms, whereas those obtained by DDPG and DDPG-HER lie in the middle range and that obtained by SA-DDPG-HER is higher than all three. The reward obtained by the proposed ICM-DDPG-HER based on the SA mechanism is the highest among the five algorithms in the dynamic-obstacle environment. The gradual increase in reward demonstrates that, on the basis of DDPG, it is both feasible and effective to improve sample utilization with the self-attention mechanism, to learn from failure experience through hindsight relabeling, and to explore more unknown areas through the curiosity mechanism.

3.3 Testing results and analysis

3.3.1 Static-obstacle environment

For a fair comparison, the models trained with static obstacles are tested in a new static environment, as shown in Fig. 5(c). The five groups of experiments described in Sects. 3.3.1 and 3.3.2 are carried out in a closed environment with static and dynamic obstacles. Different start and goal points are configured for each group to test the paths generated by the five algorithms.

Table 3

Average reward per 100 episodes during training in the dynamic-obstacle environment.

episodes	100	200	300	400	500	Avg
DQN	0.26	2.98	6.00	11.22	16.03	7.30
DDPG	12.72	15.45	15.65	17.00	18.23	15.81
DDPG-HER	13.38	15.66	20.46	18.87	24.49	18.57
SA-DDPG-HER	22.60	24.34	25.22	24.87	25.47	24.50
Proposed method	20.89	26.47	28.82	28.30	28.60	26.61

Each group of tests uses the same starting point and target point. The result of one group is shown in Fig. 8, and the paths planned by the proposed ICM-DDPG-HER in the other four groups are shown in Fig. 9. These results show that as the algorithm is progressively improved, the trajectory planned by the robot becomes shorter and smoother.

The test results show that the proposed algorithm successively overcomes the shortcomings of DDPG in static-obstacle environments. Each incremental component shortens the planned trajectory, and the proposed algorithm, which combines all three, yields the shortest and smoothest path among the five methods. This confirms that the improvements obtained during training transfer to an unseen static-obstacle testing environment, and that the learning efficiency of the robot is improved.

3.3.2 Dynamic-obstacle environment

The models trained in the dynamic-obstacle environment were tested in the environment shown in Fig. 5(d). Five groups of experiments were carried out, and the resulting path lengths are shown in Fig. 10. In Fig. 10, the horizontal-axis values 1–5 denote the five environments with distinct start and goal points. The vertical axis represents the number of moving steps, which serves as an indicator of path length, taken by different algorithms to travel from the start point to the goal point in each environment. In the dynamic-obstacle test, the path length planned by DDPG is 38, 21, 36, 39, and 28% shorter than that planned by DQN in the five groups, and the path length planned by DDPG-HER is 7, 11, 13, 15, and 21% shorter than that planned by DDPG. The paths planned by SA-DDPG-HER are generally shorter than those planned by DDPG-HER, although one counterexample occurs in the third group: the robot encountered a dynamic obstacle during path planning and the resulting detour made its path slightly longer than that planned by DDPG-HER.

The path planned by the proposed algorithm is shorter than that planned by SA-DDPG-HER by 40, 7, 22, 4, and 19% in the five groups. Overall, the average path length obtained by the proposed algorithm is approximately one-third of that obtained by DQN. These results show that the ICM-DDPG-HER algorithm based on the self-attention mechanism can also plan a shorter and smoother path in a dynamic-obstacle environment, can avoid obstacles effectively, and therefore, gives the robot strong environmental adaptability.

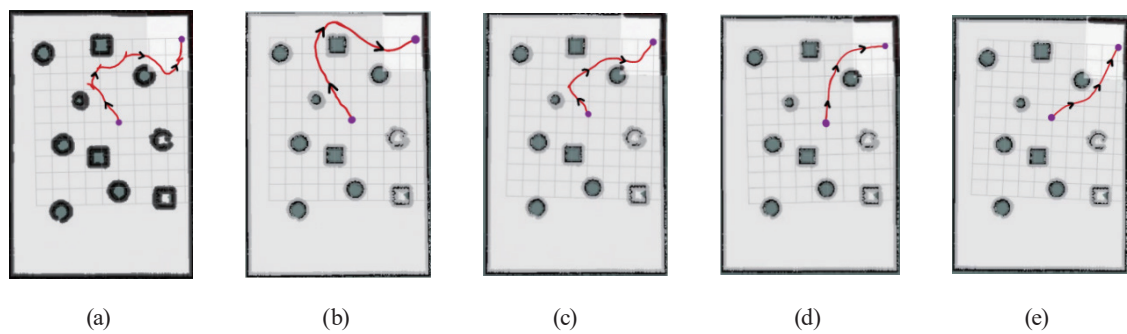


Fig. 8. (Color online) Test results of static-obstacle path planning: (a) DQN, (b) DDPG, (c) DDPG-HER, (d) SA-DDPG-HER, and (e) the proposed method.

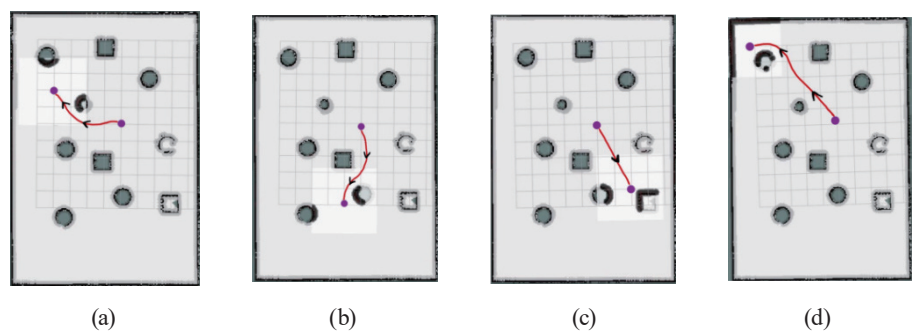


Fig. 9. (Color online) Path planning by the proposed method in the other four experiments in a static-obstacle environment.

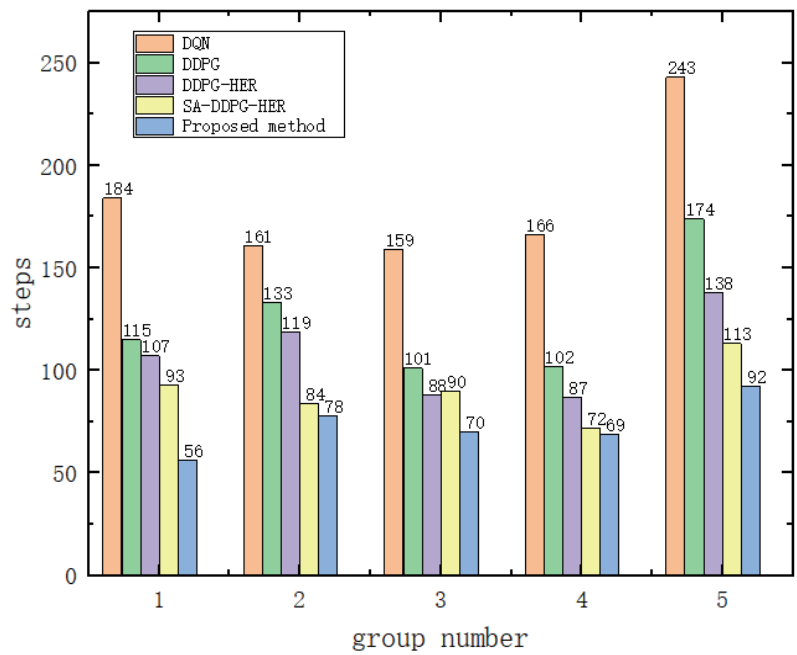


Fig. 10. (Color online) Test results of dynamic-obstacle path planning.

3.4 Discussion

The simulation results can be interpreted from three perspectives. First, the SA-based fusion module is the main source of the improvement from DDPG-HER to SA-DDPG-HER: the average episodic reward increases from 4.10 to 5.45 in the static environment and from 18.57 to 24.50 in the dynamic environment (Tables 2 and 3). By reweighting visual features in accordance with their relevance to the current decision, the module amplifies task-informative regions while suppressing redundant and noisy responses, so that the robot extracts more usable information from the same camera and LiDAR data without requiring additional interactions. This directly mitigates the inefficient exploitation of high-dimensional visual features that characterizes passive perception, and is correspondingly reflected in the stronger robustness of the sensing data against illumination variation and clutter, that is, an overall improvement in sensor performance. Second, HER contributes most in the early stage of training. By relabeling failed episodes with hypothetical goals, it converts otherwise useless transitions into effective supervision, raises the average reward from 1.01 to 4.10 in the static environment, and substantially shortens the time needed to obtain the first successful episode, thereby alleviating the sparse-reward problem. Third, the curiosity-driven intrinsic reward provides the additional gain from SA-DDPG-HER to the proposed method, mainly in the dynamic environment, where passive perception leaves large parts of the state space unvisited; it raises the average reward from 24.50 to 26.61 and yields paths that are 4–40% shorter than those of SA-DDPG-HER. Because the three mechanisms act on distinct stages of the learning pipeline, namely, feature representation, sample supervision, and environmental exploration, their benefits are largely complementary rather than redundant, which explains why their combination consistently outperforms every partial variant.

4. Conclusions

In this paper, we proposed ICM-DDPG-HER, an improved DDPG path-planning algorithm for vision-based robots, which addresses three limitations of conventional intelligent perception: passive perception, degraded perception capability in complex scenarios, and low sample-utilization efficiency. Building on the DDPG framework, the algorithm introduces three complementary components that together form a unified sensor-driven perception pipeline. First, an SA-based multimodal sensor fusion method is developed to integrate heterogeneous information acquired from multiple onboard sensors. Specifically, the raw image frames captured by the camera sensor are first processed by an SA module to strengthen visual feature representation, and the enhanced visual features are then fused with information from the LiDAR sensor. Second, HER combined with a progressive learning strategy injects hypothetical goals into the experience buffer, turning invalid interaction trajectories into effective training samples and alleviating reward sparsity. Third, a curiosity-driven intrinsic reward is combined with the extrinsic reward into a dual reward framework that balances exploitation and exploration, prompting the robot to actively perceive unknown scenarios and reduce over-reliance on passive sensing.

Simulation results show that the proposed algorithm converges faster and is more robust to sensing interference—e.g., partial occlusion or noise in the camera and ranging modalities—enabling efficient obstacle avoidance and target reaching in unknown complex environments. In dynamic obstacle tests, the average planned path length is reduced to about one-third of that of DQN. These results suggest that the proposed method offers a practical approach for improving visual sensing and intelligent perception in robotic navigation.

Future work will proceed along the following lines: transferring the algorithm to physical robotic platforms to quantify the sim-to-real gap and validate robustness under real sensor noise and dynamics conditions; benchmarking against a broader set of modern off-policy methods; and reporting additional metrics such as success rate, collision rate, and computational cost per decision. The scalability of the fusion framework in crowded dynamic scenes and the integration of large pretrained vision-language models are further promising directions that we plan to explore.

Acknowledgments

This work was supported by Jiangsu Province Industry University-Research Cooperation Projects (BY20251033 and BY20231232), Suqian Science and Technology Plan Guiding Projects (Z2025040 and Z2023138) and Suqian University Talent Introduction Research Start-up Fund Project (University 2022XRC091).

References

- 1 S. Li, F. Yang, X. Y. Wang, C. D. Guo, L. L. Tong, and Q. Hu: *Acta. Automatica Sinica*. **48** (2022) 1869. <https://doi.org/10.16383/j.aas.c210263>
- 2 A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W. Y. Lo, P. Dollar, and R. Girshick: *Proc. 2023 IEEE/CVF Int. Conf. on Computer Vision*. (IEEE, 2023) 3992. <https://doi.org/10.1109/ICCV51070.2023.00371>
- 3 K. C. Wang, V. Saxena, S. Cheng, Y. Koga, and D. F. Xu: *arXiv:2606.23625* (2026). <https://arxiv.org/abs/2606.23625>
- 4 L. Zhang, K. X. Xie, X. H. Zheng, Y. S. Song, C. J. Wang, and Y. Z. Zhao: *CAAI Trans. Intell. Syst.* **19** (2024) 767. <https://doi.org/10.11992/tis.202306001>
- 5 C. Y. Gao, Q. Liu, M. H. Li, and X. Liu: *CAAI Trans. Intell. Syst.* **21** (2026) 834. <https://doi.org/10.11992/tis.202510008>
- 6 J. J. Padalunkal, C. Salvato, V. Bomnale, J. K. Wang, and Y. Xiang: *arXiv:2410.09072* (2024). <https://arxiv.org/abs/2410.09072>
- 7 F. Q. Pu, L. Jiang, and W. M. Yang: *arXiv:2603.00615* (2026). <https://arxiv.org/abs/2603.00615>
- 8 V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller: *arXiv:1312.5602* (2013). <https://arxiv.org/abs/1312.5602>
- 9 H. van Hasselt, A. Guez, and D. Silver: *Proc. 30th AAAI Conf. Artificial Intelligence (AAAI, 2016)* 2094. <https://doi.org/10.1609/aaai.v30i1.10295>
- 10 V. R. Konda and J. N. Tsitsiklis: *SIAM J. Control Optim.* **42** (2003) 1008. <https://doi.org/10.1137/S0363012901385691>
- 11 T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra: *arXiv:1509.02971* (2016). <https://arxiv.org/abs/1509.02971>
- 12 Y. Zhang and H. Qiu: *Sensors* **22** (2022) 5020. <https://doi.org/10.3390/s22135020>
- 13 L. Huo and Y. Tang: *Appl. Sci.* **13** (2023) 325. <https://doi.org/10.3390/app13010325>
- 14 M. Li, T. Huang, and W. Zhu: *Pattern Recognit.* **131** (2022) 108875. <https://doi.org/10.1016/j.patcog.2022.108875>
- 15 J. W. Wang, T. Wang, Z. C. He, W. Z. Cai, and C. Y. Sun: *Appl. Intell.* **53** (2023) 6195. <https://doi.org/10.1007/s10489-022-03503-6>

- 16 M. G. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos: Proc. 30th Int. Conf. Neural Information Processing Systems (NIPS, 2016) 1. <https://arxiv.org/abs/1606.01868>
- 17 D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell: Proc. 2017 IEEE Conf. Computer Vision and Pattern Recognition Workshops (IEEE, 2017) 488. <https://doi.org/10.1109/CVPRW.2017.70>
- 18 W. Y. Zhang, Q. Chen, J. Y. Yan, S. Zhang, and J. Y. Xu: Energy **236** (2021) 121492. <https://doi.org/10.1016/j.energy.2021.121492>
- 19 Y. D. Wang, Y. L. Jia, Y. H. Tian, and J. Xiao: Expert Syst. Appl. **200** (2022) 117013. <https://doi.org/10.1016/j.eswa.2022.117013>
- 20 W. J. Li, M. Yue, J. Y. Shanguan, and Y. Jin: Int. J. Control Autom. Syst. **21** (2023) 563. <https://doi.org/10.1007/s12555-021-0642-7>
- 21 D. G. Thomas, D. Olshanskyi, K. Krueger, T. Wongpiromsarn, and A. Jannesari: arXiv:2105.12254 (2021). <https://arxiv.org/abs/2105.12254>
- 22 X. R. Pan, C. J. Ge, R. Lu, S. J. Song, G. F. Chen, and Z. Y. Huang: Proc. 2022 IEEE/CVF Conf. Computer Vision and Pattern Recognition (IEEE, 2022) 805. <https://doi.org/10.1109/CVPR52688.2022.00089>
- 23 Y. W. Zhou and W. Z. Zhang: Front. Neurorobot. **19** (2025) 1646336. <https://doi.org/10.3389/fnbot.2025.1646336>
- 24 X. J. Zhao, H. Ye, H. Li, and Z. X. Sun: Comput. Sci. **52** (2025) 306. <https://doi.org/10.11896/jsjx.240500099>
- 25 M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba: arXiv:1707.01495 (2017). <https://arxiv.org/abs/1707.01495>

About the Authors



Fengying Wang received her B.S. degree from the University of Science and Technology Liaoning in 2001 and her M.S. degree from Shenyang Jianzhu University in 2004. She worked at Shenyang Jianzhu University from 2004 to 2022 and has since been an associate professor at Suqian University and a part-time master's supervisor at Shenyang Jianzhu University. Her research interests include machine learning, intelligent software engineering, and visual inspection. (wangfengying@squ.edu.cn)



Ying Chen received her B.S. degree from Zhengzhou University of Aeronautics, China, in 2021 and her M.S. degree from Shenyang Jianzhu University, China, in 2024. Since 2024, she has been an employee of the Nanyang Branch, Henan Co., Ltd., China Mobile Communications Group. Her research interest is deep reinforcement learning. (644316491@qq.com)



Liming Du received his B.S. and M.S. degrees from Shenyang Jianzhu University in 2000 and 2003, respectively, and his Ph.D. degree from Northeastern University, China, in 2015. He worked at Shenyang Jianzhu University from 2003 to 2022 and has since been an associate professor at Suqian University and a part-time master's supervisor at Shenyang Jianzhu University. His research interests include machine learning and intelligent software testing. (duliming@squ.edu.cn)