A Hybrid Technique for Active SLAM Based on RPPO Model with Transfer Learning

Shuhuan Wen (swen@ysu.edu.cn)
Yanshan University https://orcid.org/0000-0002-7646-4958
Zhixin Ji
Yanshan University
Ahmad B. Rad
Simon Fraser University
Zhengzheng Guo
Yanshan University

Research Article

Keywords: Active SLAM, RPPO, Deep Separable Convolution, Data Batch Processing, Transfer Learning

Posted Date: January 11th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1229897/v1

A hybrid technique for Active SLAM Based on RPPO model with Transfer Learning

Shuhuan Wen[1,2*], Zhixin Ji[1,2], Ahmad B. Rad[3] and Zhengzheng Guo[1,2]

[1*]Engineering Research Center of the Ministry of Education for Intelligent Control System and Intelligent Equipment, Yanshan University, 438 West Hebei Street, Qinhuangdao, 066004, Hebei province, China.

[2]Key lab of Industrial Computer Control Engineering of Electrical Engineering of Heibei Province, Yanshan University, 438 West Hebei Street, Qinhuangdao, 066004, Hebei province, China.

[3]School of Engineering Sciences, Simon Fraser University, 250-13450,102 Avenue, Surrey, V3T 0A3, BC, Canada.

*Corresponding author(s). E-mail(s): swen@ysu.edu.cn;
Contributing authors: jzxin121@qq.com; arad@sfu.ca;
gzz@stumail.ysu.edu.cn;

Abstract

The problem of exploration in unknown environments is still a great challenge for autonomous mobile robots due to the lack of a priori knowledge. Active Simultaneous Localization and Mapping (SLAM) is an effective method to realize obstacle avoidance and autonomous navigation. Traditional Active SLAM is usually complex to model and difficult to adapt automatically to new operating areas. This paper presents a novel Active SLAM algorithm based on Deep Reinforcement Learning (DRL). The Relational Proximal Policy Optimization (RPPO) model with deep separable convolution and data batch processing is used to predict the action strategy and generate the action plan through the acquired environment RGB images, so as to realize the autonomous collision free exploration of the environment. Meanwhile, Gmapping is applied to locate and map the environment. Then, based on Transfer Learning, Active SLAM algorithm is applied to complex unknown environments with various dynamic and

static obstacles. Finally, we present several experiments to demonstrate the advantages and feasibility of the proposed Active SLAM algorithm.

Keywords: Active SLAM, RPPO, Deep Separable Convolution, Data Batch Processing, Transfer Learning

1 Introduction

If the environmental prior information is known in advance, the mobile robot can freely interact with the physical environment and navigate in it. However, in an unpredictable environment, the basic capabilities of mobile robots, such as autonomous navigation, positioning and mapping, still presents a severe challenge. To explore in the unknown real environment, mobile robots need to have the ability to avoid obstacles, as well as accurately locate themselves according to the environmental information and to ensure avoiding collisions with obstacles. However, in a dynamic or an unknown environment, it is difficult to generate an effective motion plan. Therefore, to achieve the Active SLAM [1], the robot's exploration of the environments not only includes the method of avoiding obstacles, but also have the ability of detecting its position and state by obtaining local environmental information from the vicinity in space and time, and at the same time transforming the environmental information into its own coordinate system to map the surrounding environment. Several types of sensors can be employed to obtain information in an environment, such as laser scanners, lidars and depth cameras. Using these sensors to perceive the data, it is possible for robots to decide how to explore in an unknown environment. In practical applications, with the ability of Active SLAM, robots can overcome the space restrictions and expand their application areas such as patrol surveillance and space search [2–5].

The core problem of Active SLAM is mobile robots' autonomous collision-free navigation and solving SLAM, which requires mobile robots to explore as many unknown areas as possible under the premise of ensuring the accuracy of pose estimation in unknown environments, and complete the construction of explored area maps at the same time. In an uncertain environment, autonomous obstacle avoidance navigation of the robot is usually solved through the use of ranging sensors and the application of predetermined logic [6]. Sensors are used to collect environmental information to detect obstacles in the environment and the free space where the robot can move. Here, the visual SLAM approach can be used to understand the environment and navigate [7–9]. Segmentation can identify the traversable plane that the robots' advances from the visual image or point cloud [10–13]. Laser, sonar and other ranged sensors can also detect obstacles and their shape and position [14, 15]. After the obstacle is detected, the motion plan is selected according to the strategy, then during the movement of the robot, the odometer information can be used to obtain the preliminary estimation of pose. At the same time, the

environmental information obtained by the sensors is transferred to the robot coordinate system for accurate modification of the robot pose, so the accurate positioning of the robot is obtained. Finally, based on the precise positioning, the sensor data is added to the map to complete the construction of the entire map.

In this paper, we propose an active SLAM algorithm to overcome these challenges. With the proposed algorithm, mobile robots can generate motion strategies to avoid collisions and explore faster by using real-time environmental information collected in the simulation and real unknown environments, and complete SLAM in the process of motion. More specifically, the contribution of this research study is twofold as follows:

- An Active SLAM algorithm based on the RPPO model is proposed to realize the robot's exploration and mapping of the unknown environment. In addition, data batch processing is added to deal with the possible sample correlation problems, and deep separable convolution is used to reduce computing costs to improve the adaptability of complex environment.
- Based on transfer learning to share network model parameters to accelerate the learning speed of target tasks in complex environments, the robot has stronger learning and reasoning capabilities after being trained in the complex environment of the target domain to adapt to the unknown environments of various dynamic and static obstacles. And the feasibility and high performance of the proposed algorithm are proved through simulation and experiment.

The rest of the paper is organized as follows. In Section 2 related works are reviewed. The proposed SLAM framework is described in Section 3. Followed by experimental results given in Section 4. Finally, conclusions are discussed in Section 5.

2 Related Work

In an ideal condition, the robot can be aware of its surroundings in advance and obtain some information about the environment. In this case, there are a variety of possible algorithms, which can realize the robot's exploration of the motion plan, and then complete the update of the map. Blaer and Allen [16] use sensor information to establish a simple map model to solve the robot's path planning, and then use an algorithm to select feature points from the three-dimensional space to optimize the simple map model, thereby establishing a perfect environment map. Okada et al. [17]optimizes the preliminary map established by locating the information area on a simple two-dimensional map model, and then planning the viewpoint to observe the entire area. These methods can realize the exploration and mapping of the environment, but part of the environmental information is necessary to establish a preliminary model of the environment, so the expected effect cannot be achieved for a completely unknown environment.

In an unknown or uncertain environment, the active SLAM algorithm firstly requires that the mobile robot can realize autonomous obstacle avoidance navigation by using sensor data and applying preset algorithms to generate motion plans. In addition, in the process of exploration, the robot can follow The environment state and odometer information obtained by the camera or other sensors are used to realize the pose estimation, and the environment information is drawn as a map. Chen [18] proposed an active SLAM framework, based on a model predictive control framework, by introducing a control switching mechanism to solve the uncertainty and area coverage tasks of SLAM. Kim and Eustice [19] proposed an active visual SLAM method, which intelligently navigates after acquiring environmental information, and controls the robot's exploration and the return visit of the explored area through a reward mechanism to achieve environmental map construction. Meng [20] proposed an intelligent robot system. The robot explores the environment through view planning and navigation infrastructure to generate effective motion plans, and gradually builds the entire environment map through online 3D SLAM. These methods address some of the shortcomings of the traditional SLAM algorithm, and enhance the robot's autonomy and environmental adaptability in unknown scenarios. However, these algorithms generally do not perform well in the face of dynamic obstacle environments, and they usually require complex modeling, generating and processing a large number of parameters. In addition, it is also a challenge for them to adapt to the new mission environment.

Nowadays, deep learning has shown good performance in autonomous robot navigation tasks. With the development of deep learning, some end-to-end supervised deep learning algorithms have been produced. These algorithms can directly generate control strategies from raw data collected by sensors and don't need to carry out the complex modeling and parameter adjustment process of traditional autonomous obstacle avoidance navigation algorithms. Lei [21] proposed a highly compact network structure, taking the original depth image as input to generate control commands as the network output, and experiments show the effectiveness of the hierarchical structure composed of fusion convolutional neural network (CNN) and decision-making process. However, due to their supervisory nature, if tags are manually marked, a lot of time and energy will be consumed. Even if the self-supervised learning method of automatically generating tags through data sets is adopted without manual supervision, the learning strategy will also be limited by the tag generation strategy. Reinforcement learning explores strategies through continuous interaction with the environment, and then solves a series of problems caused by label generation in supervised deep learning. It has shown great potential in generating good strategies and has tuned strategies in various application areas, such as playing video games [22, 23], financial market analysis [24, 25], Go game [26], And various aspects applied to robots [27–29]. In the application of robot navigation and path planning, reinforcement learning has also been widely used. It can learn autonomous navigation and collision avoidance

from input signals. Shih and Chen [30] proposed an obstacle avoidance and path correction method to determine the behavior of the robot. By detecting environmental information, it continuously interacts with the environment during the movement, so that it can automatically avoid obstacles and effectively move to the target area. Michels [31] proposed an obstacle avoidance method using reinforcement learning. First, the depth estimation of the monocular image is obtained through supervised learning, and then reinforcement learning/strategy search is used to learn a control strategy, and the steering direction is selected according to the output of the visual system.

Deep learning builds a neural network model to identify and analyze the acquired data to generate action strategies, so that the machine can obtain the same learning ability as humans. The application of reinforcement learning enables the robot to optimize the learning strategy through the reward function feedback reward value in the continuous interaction with its surrounding environment, and then continuously optimize and improve its own behavior to achieve the preset goal.The application of the deep reinforcement learning algorithm of the combination of the two in the mobile robot navigation system is to use the sensor data of the robot as input information, and then analyze the state of the robot through the deep learning method, and further use the reinforcement learning algorithm to optimize the robot's decision-making strategy by interacting with the environment to feedback the reward value, and then generate more effective actions.Through deep reinforcement learning algorithm, the restriction of manual tag labeling and tag generation strategy is removed. By presetting the strategic decision goal (no collision, exploring more areas, shortest navigation path, etc.), the robot learns the target task in continuous interaction with the scene, and then iteratively optimizes the structure of the neural network. Thus, the process of acquiring environmental data from sensors to generating control actions is realized.

However, deep reinforcement learning needs to optimize control decisions by exploring strategies in continuous interaction with the environment. Compared with supervised learning, it is less efficient and requires more training data to achieve a good effect. And because of the high cost of trial and error in the real environment, it is usually not for exploration experiments in the real world, but to use a simulation environment with failure and recovery mechanisms for [32]. A large number of practices [33–36] have proved that deep learning networks can be trained in a virtual simulation environment through reinforcement learning and then transferred to the real world. Xie [33] proposed a method of using D3QN (Dueling Deep Double Q network) to avoid obstacles. By obtaining the depth prediction from the monocular depth camera, the neural network trained in the simulation environment is further transferred, and the task of exploration and obstacle avoidance of the robot in the real scene is completed.Tai [36] proposed a mapping-free motion planner based on learning. Through asynchronous deep reinforcement learning methods, the mapping-free motion planner can be trained end-to-end, after the training is completed, it can be directly applied in virtual and real environments to realize the mobile

robot autonomously navigate to the target without colliding with any obstacles.After the mobile robot can autonomously navigate without collision, it can start to move from any position in the unknown environment, and in the process of autonomous navigation, through the analysis of the environmental characteristics obtained by the sensor to predict the action and estimate the pose, and then record the environment characteristics according to the acquired pose information to build a map, so as to realize active SLAM.

3 Active SLAM algorithm based on RPPO model

To tackle the problem of when and how to avoid obstacles and explore the environment from the environmental information obtained by the camera as well as estimate positions and build maps based on the information data obtained by other sensors in the absence of pre-existing maps, we designed an Active SLAM algorithm based on the RPPO model. The RPPO model is used to process real-time visual information to generate motion strategies to explore the environment and avoid obstacles. Specifically, the performance of the neural network is improved by adding Relational network [23] to the network. In addition, data batch processing is added to the algorithm to deal with possible sample correlation problems, and deep separable convolution is used to reduce computing costs. Finally, based on transfer learning, the proposed SLAM algorithm is applied to a more complex environment for training to improve learning and reasoning ability.

3.1 Active SLAM algorithm framework

The proposed Active SLAM in this paper is shown in Figure 1. During the movement of the robot, the initial estimation of robot pose can be obtained by using the odometer information and the motion model of the robot to calculate the trajectory of navigation. Then, the sensor data acquired by the robotic lidar combine with the observation model (laser scanning matching) accurately correct the pose of the robot and match the robot in the map. Finally, on the basis of accurate positioning, the laser data can be added to the map being generated. At the same time, the robot uses the real-time environment information obtained by the kinect camera, based on the goal of faster exploration and autonomous collision-free navigation, through the trained deep reinforcement learning network to predict the action strategy and then generate the motion plan to realize the robot movement in the unknown environment. Repeat this to update the map, and finally complete the construction of the entire map.

This article takes Turtlebot2, a wheeled differential drive robot, as the research object, so the robot has an odometer sensor to record the change of its own motion control. This article uses a simple and abstract mathematical model to represent the motion model of the robot to express the correlation between the robot motion control and the pose conversion, as shown in

equation (1):

$$R_t = f(R_{t-1}, u_t, n_t) = R_{t-1} + u_t + n_t. \quad (1)$$

where $R_{t-1} = [x, y, \theta]_{t-1}$ is the position of the robot at the time $t-1$, where the pose of the robot is simply described by two position parameters and one rotation angle parameter. $u_t = [\Delta x, \Delta y, \Delta\theta]_t$ is the amount of movement control at the time t. R_t represents the pose of the robot at the time t. n_t refers to the motion noise conforming to gaussian distribution. Figure 2 is a simple description of observation model and movement model.

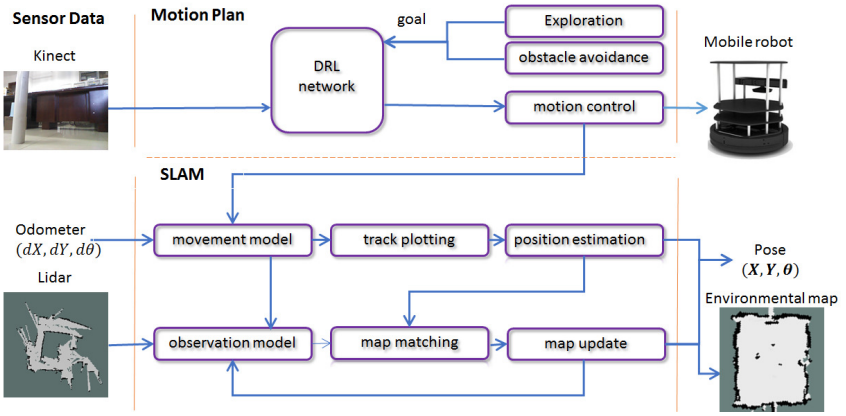

Fig. 1 The Active SLAM algorithm framework proposed

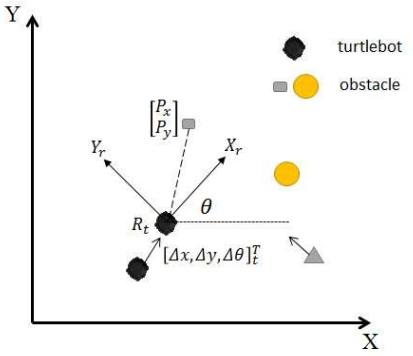

Fig. 2 A simple description of observation model and movement model

The observation model of the robot is used to describe the result of the robot's perception of the external environment, that is, the position of the detected object relative to the robot. In the process of SLAM, the sensor used by the robot is lidar. When the laser sensor scans and observes the feature

points of the obstacle in the plane, the position of the feature point relative to the robot coordinate system can be obtained.

3.2 DRL elements of movement strategy

Obstacle avoidance and exploration based on real-time visual information can be regarded as a decision-making process in which robots interact with the environment through cameras and improve their action prediction strategies in the process of interaction. In order to realize the interaction with the environment and optimize the action strategy, we designed the following DRL elements. The state space and action space of the robot can enable the robot to explore in the environment, while the reward function is to improve the action strategy so that the robot can navigate autonomously to avoid obstacles and explore faster.

State space representation

The state space of the robot includes the state of the robot itself and the three-dimensional information of the environment obtained by the robot. The robot's own sensors and odometer information can be used to obtain the robot's own state, such as the current speed, current position, and heading angle of the robot. The kinect camera can be used to obtain visual information of the environment, such as dynamic and static obstacles or other environmental features.

Action space representation

In order to train the network to generate feasible control strategies, the robot's actions need to be defined correctly. The actions in our network are classified as a feasible sequence of linear and angular velocities in a continuous space, rather than simple commands like "forward", "turn left or right". Set the value of the linear velocity υ of the robot as [0.1, 0.5], and the value of the angular velocity ω as the set$[-\frac{\pi}{3}, \frac{\pi}{3}]$. Defining the action correctly can not only help the robot to effectively complete the task of environment exploration, but also improve the operation efficiency of the deep reinforcement learning algorithm.

Reward function

Reward function is the key to evaluate whether a reinforcement learning algorithm can converge successfully. In the process of robot autonomous exploration forward, it observes environment status s_t at the time t, then the robot will get actions a_t through the DRL network to predict action strategies, which will be rewarded or punished by the reward function on how it walks. Three conditions define the reward function in our DRL network:

$$r\left(s_t, a_t\right) = \begin{cases} r_s & if \ \ rotation = True \\ r_c & if \ \ collision = True \\ \alpha * v * \cos\ \omega & \end{cases} \tag{2}$$

if the robot does a simple rotation, the reward function will punish the robot by giving it a negative reward r_s, in addition, if a robots collision with an obstacle is detected, negative reward r_c is applied. When the robot is in the normal course of exploration, where v and ω are local linear and angular velocity respectively and α is the time for each training loop. When the robot moves as fast as possible, i.e. the linear velocity is greater, and the absolute value of the angular velocity of the robot is smaller, the robot can obtain a higher instantaneous reward value. The episode reward is the accumulation of instantaneous rewards for all the steps in the episode.When the robot does not simply rotate and collide with an obstacle in a episode, by setting the maximum number of exploration steps in the episode, when the robot reaches the maximum number, it will end this episode and merge the initialization state.

3.3 RPPO algorithm model

In order to achieve in unknown environments through the visual image for obstacle avoidance and exploration, we designed a model based on PPO (Proximal Policy Optimization) algorithm [22]. PPO is a kind of strategy gradient algorithm based on the actor-critic framework , the algorithm puts forward a new kind of "alternative" objective function achieves the small batch update in multiple training steps and solves the problem of policy gradient in step difficult to determine. PPO algorithm is currently the best comprehensive base AC framework algorithm, which is OpenAI's default deep reinforcement learning algorithm. In order to further improve the exploration performance and training efficiency as well as the generalization performance applied to different unknown environments of PPO algorithm, this section introduces an advanced network structure of RPPO algorithm.

The goal of our algorithm design is to select the optimal action instruction a for the robot through the neural network prediction strategy when the robot perceives the state of the surrounding environment as s, and controls its navigation to achieve Active SLAM.The neural network structure designed to achieve the algorithm goal is shown in Figure 3. The input of the neural network is the real-time environmental RGB image collected by the kinect camera, which is simply processed into a multi-dimensional depth image pixel value array. The input array is first convolved through two conventional convolution layers with different convolution kernels and different step sizes. Then input to a deep separable convolutional layer for convolution [37]. The processing of the convolutional layer can help the network retain the main features of obstacles or other environmental information in the environmental image information and reduce parameters. In the process of obtaining the output characteristics, a nonlinear unit (ReLU) is used as the activation function. Depth separable convolution performs feature extraction by making simple changes to conventional convolution, and separates the convolution process of size adjustment and channel adjustment from input data to output data. Through depthwise convolution, the multi-channel input data is split into multiple single-channel

feature maps, and each channel data is convolved to realize the adjustment of the feature size. Pointwise convolution uses a 1×1 size convolution kernel to perform secondary convolution on the data obtained by depthwise convolution, and adjusts the number of channels by adjusting the number of filters. Compared with the conventional convolution, it can significantly reduce the amount of parameters and the computational cost.

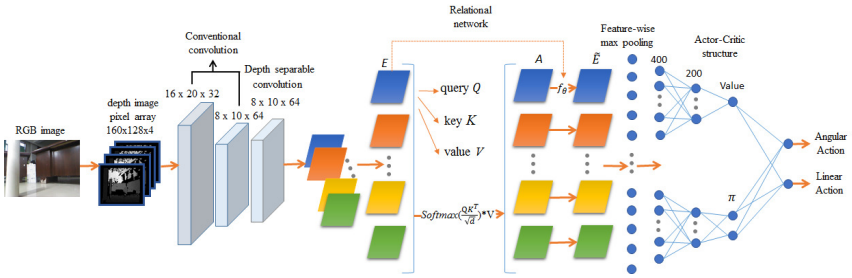

Fig. 3 The proposed RPPO algorithm network structure

Then connect the output of the convolutional layer with the relational network [23]. By combining reinforcement learning with relational learning, using a shared function to calculate the interaction relationship of the environmental state feature entities collected by the camera, and through iterative calculations to perceive high-order interactions between the inference entities, the performance of the neural network can be improved. Specifically, for the flattened input feature map information of the convolutional layer, the relational network uses the MHDPA (multi-head dot-product attention) to process the entity feature set, the process is shown in Figure 4.

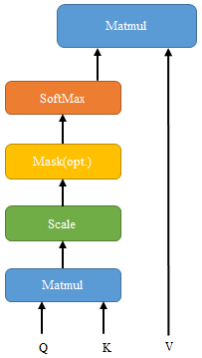

Fig. 4 The MHDPA process

By using a self-attention mechanism to iterate and reason the relationship between features in the environment, the efficiency, generalization ability and interpretability of reinforcement learning are improved. Among a

series of related methods for calculating non-local interactions [38], this paper chooses a computationally efficient attention mechanism [23]. This mechanism is similar to graph neural networks and more general message passing calculations [39, 40]. Using this mechanism, the interaction between entities can be expressed in a certain way of calculation.

Because the image collected by the robot is unstructured input, it is necessary to first show the characteristic entities of the collected image pixels in some way. The output image information of the convolutional neural network is processed into k feature maps with a size of nxn, where k is the number of output channels of the convolutional neural network. Then, the x and y coordinates are connected to each k-dimensional pixel feature vector to indicate the location of the pixel in the scene. Next, the obtained feature vector of $n \times n$ pixels is compiled into an $n \times n \times k$ matrix E as the entity set. In this way, an efficient and flexible learning method is provided to represent related entities. Input the entity set matrix E of the feature map into a feedforward neural network with k nodes to obtain the query vector matrix Q. The key vector matrix K is the matrix obtained by inputting Q into the neural network with k nodes, and the value vector matrix V is the matrix obtained by K in the same way, and normalizes them separately. The query vector matrix is multiplied by the transpose of the dot product and the key vector matrix of all entities, and adjusted by the scale factor to avoid excessive inner product, and then normalized to weight. For each entity, the cumulative interaction is calculated by the weighted mixed calculation of the value vector matrix of all entities, which can be specifically expressed by equation (3) [23]:

$$A = Softmax(\frac{QK^T}{\sqrt{\mathrm{d}}}) * V. \tag{3}$$

where d is the dimension of the key vector as the scale factor. Then pass the cumulative interaction matrix A to the two-layer MLP(Multilayer Perceptron) with RELU nonlinear activation with the same layers sizes as E, as shown in Figure 5, sum with E, and transform via layer normalization, to produce an output.

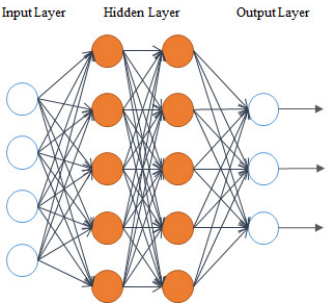

Fig. 5 The MLP structure

Because the PPO algorithm is used in this article to train the robot, the neural network uses the Actor-Critic structure, as shown in Figure 3, so the data processed by the relational network has two branches, the Actor network branch and the Critic network branch. The two branches have two identical hidden layers, which are fully connected layers with 400 neurons and 200 neurons, and both use the ReLU nonlinear function as the activation function. However, the Actor network and the Critic network are two independent networks. When the strategy is updated, the two network branches will not share parameters with each other. Then the features extracted by the hidden layer pass to the output layer, where the branch output of the Actor network is the two vectors under the environmental state s at time t, which are the mean and the variance. The control strategy is obtained by sampling the Gaussian distribution, and the output of the network is the angular velocity and linear velocity motion prediction strategy π in the continuous space can obtain the linear velocity control command and angular velocity control command of the mobile robot at this time. The range of linear velocity is limited to [0.1, 0.5] and the range of angular velocity is limited to $[-\frac{\pi}{3}, \frac{\pi}{3}]$. The output of the critic network branch is the state value function $V(s_t)$ of state s at time t.

After obtaining the action prediction strategy through the designed network, in order to obtain the optimal action, we use PPO algorithm to optimize the strategy here [22]. Use $r_t(\theta) = \frac{(\pi_\theta(a_t|s_t)))}{(\pi_{\theta_{old}}(a_t|s_t))}$ represents the ratio of robot strategy before and after the update, the way we choose the clip [22]. The main objective is the following [22]:

$$L^{PPO-CLIP}(\theta) = E_t[min(r_t(\theta)A_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)A_t)]. \tag{4}$$

where A_t is an estimator of the advantage function at timestep t and ϵ is a hyperparameter to constrain the lower and upper limits of policy updates. We set ϵ initial value is 0.2, and then use the dynamic parameters instead of the specified parameters, along with the increase in training round slowly decreasing ϵ value to get a better effect of constraints. Moreover, computing variance-reduced advantage-function estimators usually use the learned state-value function $V(s_t)$, in addition, this objective can further be augmented by adding an entropy bonus to ensure sufficient exploration. Combining these terms, we obtain the following objective [22]:

$$L_t^{CLIP+VF+S}(\theta) = E_t[L_t^{CLIP}(\theta) - c_1 L_t^{VF}(\theta) + c_2 S[\pi^\theta](s_t)]. \tag{5}$$

where c_1, c_2 are coefficients, and S denotes an entropy bonus, and $L^V F(\theta)$is a squared-error $(V(s_t) - V_t^{target})^2$.

When using sampled data to train a neural network in a deep reinforcement learning model, it is hoped that the training data is independent of each other. However, because of the continuity of actions during the autonomous navigation process of the robot, there is usually a certain correlation between

the samples obtained through sampling before and after. Therefore, we use the method of data batch processing. The sample data obtained by the interaction between the robot and the envieonment is stored, and the stored samples are further shuffled and repeated, and finally a batch of samples are randomly taken out to train and optimize the neural network parameters, as shown in Figure 6.

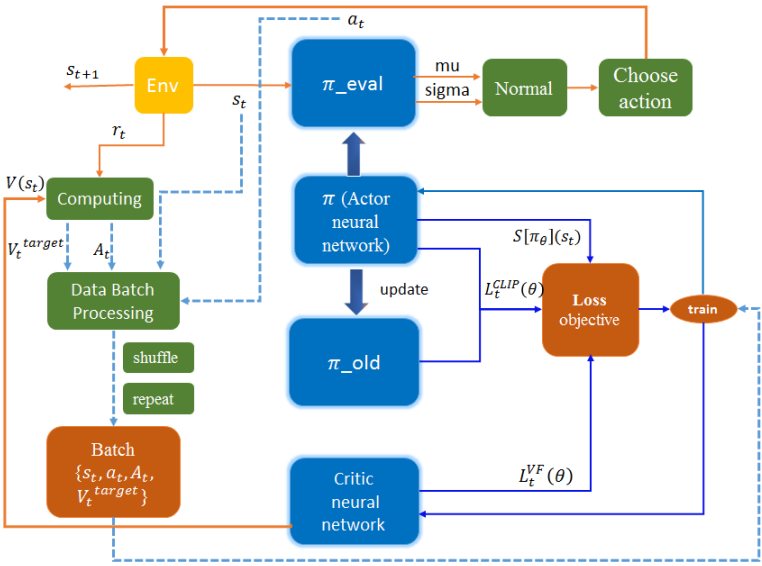

Fig. 6 Strategy optimize with Data Batch Processing

The main function of using batch data processing is to overcome the problems of correlation and non-stationary distribution of continuously collected sample data. On the one hand, a sample is more likely to be selected after repeated operations, and may be used multiple times, thus increasing the data usage rate. On the other hand, the sample will be shuffled, because the continuity of the action will cause the correlation when storing the sample to become much larger, which will increase the variance when the strategy is updated. When the stored sample data is shuffled, the variance can be greatly reduced. To reduce the correlation, in addition, the selection of batch data is also random, which will further reduce this correlation.In particular, when the environmental state s_t at time t is input to the design of neural network layer, by constantly updating π_{eval} network and critic network get action a_t and state-value function $V(s_t)$. After the robot performs an action a_t and gets a immediate reward r_t, then goes into the next state. At the same time V_t^{target} and advantage function A_t can be calculated. The collected data is not trained directly, but is shuffled and repeated through Data Batch Processing, from which a batch of data is extracted for network updates.

The process of using RPPO algorithm to conduct autonomous obstacle avoidance exploration training for mobile robots is shown in Algorithm 1.

Algorithm 1 The training process of RPPO model with Data Batch Processing

Require: Maximum number of training episode $MAX\_EPOSIDE$, maximum number of training steps per episode MAX_T, sample size of network updates K, learning rate lr

Initialize strategy π and state-value function $V(s)$
Set $t = 0$, $episode = 0$
while $episode < MAX\_EPOSIDE$ **do**
    Initialize the input environment to get the initial state s_0, set $reset =$ False
    while step $t < MAX\_T$ and not $reset$ **do**
        According to the observed state s_t and current strategy obtained π_{eval}, the neural network selects the actions a_t to be executed
        Apply the selected action to the current environment for immediate rewards r_t and new environment status s_{t+1}
        Calculate V_t^{target} and advantage function A_t
        Storage sample $\{s_t, a_t, r_t, V_t^{target}, A_t\}$
        if $t = K$ **then**
            Update the old strategy π_{old}
            Data batch processing for the stored samples, and acquire batch$\{s_t, a_t, r_t, V_t^{target}, A_t\}$
            Calculate the network loss $L_t^{CLIP+VF+S}(\theta)$
            Using ADAM optimizer to optimize neural network parameters
            $L_t^{CLIP+VF+S}(\theta)$ //Update network parameters with learning rate lr
            Empty the stored samples
            Set $t = 0$
        end if
        t+=1
    end while
    $eposide$+=1
end while

3.4 Transfer learning

In simple terms, transfer learning is to find the connection between the learned knowledge and the new learning task, and then use what has been learned from the completed learning task to help learn new knowledge [41]. The key to transfer learning is how to reasonably find the correlation between the learned experience and the new learning task, and to improve the efficiency of the new learning task by looking for the correlation. Specifically, in transfer

learning, the learned experience and knowledge are the source domain, and the new learning task to be performed is the target domain. As shown in Figure 7, what transfer learning needs to do is to find the relationship between the knowledge and experience system of the source domain and the new task of the target domain, and then transfer the knowledge reserve of the source domain to the target domain to realize the establishment of the target domain learning system.

Under ideal circumstances, in a successful transfer learning application, higher initial performance, faster acceleration rate, or better model convergence performance will be obtained. This process can also be applied to deep learning, as shown in Figure 8, because the neural network needs to train the target task through the sample data obtained, and the supervision process from input to specific output needs to train the corresponding network parameters and record each parameter as corresponding weight [42]. In similar learning tasks with the same network model, the weights corresponding to the parameters of the network can be extracted and transferred to the deep learning model of the new learning task to "transfer" the learning experience, and then fine-tuned to achieve the goal the learning task, thus avoiding the process of retraining the parameters of the neural network. By using the trained neural network parameters, perhaps only a small set of weights corresponding to the parameters can greatly improve the training efficiency.

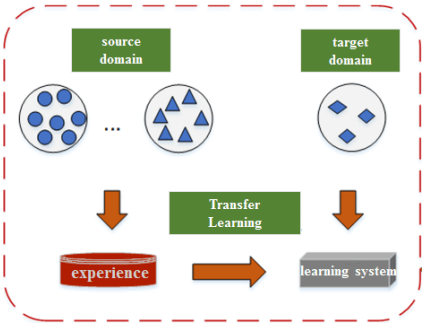

Fig. 7 Transfer learning process

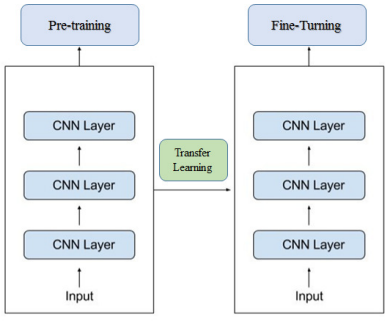

Fig. 8 Neural network transfer process

Considering that the tasks of active SLAM in different unknown environments are relevant, no matter it is a simple or complex environment, the active SLAM algorithm framework has shared model parameters. Therefore, transfer learning can be used to initialize the new model by extracting network weights to speed up the learning progress, instead of collecting and training a large number of sample data from the beginning to perform the learning task when training the network for the first time. As shown in Figure 9, the learning task of the robot in a simple training environment with only static obstacles is the source domain, and the learning task of the robot in an unknown and complex new environment with various dynamic and static obstacles is the

target domain. In the source domain, the robot learns the action prediction strategy based on the designed algorithm model, and obtains the optimized action prediction strategy neural network parameters θ after training.In order to train a neural network that can adapt to more complex environments to generate action prediction strategies, it is no longer necessary to retrain those neural networks used to distinguish static obstacles and free space. Instead, by introducing transfer learning and extracting the weights of network nodes, the network gains the ability to identify simple static obstacles and explore, and then continue to fine-tune training, so that it can identify and analyze different color types of dynamic and static complex obstacles and predict actions to adapt to complex environments.Because the environment setting in the target domain is more complicated, the robot in the target domain has stronger learning and reasoning ability after training to adapt to various complex unknown environments.

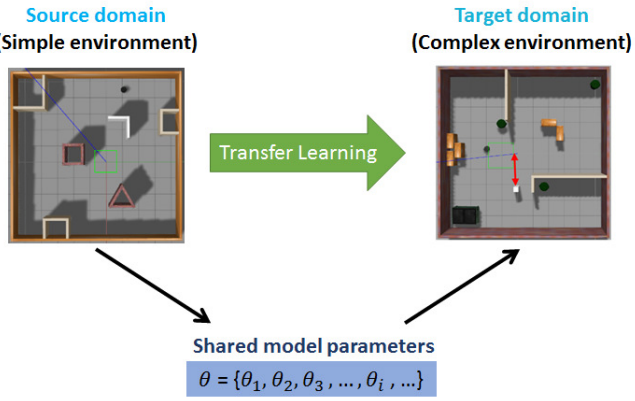

Fig. 9 The transfer process of SLAM algorithm

4 Experimental Results

In this section, we will evaluate the proposed Active SLAM algorithm framework in different environments. We use Gazebo Simulator [33] for training, and build different environments with different types of static and moving obstacles. First, the robot is used to train the RPPO model in the relatively simple simulation environment as shown in Figure 14(a). Then, based on the training model parameters of the simple environment, Transfer Learning is used to carry out further training in the complex environment with various dynamic and static obstacles as shown in Figure 15(a). Next the Active SLAM framework is used to test in simulated environments and unknown environments in the real world. Here, Turtlebo2 robot is used for training and real-time testing of the framework. Meanwhile,Kinect camera is used to enable the robot to obtain three-dimensional information of the environment, and lidar is used

to complete simultaneous locatation and mapping. In addition, we use the gmapping algorithm to map the environment [43].

4.1 Advanced performance of the RPPO model

First, we analyze the advantages of deep separated convolution. As shown in Table.1, it is a comparison of the parameter amount, the amount of calculation and the occupation of system resources when using the deep separable convolution layer and the conventional convolution layer. It can be seen that due to the reduction in the amount of parameters and calculations, the calculation cost is reduced, so the proportion of system resources occupied by the depth separable convolution training is relatively low. Figure 10 is the time of predicting the action from the input environment information through the neural network using the depth separable convolution and the conventional convolution layer respectively. As can be seen from the figure, compared with the conventional convolutional network, the prediction time of the depth separable convolution is significantly reduced, thus reducing the robot's requirements for the upper-level computer control equipment. In actual experiments, the robot's response speed can also be guaranteed in the face of equipment with poor computing capabilities.

Table 1 The comparison of the computing costs of two different neural networks

	parameter amount	calculated amount	resources occupation rate
Conv	36864	1769472	87
Se Conv	4672	355328	**76%**

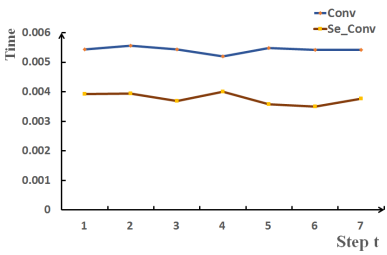

Fig. 10 The action prediction time of two different neural networks

In addition, deep separable convolutional and conventional convolutional neural networks are used to train the improved RPPO algorithm model, as shown in Figure 11, then the accumulated rewards of 18 collision-free episodes are randomly selected, and the average is calculated. It can be seen that the performance of the two networks is almost the same, and even the effect

of the deep separable convolutional layer network is slightly better than the conventional convolutional.

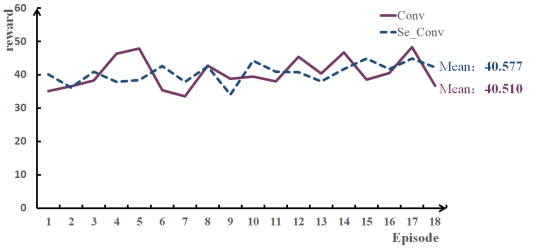

Fig. 11 The accumulated rewards of two different neural networks

Further, in order to analyze the training efficiency and model performance of the designed RPPO, as well as the advantages of adding the relational network, compared with the PPO model [22] and the D3QN model [33] that performs well in the field of robot obstacle avoidance, three algorithms use the same input and reward function. Through training in the same simulation environment designed, and recording the cumulative rewards of each episode in the training process, as shown in Figure 12 . It can be seen that no matter which deep reinforcement learning algorithm is used, as the robot continues to interact with the environment, it can train neural networks and optimize action strategies, and the cumulative reward curve of the episode has a gradually rising trend.

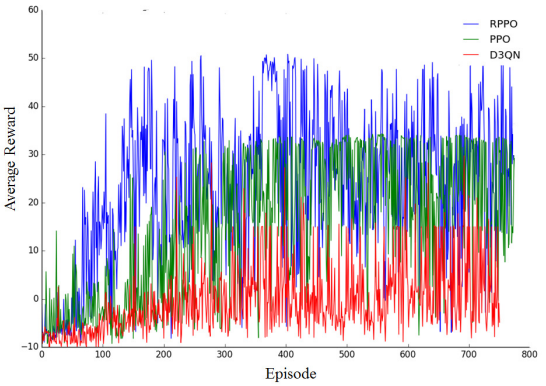

Fig. 12 The episode cumulative rewards curve of the three algorithm models

However, compared to algorithms based on PPO, the D3QN algorithm has a poor ability to predict action strategies in a continuous action space, which will lead to rough navigation behavior. Although it can enable the robot to

explore, it is difficult to learn stable and effective obstacle avoidance strategy. The two algorithm models based on PPO have learned effective collision-free exploration strategies, but compared with the PPO algorithm with an average stability threshold reward of only 30, the model trained based on the improved RPPO algorithm can achieve a higher learning reward strategy through learning the interaction between features, and its training speed is much faster than the PPO algorithm model, which means that the improved RPPO algorithm can make the robot learn obstacle avoidance exploration strategies faster, and the trained robot has a faster speed in environmental exploration.

Furthermore, by randomly selecting the cumulative reward value of some non-collision rounds of the robot under the three algorithms, it can be seen from Table 2 that the optimized RPPO algorithm can accumulate a higher reward value in a non-collision round. In addition, as shown in Figure 13 , the robot uses the optimized RPPO algorithm to have a higher Average instantaneous reward. In other words, the robot has a larger running speed during the exploration process, which can realize faster exploration of the environment.

Table 2 Cumulative rewards for non-collision eposides

	1	2	3	4	5	average reward
D3QN [33]	17.6806	19.7211	15.1624	21.0141	14.9411	17.7039
PPO [22]	33.1864	25.9548	33.0303	32.6468	30.8146	31.1266
RPPO **(ours)**	39.0678	46.5920	49.6317	46.1649	48.0439	**45.9001**

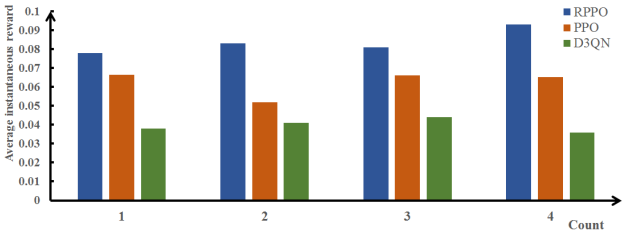

Fig. 13 Average instantaneous rewards of the three algorithm models

4.2 Simulation Environment Tests

First, we test in the trained simulation environment whether the robot can predict a reasonable action through the trained RPPO model to avoid obstacles and explore, and complete the construction of the environment map, and then test the robot in a completely unknown simulation environment.

As shown in Figure 14, Figure 15 and Figure 16, the robot can successfully complete the task of active SLAM based on the goal of exploration and obstacle avoidance in the simple and complex environment that has been trained, as well as the completely unknown environment.

Specifically, Figure 14(b), Figure 15(b) and Figure 16(b) record the movement trajectory of the robot when it carries out Active SLAM tasks in different environments with different types of multiple obstacles. In addition to various types of wall obstacles, other static obstacles include five cabinets (yellow rectangles), three garbage bins (blue circles) and one box (black rectangles), and dynamic obstacles are square columns that move periodically (gray squares).From the results we can see that the robot can successfully avoid obstacles and explore the environment. As shown in Figure 14(c), Figure 15(c) and Figure 16(c), The robot can explore the environment from the environment of different shapes with different dynamic and static obstacles, and construct an environment map. As shown in Figure 14(d), Figure 15(d) and Figure 16(d), in the process of robot movement, according to the collected RGB image information of the environment in different states, different linear velocity and angular velocity actions are predicted according to the RPPO model. From which we can see that when observed obstacles, the robot will produce different angular velocity to turn left or right of action to avoid obstacles, in addition, since the reward function defined in the training phase based on the purpose of the exploration setting robot run faster can gain greater rewards, so the robot in the process of completing the task is to keep the largest linear speed.

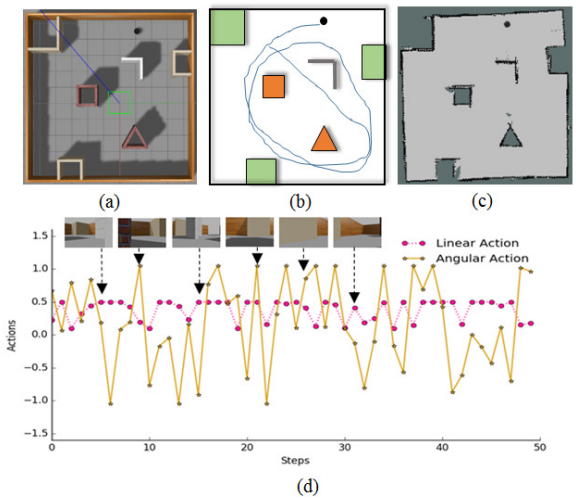

Fig. 14 Simple simulation environment and test results

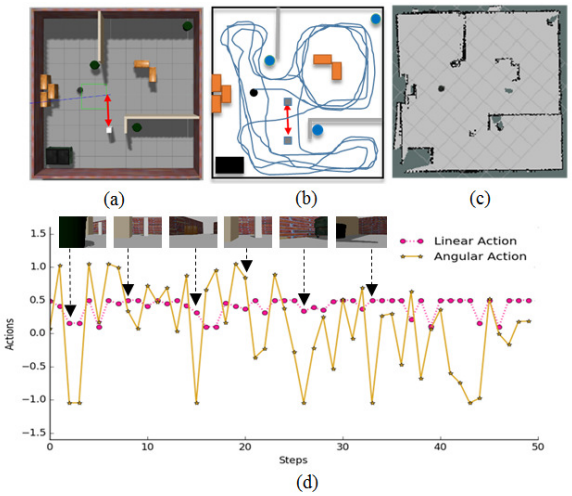

Fig. 15 Complex simulation environment and test results

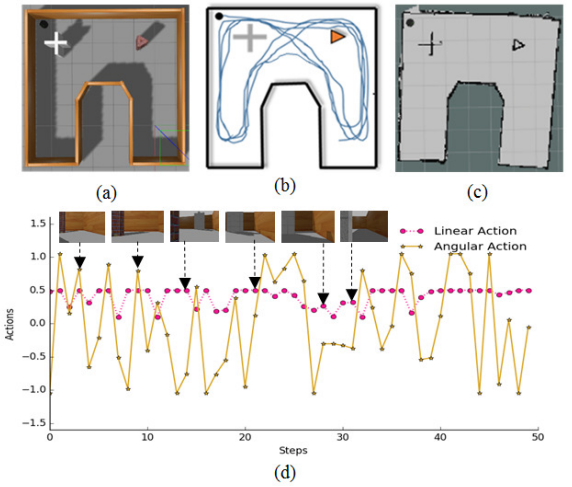

Fig. 16 Unknown simulation environment and test results

4.3 Real World Tests

Several environments are built to directly test the Active SLAM algorithm framework in real world. Figure 17 records the test results of the robot in a simple environment with only one kind of white cylinder obstacle. Then, we change the shape of the environment and add a green cylinder obstacle, a light blue trash can and a box to construct a relatively complex unknown environment. The environment and test results are shown in Figure 18. Finally, we change the position of the obstacle in the environment shown in Figure 18, and add a pedestrian whose position would change to construct a complex

unknown environment with dynamic and static obstacles. The environment and test results are shown in Figure 19.

Specially, Figure 17(a), Figure 18(a) and Figure 19(a) are the environment we constructed. Figure 17(b), Figure 18(b) and Figure 19(b) show that the robot applies the proposed active SLAM algorithm framework without prior information about the environment, and successfully realizes obstacle avoidance and exploration of the environment, and finally completed Construction of environmental maps. The black circle is the robot model, the red dashed line is the trajectory of the robot in continuous time, and the blue ellipse is the information recorded by the robot when the pedestrian is at different positions. Figure 17(c), Figure 18(c) and Figure 19(c) are the real-time RGB images of the environment collected by the camera during the continuous movement of the robot, and the action prediction strategy is generated through the network trained by the RPPO algorithm model, resulting in different linear velocity actions and angular velocities actions.

From the results, we can see that the robot can smoothly avoid obstacles based on the collected three-dimensional information of the environment, and try to maintain the maximum linear velocity during the movement to achieve faster exploration of the environment. In short, we further verify the effectiveness and robustness of the algorithm by setting up a test environment in the real world.

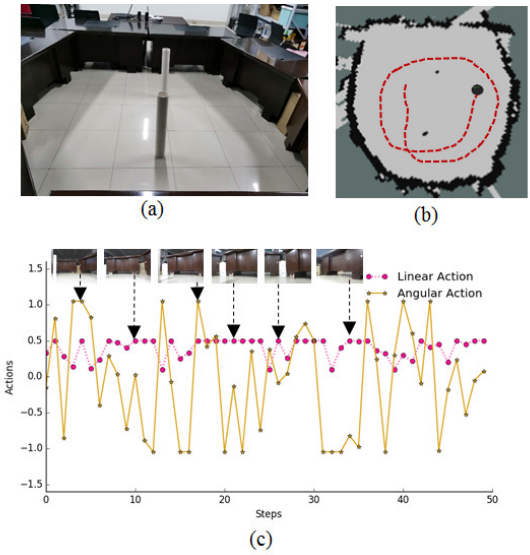

Fig. 17 Unknown simple real environment and test resultss

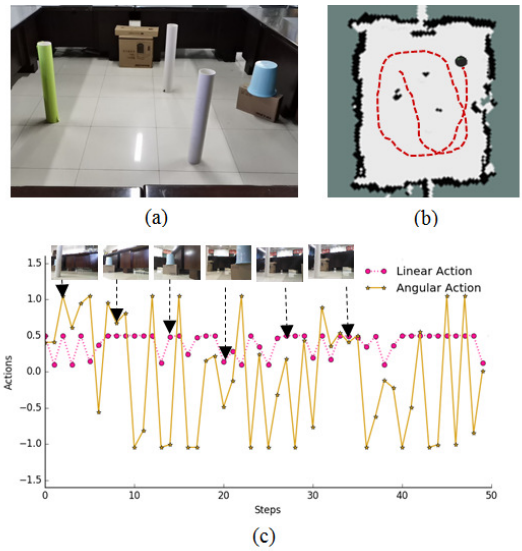

Fig. 18 Unknown complex real environment and test results

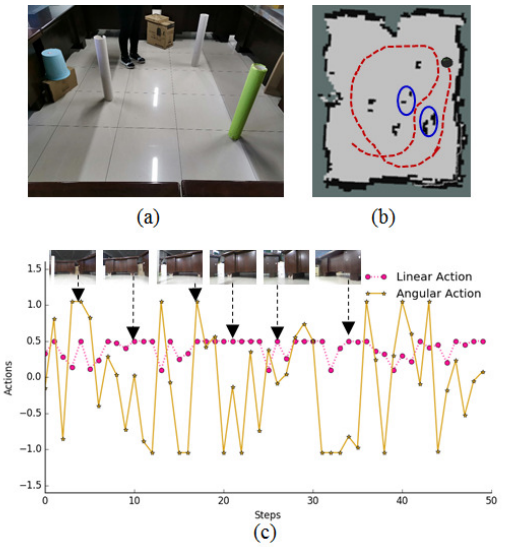

Fig. 19 Unknown real environment with dynamic obstacle and test results

5 Conclusion

In this paper, an active SLAM algorithm framework based on RPPO model is proposed to realize the robot to explore the unknown environment and map. The RPPO algorithm model based on the relational network shows

higher learning efficiency than PPO and other reinforcement learning algorithms such as D3QN to implement the robot from the environment state input to the motion plan to avoid obstacles and explore, in addition, data batch processing is used to solve possible data relevance problems, and deep separable convolution is used to reduce computing costs and improve environmental adaptability. Then, based on transfer learning, use the network model parameters trained in a simple environment to extract primary features, and then fine-tune the training in a complex simulation environment to quickly adapt to the complex environment of the target domain. After the training is completed, the network structure will have stronger learning and reasoning ability to realize active SLAM in various complex unknown environments. In addition, a large number of simulations and experiments show the feasibility and effectiveness of the proposed active SLAM algorithm framework from virtual to real.

The algorithm designed in this paper focuses on the end-to-end action output process of the robot according to the observed environmental state, and focuses on the learning and optimization of the robot action prediction strategy. It is inclined to allow robots to explore faster, and does not consider the trade-off between exploration and revisit in active SLAM. This issue will be considered in future work. In addition, we have designed a reward function based on faster exploration and obstacle avoidance. There may be some situations that influence the SLAM process. In our future research, we will add SLAM factor to the design of the reward function and optimize the reward function to realize the optimization of active SLAM process.

Compliance with Ethical Standards

- Ethical approval: This research did not involve animal or human participants, followed accepted principles of ethical and professional conduct.
- Funding details: The work was partly supported by the National Natural Science Foundation of China (NSFC, Project No.61773333), the National Natural Science Foundation of China and the Royal Society of Britain(NSFC-RS, Project No.62111530148) and the China Scholarship Council (CSC, Project No.201908130016).
- Conflict of interest: This work did not involve conflict of interest and complied with ethical standards and professional norms.
- Informed Consent: This work did not involve human participants, so informed consent was not required.

Declarations

- Competing interests: The authors have no relevant financial or non-financial interests to disclose.
- Authors' contributions: All authors contributed to the study conception and design. Formula derivation, experiment design and analysis were performed by Shuhuan Wen, Zhixin Ji and Ahmad B.Rad. The first draft of

the manuscript was written by Zhengzheng Guo and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

- Data availability: The datasets are too big and analysed during the current study are not publicly available but are available from the corresponding author on reasonable request.

References

[1] Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., Leonard, J.J.: Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. IEEE Transactions on Robotics **32**(6), 1309–1332 (2016). https://doi.org/10.1109/TRO.2016.2624754

[2] Chen, Y., Huang, S., Fitch, R.: Active slam for mobile robots with area coverage and obstacle avoidance. IEEE/ASME Transactions on Mechatronics **25**(3), 1182–1192 (2020). https://doi.org/10.1109/TMECH.2019.2963439

[3] Qin, H., Meng, Z., Meng, W., Chen, X., Sun, H., Lin, F., Ang, M.H.: Autonomous exploration and mapping system using heterogeneous uavs and ugvs in gps-denied environments. IEEE Transactions on Vehicular Technology **68**(2), 1339–1350 (2019). https://doi.org/10.1109/TVT.2018.2890416

[4] Wen, S., Zheng, W., Zhu, J., Li, X., Chen, S.: Elman fuzzy adaptive control for obstacle avoidance of mobile robots using hybrid force/position incorporation. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) **42**(4), 603–608 (2012). https://doi.org/10.1109/TSMCC.2011.2157682

[5] Ekvall, S., Jensfelt, P., Kragic, D.: Integrating Active Mobile Robot Object Recognition and SLAM in Natural Environments. Paper presented at 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, 9–15 October 2006 (2006)

[6] Gao, W., Tang, Q., Ye, B., Yang, Y., Yao, J.: An enhanced heuristic ant colony optimization for mobile robot path planning. Soft Computing **24**(2), 6139–6150 (2020)

[7] Taketomi, T., Uchiyama, H., Ikeda, S.: Visual slam algorithms: a survey from 2010 to 2016. Ipsj Transactions on Computer Vision & Applications **9**(1), 16 (2017)

[8] Ruiz-Ascencio, Jose, Fuentes-Pacheco, Jorge, Manuel, Rendon-Mancha, Juan: Visual simultaneous localization and mapping: a survey. Artificial

Intelligence Review: An International Science and Engineering Journal **43**(1), 55–81 (2015)

[9] Cuzzocrea, Alfredo, Mumolo, Enzo, Grasso, Giorgio, Mario: Advanced pattern recognition from complex environments: a classification-based approach. Soft computing: A fusion of foundations, methodologies and applications **22**, 4763–4778 (2018)

[10] Wu, X., Chen, H., Wu, X., Wu, S., Huang, J.: Simultaneous localization and mapping of medical burn areas based on binocular vision and capsule networks. Soft Computing **24**(14), 18155–18171 (2020)

[11] Zhang, Y., Chen, H., He, Y., Ye, M., Cai, X., Zhang, D.: Road segmentation for all-day outdoor robot navigation. Neurocomputing **314**(NOV.7), 316–325 (2018)

[12] Niijima, S., Sasaki, Y., Mizoguchi, H.: Real-time autonomous navigation of an electric wheelchair in large-scale urban area with 3d map*. Advanced Robotics **33**(1), 1–13 (2019)

[13] Yang, H., Wang, Q., Li, H., Fang, F., Kadry, S.N.: Maritime moving object localization and detection using global navigation smart radar system. Soft Computing **25**(11), 11965–11974 (2021)

[14] Pham, H., Smolka, S.A., Stoller, S.D., Phan, D., Yang, J.: A survey on unmanned aerial vehicle collision avoidance systems. Preprint at https://arxiv.org/abs/1508.07723 (2015)

[15] Hoy, M., Matveev, A.S., Savkin, A.V.: Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey. Robotica **33**(3), 463–497 (2015)

[16] Blaer, P.S., Allen, P.K.: Data acquisition and view planning for 3-D modeling tasks. Paper presented at 2008 IEEE/RSJ International Conference on Intelligent Robots & Systems(IROS), Nice, 22–26 September 2008 (2008)

[17] Okada, Y., Miura, J.: Exploration and observation planning for 3D indoor mapping. Paper presented at 2015 IEEE/SICE International Symposium on System Integration, Meijo University, Nagoya, 12–13 December 2015 (2015)

[18] Chen, Y., Huang, S., Fitch, R.: Active slam for mobile robots with area coverage and obstacle avoidance. IEEE/ASME Transactions on Mechatronics **25**(3), 1182–1192 (2020)

[19] Kim, A., Eustice, R.M.: Perception-driven navigation: Active visual

SLAM for robotic area coverage. Paper presented at 2013 IEEE International Conference on Robotics and Automation, Germany, 6–10 May 2013 (2013)

[20] Meng, Z., Hao, S., Qin, H., Chen, Z., Ang, M.H.: Intelligent robotic system for autonomous exploration and active SLAM in unknown environments. Paper presented at 2017 IEEE/SICE International Symposium on System Integration (SII), Taipei, 11–14 December 2017 (2017)

[21] Lei, T., Li, S., Ming, L.: A deep-network solution towards model-less obstacle avoidance. Paper presented at 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems(IROS), Daejeon, 9–14 October 2016 (2016)

[22] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal Policy Optimization Algorithms. Preprint at https://arxiv.org/abs/1707.06347v1 (2017)

[23] Zambaldi, V., Raposo, D., Santoro, A., Bapst, V., Li, Y., Babuschkin, I., Tuyls, K., Reichert, D., Lillicrap, T., Lockhart, E.: Relational Deep Reinforcement Learning. Preprint at https://arxiv.org/abs/1806.01830v1 (2018)

[24] Li, Q., Tan, J., Wang, J., Chen, H.: A multimodal event-driven lstm model for stock prediction using online news. IEEE Transactions on Knowledge and Data Engineering **33**(10), 3323–3337 (2021). https://doi.org/10.1109/TKDE.2020.2968894

[25] Chung, S.T., Morris, R.L.: Deep learning for stock prediction using numerical and textual information. Paper presented at 15th International Conference on Computer and Information Science (ICIS), Okayama, 26–29 June 1978 (2016)

[26] Babbar, S.: Review - mastering the game of go with deep neural networks and tree search. Nature **529**(7587), 484–489 (2016)

[27] Wen, Shuhuan, Chen, Xiao, Hua, Shaoyang, Lam, H., K., Chunli: The q-learning obstacle avoidance algorithm based on ekf-slam for nao autonomous walking under unknown environments. Robotics and Autonomous Systems **72**, 29–36 (2015)

[28] Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D.: Continuous control with deep reinforcement learning. Preprint at https://arxiv.org/abs/1509.02971 (2015)

[29] Sünderhauf, N., Brock, O., Scheirer, W., Hadsell, R., Fox, D., Leitner, J.,

Upcroft, B., Abbeel, P., Burgard, W., Milford, M.: The limits and potentials of deep learning for robotics. The International journal of robotics research **37**(4-5), 405–420 (2018)

[30] Shih, B.Y., Chen, C.Y., Chou, W.C.: Obstacle avoidance using a path correction method for autonomous control of a biped intelligent robot. Journal of Vibration & Control **17**(10), 1567–1573 (2011)

[31] Michels, J., Saxena, A., Ng, A.Y.: High speed obstacle avoidance using monocular vision and reinforcement learning. Paper presented at the Twenty-Second International Conference on Machine Learning (ICML), Bonn, 7-11 August 2005 (2005)

[32] Mirowski, P., Pascanu, R., Viola, F., Soyer, H., Hadsell, R.: Learning to Navigate in Complex Environments. Preprint at https://arxiv.org/abs/1611.03673 (2016)

[33] Xie, L., Wang, S., Markham, A., Trigoni, N.: Towards Monocular Vision based Obstacle Avoidance through Deep Reinforcement Learning. Preprint at https://arxiv.org/abs/1706.09829 (2017)

[34] James, S., Johns, E.: 3D Simulation for Robot Arm Control with Deep Q-Learning

[35] Rusu, A.A., Večerík, M., Rothörl, T., Heess, N., Pascanu, R., Hadsell, R.: Sim-to-Real Robot Learning from Pixels with Progressive Nets. Paper presented at 1st Annual Conference on Robot Learning, California, 13-15 Nov (2017)

[36] Chollet, F.: Virtual-to-real Deep Reinforcement Learning: Continuous Control of Mobile Robots for Mapless Navigation. Paper presented at 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, 24-28 Sept (2017)

[37] Chollet, F.: Xception: Deep Learning with Depthwise Separable Convolutions. Paper presented at 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, 21–26 July 2017 (2017)

[38] Wang, X., Girshick, R., Gupta, A., He, K.: Non-local Neural Networks. Preprint at https://arxiv.org/abs/1711.07971 (2017)

[39] Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. IEEE Transactions on Neural Networks **20**(1), 61–80 (2009). https://doi.org/10.1109/TNN.2008.2005605

[40] Niepert, M., Ahmed, M., Kutzkov, K.: Learning convolutional neural networks for graphs. In: Proceedings of the 33rd International Conference

on International Conference on Machine Learning. ICML, vol. 48, pp. 2014–2023. JMLR.org, New York (2016)

[41] Perlich, C., Dalessandro, B., Stitelman, O., Raeder, T., Provost, F.: Machine learning for targeted display advertising: Transfer learning in action. Social Science Electronic Publishing **95**(1), 103–127 (2014)

[42] He, W., Chen, Y., Yin, Z.: Adaptive neural network control of an uncertain robot with full-state constraints. IEEE Transactions on Cybernetics **46**(3), 620–629 (2017)

[43] Grisetti, G., Stachniss, C., Burgard, W.: Improved techniques for grid mapping with rao-blackwellized particle filters. IEEE Transactions on Robotics **23**(1), 34–46 (2007)