Journal of Ambient Intelligence and Humanized Computing (2024) 15:2925–2939
https://doi.org/10.1007/s12652-024-04787-x

ORIGINAL RESEARCH

A path planning method based on deep reinforcement learning for crowd evacuation

Xiangdong Meng[1,2] · Hong Liu[1,2] · Wenhao Li[1,2]

Received: 6 August 2021 / Accepted: 13 March 2024 / Published online: 18 April 2024

Abstract
Deep reinforcement learning (DRL) is suitable for solving complex path-planning problems due to its excellent ability to make continuous decisions in a complex environment. However, the increase in the population size in the crowd evacuation path-planning problem causes a substantial computational burden for the algorithm, which leads to an unsatisfactory efficiency of the current DRL algorithm. This paper presents a path planning method based on DRL for crowd evacuation to solve the problem. First, we divide crowds into groups based on their relationship and distance from each other and select leaders from them. Next, we expand the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) to propose an Optimized Multi-Agent Deep Deterministic Policy Gradient (OMADDPG) algorithm to obtain the global evacuation path. The OMADDPG algorithm uses the Cross-Entropy Method (CEM) to optimize policy and improve the neural network's training efficiency by applying the Data Pruning (DP) algorithm. In addition, the social force model is improved, incorporating the relationship between individuals and psychological factors into the model. Finally, this paper combines the improved social force model and the OMADDPG algorithm. The OMADDPG algorithm transmits the path information to the leaders. Pedestrians in the environment are driven by the improved social force model to follow the leaders to complete the evacuation simulation. The method can use a leader to guide pedestrians safely arrive the exit and reduce evacuation time in different environments. The simulation results prove the efficiency of the path planning method.

Keywords Path planning · Deep reinforcement learning · Crowd evacuation · Optimized multi-agent deep deterministic policy gradient

1 Introduction

With the improvement of people's living standards, people gather in large public places such as supermarkets and parks for recreational activities in their free time. Especially during peak periods of pedestrian traffic, such as holidays, these public places are often crowded. When an emergency occurs, pedestrians easily cause secondary injuries (Tang et al. 2019). In severe cases, pedestrians will cause heavy casualties (Zhou et al. 2019). Reducing the risk of crowd evacuation in an emergency has become an urgent problem. Therefore, crowd evacuation simulation has attracted widespread attention.

✉ Hong Liu
lhsdcn@126.com

✉ Wenhao Li
liwh85@126.com

1 School of Information Science and Engineering, Shandong Normal University, Jinan 250014, China

2 Shandong Provincial Key Laboratory for Novel Distributed Computer Software Technology, Jinan 250014, China

Researchers have proposed many simulation models to solve the problem, such as the social force model (Helbing and Molnár, 1995), cellular automata model (Gao et al. 2021), potential field model (Chen et al. 2021), and agent-based model. These models attracted widespread attention when they were proposed, and many researchers improved them (Zhang et al. 2018; Ji and Gao 2007; Jiang et al. 2020; Liu et al. 2018). As a pedestrian dynamics simulation model, the social force model considers various factors in pedestrian movement, which can reproduce the self-organization phenomenon in crowd movement. The pedestrian in this model is regarded as a particle that moves under the drive of mental factors and physical factors, which is in line with the real crowd movement.

In addition, researchers proposed different solutions to various problems in crowd evacuation simulation. Küllü et al. (2017) proposed a communication model for crowd simulation that can be useful for emergency crowd evacuation. Oğuz et al. (2010) proposed a crowd simulation model to simulate emergency crowd simulation. However, these models must correctly guide the crowd to choose the exit to avoid pedestrians and obstacles in the forward direction due to the lack of guidance in global path planning when dealing with large-scale crowd evacuation. Moreover, crowd movement is a continuous process in reality, and pedestrians will adjust the target exit and movement direction according to the actual situation (Yao et al. 2020). The path planning algorithm (Zhao et al. 2021) can get the optimal global evacuation path and guide pedestrians to the exit. In order to solve the global path planning problem, the path planning algorithm is introduced into the crowd evacuation simulation.

As a DRL algorithm, the MADDPG algorithm (Lowe et al. 2017) can utilize the deep neural network to solve decision-making problems with continuous action space and learn effective policies in an observation space based on joint observation. Compared with other path planning algorithms, the algorithm has a more robust performance in solving the multi-objective path planning problem with continuous action space. However, the increase in the population size in the crowd evacuation path planning problem causes a substantial computational burden for the algorithm, which leads to an unsatisfactory efficiency of the current DRL algorithm. In addition, the deterministic policy of the algorithm also is an important factor, which can influence the evacuation results.

To figure out these problems, we present a path-planning method based on deep reinforcement learning for crowd evacuation. First, this paper proposes an OMADDPG algorithm. The algorithm will be affected by noise during the training process, which leads to a decrease in the ability of the algorithm to learn effective policies. Therefore, the OMADDPG algorithm uses CEM to optimize the deterministic policy of the algorithm to improve the ability to learn effective policies. Since the MADDPG algorithm uses the first-in-first-out order to remove experiences, it accumulates redundant data and affects the algorithm's convergence. Therefore, the algorithm reduces the data relevance of the MADDPG algorithm to improve the training efficiency of the neural network by using the DP algorithm. Next, an improved social force model is proposed to address the problem of insufficient description of the influence between pedestrians in motion in the social force model. Pedestrians inevitably influence each other in motion, and this influence is manifested in the movement as an effect on the desired speed of pedestrians. Since the social force model lacks consideration of these influences, this paper improves the social force model by incorporating the influence of surrounding pedestrians and leaders into modeling the social force model. Therefore, pedestrians can adjust their desired speed and direction according to surrounding individuals to avoid obstacles and follow the leader to the exit. Finally, to realize the path planning method, we combine the OMADDPG algorithm and the improved social force model to complete the simulation. The main contributions of this paper are as follows:

1. We use the CEM to eliminate the influence of bad samples on the MADDPG algorithm to optimize the deterministic policy, which can improve the efficiency of learning effective policies. The data correlation of the MADDPG algorithm is reduced by the DP algorithm, which improves the neural network's training efficiency and enhances the samples' representativeness.
2. The social force model is improved. The cohesive force and the excitement are added to the model. In this model, pedestrians can change the desired speed according to the excitement of surrounding pedestrians and attract others who have close relationships. Pedestrians can dynamically adjust the desired speed and direction so that they can form the pedestrian stream to avoid obstacles to accelerate the evacuation.
3. A path planning method is proposed for crowd evacuation simulation. This method uses the OMADDPG algorithm to transmit the global path to the agent to complete the evacuation, which can provide better decision support for crowd evacuation. A multi-agent particle interactive environment is built to realize the path planning method. The improved social force model is introduced to the environment to avoid obstacles and model crowd behavior.

The general organization of this paper is as follows. Section 2 briefly reviews the related research about the path planning method and DRL. Section 3 describes the path planning method in detail. Section 4 presents the experimental results, while Sect. 5 concludes.

2 Related work

2.1 Path planning

In large-scale evacuation simulation, path planning algorithm is an indispensable part. In reality, the crowd will dynamically adjust the movement direction to avoid congestion, and will follow other individuals that have target (Ji and Gao 2007). The ordinary evacuation simulation models cannot effectively plan the path for the crowd when carrying out large-scale crowd evacuation simulation, resulting in the further decline of evacuation efficiency, which is not in line with the real situation. Path planning algorithms can obtain global evacuation paths to lead pedestrians arrive the exit, which can reduce the evacuation time and improve the efficiency of crowd evacuation. Therefore, the path planning algorithm was introduced to the evacuation simulation. Currently, path planning algorithms are mainly divided into swarm intelligence algorithms and RL algorithms. Other non-learning-based path planning methods, such as probabilistic roadmap algorithm (PRM) (Xu et al. 2021), are only suitable for solving path planning problems in static environment. When solving path planning problems in dynamic environment, this algorithm cannot get the optimal path because of the randomness of the sampling process. Swarm intelligence algorithms, such as Artificial Bee Colony algorithm (ABC) (Sun et al. 2021) and Particle Swarm Optimization algorithm (PSO) (Gul et al. 2021), often end prematurely and cannot obtain the global path because the solving strategy is relatively simple.

RL algorithm contains four elements: state, action, reward and environment. RL algorithms judge whether an action is good or bad according to the environmental feedback, and obtain the optimal solution during the learning process. The RL have been widely used (Madani et al. 2020) because of the advantages of RL are online learning and the adaptability to the environment. Yao et al. (2020) proposed a crowd simulation method, which combined deep learning's perceptual ability and RL. The method can learn pedestrian's behavior from real data and apply it to various scenes. Jardine and Givigi (2021) improved the cooperative reinforcement learning by sharing their experience, which can results in faster learning without any loss of performance in complex environments. Yao et al. (2019) proposed a reinforcement learning based data-driven crowd evacuation model. It can simulate crowd movement more in line with real data. But RL algorithms still have limitation when dealing with the problem of the continuous action space.

In recent years, researchers have applied DRL algorithms to solve path planning problems. DRL's perceptual ability can deal with the high-dimensional continuous state-action space problem better in the dynamic environment. Yan et al. (2020) used the Dueling Double Deep Q Network (D3QN) algorithm to guide the unmanned aerial vehicles to reach the target in complex environments with unknown potential threats. Jiao and Oh (2019) proposed an algorithm, which can learn a single continuous control policy from raw sensor data. Hu et al. (2020) combined Deep Deterministic Policy Gradient (DDPG) with prior data and used demonstration data to make the algorithm better adapts to the environment. Wan et al. (2020) improved the learning technologies of the DDPG algorithm that can carry out path planning in a dynamic uncertain environment.

The path planning problems for crowd evacuation usually have continuous action space and complex environment, so swarm intelligence algorithms and RL algorithms cannot effectively deal with them. DRL algorithms provide a good idea for solving the kind of problems. These above researches also prove the ability of DRL algorithms for solving path planning problems in various environments. Currently, literature that combines the DRL algorithm and dynamic crowd simulation model to solve the problem of large-scale crowd simulation is still limited, so this topic has value of discussion.

2.2 Deep reinforcement learning

As a branch of RL (Talaat et al. 2020), DRL also is a kind of learning from state to action. The learning process only requires rewards, and maximizing future rewards is the learning goal of the algorithm. Practical problems often have complex realistic attributes and various abstract problems, which make RL algorithms fall into the dimensional explosion problem easily and unable to converge. DRL combines the perceptual ability of deep learning, which can handle high-dimensional state and action that may be contained in complex systems, providing ideas about solving complex real-world problems.

DRL algorithms are mainly divided into the value-based DRL and the DRL based on the policy gradient. Deep learning has a nonlinear fitting ability so that the neural network can better fit the value function. The Q function trained from the neural network can more accurately measure the performance of an action. Deep Q Network (DQN) (Mnih et al. 2015), as a value-based DRL algorithm, combines neural networks and reinforcement learning. It starts a new DRL era. The DRL algorithms based on the policy gradient can better deal with practical problems with large action space or continuous behavior compared with other value-based DRL algorithms. DDPG (Xu et al. 2019) is a deterministic policy gradient algorithm. Compared with other stochastic algorithms, the DDPG algorithm requires fewer samples and efficiency. However, these algorithms are not effective when dealing with the multi-objective optimization problem with an unfamiliar environment. The path planning problem in crowd evacuation simulation often needs to plan global path for multi-object in a dynamic environment, so single-agent DRL algorithms cannot solve this problem well.

The single-agent DRL algorithm is inefficient in dealing with multi-objective problems, so Lowe et al. (2017) proposed the MADDPG algorithm to solve the problem. The MADDPG algorithm parameterizes the policy to obtain an objective function, and directly optimizes the objective function to obtain the optimal policy through the gradient ascent. But when dealing with large-scale crowd evacuation, the training efficiency of the algorithm is dissatisfactory since the state space increases exponentially with an increasingly number of population. To improve the efficiency of the MADDPG algorithm, Jiao and Oh (2019) proposed a Rec-MADDPG algorithm. The Rec-MADDPG algorithm represents the externally observed state of each agent as a low-dimensional embedding. It proposes a distributed training paradigm combining parameter sharing and an A3C-based asynchronous framework, which reduces the algorithm's complexity and effectively improves the algorithm's adaptability to complex tasks; however, it reduces the learning efficiency of the algorithm. The proposed OMADDPG algorithm uses CEM to optimize the algorithm's deterministic policy to improve the algorithm's ability to learn effective policies. The DP algorithm is used to reduce the data correlation of the MADDPG algorithm, enhance the representativeness of the samples, and improve the training efficiency of the neural network.

3 A path planning method based on deep reinforcement learning for crowd evacuation

In this section, we elaborate on the method in detail. First, the method initializes environmental setting and the parameters of crowd. Then, the approach groups the crowd and selects leaders. Next, the method trains the OMADDPG algorithm and gets the completed model. Then, the method uses the OMADDPG algorithm to obtain the global path. The algorithm passes the global path to the leader in the environment. The leader guides pedestrians in the environment to avoid obstacles and select exit. Finally, the method outputs the visualization results. A detailed explanation is given in the following subsections.

3.1 Optimized multi-agent deep deterministic policy gradient algorithm

This paper improves the MADDPG algorithm to propose an OMADDPG algorithm. The OMADDPG algorithm uses the CEM to optimize the deterministic policy of the algorithm, which improve the efficiency of learning effective policies. In order to reduce the correlation between sample data, the algorithm uses the DP algorithm to delete the correlation experience randomly to improve the training efficiency of neural network.

The OMADDPG algorithm has two neural networks: an actor network and a critic network. Each of these networks has two different sections: online network and target network. The network's parameters of the critic network are expressed θ^Q and $\theta^{Q'}$. The actor network's parameters are expressed as θ^μ and $\theta^{\mu'}$. An replay buffer is set up to store the experience {current state x, current action $a_1, \ldots, a_n$, reward $r_1, \ldots, r_n$, next state x_} of all agents at each step, where $x = (o_1, o_2, \ldots, o_n)$ contains the observation state of all agents. θ^Q and θ^μ are initialized and copied to the corresponding parameters $\theta^{Q'}$ and $\theta^{\mu'}$. The replay buffer is initialized, S experiences are randomly selected from the replay buffer as a mini-batch of training data for the online network, the target Q value y is calculated, and then the mean square error loss function $L(\theta_i)$ is calculated through parameter y:

$$y = r_i + \gamma Q_i^{\mu'}\left(x', a_1', \ldots, a_n'\right)|_{a_i' = \mu_i'(o_i)} \tag{1}$$

$$L(\theta_i) = E_{x,a,r,x\_}[(Q_i^{\mu}(x, \ldots, a_n) - y)^2] \tag{2}$$

$$\nabla_{\theta^i} J \approx E_{x,a\sim D}\left[\nabla_{\theta^i}\mu_i(a_i|o_i)\nabla_{a^i}Q_i^{\mu}\left(x, a_1, \ldots, a_n\right)|_{a_i=\mu_i(o_i)}\right] \tag{3}$$

where $x = (o_1, o_2, \ldots, o_n)$ is the observation state, i represents the number of updates, D is the distribution function of state x, x includes the observation status of all agents, and $Q_i^{\mu'}$ represents the target network $\mu' = [\mu_1', \ldots, \mu_n']$. In addition, the function $Q_i^{\mu'}$ denotes the critic online network. The neural network is used to calculate the critic network's policy gradient. Then, two neural networks' parameters will be updated. Since the target network parameters delay update, the update formula adding the balance factor τ is:

$$\theta_i' \leftarrow \tau\theta_i + (1-\tau)\theta_i' \tag{4}$$

The method optimizes the policy of the actor network by using the CEM to obtain the optimized policy. The actor network outputs a set of noise-added action values a according to the formula, where μ_{θ_i} means the current policy and P_t represents the random noise. The actor network input them into the environment for execution to obtain a return value r and the next state x_.

$$a_i = \mu_{\theta_i}(o_i) + P_t \tag{5}$$

The algorithm deletes this experience $\{x, a_1, \ldots, a_n, r_1, \ldots, r_n, x\_\}$ randomly according to the probability of removal. N training episodes are repeated until the maximum number of training episodes is reached.

```
Input: information of $P^n$ pedestrians, the setting of environmental parameters
Output: deterministic policy $\mu_{\theta_i}$
Step 1: Initialize a random noise $P_t$ for action exploration and receive initial state x.
Step 2: For each agent, select action $a_i=\mu_{\theta_i}(o_i)+P_t$.
Step 3: Execute actions $a=(a_1,\ldots,a_n)$ and observe reward r and new state x_, set x←x_, calculate the unique value $u_i$ and
    P(remove_$e_i$) when the algorithm generates an experience, store experiences according to the probability.
Step 4: For each agent, randomly select S experiences $(x^j,a^j,r^j,x^j\_)$ from replay buffer, calculate $y^j$ according to (1) and update
    the critic online network according to (2), update the parameter of actor online network according to (3) and optimize the
    policy by using the cross-entropy method.
Step 5: Update target network parameters for each agent according to (4).
Step 6: If the algorithm reaches the max-episode-length, the algorithm is completed; otherwise, return to Step 2.
```

Optimized Multi Agent Deep Deterministic Policy Gradient Algorithm

In this paper, an OMADDPG algorithm is proposed. During the training process of the MADDPG algorithm, the algorithm is inevitably affected by noise, which leads to a decrease in the ability of the algorithm to learn effective policies efficiently. Therefore, the OMADDPG algorithm uses CEM to optimize the deterministic policy of the algorithm, which improves the ability of the algorithm to learn effective policies. Since the MADDPG algorithm uses a first-in-first-out order to remove the experience pool data, it leads to the accumulation of redundant data, which affects the convergence of the algorithm. Therefore, the algorithm uses the DP algorithm to reduce the data relevance of the MADDPG algorithm, enhance the representativeness of the samples, and improve the training efficiency of the neural network.

3.1.1 Definitions

Definition 1 (*Observation state*) The algorithm uses *o* to represent the observation state, and the observation state includes the external observation state and internal observation state. The external observation state includes the relative distance between the agent with other adjacent entities in real space and the Euclidean distance of the center of mass for the agent and the target position in real space. The internal observation state includes the coordinates and direction of the agent.

Definition 2 (*Action*) The algorithm uses the parameter *a* to represent the action at the next moment according to the current observation state *o*. The algorithm sets the agent's current coordinate to specific coordinate in the scene. And the algorithm set the next move direction of the agent to be expressed as (l_1, l_2), which is the vector of the next move direction of the agent.

Definition 3 (*Reward function*) The algorithm uses *r* to represent the immediate reward in the OMADDPG algorithm. This paper considers the various factors that affect the evacuation of people and considers the distance between pedestrians and targets, whether pedestrians collide with obstacles or each other as the design standard. In addition, this paper defines that when the distance between the agent and other entities is less than the sum of their radii, it means that they have collided. n_i is the distance between the pedestrian and the target position, the leader's specific coordinate is (x_i, y_i), and the target position is (x_i, y_i). We can obtain

$$n_i = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{6}$$

There is a collision when $r_p = -10$, and when $r_p = 0$ there is no collision, R based on (7):

$$R = -\min(n_i) + r_p \tag{7}$$

3.1.2 Cross-entropy method

The CEM is a Monte Carlo method that is mainly used for optimization and importance sampling. In the training process of MADDPG algorithm, it is inevitably affected by bad samples so that it cannot converge quickly. Although several hybrid loss functions (Göçeri. 2021a, b) have been proposed to solve different problems with deep network architectures, we preferred the cross-entropy loss function due to avoid the high computational cost and its efficiency in the proposed architecture. Therefore, the algorithm uses the CEM to optimize the deterministic policy.

Definition 4

(*Gaussian distribution*) The CEM uses Gaussian distribution $N(\mu_{\theta_i}, \sigma_i^2)$ to perform importance sampling. The method uses the elite samples selected by the critic network to update the deterministic policy. The parameters are adjusted by calculating the weight of the elite samples to better optimize the algorithm policy. The formula is defined as follows:

$$w_j = e^{Q_i^{\mu}(x,a)} \tag{8}$$

$$\mu_i = \sum_{j=1}^{M} w_j a_j / \sum_{j=1}^{M} w_j \tag{9}$$

$$\sigma_i^2 = \sum_{j=1}^{M} w_j (a_j - \mu_i)^2 / \sum_{j=1}^{M} w_j \tag{10}$$

where $Q_i^{\mu}(x,a)$ is the evaluation function obtained by the critic network according to sample *A*, w_j is the weight of the elite sample, and μ_{θ_i} and σ_i^2 are two parameters of the Gaussian distribution. Since the value obtained by $Q_i^{\mu}(x,a)$ is negative, the exp function is used to calculate the weight of the parameters to increase the proportion of elite samples of the policy. Thus, we can get the optimized deterministic policy.

```
Input: deterministic policy $\mu_{\theta_i}=\pi_\theta^i(x)|_{x\sim D}$
Output: optimized deterministic policy $\mu_{\theta_i}=\pi_\theta^i(x)|_{x\sim D}$
Step 1: When the algorithm completes a training episode, let the actor network output the deterministic policy $\mu_{\theta_i}=\pi_\theta^i(x)|_{x\sim D}$ for
    each agent and initialize $\sigma_i^2$, where D is the distribution function of state x.
Step 2: Calculate the Gaussian distribution $N(\mu_{\theta_i},\sigma_i^2)$ of the action output by each policy in the current state according formula
    (8) (9) (10).
Step 3: Sample T actions $(a_1,\ldots,a_T)$ from the Gaussian distribution $N(\mu_{\theta_i},\sigma_i^2)$.
Step 4: Let $A=\{\mu_{\theta_i}\} \cup \{a_1,\ldots,a_T\}$, input A into the critic network's Q function $Q_i^{\mu}(x,a)$.
Step 5: Sample M elite actions from A according to the critic network $Q_i^{\mu}(x,a)$, and update $\mu_{\theta_i}$ and $\sigma_i^2$ based on these elite
    actions.
Step 6: If the algorithm reaches the max-episode-length, the algorithm is completed; otherwise, return to Step 2.
```

Cross Entropy Method

3.1.3 Data pruning

When the replay buffer capacity reaches the limit, the MADDPG algorithm chooses to store data in a queue and deletes redundant data in an order manner, which leads to the accumulation of highly relevant data in the replay buffer and affects the algorithm's convergence. Therefore, the method uses the DP algorithm to process experience in the replay buffer. The DP algorithm is a data preprocessing method. By removing some redundant experiences in the calculation process, it can improve the training efficiency of neural network. This paper sets a unique value to evaluate whether the experience should be deleted.

Definition 5

(*Unique value*) The algorithm designs the unique value as a standard to evaluate the correlation of experience. According to the unique value, the high relevance experiences are deleted. The data is generated from the training process and then the data features are extracted. The probability of deleting the data is obtained by the formula and the data is deleted randomly according to the probability. The formula is defined as follows:

$$u_i = \sum_{j=1}^{d} S(\Phi_i, \Phi_j) \tag{11}$$

$$P\left(\mathrm{remove}_{e_i}\right) = u_i^{-1} / \sum\nolimits_{i=1}^{d} u_i^{-1} \tag{12}$$

where $\Phi_i = \{x_i, a_i, r_i, x_i\}$ is the experience in the replay buffer, d is the size of the replay buffer, and $S(\Phi_i, \Phi_j)$ is the distance function. The algorithm deletes experience randomly according to the probability *P*.

```
Input: a set of experience $e_i$
Output: the probability of removal P
Step 1: When a set of experience $e_i=\{x,a_1,\ldots,a_n,r_1,\ldots,r_n,x\_\}$ is generated, extract the feature $\Phi_i=\{x_i,a_i,r_i,x\_i\}$.
Step 2: Calculate the unique value $u_i$ and the probability of removal according to formula (11) (12).
Step 3: Randomly discards the experiences according to the probability of removal.
Step 4: If the algorithm reaches the max-episode-length, the algorithm is completed; otherwise, return to Step 1.
```

Data pruning algorithm

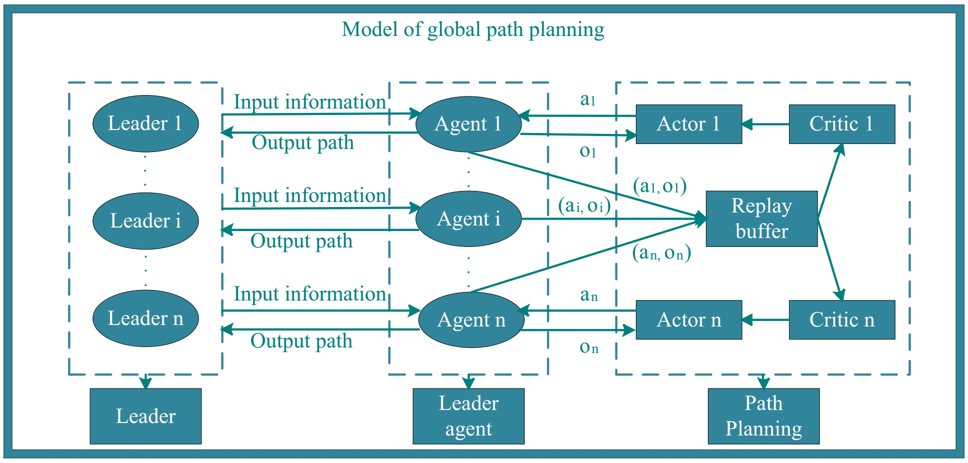

Fig. 1 The structure of the global path planning algorithm

3.2 Improved social force model

To improve the efficiency and authenticity of simulation, micro model is necessary. In this paper, the social force model (Helbing et al. 1995) is used to avoid obstacles. In this model, the pedestrian is regarded as a particle that moves under the driving of mental factors and physical factors. Thus, the formula is defined as follows:

$$m_i \frac{d\overrightarrow{v_i(t)}}{dt} = \overrightarrow{f_i^0} + \sum_{j(\neq i)} \overrightarrow{f_{ij}} + \sum_{w} \overrightarrow{f_{iw}} \tag{13}$$

where $\overrightarrow{v_i(t)}$ is the movement speed of the individual during evacuation, and $\overrightarrow{f_i^0}$ is the self-driving force generated by the individual being affected by the target position. $\overrightarrow{f_{ij}}$ is the force that individual *i* receives from individual *j* to maintain a safe distance. $\overrightarrow{f_{iw}}$ is the force of a wall or obstacle on an individual, which is used to avoid obstacles.

Pedestrians are inevitably disturbed by various psychological factors in the process of moving. The surrounding pedestrians and the environment will affect pedestrian excitement. We improve the expected speed $v_i^0(t)$ of the model and use the parameter $h_i(t)$ to reflect the excitement of pedestrians, as shown in the formula:

$$h_i(t) = x_1 * g\left(\left(v_i^0(t) - \overrightarrow{v_i(t)}\right)/v_i^0(t)\right) + x_2 * g\left(\left(\overline{v_0(t)} - \overrightarrow{v_i(t)}\right)/\overline{v_0(t)}\right) \tag{14}$$

where x_1 and x_2 are constants between 0 and 1, and $x_1 + x_2 = 1$, $g(x)$ is a piecewise function. In the formula, the first term on the right of the equal sign reflects the impact of pedestrian real-time desired speed. The second term on the right of the equal sign reflects the influence of surrounding pedestrians. $\overline{v_0(t)}$ is the average speed of other pedestrians in the same group, $\overrightarrow{v_i(t)}$ is the component of pedestrian speed in the direction of desired speed. If $\overline{v_0(t)}$ exceeds the speed of the pedestrian, the excitement of the pedestrian may be aggravated, and the desired speed will increase. The real-time update formula of the desired speed is:

$$v_i^0(t) = \left(1 - h_i(t)\right) * v_i^{min} + h_i(t) * v_i^{max} \tag{15}$$

where v_i^{min} and v_i^{max} refer to the minimum expected speed and the maximum expected speed, respectively. This paper sets $v_i^{min} = 0.6$ m/s, and $v_i^{max} = 2.4$ m/s according to the reference (Weidmann 1993).

In the real world, all individuals have a certain relationship, and completely isolated individuals are nonexistent. Therefore, we introduce the group cohesive force to extend the social force model. Individuals would attract others who have close relationships with them under the influence of group cohesion. The formula is defined as follows:

$$\overrightarrow{f_{ij}^{rel}} = R_{ij} E_i \exp((r_{ij} - d_{ij})/G_i) \overrightarrow{n_{ij}} \tag{16}$$

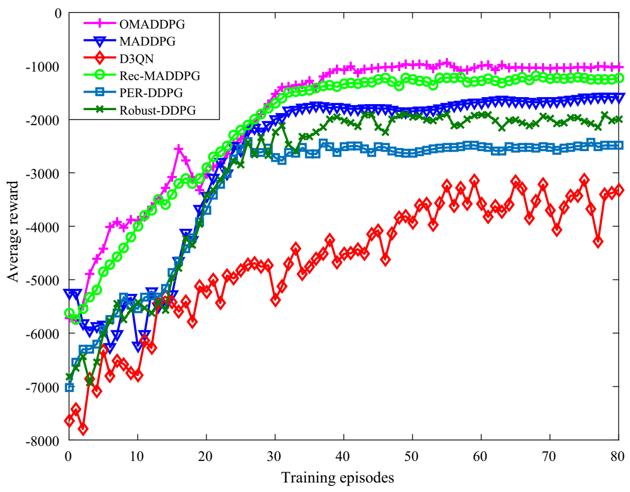

Fig. 2 Average reward of the six methods

where E_i represents the coefficient quantifying the relationship between pedestrians and G_i is the minimum safe distance of pedestrian movement. R_{ij} is the quantization constant for the different relationships between pedestrians. $\overrightarrow{n_{ij}}$ is the vector direction between pedestrians. These relations are defined in ρ_{rl}. The larger the value of ρ_{rl}, the closer the relationship between pedestrians.

3.3 Multi-agent particle interactive environment

This paper builds build a multi-agent particle interactive environment in the OpenAI that combined with the improved social force model to realize our method. The multi-agent particle interactive environment supports a variety of variations that can comprehensively demonstrate our method by changing the numbers and positions of exits and obstacles in this environment.

When an evacuation occurs, most pedestrians will move in the direction of other pedestrians under the influence of other pedestrians and obstacles in complex environments. Therefore, different groups of people (Zhang et al. 2018) will form. According to this phenomenon, Ji and Gao (2007) proposed a leader–follower model to reproduce this phenomenon. The model provided an idea to solve the problem of low efficiency in simulation of large-scale crowd evacuation. This paper uses K-Medoids algorithm to group pedestrians. The relationship between individuals are necessary for the partition of group, so the relationship between individuals is added to the algorithm as the eigenvalue.

Definition 6 (*Eigenvalue function*) Individuals tend to have close relationships with other individuals in crowd evacuation. Considering the impact of these practical factors, the relationship between individuals is added to the algorithm as the eigenvalue:

$$F(i,j) = w_1 d_{ij} + w_2/(R_{ij} + u) \tag{17}$$

where $w_1+w_2 = 1$, $F(i,j)$ is the eigenvalue function, d_{ij} is the distance of pedestrian *i* and pedestrian *j*, R_{ij} is the quantization constant for the different relationships, and *u* is a constant. This paper sets the constant $u=0.5$ and the $k = \sqrt{P^n}/2$.

```
Input: information of $P^n$ pedestrians
Output: k clustering clusters
Step 1: Randomly select $k=\sqrt{P^n}/2$ objects as initial cluster centers.
Step 2: Calculate the eigenvalue between pedestrian particles and k cluster centers.
Step 3: Group other particles according to the size of the eigenvalue.
Step 4: Calculate the average eigenvalue of all pedestrian particles in the cluster.
Step 5: Select new cluster center based on the average eigenvalue of the objects in each cluster.
Step 6: If clusters no longer change, the algorithm is completed; otherwise, return to Step 2.
```

Improved K Medoids Algorithm

Table 1 Comparison of the average reward in a scene with obstacles

Methods	Training episodes						
	0	10,000	20,000	30,000	40,000	50,000	60,000
Our method	−5271.28	−3875.37	−3317.64	−1706.03	−1056.33	−971.91	−1039.32
MADDPG	−5244.5	−5338.55	−3673.35	−2099.56	−1770.77	−1855.34	−1682.36
D3QN	−7635.28	−6742.31	−5127.4	−4729.5	−4677.96	−3813.44	−3153.83
Rec-MADDPG	−5624.74	−4207.48	−3102.51	−1847.27	−1394.71	−1221.81	−1387.51
PER-DDPG	−7204.31	−5547.19	−3697.24	−2717.31	−2517.56	−2638.01	−2517.21
Robust-DDPG	−6821.74	−5424.51	−3412.23	−2237.74	−2017.45	−1949.31	−1911.12

Table 2 Comparison of the success rates in a scene with obstacles

Methods	Training episodes		
	40,000	60,000	80,000
OMADDPG	0.937	0.941	0.942
OMADDPG without CEM	0.854	0.872	0.874
MADDPG	0.784	0.852	0.861

The group will follow the leader to complete evacuation simulation. Therefore, this paper sets the following conditions based on the actual situation. Leaders should meet these conditions:

1. Every leader should know at least one door closest to him or her.
2. When evacuating, the leader should lead the evacuation in front of the group.
3. When pedestrians start moving, the pedestrian closest to the exit should be given priority as the leader.

The leader's speed formula is defined as follows:

$$\overrightarrow{v_i^l}=v_i^0(t)\overrightarrow{b_i^0(t)} \tag{18}$$

where $\overrightarrow{b_i^0(t)}$ is the vector representation of the next movement direction. In addition, the desired direction of follower becomes the leader's movement direction:

$$\overrightarrow{e_i^0}=(\overrightarrow{p_i}-\overrightarrow{p_j})/d_{ij} \tag{19}$$

where $\overrightarrow{p_i}$ and $\overrightarrow{p_j}$ represent the positions of the follower and the leader, d_{ij} is the distance of the follower and the leader.

Definition 7 (*Leader agent*) The method uses a six tuple < Aid, Input, Communication, Output, Goal, Trigger > to represent the leader agent. Aid is the number of leader agents. The input interface component obtains the leader's current observation state and reward information, and the message is transmitted to the agent through a communication component. The path planning model transmits the generated next path information to the output interface component of the leader agent, and then returns it to the leader. The goal is to choose the best evacuation route. Trigger is the component of activity triggering, which determines whether to activate the leader agent to start the evacuation.

Figure 1 shows the structure of the global path planning algorithm. The leader, as the mapping of the leader agent, is responsible for obtaining the current observation state in the environment and transmitting the summary information to the leader agent. Next, the leader agent transmits the observation state into the path planning module. At this time, the path planning module will output the deterministic policy, and then the deterministic policy will output the optimized path by the action to the leader agent. The leader will execute the action and get next observation state. Finally, the leader agent inputs the current experience into the replay buffer as the training data.

4 Simulation experiment and analysis

In this section, this paper presents experiments for evaluating the OMADDPG algorithm and testing the path planning method for crowd evacuation simulation as follows.

1. The experiments in Sect. 4.1 illustrate that the OMADDPG algorithm has a faster convergence rate and better stability than other DRL algorithms since the policy is optimized.
2. The experiments in Sect. 4.2 illustrate that our path planning method can effectively complete the evacuation simulation with a large population in an obstacle-free scene when compared with other path planning methods.
3. The experiments in Sect. 4.3 illustrate that our path planning method can effectively complete the evacuation simulation with a large population in an obstacle scene when compared with other path planning methods.
4. The simulation results in Sect. 4.4 illustrate that our method has good adaptability and matches the movement of the crowd in real world.

In this paper, the simulation experiment is based on the Windows 7 operating system environment. All of the reported results are obtained on a machine with a 3.40 GHz Intel Core 2 Duo CPU with 16 GB of RAM.

4.1 The policy performance comparison of different DRL algorithms

In this experiment, this paper compares the OMADDPG algorithm with existing DRL algorithms to verify the performance and the robustness of the proposed algorithm. The main purpose of the experiment is to verify the performance of the OMADDPG algorithm by comparing the average reward of different DRL algorithms. In order to reduce the influence of other environmental factors, different DRL algorithms need to be trained in the same benchmark-testing environment. The evacuation scene is a 50*30 m room. The algorithm sets the population size is 200. In this experiment, Adam optimizer is used to update network parameters and the activation function is ReLU. The algorithm chooses

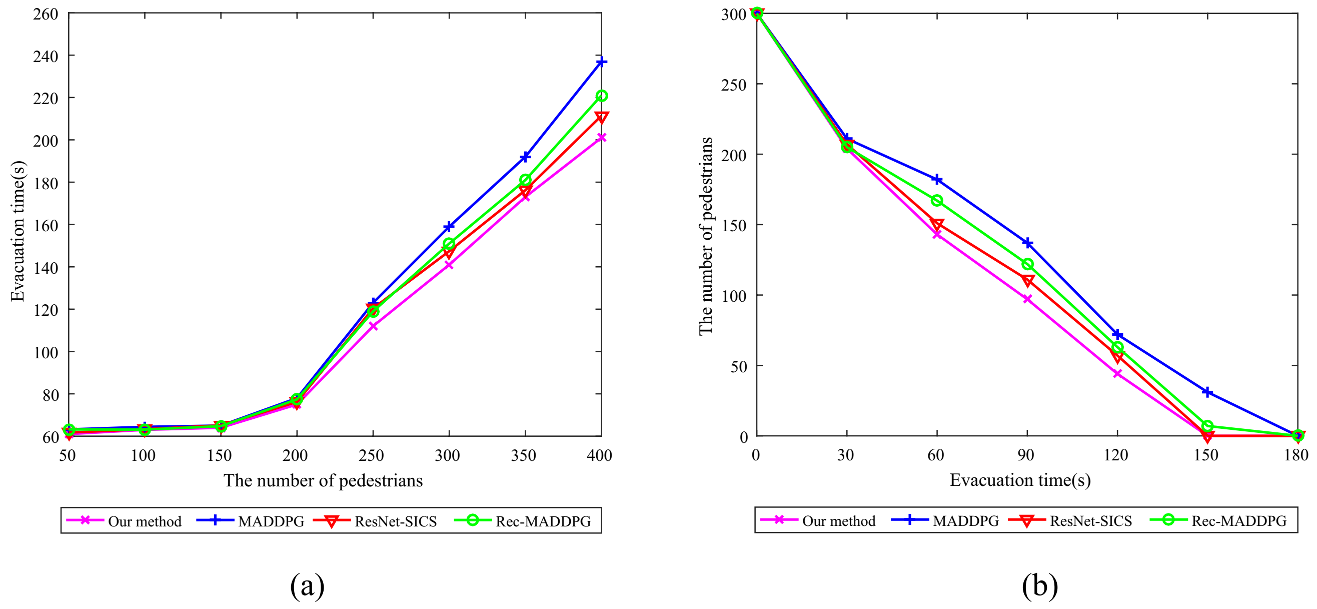

Fig. 3 Comparison of evacuation time between the four methods

Table 3 Evacuation time(s) of the four methods in a scene with obstacles

Methods	The number of pedestrians							
	50	100	150	200	250	300	350	400
Our method	61.1	63.7	64.4	75.7	112.1	141.2	173.7	201.4
MADDPG	63.2	64.4	65.1	78.2	123.2	159.7	192.1	237.5
ResNet-SICS	61.7	63.3	65.1	76.2	120.4	147.4	176.2	211.4
Rec-MADDPG	63.2	64.1	64.7	77.4	119.7	151.6	181.1	221.7

Table 4 Variations in the number of pedestrians in a scene with obstacles

Methods	Evacuation time (s)						
	0	30	60	90	120	150	180
Our method	300	204	143	97	44	0	0
MADDPG	300	211	182	137	72	31	0
ResNet-SICS	300	207	151	111	57	0	0
Rec-MADDPG	300	205	167	122	63	7	0

80,000 episodes as the end. The algorithm sets $\gamma = 0.97$ and $\tau = 0.01$. The learning rate of the Adam optimizer is 0.01. The size of the replay buffer is 106 and the size of mini-batch is 1024 (Lowe et al. 2017). In this section, the experiments demonstrate the performance and robustness of the proposed algorithm by comparing the average reward. As a significant criterion in reinforcement learning, average reward intuitively describes the change of the algorithm in the environment. The experiment compares our algorithm with the D3QN method (Yan et al. 2020), the Rec-MADDPG algorithm (Jiao and Oh 2019), the PER-DDPG algorithm (Hu et al. 2020), the Robust-DDPG algorithm (Wan et al. 2020) and the MADDPG algorithm (Lowe et al. 2017) in a scene that consists of a door and five obstacles. Because the total number of training episodes is difficult to limit, the experiment records the average reward for every 1000 training episodes.

As shown in Fig. 2 (Table 1), the average reward of the OMADDPG is better than other methods in the same environment setting. Because of the setting of exit distance and crowd collision in the reward function, a negative reward is inevitable. In addition, since the addition of noise perturbation and neural network to the DRL algorithm, the algorithm will inevitably be affected when outputting actions and will not achieve optimal results. In the Fig. 2 (Table 1), the proposed algorithm has a higher reward growth rate and can converge to a higher location compared with other algorithms. Between 40,000 episodes and 80,000 episodes, the proposed algorithm converges, and the fluctuation range of our algorithm tends to be stable. The average reward of the D3QN algorithm and the Robust-DDPG algorithm have a high amplitude, and the Robust-DDPG algorithm does not converge. The Rec-MADDPG algorithm and the PER-DDPG algorithm are close to convergence after 40,000 training episodes. Since the DP algorithm, our algorithm converges faster and more stable. At this time, the average rewards of other algorithms are lower than the proposed algorithm. The experimental result illustrates that the OMADDPG algorithm can deal with the multi-objective path planning problem better than other DRL algorithms.

To analyze the influence of different methods for our algorithm, this paper compares the success rate of 200 episodes run by different versions of MADDPG algorithm at the end of different number of episodes in the same scene. The experiment is set to be successful when all pedestrians successfully evacuate from the scene, and record the success rate of different methods. This experiment sets the population size is 100. Table. 2 summarizes the success rate of every different version of MADDPG algorithm. Compared with other algorithms, our algorithm considers the influence

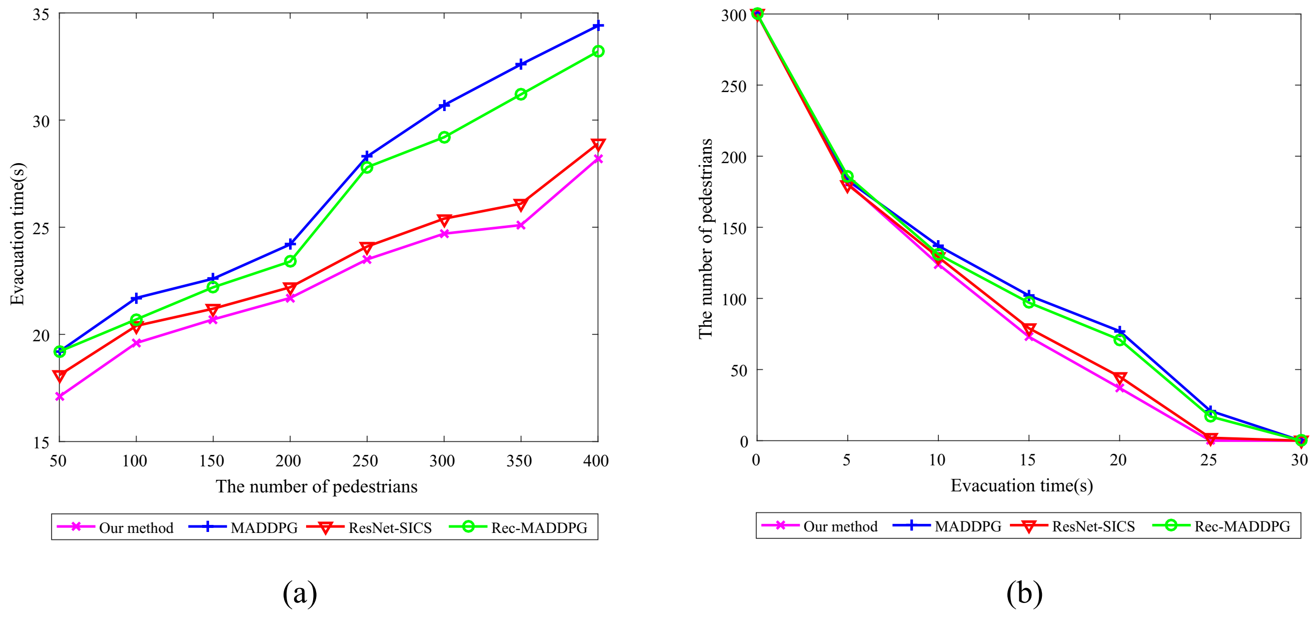

Fig. 4 Variation of the four methods in a scene without obstacles

Table 5 The evacuation time (s) of different methods in a scene without obstacle

Methods	The number of pedestrians							
	50	100	150	200	250	300	350	400
Our method	17.1	19.6	20.7	21.7	23.5	24.7	25.1	28.2
MADDPG	19.2	21.7	22.6	24.2	28.3	30.7	32.6	34.4
ResNet-SICS	18.1	20.4	21.2	22.2	24.1	25.4	26.1	28.9
Rec-MADDPG	19.2	20.7	22.2	23.4	27.8	29.2	31.2	33.2

Table 6 Variations in the number of pedestrians in a scene without obstacle

Methods	Evacuation time (s)						
	0	5	10	15	20	25	30
Our method	300	181	124	73	37	0	0
MADDPG	300	183	137	102	77	21	0
ResNet-SICS	300	180	129	79	45	2	0
Rec-MADDPG	300	186	131	97	71	17	0

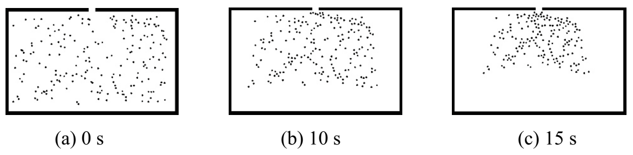
Fig. 5 Simulation result in an obstacle-free scene with one exit

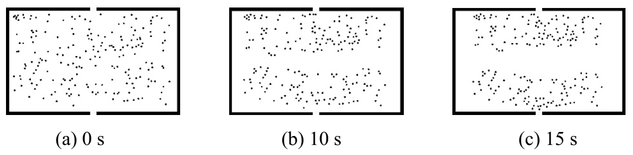
Fig. 6 Simulation result in an obstacle-free scene with two exits

of unfamiliar environments on the DRL algorithm, uses the CEM to balance the relationship between exploration and utilization of DRL. As we can see from Table 2, the success rate of our algorithm has reached stability at about 40,000 episodes, which indicates the effectiveness of the DP algorithm. Compared with the algorithm without CEM, the OMADDPG algorithm successfully completed evacuation tasks 188.4 times in average and the success rate reach to 94.2% after 80,000 episodes. The result proves that the algorithm can effectively learn policy based on joint observation.

4.2 Experiments in a scene with obstacles

To verify the evacuation ability of our method, we conducted experiments in a scene with four exits. The focus of crowd evacuation simulation is to use evacuation simulation instead of actual evacuation drills and to provide decision support for real crowd evacuation. How to complete the evacuation simulation in a safe and efficient manner is the key. The purpose of the simulation experiment is to verify the performance of our method. The changes of evacuation time and the number of pedestrians during evacuation can directly reflect the performance of the method in crowd evacuation simulation. Therefore, we verify the evacuation capability of our method by comparing the changes in evacuation time and number of pedestrians of the four methods under different conditions in an obstacle scene. The experiments compare our method with the ResNet-SICS method in the literature (Yao et al. 2020), the Rec-MADDPG algorithm in the literature (Jiao and Oh 2019) and the MADDPG algorithm. First, we verify the evacuation ability of our method in a large-scale evacuation simulation by varying the number of pedestrians and the result is shown in Fig. 3a (Table 3). Next, we verify the stability of our method in a scene with obstacles by showing the change of the number of pedestrians in the same condition for the four methods and the result is shown in Fig. 3b (Table 4).

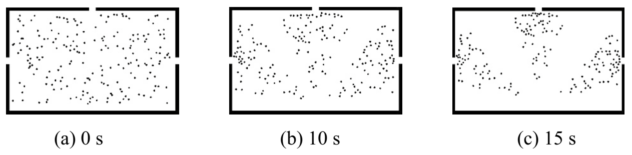
Fig. 7 Simulation result in an obstacle-free scene with three exits

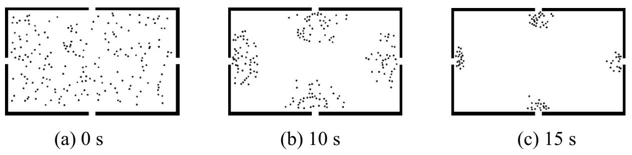
Fig. 8 Simulation result in an obstacle-free scene with four exits

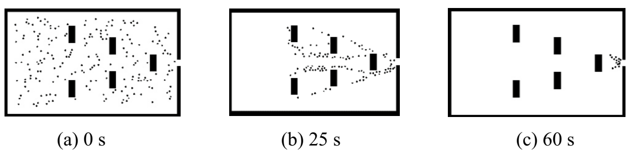
Fig. 9 Simulation result in an obstacle scene with one exit

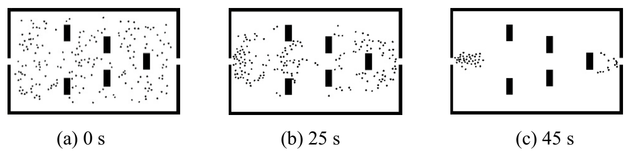
Fig. 10 Simulation result in an obstacle scene with two exits

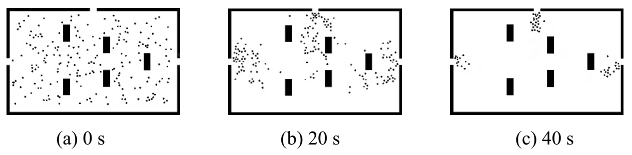
Fig. 11 Simulation result in an obstacle scene with three exits

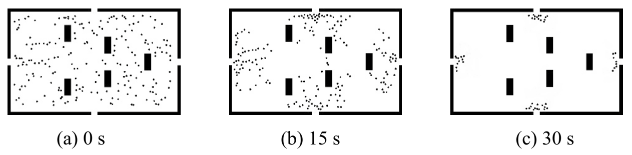
Fig. 12 Simulation result in an obstacle scene with four exits

The complexity of crowd evacuation will increase with a growing number of populations. Figure 3 is a comparison of the evacuation time between the four methods under a five-obstacle scene. It can be explained from Fig. 3a (Table 3) that the evacuation time required by each method will

Table 7 The number of pedestrians of the four methods

Scene	The number of exits			
	1	2	3	4
Obstacle scene	75.7	51.2	45.2	32.4
Obstacle-free scene	47.2	35.7	24.1	21.7

increase with a growing number of pedestrians. Compared with the ResNet-SICS, the Rec-MADDPG and the MADDPG algorithm, our method has obvious advantage when evacuating more than 200 people. Figure 3b (Table 4) is the comparison of the changes of evacuated individuals over time in the same scene. We set the population size is 300. The evacuated individuals of the four methods have a more obvious downward trend. When the time exceeds 60 s, there is a large difference in the number of evacuated individuals in the four methods. Our method, compared to other methods, will evacuate smoothly over time. In 150 s, our method and the ResNet-SICS completed the evacuation.

The proposed method balances exploration and utilization of the algorithm and effectively enhances the efficiency of learning policy. The result demonstrates that our method has good performance when the number of evacuated individuals is increasing and can quickly plan suitable evacuation routes for pedestrians to speed up evacuation. Combined with the analysis in the previous section, we can prove the performance of our method in solving the problem of large-scale crowd evacuation and that our method is superior to other path planning methods in stability.

4.3 Experiments in a scene without obstacle

To verify the evacuation ability of our method, we conducted experiments in an obstacle scene. The ability of our method is illustrated by comparing the variation in evacuation time and the number of pedestrians for four methods in a scene without obstacle under different conditions. The experiments compare our method with the ResNet-SICS method in the literature (Yao et al. 2020), the Rec-MADDPG algorithm in the literature Jiao and Oh 2019) and the MADDPG algorithm. First, we verify the evacuation ability of our method in a large-scale evacuation simulation by varying the number of pedestrians the result is shown in Fig. 4a (Table 5). Next, we verify the stability of our method in a scene without obstacle by showing the change of the number of pedestrians in the same condition for the four methods and the result is shown in Fig. 4b (Table 6).

The probability of collision in crowd movement increases with population growth. Figure 3 illustrates the variation in the four methods for a scene without obstacles. From Fig. 4a (Table 5), it can be seen that the evacuation time required for each method increases as the number of pedestrians increases. Each path planning method quickly plans a global path when the number of pedestrians is less than a certain level. When the number of pedestrians exceeds 250, our method can still quickly select an exit for the evacuating crowd based on an optimized deterministic policy that reasonably uses the exit spaces to avoid collisions, while the other methods cannot quickly plan paths for the evacuating crowd due to the increase in the number of evacuees. Compared with other methods, this method can effectively complete the evacuation simulation with a large population in an obstacle-free scene and can quickly help the crowd complete evacuation and avoid collisions.

Figure 4b (Table 6) shows the comparison in terms of the number of evacuated individuals over time for the same scene. We set the population size to 300. When the evacuation starts, there is no significant difference in the decreasing trend of evacuating individuals among the four evacuation methods. When the time exceeds 15 s, the four evacuation methods start to show significant differences due to the different evacuation strategies, and our method evacuates the individuals in a faster and more stable manner. The other methods are influenced by other individuals when planning the path for individuals to collide and cannot help the crowd evacuate quickly and avoid collisions.

4.4 Crowd evacuation simulation under different exits

To demonstrate the adaptability and authenticity of our method, we conduct experiments by varying the number of exits in different scenes. Considering the various situations that may occur in actual evacuation, this paper tests the effectiveness of our method in scenes with obstacles as well as scenes without obstacle. In this experiment, we set the number of pedestrians to 200. By comparing the simulation results in different scenes, the authenticity and the adaptability of our method can be demonstrated.

The experimental results are shown in Figs. 5, 6, 7, 8, 9, 10, 11 and 12 (Table 7). We can see that the number of exits influences evacuation efficiency. The more exits there are, the lower the probability of congestion and the higher the evacuation efficiency. In addition, the evacuation time of our method in different scenes decreases steadily, which proves the stability of our method. Pedestrians in complex environments will form many groups (Zhang et al. 2018) under the influence of individuals and obstacles in the line of sight for moving. When different groups move towards the same exit, pedestrians will form a pedestrian flow to avoid obstacles in the forward direction. In the obstacle scene, our method divides pedestrians into different groups and selects the exit to complete the evacuation according to the actual situation.

Our method leads pedestrians to different exits to complete evacuation in the obstacle-free scene. The simulation results consistent with the actual pedestrian movement demonstrate the simulation ability of our method.

The simulation results also illustrate that our method can correctly select an exit for pedestrians to complete the evacuation and avoid the occurrence of collision to the maximum extent in the obstacle-free scene. In the scene with obstacles, our method guides pedestrians to avoid obstacles in the forward direction. It fully uses exit spaces to reduce collisions, improving evacuation efficiency. The simulation results in two types of scenes show that our method can guide the crowd to the exit to complete the evacuation in two different types of evacuation simulation scenes, which verifies the adaptability of our method. In addition, our method can lead the crowd to choose the exit and avoid the obstacles to complete the evacuation in different scenes by the leader, which is consistent with the crowd movement in real world.

5 Conclusion

This paper proposes a path planning method based on deep reinforcement learning for crowd evacuation. In addition, this paper proposes an OMADDPG algorithm, which expands the MADDPG algorithm by applying the DP algorithm and CEM to optimize policy and improve the training efficiency. Next, the method incorporates the relationship between individuals and psychological factors into the social force model. Finally, the method combines the OMADDPG algorithm with the improved social force model to complete the evacuation simulation. The OMADDPG algorithm transmits next path information to the leader in the environment. Pedestrians in the environment are driven by the improved social force model to follow the leader to avoid obstacles and arrive the exit.

According to our experimental results, we can draw the following conclusions:

a. The OMADDPG has better performance and stronger robustness with other DRL algorithms. The CEM optimizes the deterministic policy of the OMADDPG algorithm. The DP algorithm reduces the correlation between samples to enhance the speed of convergence.
b. The path planning method can quickly plan suitable evacuation routes for pedestrians with an increasing number of evacuated population to speed up evacuation. The method can plan global paths dynamically for pedestrians in different environments. The results prove that our method can obtain global path in various environments and reduce evacuation time.

We will continue DRL research and attempt to apply unsupervised learning methods to DRL. In the future, we hope we can extend the model to more complex evacuation problems.

Funding This work was supported by the National Natural Science Foundation of China (62276156, 61876102, 61972237).

Code or data availability The data and code used and analysed during the study are available from the first author on reasonable request.

Declarations

Conflict of interest The authors have no conflicts of interest to declare that are relevant to the content of this article.

Consent to participate All authors have read and agreed to publish this work.

Consent for publication All authors have read and agreed to publish this work.

References

Chen AY, He JT, Liang MC, Su GF (2021) Crowd response considering herd effect and exit familiarity under emergent occasions: a case study of an evacuation drill experiment. Phys A Stat Mech Appl 556(15):124654
Gao J, Gong J, Qing Q (2021) Coupling evacuation model of air-supported membrane buildings subjected to air-leakage based on multi-velocity cellular automaton. Simul Model Pract Theory 108:102257
Göçeri E (2021a) An application for automated diagnosis of facial dermatological diseases. İzmir Katip Çelebi Univ Fac Health Sci J 6(3):91–99
Göçeri E (2021b) Diagnosis of skin diseases in the era of deep learning and mobile technology. Comput Biol Med 134:104458
Gul F, Rahiman W, Alhady SSN et al (2021) Meta-heuristic approach for solving multi-objective path planning for autonomous guided robot using PSO-GWO optimization algorithm with evolutionary programming. J Ambient Intell Human Comput 12:7873–7890
Helbing D, Molnár P (1995) Social force model for pedestrian dynamics. Phys Rev E 51(5):4282–4286
Hu J, Niu H, Carrasco J, Lennox B, Arvin F (2020) Voronoi-based multi-robot autonomous exploration in unknown environments via deep reinforcement learning. IEEE Trans Veh Technol 69(12):14413–14423
Jardine PT, Givigi S (2021) Improving control performance of unmanned aerial vehicles through shared experience. J Intell Robot Syst 102:68
Ji Q, Gao C (2007) Simulating crowd evacuation with a leader-follower model. IJCSES Int J Comput Sci Eng Syst CSES Int 411:63–73
Jiang Y, Chen B, Li X, Ding Z (2020) Dynamic navigation field in the social force model for pedestrian evacuation. Appl Math Model 80:815–826
Jiao Z, Oh J (2019) End-to-end reinforcement learning for multi-agent continuous control. In: Proceedings of 18th IEEE international conference on machine learning and applications, ICMLA 2019, pp 535–540

Küllü K, Güdükbay U, Manocha D (2017) ACMICS: an agent communication model for interacting crowd simulation. Auton Agent Multi-Agent Syst 31(6):1403–1423

Liu H, Liu BX, Zhang H, Li L, Qin X, Zhang GJ (2018) Crowd evacuation simulation approach based on navigation knowledge and two-layer control mechanism. Inf Sci (NY) 436–437:247–267

Lowe R, Wu Y, Tamar A et al (2017) Multi-agent actor-critic for mixed cooperative-competitive environments. Adv Neural Inf Process Syst 30:6379–6390

Madani Y, Ezzikouri H, Erritali M et al (2020) Finding optimal pedagogical content in an adaptive e-learning platform using a new recommendation approach and reinforcement learning. J Ambient Intell Human Comput 11:3921–3936

Mnih V, Kavukcuoglu K, Silver D, Rusu AA et al (2015) Human-level control through deep reinforcement learning. Nature 518(7540):529–533

Oğuz O, Akaydın A, Yılmaz T, Güdükbay U (2010) Emergency crowd simulation for outdoor environments. Comput Graph 34(2):136–144

Sun W, Zang W, Liu C et al (2021) Motion pattern optimization and energy analysis for underwater glider based on the multi-objective artificial bee colony method. J Mar Sci Eng 9(3):327–345

Talaat FM, Saraya MS, Saleh AI et al (2020) A load balancing and optimization strategy (LBOS) using reinforcement learning in fog computing environment. J Ambient Intell Human Comput 11:4951–4966

Tang TQ, Zhang BT, Xie CZ (2019) Modeling and simulation of pedestrian flow in university canteen. Simul Model Pract Theory 95:96–111

Wan K, Gao X, Hu Z, Wu G (2020) Robust motion control for UAV in dynamic uncertain environments using deep reinforcement learning. Remote Sens 12(4):640–661

Weidmann U (1993) Transporttechnik der fußgänger: transporttechnische eigenschaften des fußgängerverkehrs, literaturauswertung. IVT Schriftenreihe. https://doi.org/10.3929/ethz-b-000242008

Xu H, Wang N, Zhao H, Zheng Z (2019) Deep reinforcement learning-based path planning of under actuated surface vessels. Cyber Phys Syst 5(1):1–17

Xu Z, Deng D, Shimada K (2021) Autonomous UAV exploration of dynamic environments via incremental sampling and probabilistic roadmap. IEEE Robot Autom Lett 6(2):2729–2736

Yan C, Xiang X, Wang C (2020) Towards real-time path planning through deep reinforcement learning for a UAV in dynamic environments. J Intell Robot Syst 98:297–309

Yao ZZ, Zhang GJ, Lu DJ, Liu H (2019) Data-driven crowd evacuation: a reinforcement learning method. Neurocomputing 366:314–327

Yao ZZ, Zhang GJ, Lu DJ, Liu H (2020) Learning crowd behavior from real data: a residual network method for crowd simulation. Neurocomputing 50:2633–2646

Zhang H, Liu H, Qin X, Liu BX (2018) Modified two-layer social force model for emergency earthquake evacuation. Phys A Stat Mech Appl 492:1107–1119

Zhao Y, Liu H, Gao KZ (2021) An evacuation simulation method based on an improved artificial bee colony algorithm and a social force model. Appl Intell 51:100–123

Zhou M, Dong HR, Ioannou PA, Zhao Y, Wang FY (2019) Guided crowd evacuation: approaches and challenges. IEEE/CAA J Autom Sin 6(5):1081–1094