Computers & Industrial Engineering 161 (2021) 107621

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Deep deterministic policy gradient algorithm for crowd-evacuation path planning

Xinjin Li [a,b], Hong Liu [a,b,*], Junqing Li [a,c], Yan Li [a,b]

[a] School of Information Science and Engineering, Shandong Normal University, Jinan 250014, China
[b] Shandong Provincial Key Laboratory for Distributed Computer Software Novel Technology, Jinan 250014, China
[c] College of Computer Science, Liaocheng University, Liaocheng 252059, China

ARTICLE INFO

Keywords:
Deep reinforcement learning
Multi-agent reinforcement learning
Path planning
Crowd simulation for evacuation

ABSTRACT

In existing evacuation methods, the large number of pedestrians and the complex environment will affect the efficiency of evacuation. Therefore, we propose a hierarchical evacuation method based on multi-agent deep reinforcement learning (MADRL) to solve the above problem. First, we use a two-level evacuation mechanism to guide evacuations, the crowd is divided into leaders and followers. Second, in the upper level, leaders perform path planning to guide the evacuation. To obtain the best evacuation path, we propose the efficient multi-agent deep deterministic policy gradient (E-MADDPG) algorithm for crowd-evacuation path planning. E-MADDPG algorithm combines learning curves to improve the fixed experience pool of MADDPG algorithm and uses high-priority experience playback strategy to improve the sampling strategy. The improvement increases the learning efficiency of the algorithm. Meanwhile we extract pedestrian motion trajectories from real motion videos to reduce the state space of algorithm. Third, in the bottom layer, followers use the relative velocity obstacle (RVO) algorithm to avoid collisions and follow leaders to evacuate. Finally, experimental results illustrate that the E-MADDPG algorithm can improve path planning efficiency, while the proposed method can improve the efficiency of crowd evacuation.

1. Introduction

Effectively organizing the evacuation of crowds from crowded places has always been an important issue. When dangerous accidents occur in crowded places, pedestrians tend to rush to escape the scene in panic to avoid the danger. In this case, the evacuation process is usually crowded, which may cause collision and trampling accidents resulting in secondary injuries (Zhang et al., 2020; Liu et al., 2018; Zhang et al., 2018). Therefore, it is challenging to plan evacuation paths in crowded places effectively, increase evacuation speed, and reduce the incidence of injuries (Peng et al., 2019; Liu et al., 2018). But, the evacuation of crowds is a complex process. The evacuation experiments have also been hampered by various problems such as difficult organizations, high costs, and personnel safety issues (Liu et al., 2018). Therefore, computer simulation technology has become the means of simulating evacuation processes and exploring how evacuations evolve (Tian et al., 2020). The social force model (SFM) and meta-automata model have universal application in the research of crowd evacuation simulation. But, SFM has many problems such as low efficiency and weak obstacle avoidance ability when carrying out crowd evacuation (Zhang et al., 2020; Zhang et al., 2018). The application of the meta-automata model in crowd evacuation requires the definition of complex rules and the definition of state update rules is more difficult. More importantly, SFM and meta-automata model don't consider the role of macro factors (Jiang et al., 2020). Therefore, none of these models can well solve the problem of large-scale crowd evacuation simulation.

Compared with SFM and meta-automorphic models, reinforcement learning (RL) is a macroscopic approach. RL has the advantages of simple rule definition, high computational power, high learning efficiency, and the ability to obtain optimization strategies through learning (Singh et al., 2021; Vinyals et al., 2019). Therefore, RL can better solve the problem of crowd evacuation simulation. Yao et al. (Yao et al., 2020) proposed a crowd evacuation method based on RL and data driven, which successfully applied RL to crowd evacuation. The method improves the realism of crowd evacuation, but suffers from the problem of inefficient evacuation. Bi et al. (Bi et al., 2019) proposed a RL-based

* Corresponding author at: School of Information and Engineering, Shandong Normal University, Jinan 250014, China and Shandong Provincial Key Laboratory for Distributed Computer Software Novel Technology, Jinan 250014, China.
E-mail address: lhsdcn@126.com (H. Liu).

https://doi.org/10.1016/j.cie.2021.107621

Available online 13 August 2021

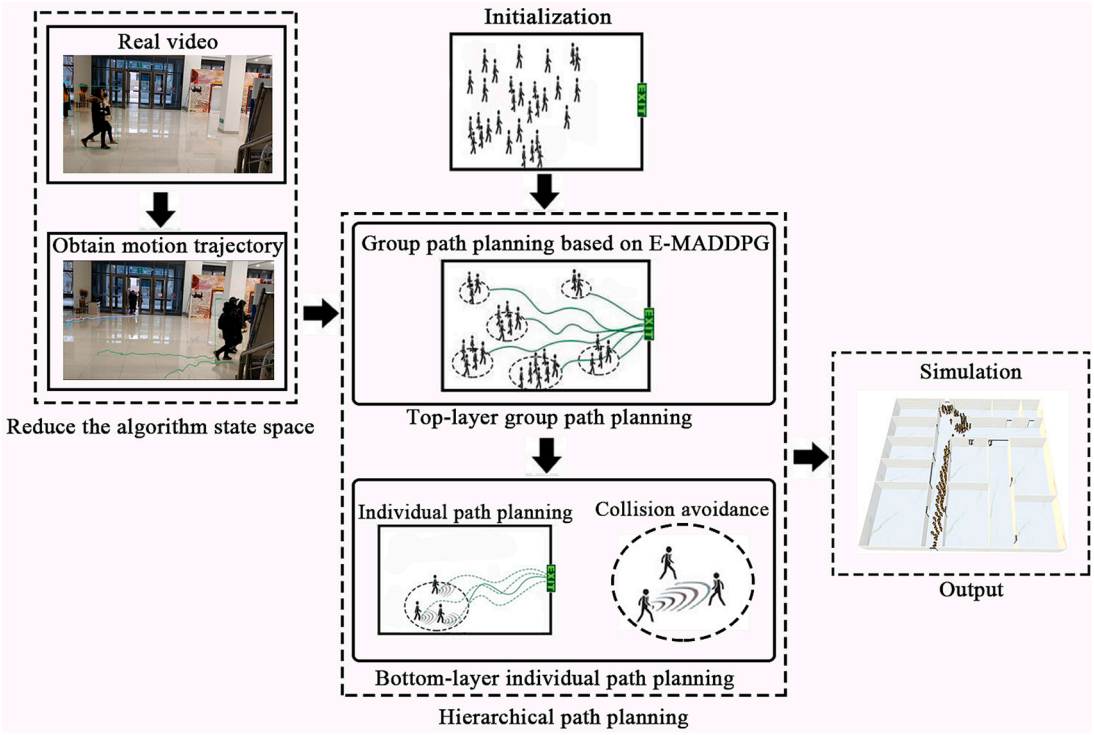

Fig. 1. Method framework.

evacuation path selection method, which improves the efficiency of model training and the overfitting problem in the learning process, and can generate the optimal evacuation path. However, the method suffers from problems such as inability to cope with complex environments. Current RL-based evacuation methods are commonly used in simpler environments, however, near-real-life evacuations are often accompanied by complex evacuation environments and large crowds, and it is clear that traditional RL cannot cope with this problem. To solve this problem, Zhang et al. (Zhang et al., 2021) introduced deep reinforcement learning (DRL) in crowd evacuation and proposed the Dyna-Q algorithm that combines the advantages of model-based RL methods and model-free RL methods, which can determine the most effective evacuation strategy based on cumulative rewards and can obtain the fastest evacuation paths in complex environments. Sharma et al. (Sharma et al., 2020) proposed a crowd evacuation method based on an improved DQN algorithm for real fire situations, which first pre-trained the network weights of DQN to integrate the shortest path for successful evacuation, and the method successfully achieved efficient evacuation in several complex environments. Both of these methods can be applied to more complex environments, but both consider only a small number of individual evacuations; real evacuations tend to have large crowds. The multi-agent deep deterministic policy gradient (MADDPG) algorithm proposed by Lowe et al. (Lowe et al., 2017) achieves better performance when searching for a global optimal solution for multi-agent. However, the learning efficiency of this algorithm is relatively low due to the unchangeable size of the experience pool and random experience replay. Zheng et al. (Zheng and Liu, 2019) proposed a crowd evacuation method based on an improved MADDPG algorithm to achieve larger scale crowd evacuation. However, the method also suffers from low learning efficiency and large state space. While DRL can cope well with crowd evacuation, the increasing population base and the increasing complexity of the environment still present a significant challenge. To achieve more realistic crowd evacuation, we must address the problems caused by the increasing population base and the increasing complexity of the environment.

To manage these problems, the E-MADDPG algorithm is proposed. Combined the MADDPG algorithm with the learning curve, the size of the algorithm experience pool increases with increasing learning experience to improve learning efficiency. Concurrently, the proposed algorithm uses the high-priority experience replay policy which improves learning effectiveness. Then, we reduce the state space by extracting pedestrian trajectories from real videos. Then, this study proposes a two-layer path planning method for crowd evacuations. Fig. 1 is the proposed method's framework. The first step is to set the initial parameters. The second step is to use our proposed hierarchical path planning method to guide evacuation. The proposed method divides pedestrians into leaders and followers. In the top layer, the leader uses the E-MADDPG algorithm for path planning. Concurrently, we extract pedestrian tracks from real videos to reduce the state space. In the bottom layer, followers use RVO algorithm to avoid conflicts and follow the leader to evacuate. The third step is to visualize the evacuation results. The following are the main contributions of the paper:

(1) The E-MADDPG algorithm is proposed. The combination of learning curve and high-priority experience replay strategy is adopted to improve the experience pool and sampling strategy of the MADDPG algorithm, thereby improving the learning efficiency algorithm.
(2) The state space of the algorithm is reduced. Extracting motion trajectories from real pedestrian videos reduces the state space and effectively solves the problem of dimensional disasters.
(3) A crowd evacuation method based on E-MADDPG is proposed, which uses a two-layer path planning method for crowd evacuations. This method can better guide the crowd evacuation in dangerous situations.

The rest of the paper is organized as follows: Section 2 contains knowledge and related studies. Section 3 contains related algorithms. Section 4 contains the E-MADDPG algorithm and the two-layer path planning method based on the E-MADDPG algorithm. Section 5 contains the experimental results and simulation results. Last, section 6 contains the conclusions and proposes future research directions.

2. Background and related studies

2.1. Path planning

Path planning is the primary topic of existing crowd evacuation simulation study. Swarm intelligence algorithms are widely for path planning (Zhao et al., 2021). Swarm intelligence is an artificial intelligence model proposed inspired by the phenomenon of intelligence exhibited by groups of organisms in nature. The swarm intelligence algorithm simulates the behavior of biological groups in nature, which makes the algorithm with good parallelism and autonomous exploration, and provides new ideas and means to solve complex problems (Li et al., 2021; Fan et al., 2013). Although the swarm intelligence algorithm may not be able to obtain an optimal solution quickly for a given problem, a solution can always be obtained by providing enough random solutions and iterations (Chen et al., 2016; Li, 2020; Li, 2021). Zhao et al. (Zhao et al., 2021) proposed the MABCM algorithm, which can obtain the optimal evacuation plan according to the change of the number of pedestrians during the evacuation process, markedly reducing the time required for crowd evacuation. Based on the PSO algorithm, Wong et al. (Wong et al., 2015) performed path planning with pedestrians by considering the average movement time and average distance of pedestrians to complete crowd evacuation path planning. Liu et al. (Liu et al., 2016) introduced quantum representation based on the AC algorithm and proposed the QAC algorithm, which updates the pheromones of ant colonies via quantum representation, and can avoid dangerous areas more effectively and plan paths more safely. Based on the ant colony algorithm, Goel et al. (Goel and Maini, 2018) introduced the firefly algorithm to explore the unsearched solution space, successfully solved the problem of a local optimal solution and was able to plan paths more effectively. Although swarm intelligence algorithms have been widely used in path planning, their application is limited by the need for more information, their poor adaptability to changes in the environment (Li, 2020), their high computation costs, their poor real-time performance and research stagnation.

Therefore, path planning methods based on RL have emerged. Compared to the traditional swarm intelligence algorithm, the RL algorithm can perceive the environment in real time, and can adapt to the dynamic environment somewhat. RL is more practical used in path planning. Zhao et al. (Zhao et al., 2018) proposed a path planning method of asynchronous RL based on a parallel architecture. This method can dynamically change its action strategy according to the environment, improve the learning efficiency and can realize the path planning in discrete space more accurately. Low et al. (Low et al., 2019) optimized the initialization process based on Q-learning, accelerated the convergence speed in a complex environment, and was able to plan paths in a complex environment more accurately. Kim et al. (Kim and Pineau, 2016) used an inverse RL algorithm to achieve universal path planning in dynamic environments. Yao et al. (Yao et al., 2019) proposed a method that combines RL and data-driven, that markedly improved the authenticity and efficiency of path planning. RL can perform path planning well, but in crowd evacuation, path planning is more complicated because there are many pedestrians. To apply RL to crowd evacuation simulation, cooperation among multi-agent must be considered.

2.2. Multi-agent deep reinforcement learning

RL can formulate an optimal strategy for a specific problem, so that the strategy obtains the optimal reward. But, the solution of many complex problems requires cooperation between multi-agent. For example, interactions between pedestrians must be considered in crowd evacuation simulations. Multi-agent reinforcement learning (MARL) used multi-agent system (MAS) to improve RL, realized information sharing among multi-agent, and solved the problem of cooperation among multi-agent (Buşoniu et al., 2008). It improves the cooperation and processing speed among agents, and performs effective information sharing among multi-agent. Cruz et al. (Cruz and Yu, 2017) used neural networks to estimate an unavailable state, improved the MARL algorithm, and realized path planning for an unknown environment. Cui et al. (Cui et al., 2020) successfully introduced shared space, enabling agents to share information and perform path planning more effectively. In crowd evacuation simulations, most evacuation scenes are complex scenes and similar to real scenes, and traditional RL methods cannot manage this problem effectively.

Hinton et al. (Hinton et al., 2006) proposed the concept of DL. The typical neural network structures for deep learning include convolutional neural networks (CNN) and deep belief nets (DBN) (Liu et al., 2011). Sun et al. (Sun et al., 2020) conducted an in-depth study of convolutional neural networks, and demonstrated a more efficient processing of high-dimensional input by convolutional neural networks compared to other networks. Traditional RL is difficult to managing high-dimension problems, while DL achieves superior performance with high-dimension problems. Therefore, DL can solve high-dimensional problems that are typically difficult for RL to solve.

Mnih et al. pioneered the deep Q-learning network (DQN) algorithm (Mnih et al., 2015), it introduces DL into RL and greatly improves the performance of the algorithm. But, the target Q value of DQN is directly

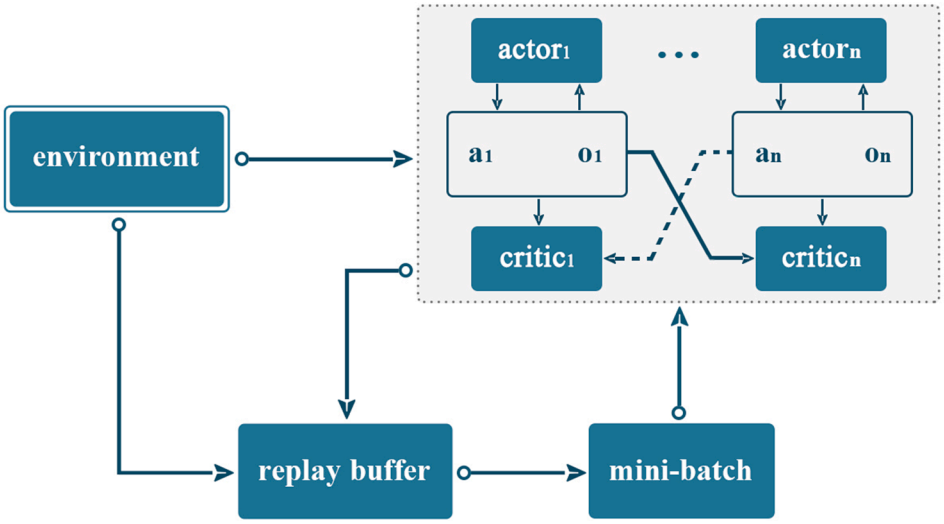

Fig. 2. Algorithm structure of MADDPG.

obtained via the greedy method, thus the Q value estimation will be high (Wang et al., 2016). Van et al. (Van Hasselt et al., 2016) proposed the double DQN (DDQN) algorithm, which achieved the selection and evaluation of actions with different value functions, and successfully mitigated issues with excessive Q. Based on maintaining the primary network unchanged, Wang et al. (Wang et al., 2016) changed the single output in the original network to two outputs, and obtained a more accurate Q. But, these algorithms can only output discrete actions, not continuous actions. Lillicrap et al. (Lillicrap et al., 2016) proposed the DDPG algorithm, which can output continuous actions based on the DPG algorithm (Wang et al., 2020) and the advantage of the single-step update of the policy gradient of actor-critic (Bhatnagar et al., 2009). Different from DQN and DDQN, DDPG is a policy-based learning algorithm. DDPG uses a deterministic policy to select actions, thus the output is not the probability of the behavior but the specific behavior. But, the DDPG algorithm cannot solve complex tasks based on multi-agent. Then, Lowe et al. (Lowe et al., 2017) proposed MADDPG algorithm which improves the critic part, so that the critic part of each agent can be aware of the action information of all other agents, thus MADDPG can adapt to the multi-agent environment. Zheng et al. (Zheng and Liu, 2019) introduced the mean field network based on the MADDPG algorithm to maximize the performance of all agents in the cooperative planning task during the training period, but the resulting efficiency was low. Zhang et al. (Zhang et al., 2020) used group target network smoothing and dual-centered critic to improve MADDPG, reducing network errors and optimizing the final strategy. This process improved system adaptability to manage complex tasks effectively, but reduced the learning efficiency.

3. Preliminary

This section introduces the algorithms used in this paper, primarily including the MADDPG algorithm, learning curve, and RVO.

3.1. MADDPG algorithm

In a complex environment, the RL algorithm based on single agent cannot share information with other agents, which often leads to failure of the training model. To realize the information sharing between agents, the actor-critic structure of algorithm must be improved. The algorithm structure of MADDPG is shown in Fig. 2. Among them, actor and critic are the network of algorithm. $\{a_1 \cdots a_n\}$ represents the actions of n agents. $\{o_1 \cdots o_n\}$ represents the state of n agents. *mini* $-$*batch* is the number of samples to be sampled.

MADDPG is the centralized-training and decentralized-execution modes. In centralized-training mode, the input to the critic network of agents include the action state information. In decentralized-execution mode, the input of each agent's actor network is the agent's own information. Because the centralized critic network uses the strategies of other agents to improve the efficiency of learning, the agents can collaborate to solve problems more effectively.

3.2. Learning curve

The learning curve is a dynamic evaluation technique, which refers to the change curve, in which the person-hours used to manufacture a single product gradually decrease as the production increases over many production cycles (Saraswat and Gorgone, 1990). The learning curve reflects the process in which learning efficiency increases as experience increases. The learning curve typically has two definitions: broad and narrow. The generalized learning curve refers to the life cycle curve of a product or a certain industry. This curve is a learning curve that integrates production technology and management level. A narrow learning curve refers to the learning curve of individual operators only, which is introduced in detail in 4.1.1.

3.3. RVO algorithm

The RVO algorithm is obtained based on improved speed obstacle algorithm. By adjusting the speed of individuals, collisions between any individuals in the scene can be effectively avoided. The algorithm reduces the degree of speed deviations, ensures the avoidance of local obstacles without jitter between groups, and can more realistically simulate collision avoidance between multiple individuals in dense and complex scenes. RVO can avoid conflicts between agents and does not require global coordination(Den Van Berg et al., 2008). This algorithm abstracts the object into a circular object with a radius and a speed, which are used to predict whether two objects collide. If the prediction result is that there will be a collision, their directions will change to avoid obstacles. The RVO algorithm is described as follows. For individuals M and N, $VO^{\sigma}_{M|N}$ is a relative space of individual M with respect to the speed V_M of individual N when the speed of N is V_N, thus M and N collide within time σ:

$$D(c,r) = \{q||c-q|| < r\} \tag{1}$$

where $D(c,r)$ is a circle with c as the center and r is the radius. q is a variable that represents the range of distance where the collision will occur.

$$VO^{\sigma}_{M|N} = \{v|\exists t \in [0,\sigma], t^*v \in D(p_N - p_M, r_M + r_N)\} \tag{2}$$

The velocity obstacle (VO) is defined as follows. Assuming that the speeds of individual M and individual N are V_M and V_N, respectively, the speed obstacle can be expressed as $V_M - V_N \in VO^{\sigma}_{M|N}$; thus, when individuals M and N are moving at their current speeds, individual M and individual N will collide within time σ.

$M \oplus N$ is expressed as $M \oplus N = \{m+n|m \in M, n \in N\}$, and the speed set of individual N is V_N. If $v_N \in V_N \wedge v_M \notin VO^{\sigma}_{M|N} \oplus V_N$, then during time σ, M and N are not going to collide, thus a collision avoidance set $C^{\sigma}_{\mathrm{M|N}}$of M and N can be obtained:

$$C^{\sigma}_{M|N} = \{v|v \notin VO^{\sigma}_{M|N} \oplus V_N\} \tag{3}$$

The RVO algorithm only obtains the speed when moving based on avoiding collisions between individuals. If the planned paths are not given, the individual may not reach the predetermined exit position smoothly, or may not reach the predetermined exit within the optimal time.

4. Crowd evacuation method based on DRL

This section introduces the E-MADDPG algorithm, then introduces the proposed method of reducing the state space, and finally introduces the two-layer path planning method.

4.1. E-MADDPG algorithm

Firstly, we introduce the improvement of algorithm that using the learning curve and then introduce the meaning of "priority experience replay" and its combination with the algorithm. Finally, the E-MADDPG algorithm is described in detail.

4.1.1. Dynamic capacity experience pool

The RL system gains experience and improves the action plan by adapting to environment. Through the learning curve, we can conclude that with more knowledge, the learning efficiency will increase. The fixed capacity experience pool in the MADDPG algorithm will undoubtedly affect the learning efficiency. Therefore, we can combine the learning curve with the MADDPG algorithm. Considering the influence of experience growth on the learning efficiency, the size of the experience pool depends on the number of learning times, so that the experience pool becomes dynamically variable, and the number of sampling

samples changes accordingly. The learning curve equation is as follows:

$$y(x) = kx^{\alpha} \tag{4}$$

$$\alpha = lgx/log2 \tag{5}$$

where x represents the x^{th} learning, k represents learning effect, $y(x)$ represents the x^{th} learning's time, and α is the learning coefficient.

This paper refers to the relationship between production and time in the learning curve, and adds change parameters β to the experience pool capacity and the number of samples in the algorithm. The algorithm's experience pool can be improved by combining the parameter β. We adjust the size of the experience pool through the parameter β, so that the size of the experience pool changes dynamically with the learning process. Therefore, the impact of an experience pool that is too small or too large on the learning efficiency during the learning process is eliminated. The change function after improvement is as follows:

$$\beta = t^{(lgt/log3)} \tag{6}$$

$$R(t+1) = R(t)*t^{(lgt/log3)} \tag{7}$$

where $R(t)$ is the current size, t is the number of learning times.

Similarly, the number of samples sampled can be improved. As learning progresses, the number of samples increase, and a fixed number of sample collections will inevitably affect the efficiency of learning. Similarly, we adjust the number of samples by changing the parameter β, and the change function after improvement is as follows:

$$N(t+1) = N(t)*t^{(lgt/log3)} \tag{8}$$

where $N(t)$ is the current number.

4.1.2. Experience replay

Traditional online RL agents update their parameters incrementally through experience. And they discard incoming data after update. But it also brings some problems, such as quickly throw away rare experiences that might be useful later. Experience replay solves these problems: by storing the experience in the experience pool, temporal correlation can be broken by mixing more or less updated experiences, while rare experiences will be used for more than one update. By using more computation and memory, experience replay can reduce the amount of experience required for learning.

But traditional experience replay of random sampling reduces the efficiency of learning. To mitigate this issues, we select priority experience replay which is to replay successful attempts or poorly performing samples more frequently, thereby improving the learning efficiency. The idea of prioritized experience replay comes from prioritized cleaning, which replays samples that are more useful for learning at a high frequency. This study uses the TD-error to measure the size of the sample value. The absolute value of the TD-error $|\delta_t|$ is used as the standard for evaluating the sample value.

$$|\delta_t| = r(s_i, a_i) + \tau Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'}) - Q(s_i, a_i|\theta^{Q}) \tag{9}$$

where $r(s_i, a_i)$ is a function of reward, $\tau \in (0,1)$is a discount parameter, $Q'(s, a|\theta^{Q'})$ is the target action value network, $Q(s, a|\theta^{Q})$ is the action value network, $\mu(s|\theta^{\mu})$ is the actor network, and θ^{Q} and θ^{μ} are the network parameters.

Experience replay frees online learning agents from processing transitions in the exact order they go through. Priority experience replay further liberates the agent from having to consider transitions at the same frequency. TD-error is the difference between the action's estimated value and the action's output value. Therefore, a sample with a larger TD-error value has a higher value. Replaying these high-value samples more frequently can help the agent improve the effectiveness of learning, thereby improving the overall performance, and "learning more from the sample". Therefore, priority experience replay can improve the learning efficiency.

4.1.3. E-MADDPG algorithm

We define the elements of proposed method, as shown below:

Definition 1. (*(State)*) *Denoted by S, where $s_t \in S$, the state represents the pedestrian's position at time t. There are two ways to express the time unit: seconds and frames. During learning, S includes the leader's current position and the set of inflection points and the straight points of the path.*

Definition 2. (*(Action)*) *Denoted by A, where, $a_t \in A$, the action represents that which the agent selects the next state to be according to the current state. $S_{t+1} = a(S_t)$, S_{t+1} represents that the agent's state at the next moment.*

Definition 3. (*(Reward Function)*) *Denoted by R, represents the instantaneous reward for the action after the environment executes action a. A good reward function is very significance for RL. For proposed method, an excellent reward function can better plan the evacuation path and reduce the evacuation time. In multi-agent path planning, two primary tasks must be completed: reaching the destination and avoiding collisions between agents. The agent position is expressed as:*

$$\left\{(\boldsymbol{x}_1, \boldsymbol{y}_1)\cdots(\boldsymbol{x}_i, \boldsymbol{y}_i), (\boldsymbol{x}_j, \boldsymbol{y}_j)\cdots(\boldsymbol{x}_n, \boldsymbol{y}_n)\right\} \in (\boldsymbol{x}, \boldsymbol{y}) \tag{10}$$

We define some distance formulas as follows:

$$\boldsymbol{d}_{dist} = \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^2 + (\boldsymbol{y}_i - \boldsymbol{y}_j)^2} \tag{11}$$

$$\boldsymbol{d}_{obs} = \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_k)^2 + (\boldsymbol{y}_i - \boldsymbol{y}_k)^2} \tag{12}$$

$$\boldsymbol{d}_{min} = \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_g)^2 + (\boldsymbol{y}_i - \boldsymbol{y}_g)^2} \tag{13}$$

where d_{dist} is the distance between the two agents. d_{obs} is the distance from the agent to the obstacle, the obstacle position is (x_k, y_k), $k \epsilon (1, n)$; d_{min} represents the minimum distance from the agent to the exit, the exit position is $\left(x_g, y_g\right)$, $g \epsilon (1, m)$.

Based on the two goals of reaching the destination and avoiding collision, this algorithm use the distance from the agent to the exit to represent the reward of reaching the destination: $R_g = -min(d_{min})$. Avoiding collisions are divided into avoiding collisions with obstacles and avoiding collisions with other agent. We define r is the radius of the agent. When $d_{dist} \leq 2r$, we assume that two agents collide, and thus, $R_{c1} = -1$. When $d_{obs} \leq r$, we assume that the agent collides with the obstacle, and thus, $R_{c2} = -1$. Thus, this study defines the reward function as:

$$\boldsymbol{R} = \boldsymbol{R}_g + \boldsymbol{R}_{c1} + \boldsymbol{R}_{c2} \tag{14}$$

Algorithm 1 E-MADDPG algorithm.

Input: agent number *i*, exit position $\left(x_g, y_g\right)$, obstruction position(x_k, y_k)

Output: Reward

1: Randomly initialize the actor network and the critic network: θ^{Q} and θ^{μ};
2: Initialize the target network : $\theta^{Q'}$ and $\theta^{\mu'}$;
3: Initialize the buffer D, hits N, the TD-error absolute minimum value $|\delta_{min}|$, and pointer P = 1;
4: **for** episode = 1, M **do**
5: Initialize a random process Φ;
6: Receive the initial observed status s_1;
7: **for** t = 1, T **do**
8: For $agent_i$, select actions based on the current policy and the explore noise $a_t = \mu(s_t|\theta^{\mu}) + \Phi_t$;
9: Perform action a_t, and return the reward value r_t and the new state s_{t+1};
10: Store the experience $e_t = (s_t, a_t, r_t, s_{t+1})$ to replay buffer D;
11: $D(t+1) = D(t)*t^{(lgt/log2)}$;

(*continued on next page*)

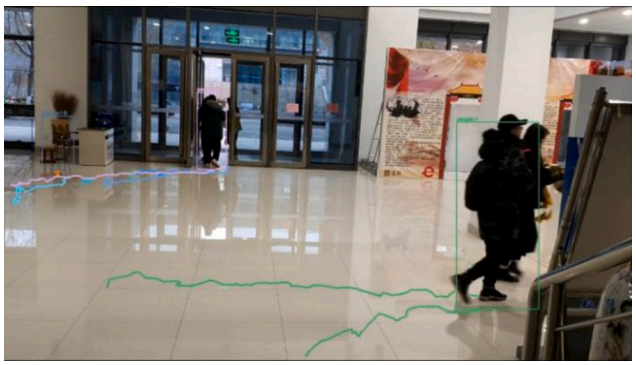

Fig. 3. Obtained motion trajectory from a video of pedestrians.

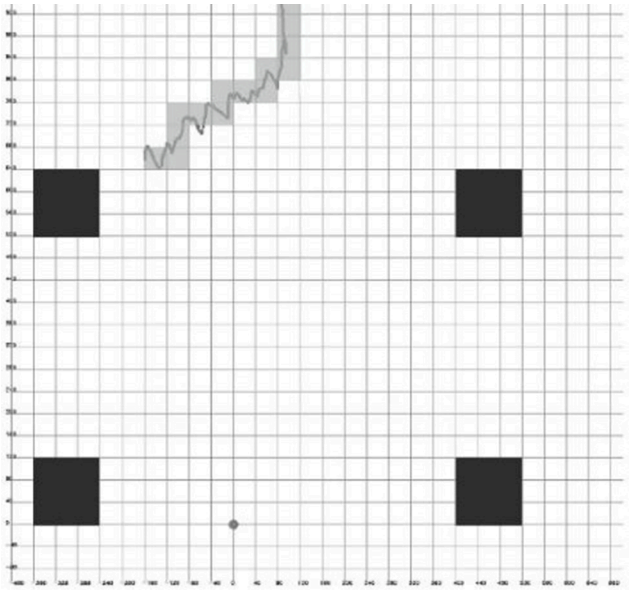

Fig. 4. Extracted straight points and inflection points from motion trajectories.

(*continued*)

Algorithm 1 E-MADDPG algorithm.

```
    Calculate the |δ_t| of sample e_t;
    if|δ_t|>|δ_min|, then;
      Insert into the e_t, query TD-error minimum value update |δ_min|;
      P = P + 1;
    end if
    for agent_i = 1, X do
      Take N samples from D;
      N(t+1) = N(t)*t^(lgt/log2);
      Set y_i = r_i + γQ'(s_{i+1}|θ^μ')|θ^Q';
      Through the minimum loss function L training critic network:
      L = (1/N)Σ[y_i − Q(s_i, a_i|θ^Q)]^2;
      Through the gradient policy to update the actor policy:
      ∇_{θ^μ} ≈ (1/N)Σ ∇_a Q Q(s, a|θ^Q)|_{s=s_i, a=μ(s_i)} ∇_{θ^μ} μ(s|θ^μ)|_{s_i};
    end for
    Update the target network;
  end for
end for
```

4.2. Reduced state space

A good state design can speed up the convergence speed. With the expansion and refinement of the region, it will certainly bring a very large state space, which is the so-called "dimensional disaster". Therefore, this paper proposes a method to reduce the state space. This paper uses YOLO V3 method to extract motion trajectory of pedestrian from pedestrian video. The video this study has used comes from a pedestrian motion video taken by surveillance in the lobby of the school's teaching building. We have collected these pedestrian motion videos by installing cameras in the lobby of the teaching building. As shown in Fig. 3, the multiple color curves are the pedestrian motion trajectory. Thus, according to the straight point and inflection point of the pedestrian trajectory in the video, the corresponding state is obtained, as shown in Fig. 4. The gray squares in figure are the state change points that are obtained via smoothing. We obtain the real situation of pedestrian movement in this scene through video. Therefore, when we are designing the state, we can reduce the state that the pedestrian does not reach in the real situation. In this way, proposed method can reduce the state space and improve the efficiency of learning.

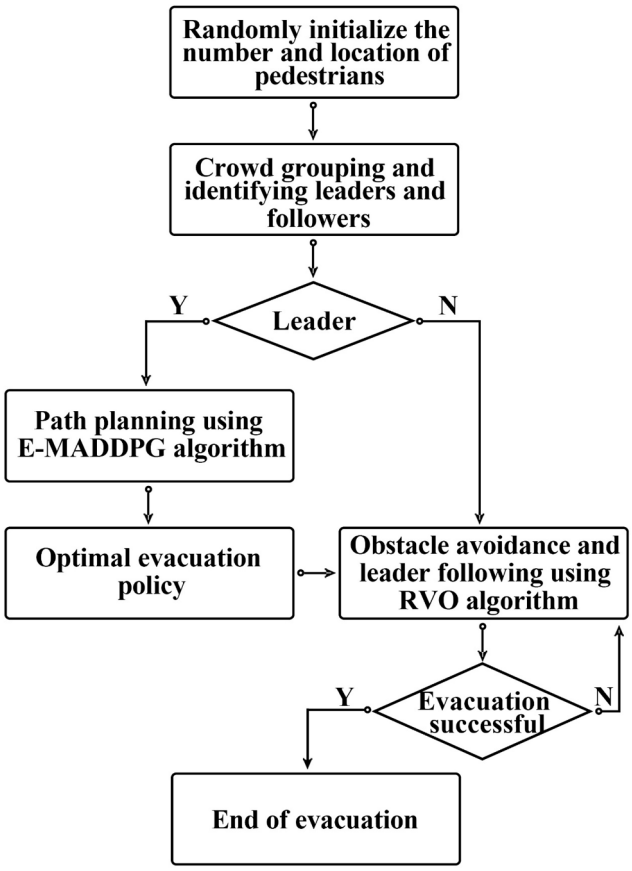

Fig. 5. Hierarchical evacuation process.

4.3. Hierarchical path planning method

The proposed method divides the crowd evacuation process into two parts: path planning and avoid collisions. At the top layer this method uses the E-MADDPG algorithm for path planning, obtain the optimal evacuation route. At the bottom layer, this method uses the RVO algorithm for collisions avoiding. The evacuation process of the method is shown in Fig. 5. First, the method randomly initializes the number of pedestrians in the crowd and the location of the crowd. Second, the crowd is grouped, and each group has a leader and several followers. After the crowd is grouped the method only needs to perform path planning for the leader, while the followers follow the leader for evacuation. Third, the method uses the E-MADDPG algorithm to perform path planning for the upper leaders to derive the set of paths where the leader evacuates successfully, and then traverses all paths to select the optimal path as the evacuation path for that leader. Fourth, the method uses the RVO algorithm to drive the followers to follow the

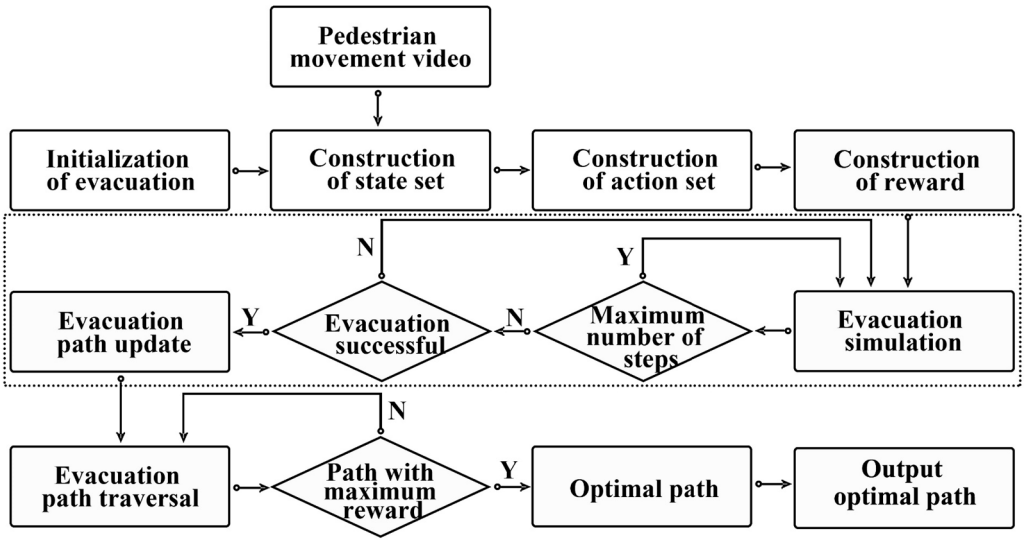

Fig. 6. Path planning process.

corresponding leader to evacuate according to the optimal evacuation path to avoid collision. Finally, the evacuation ends when all groups are successfully evacuated.

4.3.1. Path planning based on E-MADDPG

In the process of top-level path planning, model training is first required. This method uses the E-MADDPG algorithm for model training. We stipulated the total number of training times *num_episodes* and the maximum length *episodes* of a single time. Then start training. After choosing action a, agent i goes from the state s_t to the state s_{t+1}, and simultaneously obtains a new reward r. (s_t, a, r, s_{t+1}) of each step is stored in the experience pool, and then the statess_{t+1} is needed update. Concurrently, the newly generated batch(s_t, a, r, s_{t+1}) is stored. After the training is completed within the total number of steps, the trained model can be obtained.

After completing the model training, we can get the path. Fig. 6 shows the process of path planning for leaders. Next, we describe the path planning process: When the exit position p_j and leader $i$the initial position p_i^t are known, we perform the operation to reach the next position p_i^{t+1}. Then continue to perform the operation at the current position, and repeat the operation until the leader reaches the exit positionp_j. This group of position sequences are considered to be one of the path of leader$i$. Then the k paths that have been successfully evacuated are stored in the buffer$Buffer_i$. But, due to the concurrent effects of other agents, the reward of each path will be different. In this paper, the reward value R_i^k is obtained for a set of position sequences in k paths in the buffer. When traversing k paths in the $Buffer_i$, the method select one path with the largest reward R in the path buffer as the optimal evacuation path for leader i, and output the path set Path_i.

Algorithm 2 Path planning based on E-MADDPG algorithm.

Input: leader i initial position p_i^t, Exit location p_j, Path buffer $Buffer_i$**Output:** Path buffer $Path_i$

1: **for**i = 1, n **do**
2: **for** j = 1, m **do**
3: **repeat**
4: $p_i^t = p_i^{t+1}$;
5: $buffer_i^k = buffer_i^k \cup p_i^t$;
6: until $p_i^t = p_j$;
7: k++;
8: **end for**
9:**end for**
10: **for** l = 1, k **do**
11: **if** $R_i^k > R_i^{max}$;

(*continued on next column*)

(*continued*)

Algorithm 2 Path planning based on E-MADDPG algorithm.

12: $buffer_i^{max} = buffer_i^k$
13: **end if**
14: **end for**
15:$Path_i = Buffer_i^{max}$;

4.3.2. Collision avoiding based on RVO

In the lower-layer, the proposed method first proposes rules for leaders and followers: there is no requirement for the position of the leader; but, the leader must know the position of the exit. During the evacuation process, the follower should always follow the leader. Then, RVO is used to realize the process of pedestrian avoidance, which can be realized in two steps. First, the proposed method calculate speed v_c of the collision between individual i and individual j in the neighborhood. Then, calculate the collision-free speed of the individual i and select the best collision-free speedv_b, Among them, the best speed v_b is a vector with direction and magnitude; its direction is$d_i = p_i^{t+1} - p_i^t$, p_i is the optimal path of the leader obtained; and p_i^t is the position of leader $i$at time t. Once the individual's collision-free velocity is obtained, its position is updated to $x_i^{t+1} = x_i^t + \Delta t^* v_b$, and x_i^t is the current location. Thus, the proposed method makes the movement of all pedestrians are based on the optimal path of the top-layer path planning training.

5. Experimental analysis and simulation

To make the experiment more real, we set up a camera in the lobby of the teaching building to collect pedestrian motion videos. The pedestrian motion video is processed as experimental data. Then, based on the contribution of this study, we set up different experiments to verify the innovation of this study:

(1) The experiment in Section 5.1 illustrates that the proposed algorithm achieves a higher learning efficiency and can converge more quickly than other methods. Additionally, due to the effective reduction of state space, the proposed algorithm avoids unnecessary learning and can obtain higher rewards.
(2) The experiment in Section 5.2 illustrates that the propose method can effectively manage large-scale crowd evacuations in obstacle-free scenes.
(3) The experiment in Section 5.3 illustrates that in a scene with obstacles, the proposed method can plan an evacuation path

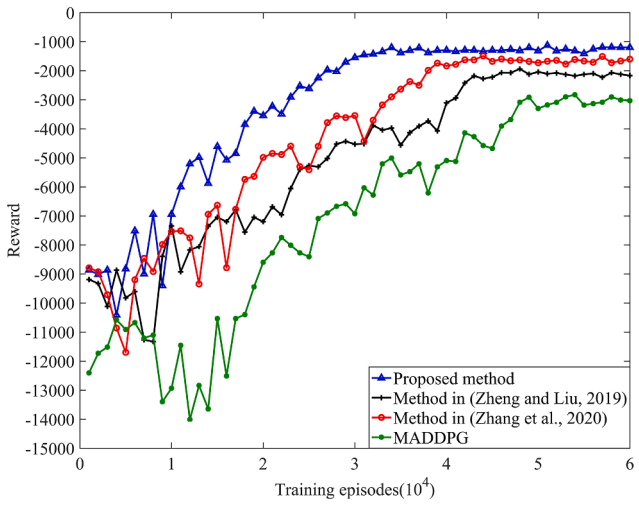

Fig. 7. Comparison of reward values for different training times.

more reasonably and that effectively improve the efficiency of evacuation.

(4) The experiment in Section 5.4 illustrates that the strong robustness of proposed method and the proposed method can complete the evacuation well when the number of exits changes.

In Section 5.5, we intuitively demonstrated the simulation results of the proposed method.

5.1. Comparison of reward values of different algorithms

Firstly, we modeled the teaching hall based on the real video, and set the corresponding scene in the Open AI Gym. Then we set up an experiment to verify the performance of the E-MADDPG algorithm. In the training scene, we set the size of population is 100, the radius of pedestrians is 0.2 m, and the mass of pedestrians is 60 kg. Then, we respectively use the E-MADDPG algorithm, the IMADDPG algorithm proposed in (Zheng and Liu, 2019), the improved MADDPG algorithm in (Zhang et al., 2020), and the MADDPG algorithm for path planning. Finally, we record the average reward value of the training results of four algorithms between 10,000 and 60,000 iterations.

As shown in Fig. 7, the blue line is the value of the proposed algorithm in this paper; the black line represents the improved MADDPG algorithm proposed in (Zheng and Liu, 2019); the red line represents the algorithm proposed in (Zhang et al., 2020) after the improvement of the MADDPG algorithm; and the green line represents the MADDPG algorithm. Because obstacles and agents are both present, the reward value obtained through the reward function is negative. From Fig. 7 (Table 1) the E-MADDPG algorithm's convergence speed is shown to differ significantly from that of the other methods, and the reward value after convergence is higher than that of the other methods. The figure also shows that the E-MADDPG algorithm has converged after 30,000 steps, and we can clearly see that it's value is higher than those of the other algorithms. The algorithm in (Zhang et al., 2020) is also near convergence after 38,000 steps of training; but, the convergence speed of it is significantly lower than the E-MADDPG algorithm, and the reward value after convergence is approximately 400 lower than the proposed algorithm. The algorithm in (Zheng and Liu, 2019) is also near convergence after 42,000 steps of training; but, the convergence speed is lower than that of E-MADDPG algorithm, the value after convergence is approximately 1000 lower than that of E-MADDPG algorithm. The MADDPG algorithm fluctuates significantly in the early stage. Although this algorithm tends to converge at 52,000 steps, its value is much lower than others. These results illustrate that E-MADDPG algorithm has higher learning efficiency and can more effectively to plan path. Therefore, the proposed algorithm achieves higher return values and achieves faster convergence speeds with the same number of training steps. With the same reward function, the proposed algorithm can plan better and faster paths in complex environments. Therefore, the proposed algorithm has a higher learning efficiency.

5.2. Comparison of different evacuation methods in obstacle-free scenes

In an obstacle-free scene with two exits, we respectively use the method based on the E-MADDPG algorithm, the method proposed in (Yao et al., 2019), the method based on the DDQN algorithm (Van Hasselt et al., 2016) and the method proposed in (Tian et al., 2020) to evacuate pedestrians. We first compare the evacuation efficiency of different methods by comparing the time to complete the evacuation of different methods. Then we compared the obstacle avoidance ability of

Table 2
Different methods for successful evacuation time in an obstacle-free scene.

Number of pedestrians	100	200	300	400	500	600
Proposed method	25.3 s	32.3 s	40.9 s	56.6 s	71.9 s	86.6 s
Method in (Yao et al., 2019)	25.5 s	32.6 s	41.8 s	58.5 s	74.6 s	89.8 s
DDQN	25.6 s	32.5 s	42.3 s	61.1 s	77.9 s	93.4 s
Method in (Tian et al., 2020)	25.5 s	32.7 s	43.4 s	60.3 s	76.1 s	91.5 s

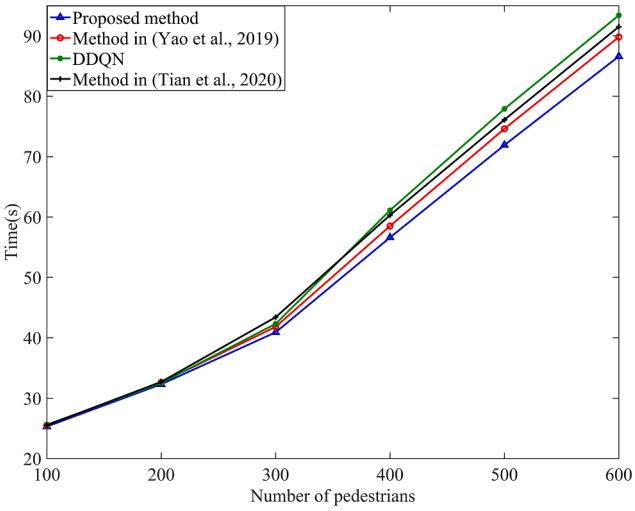

Fig. 8. Different methods for successful evacuation time in an obstacle-free scene.

Table 1
Comparison of reward values for different training times.

Training episodes	1000	10,000	20,000	30,000	40,000	50,000	60,000
Proposed method	−8863.46	−6943.74	−3546.9	−1546.52	−1295.17	−1317.74	−1198.79
Method in (Zheng and Liu, 2019)	−9188.28	−7344.76	−7197.9	−4528.26	−3109.87	−2043.75	−2168.31
Method in (Zhang et al., 2020)	−8788.06	−7534.53	−4985.27	−3545.11	−1834.97	−1728.58	−1603.37
MADDPG	−12402.5	−12931.4	−8596.42	−6421.41	−5093.69	−3300.94	−3033.47

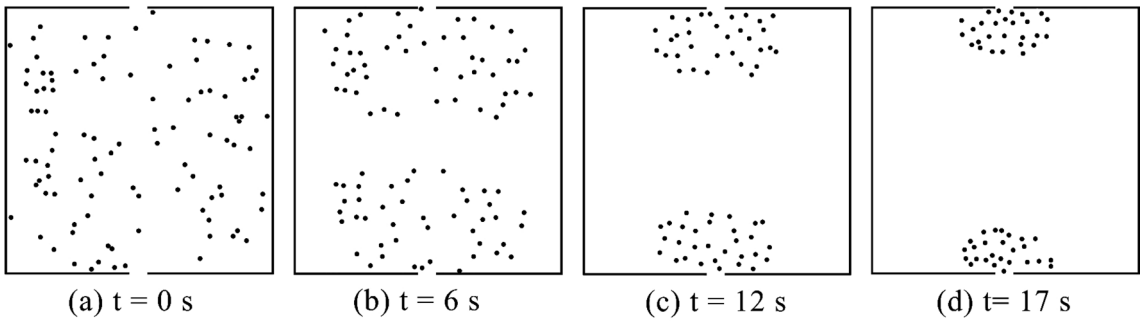

Fig. 9. Simulation based on the E-MADDPG method.

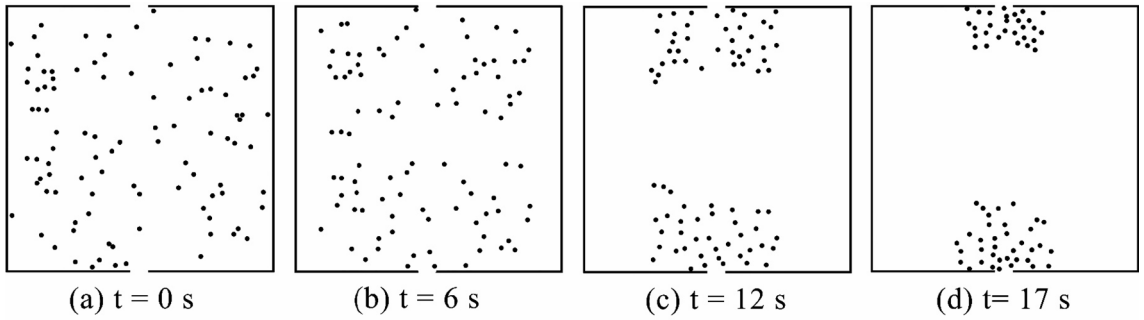

Fig. 10. Simulation based on the method in (Yao et al., 2019).

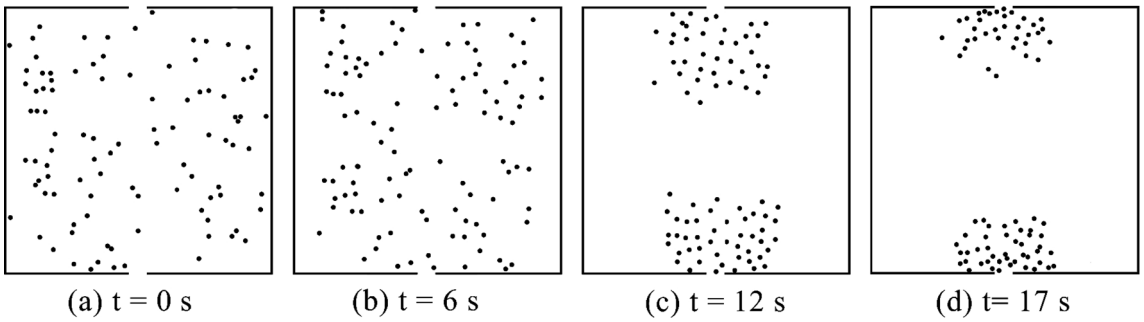

Fig. 11. Simulation based on the DDQN method.

different methods by simulating the evacuation process of 100 pedestrians. Finally, we set up an evacuation experiment with 300 pedestrians to compare the crowding situation during evacuation by comparing the change of the number of pedestrians during evacuation by different methods.

First, in an obstacle-free scene with two exits, we set up respectively evacuation experiments for 100 to 600 people to compare the evacuation times of the four methods. Table 2 and Fig. 8 show the evacuation time of above crowd evacuation methods. In Fig. 8, the blue line represents the time taken of the evacuation by the proposed method; the red line represents the time taken of the evacuation using the method in (Yao et al., 2019); the green line represents the time taken of the evacuation by the method based on the DDQN algorithm; and the black line represents the time taken of the evacuation using the method in

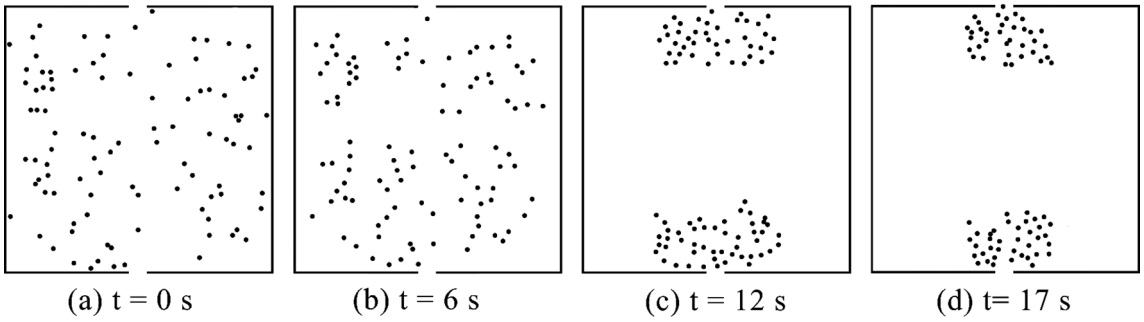

Fig. 12. Simulation based on the method in (Tian et al., 2020).

Table 3
Changes in the number of pedestrians during the evacuation process in an obstacle-free scene.

Time(s)	0	10	20	30	40	41	50
Proposed method	300	240	190	90	3	0	0
Method in (Yao et al., 2019)	300	241	185	100	8	3	0
DDQN	300	241	200	116	13	6	0
Method in (Tian et al., 2020)	300	243	193	104	10	4	0

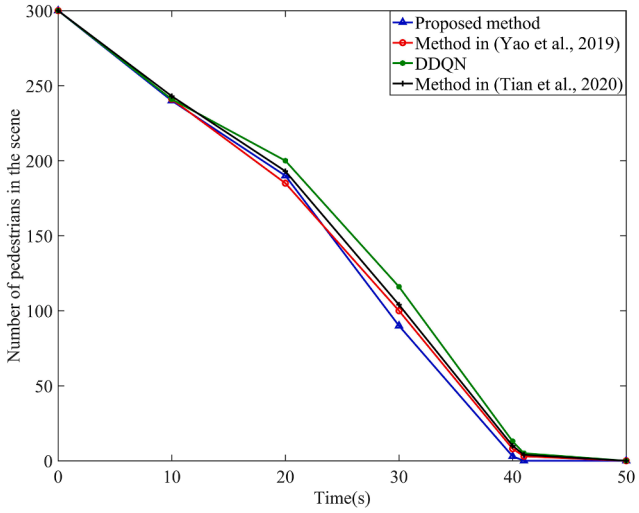

Fig. 13. Changes in the number of pedestrians during the evacuation process in an obstacle-free scene.

(Tian et al., 2020). Because there are no obstacles in this scene, no congestion collisions occur when the population is small, and the evacuation time of the four algorithms are relatively close. But, as the population size increases, congestion and collisions inevitably occur, and the evacuation time of the four algorithms begins to diverge. It can be seen from the figure that as the number of pedestrians increases, the time to complete the evacuation increases for the four methods. However, the increase of the proposed method is much smaller compared with the other three methods, which indicates that the proposed method has a more reasonable path planning. In the experiment of evacuating 600 people, the time to complete the evacuation by the method is significantly less than the other three methods, which indicates that the proposed method can better solve the large-scale crowd evacuation problem. Therefore, the proposed method can effectively complete crowd evacuation of different scales in a simple obstacle-free scenario.

Then, to test the ability of pedestrians to avoid collisions, we used four methods to simulate an evacuation with 100 pedestrians, as shown in Fig. 9, Fig. 10, Fig. 11 and Fig. 12. As can be seen from the figures, the evacuation speed of the proposed method is higher than the other three methods. During evacuation, the method in (Yao et al., 2019) and the DDQN-based method both experience collisions with building, while the proposed method effectively avoids collisions with building. In crowded situations, the proposed method does not cause pedestrian collisions during evacuation, while several other methods cause collisions during evacuation. Therefore, the method can better avoid collisions.

Finally, we evacuated 300 pedestrians in an obstacle-free scene using different evacuation methods, and then compared the change in the number of people stranded indoors during the evacuation. Table 3 and Fig. 13 show number of pedestrians stranded indoors during the evacuation process when 300 pedestrians are evacuated in an obstacle-free scene. It can be seen from the figure that the proposed method completes the evacuation faster than the other three methods, so the proposed method evacuates better than the other three methods. In addition, the number of successful evacuees per unit time in the evacuation process of the proposed method is relatively stable, which indicates that the path planning of the proposed method is more reasonable and no serious congestion and blockage will occur; therefore, the evacuation process of the proposed method is more stable than the other three methods and can complete the evacuation more safely and effectively.

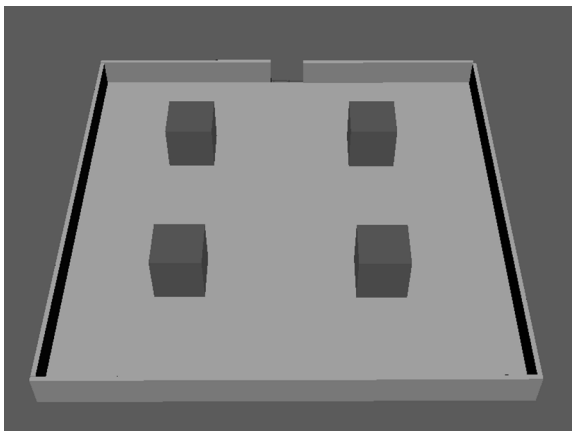

Fig. 14. Scene with one exit and four obstacles.

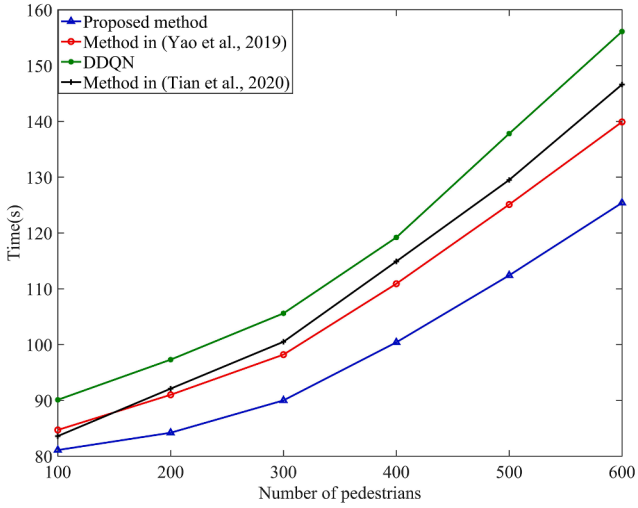

Fig. 15. Different methods for successful evacuation time in a scene with obstacle.

5.3. Comparison of different evacuation methods with obstacles

In a scene with one exit and four obstacles, as shown in Fig. 14, we respectively use the method based on the E-MADDPG algorithm, the method proposed in (Yao et al., 2019), the method based on the DDQN algorithm (Van Hasselt et al., 2016) and the method proposed in (Tian et al., 2020) to evacuate pedestrians. We first compare the evacuation efficiency of different methods by comparing the time to complete the evacuation of different methods. Then we compared the obstacle avoidance ability of different methods by simulating the evacuation process of 100 pedestrians. Finally, we set up an evacuation experiment with 300 pedestrians to compare the crowding situation during evacuation by comparing the change of the number of pedestrians during evacuation by different methods.

First, in a scene with obstacle, we set up respectively evacuation experiments for 100 to 600 people to compare the evacuation times of the four methods. Experimental results are shown in Fig. 15 (Table 4),

Table 4
Different methods for successful evacuation time in a scene with obstacle.

Number of pedestrians	100	200	300	400	500	600
Proposed method	81.1 s	84.2 s	90 s	100.4 s	112.4 s	125.4 s
Method in (Yao et al., 2019)	84.7 s	91 s	98.2 s	110.9 s	125.1 s	139.9 s
DDQN	90.1 s	97.3 s	105.6 s	119.2 s	137.8 s	156.1 s
Method in (Tian et al., 2020)	83.6 s	92.1 s	100.5 s	114.9 s	129.5 s	146.6 s

the blue line represents the time taken of the evacuation by the proposed method; the red line represents the time taken of the evacuation using the method in (Yao et al., 2019); the green line represents the time taken of the evacuation by the method based on the DDQN algorithm; and the black line represents the time taken of the evacuation using the method in (Tian et al., 2020). When the number of pedestrians is small, the evacuation times of the four methods are close because no serious congestion occurs; however, the evacuation times of the proposed method are lower than those of the other methods. As the crowd increases, the evacuation times of the four methods differ: the evacuation time of the proposed method grows slower than the other methods, indicating that the proposed method can reasonably plan the evacuation path to guide the evacuation; while the evacuation times of the other three methods increase significantly, especially the DDQN-based method grows the fastest, indicating that the other three methods are unreasonable in path planning. In addition, although the time to complete evacuation of the four methods increases with the number of pedestrians, the incremental time to complete evacuation of the proposed method is still lower than that of the other methods. Therefore, the proposed method can cope with the evacuation of large-scale crowds better than the other three methods.

Then, we used four methods to simulate the evacuation process of 100 pedestrians. Fig. 16 shows a simulation process figure based on the E-MADDPG method, Fig. 17 shows a simulation process figure based on the method in (Yao et al., 2019);Fig. 18 shows a simulation process figure based on the DDQN method, and Fig. 19 shows a simulation process figure based on the method in (Tian et al., 2020).

The simulation results show that the evacuation path planned by the proposed method is more reasonable and are effective in avoiding obstacles as well as buildings, while slowing down congestion. However, the evacuation paths of several other methods lead to different degrees

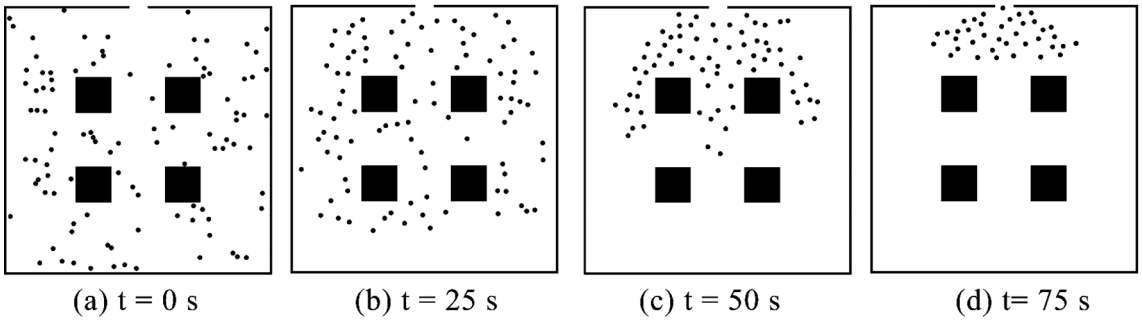

Fig. 16. Simulation based on the E-MADDPG method.

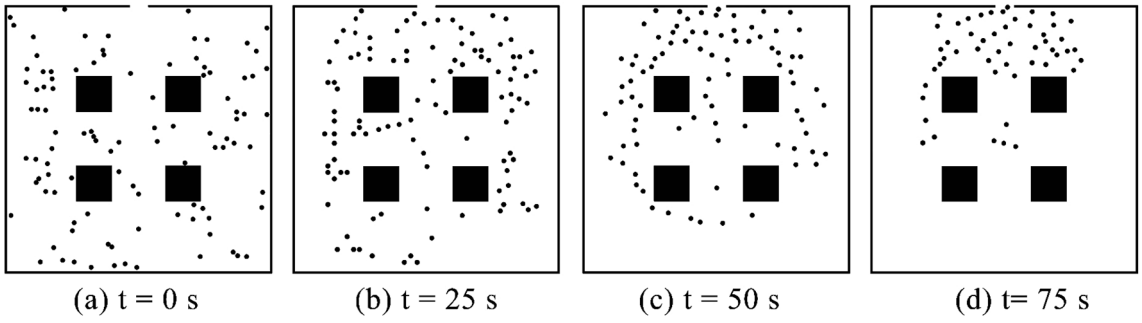

Fig. 17. Simulation based on the method in (Yao et al., 2019).

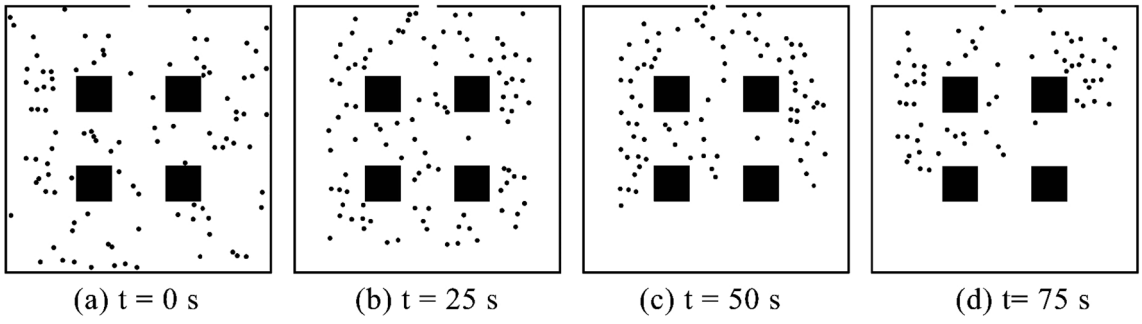

Fig. 18. Simulation based on the DDQN method.

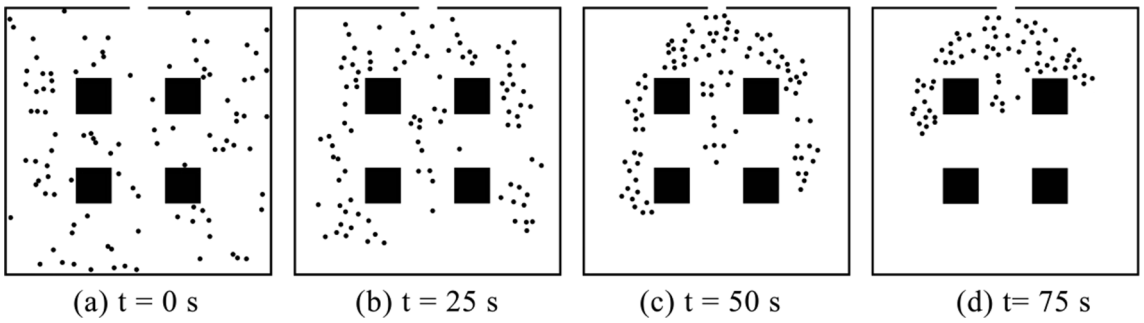

Fig. 19. Simulation based on the method in (Tian et al., 2020).

Table 5
Changes in the number of pedestrians being evacuated in a scene with obstacles.

Time(s)	0	30	60	90	120
Proposed method	300	136	62	0	0
method in (Yao et al., 2019)	300	147	90	11	0
DDQN	300	160	96	34	0
method in (Tian et al., 2020)	300	151	80	17	0

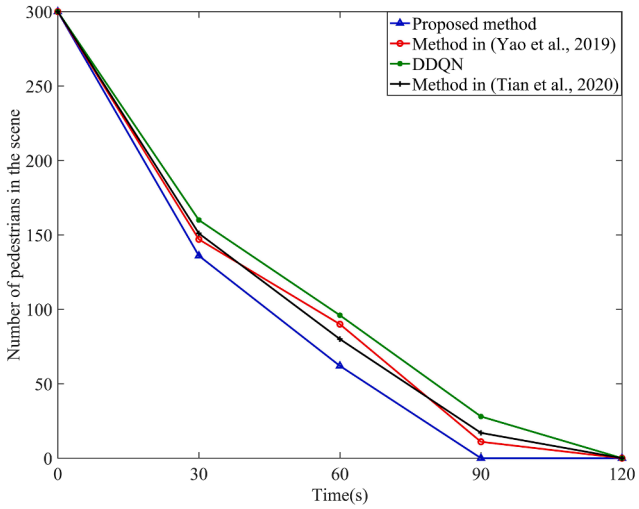

Fig. 20. Changes in the number of pedestrians being evacuated in a scene with obstacles.

Table 6
Successful evacuation time of 200 pedestrians under different exit numbers.

Number of exits	1	2	3	4
No obstacle scene	55.68 s	32.3 s	23.58 s	20.03 s
Obstacle scene	84.21 s	58.15 s	43.36 s	37.05 s

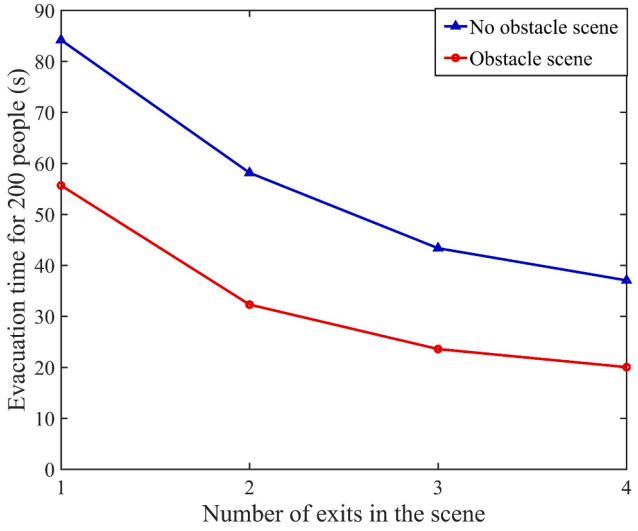

Fig.21. Ssuccessful evacuation time of 200 pedestrians under different exit numbers.

of congestion. In addition, the proposed method effectively avoids collisions between pedestrians during evacuation, while several other methods produce collisions of different degrees in crowded situations. Therefore, the proposed method can accomplish evacuation of people in scenes with obstacles more effectively, thus improving the safety of people.

Finally, we evacuated 300 pedestrians in an obstructed scene using different evacuation methods, and then compared the change in the number of people stranded inside during the evacuation.

Table 5 and Fig. 20 show change in the number of pedestrians during the evacuation process when 300 pedestrians are evacuated in the scene with obstacle. Fig. 20 shows that the proposed method completes the evacuation in approximately 90 s, the method in (Yao et al., 2019) completes the evacuation in approximately 98.2 s, the method based on the DDQN algorithm completes the evacuation in approximately 105.6 s, and the method in (Yao et al., 2019) completes the evacuation in approximately 100.5 s. Therefore, the evacuation strategy of the proposed method is more effective compared to the other three methods. As can be seen from the figure, the method in (Yao et al., 2019) has a slow decrease in the number of people until 60 s and a significant decrease in the number of people from 60 to 90 s, indicating that this method experiences severe crowding until 60 s. The number of people evacuated by the DDQN-based method before 60 s is much lower than the other three methods, indicating that this method is inefficient in path planning. The change in the number of people during evacuation in the method in (Tian et al., 2020) is similar to that of the proposed method, but the evacuation time is more than that of the proposed method. The proposed method has a relatively stable number of evacuees per unit time, which indicates that the evacuation path of the proposed method is more reasonable and there is no long time congestion. Therefore, the proposed method can better guide pedestrian evacuation in the presence of obstacles.

5.4. Comparison of evacuation process under different number of exits

To verify the robustness of the proposed method. We set the conditions of one exit, two exits, three exits, and four exits respectively in the obstacle-free scene and the obstacle scene. Then we use the proposed method to simulate the evacuation of 200 pedestrians in these scenes. Finally, we compare the evacuation time and evacuation process under different number of exits.

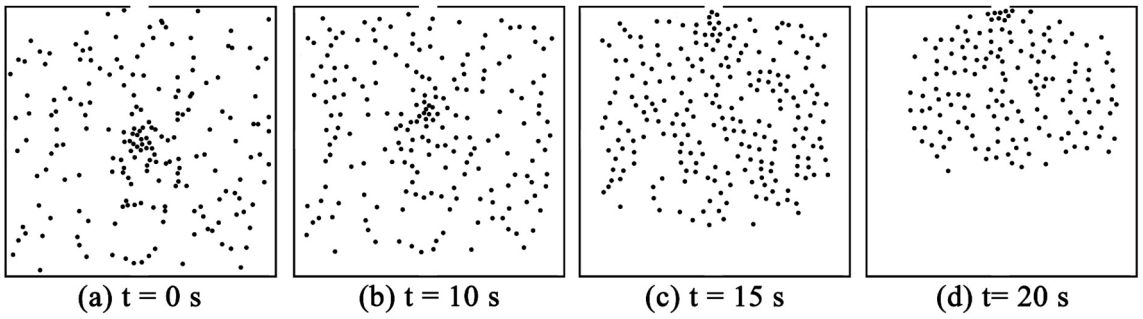

Fig. 22. The evacuation process when there is an exit in the obstacle-free scene.

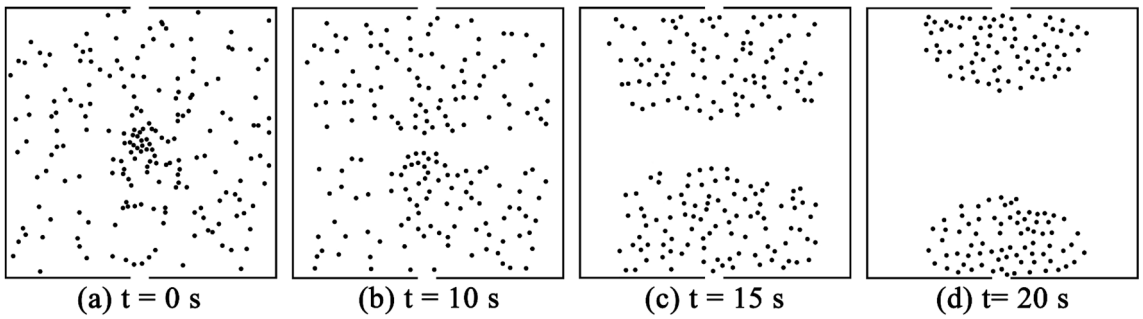

Fig. 23. The evacuation process when there are two exits in the obstacle-free scene.

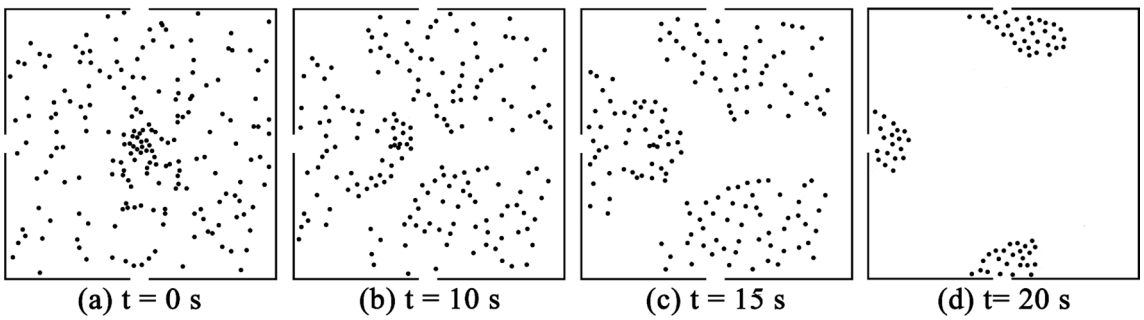

Fig. 24. The evacuation process when there are three exits in the obstacle-free scene.

Table 6 and Fig. 21 show how the evacuation time changes when the number of exits increases. In Fig. 21, the blue line represents the change in evacuation time when the number of exits increases in a scene with obstacles. The red line represents the change in evacuation time when the number of exits increases in a scene with no obstacle. With the increase in the number of exits, the evacuation time shows a downward trend, which shows that the proposed method has strong robustness. And it can be seen from the figure that in the obstacle-free scene, when one exit is increased to two, the evacuation time drops significantly, and when the number of exits continues to increase, the decline in evacuation time gradually decreases. This shows that in obstacle-free scenes, because there is no need to avoid obstacles, after the number of exits reaches a certain level, the congestion during the evacuation process will be greatly reduced, and the decline in evacuation time will have a peak. Increasing the number of exits after this peak will be less and less helpful in reducing the evacuation time. With the increase in the number of exits

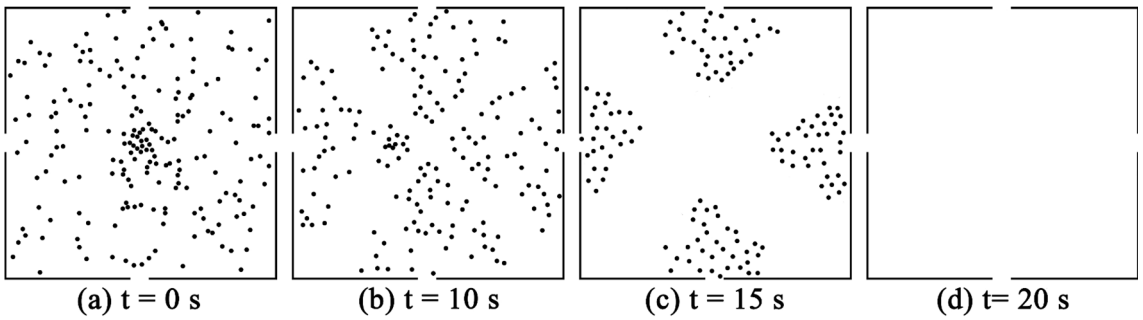

Fig. 25. The evacuation process when there are four exits in the obstacle-free scen.

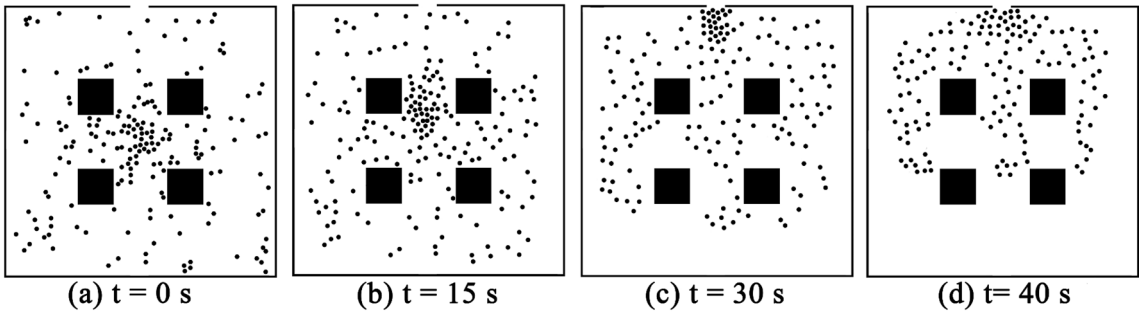

Fig. 26. The evacuation process when there is an exit in a scene with obstacles.

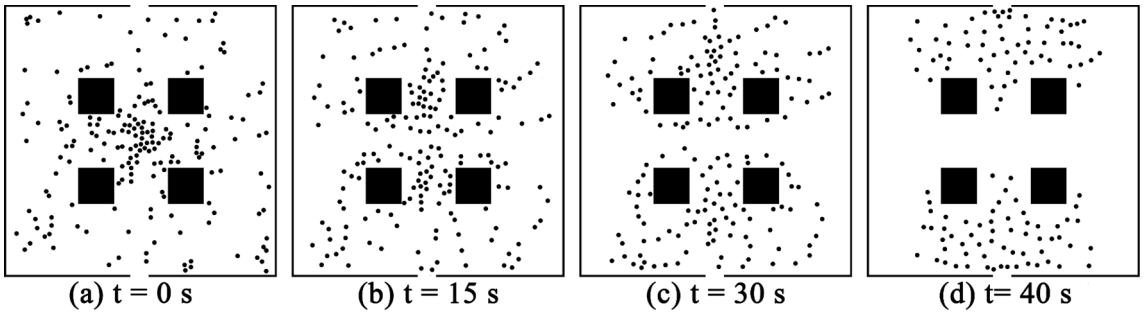

Fig. 27. The evacuation process when there are two exits in a scene with obstacles.

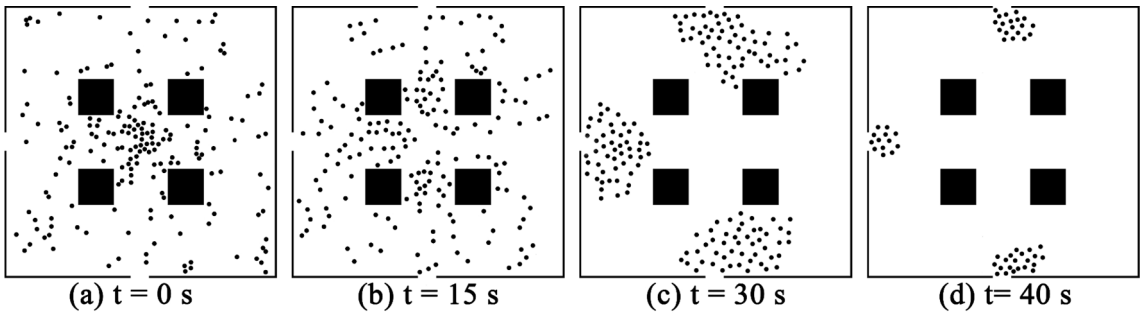

Fig. 28. The evacuation process when there are three exits in a scene with obstacles.

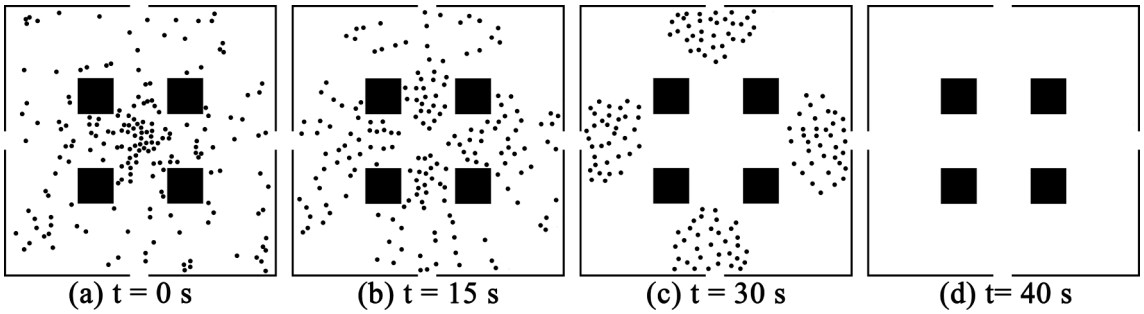

Fig. 29. The evacuation process when there are four exits in a scene with obstacles.

in the obstacle scene, the decline in evacuation time also gradually decreases, but the rate of decrease is slower. This is because in the case of obstacles, it is necessary to avoid obstacles. Therefore, each additional exit can effectively reduce the occurrence of collisions and congestion. But, after the number of exits in a scene with obstacles reaches a certain number, increasing the number of exits will become less and less helpful in reducing the evacuation time. In general, the proposed method can give different evacuation plan according to different scenes, and has strong robustness.

To more intuitively reflect the robustness of our method, we respectively simulated the evacuation process in the obstacle-free scene and the obstacle scene with different numbers of exits. Fig. 22, Fig. 23, Fig. 24, Fig. 25 are the evacuation process when there are different numbers of exits in the obstacle-free scene. Fig. 26, Fig. 27, Fig. 28, and

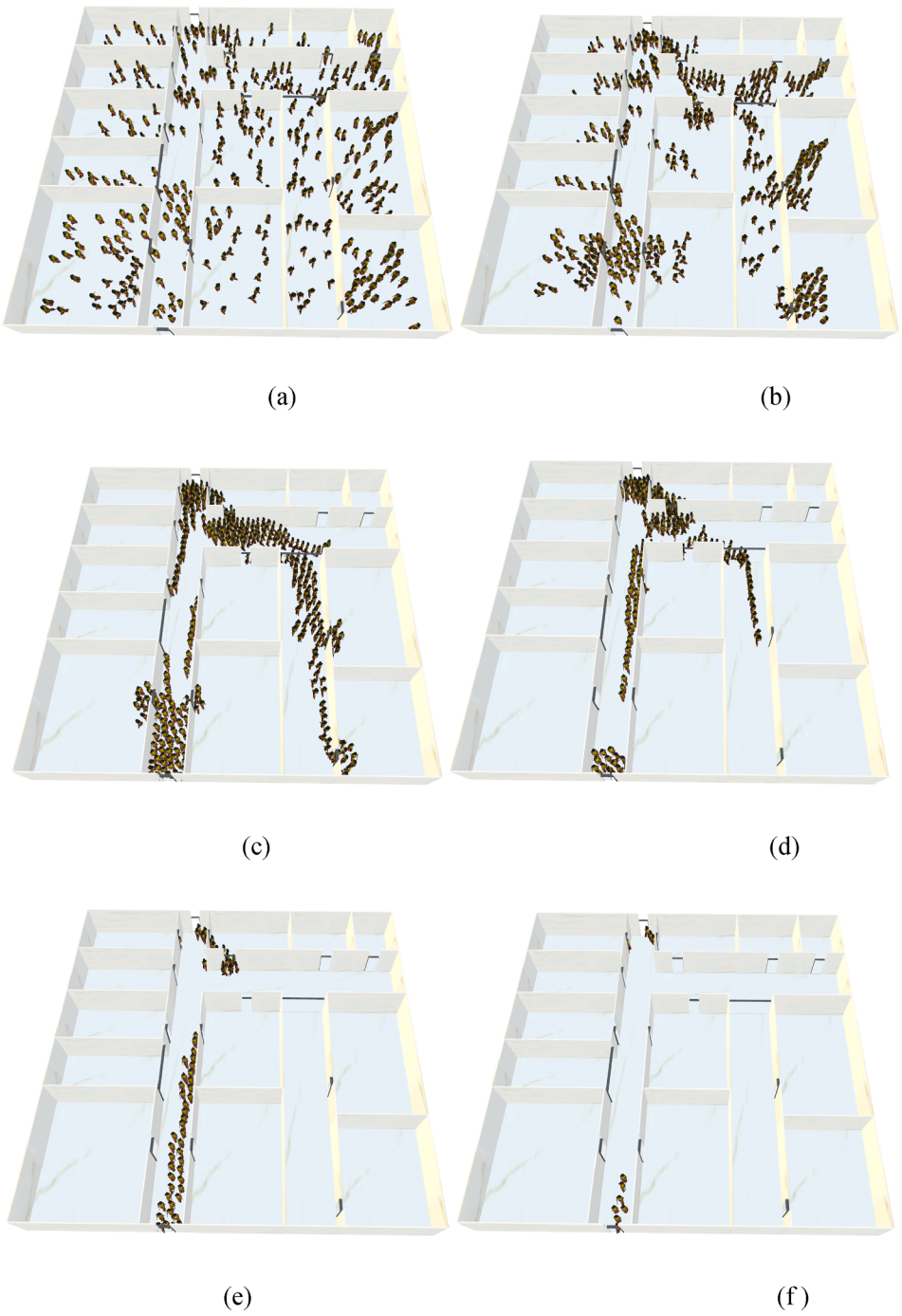

Fig. 30. Simulation based on the E-MADDPG algorithm.

Fig. 29 are the evacuation process when there are different numbers of exits in the obstacle scene. From the experimental results, the proposed method can reasonably use exits for path planning and improve the efficiency of evacuation. Simultaneously, the proposed method can achieve good path planning and evacuation guidance in different scenes and different numbers of exits. And the proposed method can make reasonable use of exits for route planning and improve evacuation efficiency. Therefore, the proposed method has strong robustness.

5.5. Simulation

In the simulation platform, we modeled based on the first floor of the teaching building. In this scene, we set the population size to 400, the pedestrian radius to r = 0.2 m, and the pedestrian mass to m = 60 kg, and then use the proposed method to evacuate the pedestrians. Fig. 30 shows the evacuation effect of the proposed method. From the experimental results, the method in this paper realizes effective evacuation in real scenes, avoids collisions during the evacuation process, makes full use of exits, and realizes safe and efficient evacuation.

6. Conclusion

In this paper, combining E-MADDPG algorithm and RVO algorithm, a method of stratified crowd evacuation is proposed. The proposed method uses the superior path planning ability of the E-MADDPG algorithm to obtain the optimal path. At the same time, it uses the RVO

algorithm to avoid obstacles and evacuates according to the optimal path during the evacuation process. The experimental results prove that the proposed method can reasonably plan the evacuation route.

Through experiments, we can draw some conclusions:

(1) The proposed algorithm achieves a higher learning efficiency and faster convergence speed than other methods, and due to the reduction of state space, the proposed algorithm also avoids irrelevant learning and can thus obtain higher rewards.
(2) When obstacles are not present, little difference is shown in the evacuation efficiencies of the proposed method and other methods when the population density is low. With increasing population density, the evacuation efficiency of the proposed method is higher than those of other methods.
(3) When obstacles are present, the proposed method can plan an evacuation path more reasonably, with a higher evacuation efficiency, and prevent long-term congestion or blockages during evacuation.
(4) The proposed method has strong robustness, can obtain the optimal evacuation path in different scenes, and improve the evacuation efficiency.

According to other research and the experimental, a reasonable evacuation strategy can accelerate evacuation. This study aims to determine an optimal evacuation strategy, guide evacuations more effectively, and ensure the safety of pedestrians when danger is present. In the proposed evacuation process, pedestrians in the crowd divide themselves into groups based on their positions and relationships with each other, and each group has a leader. Then, we simulate the evacuation process in a multi-agent system. Leaders use the E-MADDPG algorithm for global path planning, and followers use the RVO algorithm to avoid collisions and follow the leader to the evacuation point. Thus, forming a stable evacuation process and improving evacuation speed.

This study achieves only preliminary goals in this field of research. This study's experiments were performed in relatively simple scenes and did not consider the response of pedestrians to emergencies that occur during evacuation. To evacuate pedestrians more quickly when danger is present and ensure the safety of pedestrians during dangerous events, we must explore evacuations in complex environments and optimize the proposed method.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The research reported in this paper is financially supported by the National Natural Science Foundation of China (61876102, 61972237, 61472232) .

References

Bhatnagar, S., Sutton, R. S., Ghavamzadeh, M., & Lee, M. (2009). Natural actor-critic algorithms. Automatica. 10.1016/j.automatica.2009.07.008.

Bi, C., Pan, G., Yang, L., Lin, C. C., Hou, M., & Huang, Y. (2019). Evacuation route recommendation using auto-encoder and Markov decision process. Applied Soft Computing Journal, 84, 105741. 10.1016/j.asoc.2019.105741.

Buşoniu, L., Babuška, R., & De Schutter, B. (2008). A comprehensive survey of multiagent reinforcement learning. In IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews. 10.1109/TSMCC.2007.913919.

Chen, Y. H., Hong, W. C., Shen, W., & Huang, N. N. (2016). Electric load forecasting based on a least squares support vector machine with fuzzy time series and global harmony search algorithm. *Energies, 9*(2), 1–13.

Cruz, D. L., & Yu, W. (2017). Path planning of multi-agent systems in unknown environment with neural kernel smoothing and reinforcement learning. Neurocomputing. 10.1016/j.neucom.2016.08.108.

Cui, J., Liu, Y., & Nallanathan, A. (2020). Multi-Agent Reinforcement Learning-Based Resource Allocation for UAV Networks. IEEE Transactions on Wireless Communications. 10.1109/TWC.2019.2935201.

Den Van Berg, J., Lin, M., & Manocha, D. (2008). Reciprocal velocity obstacles for real-time multi-agent navigation. *Proceedings - IEEE International Conference on Robotics and Automation.* https://doi.org/10.1109/ROBOT.2008.4543489

Fan, G. F., Qing, S., Wang, H., Hong, W. C., & Li, H. J. (2013). Support vector regression model based on empirical mode decomposition and auto regression for electric load forecasting. *Energies, 6*(4), 1887–1901.

Goel, R., & Maini, R. (2018). A hybrid of ant colony and firefly algorithms (HAFA) for solving vehicle routing problems. Journal of Computational Science. 10.1016/j.jocs.2017.12.012.

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation, 18*(7), 1527–1554.

Jiang, L., Huang, H., & Ding, Z. (2020). Path planning for intelligent robots based on deep Q-learning with experience replay and heuristic knowledge. *IEEE/CAA Journal of Automatica Sinica, 7*(4), 1179–1189.

Kim, B., & Pineau, J. (2016). Socially Adaptive Path Planning in Human Environments Using Inverse Reinforcement Learning. *International Journal of Social Robotics, 8*(1), 51–66.

Li, M.-W., Wang, Y.-T., Geng, J., & Hong, W.-C. (2021). Chaos cloud quantum bat hybrid optimization algorithm. *Nonlinear Dynamics, 103*(1), 1167–1193.

Li, J. qing, Liu, Z.-M., Li, C., & Zheng, Z. (2020). Improved artificial immune system algorithm for Type-2 fuzzy flexible job shop scheduling problem. IEEE Transactions on Fuzzy Systems. 10.1109/tfuzz.2020.3016225.

Li, J. qing, Tao, X. rui, Jia, B. xian, Han, Y. yan, Liu, C., Duan, P., Zheng, Z. xin, & Sang, H. yan. (2020). Efficient multi-objective algorithm for the lot-streaming hybrid flowshop with variable sub-lots. Swarm and Evolutionary Computation. 10.1016/j.swevo.2019.100600.

Li, J. qing, Du, Y., Gao, K., Duan, P., Gong, D., & Pan, Q. (2021). A hybrid iterated greedy algorithm for a crane transportation flexible job shop problem. IEEE Transactions on Automation Science and Engineering. 10.1109/TASE.2021.3062979.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., & Wierstra, D. (2016). Continuous control with deep reinforcement learning. 4th International Conference on Learning Representations, ICLR 2016 .

Liu, Y., Zhou, S., & Chen, Q. (2011). Discriminative deep belief networks for visual data classification. Pattern Recognition. 10.1016/j.patcog.2010.12.012.

Liu, M., Zhang, F., Ma, Y., Pota, H. R., & Shen, W. (2016). Evacuation path optimization based on quantum ant colony algorithm. Advanced Engineering Informatics. 10.1016/j.aei.2016.04.005.

Liu, H., Liu, B., Zhang, H., Li, L., Qin, X., & Zhang, G. (2018). Crowd evacuation simulation approach based on navigation knowledge and two-layer control mechanism. *Information Sciences, 436–437*(88), 247–267.

Liu, B., Liu, H., Zhang, H., & Qin, X. (2018). A social force evacuation model driven by video data. *Simulation Modelling Practice and Theory, 84*, 190–203.

Liu, H., Xu, B., Lu, D., & Zhang, G. (2018). A path planning approach for crowd evacuation in buildings based on improved artificial bee colony algorithm. *Applied Soft Computing Journal, 68*, 360–376.

Low, E. S., Ong, P., & Cheah, K. C. (2019). Solving the optimal path planning of a mobile robot using improved Q-learning. Robotics and Autonomous Systems. 10.1016/j.robot.2019.02.013.

Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., & Mordatch, I. (2017). Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in Neural Information Processing Systems., 30*, 6379–6390.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature, 518*(7540), 529–533.

Peng, Y., Li, S.-W., & Hu, Z.-Z. (2019). A self-learning dynamic path planning method for evacuation in large public buildings based on neural networks. *Neurocomputing, 365*, 71–85.

Saraswat, S. P., & Gorgone, J. T. (1990). Organizational learning curve in software installation: An empirical investigation. Information and Management. 10.1016/0378-7206(90)90014-9.

Sharma, J., Andersen, P.-A., Granmo, O.-C., & Goodwin, M. (2020). Deep Q-learning with Q-matrix transfer learning for novel fire evacuation environment. *IEEE Transactions on Systems, Man, and Cybernetics: Systems, 1–19*. https://doi.org/10.1109/tsmc.2020.2967936

Singh, B., Kumar, R., & Singh, V. P. (2021). Reinforcement learning in robotic applications: a comprehensive survey. In Artificial Intelligence Review (Issue 0123456789). Springer Netherlands. 10.1007/s10462-021-09997-9.

Sun, Y., Xue, B., Zhang, M., & Yen, G. G. (2020). Evolving Deep Convolutional Neural Networks for Image Classification. IEEE Transactions on Evolutionary Computation. 10.1109/TEVC.2019.2916183.

Tian, Z., Zhang, G., Hu, C., Lu, D., & Liu, H. (2020). Knowledge and emotion dual-driven method for crowd evacuation. Knowledge-Based Systems, 208. 10.1016/j.knosys.2020.106451.

Van Hasselt, H., Guez, A., & Silver, D. (2016). Deep reinforcement learning with double Q-Learning. *30th AAAI Conference on Artificial Intelligence*.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., Oh, J., Horgan, D., Kroiss, M., Danihelka, I., Huang, A., Sifre, L., Cai, T., Agapiou, J. P., Jaderberg, M., … Silver, D. (2019). Grandmaster level in StarCraft II using multi-agent reinforcement learning. Nature. 10.1038/s41586-019-1724-z.

Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M., & De Frcitas, N. (2016). Dueling Network Architectures for Deep Reinforcement Learning. 33rd International Conference on Machine Learning, ICML 2016.

Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M., & De Frcitas, N. (2016). Dueling Network Architectures for Deep Reinforcement Learning. *33rd International Conference on Machine Learning*.

Wang, Y., Sun, J., He, H., & Sun, C. (2020). Deterministic policy gradient with integral compensator for robust quadrotor control. *IEEE Transactions on Systems, Man, and Cybernetics: Systems.*. https://doi.org/10.1109/TSMC.2018.2884725

Wong, S.-K., Tang, P.-K., Li, F.-S., Wang, Z.-M., & Yu, S.-T. (2015). Guidance path scheduling using particle swarm optimization in crowd simulation. *Computer Animation and Virtual Worlds, 26*(3-4), 387–395.

Yao, Z., Zhang, G., Lu, D., & Liu, H. (2019). Data-driven crowd evacuation: A reinforcement learning method. Neurocomputing. 10.1016/j.neucom.2019.08.021.

Yao, Z., Zhang, G., Lu, D., & Liu, H. (2020). Learning crowd behavior from real data: A residual network method for crowd simulation. *Neurocomputing, 404*, 173–185.

Zhang, Y., Chai, Z., & Lykotrafitis, G. (2021). Deep reinforcement learning with a particle dynamics environment applied to emergency evacuation of a room with obstacles. *Physica A: Statistical Mechanics and Its Applications, 571*, Article 125845. https://doi.org/10.1016/j.physa.2021.125845

Zhang, F., Li, J., & Li, Z. (2020). A TD3-based multi-agent deep reinforcement learning method in mixed cooperation-competition environment. *Neurocomputing, 411*, 206–215.

Zhang, H., Liu, H., Qin, X., & Liu, B. (2018). Modified two-layer social force model for emergency earthquake evacuation. *Physica A: Statistical Mechanics and Its Applications, 492*, 1107–1119.

Zhang, G., Wang, M., Lu, D., & Liu, H. (2020). Strategies to utilize the positive emotional contagion optimally in crowd evacuation. *IEEE Transactions on Affective Computing, 11*(4), 708–721.

Zhao, X., Ding, S., An, Y., & Jia, W. (2018). Asynchronous reinforcement learning algorithms for solving discrete space path planning problems. *Applied Intelligence, 48*(12), 4889–4904.

Zhao, Y., Liu, H., & Gao, K. (2021). An evacuation simulation method based on an improved artificial bee colony algorithm and a social force model. *Applied Intelligence, 51*(1), 100–123.

Zheng, S., & Liu, H. (2019). Improved multi-Agent deep deterministic policy gradient for path planning-based crowd simulation. IEEE Access. 10.1109/ACCESS.2019.2946659.