Simulating multi-exit evacuation using deep reinforcement learning

Dong Xu¹,² | Xiao Huang³ | Joseph Mango¹ | Xiang Li¹ | Zhenlong Li²

¹Key Laboratory of Geographical Information Science (Ministry of Education), School of Geographic Sciences, East China Normal University, Shanghai, China
²Geoinformation and Big Data Research Laboratory, Department of Geography, University of South Carolina, Columbia, SC, USA
³Department of Geosciences, University of Arkansas, Fayetteville, AR, USA

Correspondence
Xiang Li, Key Laboratory of Geographical Information Science (Ministry of Education), School of Geographic Sciences, East China Normal University, Shanghai, China.
Email: xli@geo.ecnu.edu.cn
Zhenlong Li, Geoinformation and Big Data Research Laboratory, Department of Geography, University of South Carolina, 709 Bull Street, Columbia, SC 29208, USA.
Email: zhenlong@sc.edu

Funding information
National Natural Science Foundation of China, Grant/Award Number: 41771410; Ministry of Education of China, Grant/Award Number: 19JZD023

Abstract
Conventional simulations on multi-exit indoor evacuation focus primarily on how to determine a reasonable exit based on numerous factors in a changing environment. Results commonly include some congested and other under-utilized exits, especially with large numbers of pedestrians. We propose a multi-exit evacuation simulation based on deep reinforcement learning (DRL), referred to as the MultiExit-DRL, which involves a deep neural network (DNN) framework to facilitate state-to-action mapping. The DNN framework applies Rainbow Deep Q-Network (DQN), a DRL algorithm that integrates several advanced DQN methods, to improve data utilization and algorithm stability and further divides the action space into eight isometric directions for possible pedestrian choices. We compare MultiExit-DRL with two conventional multi-exit evacuation simulation models in three separate scenarios: varying pedestrian distribution ratios; varying exit width ratios; and varying open schedules for an exit. The results show that MultiExit-DRL presents great learning efficiency while reducing the total number of evacuation frames in all designed experiments. In addition, the integration of DRL allows pedestrians to explore other potential exits and helps determine optimal directions, leading to a high efficiency of exit utilization.
Indoor pedestrian simulation in evacuation studies has been receiving a great deal of attention due to its potential for rescuing people in cases of emergency (Chen & Feng, 2009; Sun & Li, 2011). A reasonable simulation should be able to guide the pedestrian to leave quickly and safely through exits of the indoor environments. In order to achieve this objective, pedestrians need to choose the best route to pass through, usually depending on the number of sub-exits present in a particular complex. This problem can be regarded as a navigation planning problem where the shortest path can be chosen by using sampling-based algorithms (Karaman & Frazzoli, 2011; Kuffner & LaValle, 2000) or geometry-based algorithms (Geraerts, 2010; Kallmann, 2014). The pedestrians are expected to reach the final exits in the shortest time as long as they have followed the calculated path. However, these assumptions are only effective and efficient in an environment where the number of pedestrians and sub-exits is low. In reality, there are many complex indoor environments with a large number of pedestrians. In such circumstances, the evacuation process becomes more complex due to overcrowding, starting from the nearest sub-exits to the final exits. This overcrowding problem is usually due to the narrow widths of the sub-exits, making it challenging for pedestrians to leave quickly. Moreover, this problem becomes more complex if there is variation in sub-exit widths together with an uneven distribution of pedestrians. Therefore, how to reasonably allocate pedestrians to different exits is still a valid and challenging question in pedestrian evacuation simulation studies.

The existing multi-exit selection research can be broken down into two categories of studies. The first category is focused on the building design, and aims to propose more reasonable and more efficient multi-exit layouts (Choi, Choi, & Kim, 2014; Seyfried et al., 2009). However, the models designed from these studies are unable to accommodate pedestrian behaviors that are proven to be significant during the evacuation process (Bode & Codling, 2013). The second is focused on designing a multi-exit selection model using realistic experimental simulation (Haghani, Sarvi, Ejtemai, Burd, & Sobhani, 2015; Heliovaara, Kuusinen, Rinne, Korhonen, & Ehtamo, 2012; Wagoum, Tordeux, & Liao, 2017). In contrast to the first category, the models designed using the second approach often incur high preliminary costs. Diverging from these two main streams, other researchers started to utilize virtual simulation to describe the multi-exit simulation process by setting motion in models to guide pedestrian navigation via movement strategies (Hao, Bin-Ya, Chen-Fu, & Yan, 2014; Zia & Ferscha, 2009).

In the past decade, deep learning has emerged as a machine learning method to provide remarkable modeling results such as in image classification (Krizhevsky, Sutskever, & Hinton, 2012), object tracking (Bertinetto, Valmadre, Henriques, Vedaldi, & Torr, 2016), and natural language processing (Sutskever, Vinyals, & Le, 2014). One type of deep learning, deep reinforcement learning (DRL), has also enjoyed considerable success in defeating human players in many competitions such as Go (Silver et al., 2016) and Atari games (Mnih et al., 2013). Unlike both supervised learning and unsupervised learning, DRL learns a mapping (i.e., from state to action) to maximize the long-term reward (Sutton & Barto, 2018). Depending on its learning methods, DRL can be classified into two categories: Deep Q-Network (DQN) and Policy Gradient (PG). To date, there are several advanced DRL algorithms, which include: Rainbow DQN (Hessel et al., 2018), Proximal Policy Optimization (PPO; Schulman, Wolski, Dhariwal, Radford, & Klimov, 2017), Soft Actor Critic (SAC; Haarnoja, Zhou, Abbeel, & Levine, 2018), and Deep Deterministic Policy Gradient (DDPG; Lillicrap et al., 2015). All these DRL algorithms have been applied in different areas, such as robot visual navigation (Gupta, Davidson, Levine, Sukthankar, & Malik, 2017; Zhang, Springenberg, Boedecker, & Burgard, 2017). Apart from the examples of robotic applications, DRL algorithms have been used in local motion planning as well. For example, Lee, Won, and Lee (2018) proposed a DRL method based on the Actor Critic framework for crowd simulation. In the same field, Long et al. (2018) presented a decentralized collision avoidance policy by directly detecting the environment using raw sensors and feeding the measurements to a DRL network. Despite these achievements, the research on multi-exit navigation based on DRL needs to be explored further in different environments.

In this article, we focus on simulating the movement of pedestrians in multi-exit navigation within designed room micro-environments, of which all pedestrians are fully aware. Therefore, the navigation simulation process
can be regarded as a Markov decision process. A DRL algorithm is proposed to explore its potential in multi-exit navigation in the above environments. Each pedestrian is defined as a disc, characterized by a position and velocity. The simulation process in our research can be regarded as a series of ordered frames. At a given frame, the pedestrians’ positions are determined based on their velocities (described in detail in Section 3). A total of eight possible movement directions are provided for each pedestrian, and the desired direction is determined using the DRL algorithm. Optimal reciprocal collision avoidance (ORCA) is adopted to avoid collisions when pedestrians are moving. Unlike other studies, we do not explicitly assign a reasonable exit. In our method, each pedestrian is considered as an intelligent person with self-judgment, who takes desired actions to facilitate his or her evacuation after millions of iterations of interacting with the environment. To illustrate the advantages of our method, we compared our proposed method with two popular methods from Zheng, Li, Meng, Xu, and Chen (2015) and Ren-Yong and Hai-Jun (2010). The main contributions of this article are threefold, First, we designed a hierarchical model to handle multi-exit navigation simulation at micro scale, a more realistic method for describing pedestrian behaviors compared with other simulations at meso scale. Second, we developed a new method which abstracts the gray image of the room as the state space by distinguishing the target pedestrian and other pedestrians using grayscale values. Compared with the traditional raycasting methods that consider external environments (Lee et al., 2018), our method can noticeably speed up the simulation process. Third, we integrated Rainbow DQN, an effective DRL method that combines several advanced DQN algorithms. This training design greatly increases the stability of the neural network and the speed at which the convergence point is reached. Furthermore, the Rainbow DQN in our method can be updated in one step or N steps without collecting a complete training trajectory. Thus, it provides an advantage in data collection compared with the traditional PG methods.

This work is organized into seven sections. Section 2 summarizes the existing literature on multi-exit navigation and deep reinforcement learning algorithms. Section 3 presents more information about ORCA and the Rainbow DQN algorithm in network architecture, state space, action space, and reward functions. Section 4 introduces three designed scenarios to compare our proposed MultiExit-DRL model with two traditional models based on mathematical strategy navigation. It also describes the settings of the relevant parameters and computer configurations used when developing our model. Section 5 presents, analyzes and discusses the results of the proposed MultiExit-DRL model. Section 6 discusses the advantages and limitations of the proposed model. Section 7 concludes.

2 | RELATED WORK

2.1 | Pedestrian simulation

Pedestrian simulation has recently received a great deal of attention in the geographic information science (GIS) field and is widely used in path planning (Wu, Marshall, & Yu, 2007), emergency decision-making (Tashakkori, Rajabifard, & Kalantari, 2015), and human behavior analysis (Li, Claramunt, & Ray, 2010). Pedestrian simulation can be divided into two groups: discrete simulation and continuous simulation. The cellular automata (CA) model, as a classic discrete model, has been applied to simulate the movement of pedestrians, with reasonable results (Chopard & Droz, 1998). In recent decades, advances in the computational power of rough modeling have facilitated many CA related studies (Burstedde, Klauck, Schadschneider, & Zittartz, 2001; Dijkstra, Jessurun, & Timmermans, 2002; Pelechano & Malkawi, 2008). In particular, Torrens (2012) presented a complete pedestrian simulation framework using the agent-based model and CA model, which simulates behaviors at three levels: higher-, medium-, and low-level behaviors. As the movement of pedestrians is fully described, this framework has the ability to generally reproduce real-world scenarios. However, CA models have been criticized due to their roughness, as they discretize pedestrian movements, falling short of describing the dynamic behaviors of pedestrians (Cao, Chen, & Stuart, 2016).
In light of this issue, continuous pedestrian simulation has gradually become a research hotspot. Unlike the rough modeling commonly used in discrete simulations, continuous models accurately simulate the environment and pedestrians in finer detail via geometric expressions, thus allowing pedestrians to move arbitrarily within the configuration space. For instance, Helbing and Molnar (1995) proposed a classic social force model (SFM), which has received widespread attention due to its convenient and effective nature (Hou, Liu, Pan, & Wang, 2014; Karamouzas, Sohre, Narain, & Guy, 2017; Mehran, Oyama, & Shah, 2009). Nevertheless, SFM and other force-based methods fail to guarantee complete collision avoidance (Curtis & Manocha, 2014). In comparison, geometrically based methods are considered to be better at avoiding collisions and have been widely applied in robotics and motion simulation. The velocity obstacle (VO) method is a traditional geometry-based collision avoidance method which is able to forecast potential collisions by calculating the VO region (Fiorini & Shiller, 1998). As an improvement of the VO approach, a reciprocal $n$-body collision avoidance approach (ORCA) by Van Den Berg, Guy, Lin, and Manocha (2011) introduced a low-dimensional linear program for collision-free movement. As widely reviewed in the literature, the ORCA approach proved able to simulate collision-free actions for thousands of agents in a few milliseconds. Due to its efficiency, the ORCA has been deployed in many applications with different modifications (Alonso-Mora, Breitenmoser, Rufli, Beardsley, & Siegwart, 2013; Bareiss & van den Berg, 2015). Given the superiority of the ORCA method and its flexibility in different applications, we adopted it in our study as the local pedestrian simulation method to avoid collisions during the simulation of the multi-exit evacuation process.

2.2 Multi-exit selection

Evacuation simulation has always been a hot topic in the fields of safety and robotics. Compared with common pedestrian simulations, evacuation simulation focuses more on pedestrians' behaviors with various internal or external factors during the evacuation process (Ben, Huang, Zhuang, Yan, & Xu, 2013; Frank & Dorso, 2011; Parisi & Dorso, 2005; Wang, Zhang, Shi, Yang, & Hu, 2015). Further, as an important component in evacuation simulation, multi-exit selection aims to mimic or analyze pedestrians' behaviors in complex local environments. In some studies, the evacuation data are extracted from realistic scenarios, and the decision rules are specified accordingly (Guo, Huang, & Wong, 2012; Haghani & Sarvi, 2016; Ren-Yong & Hai-Jun, 2010). Such approaches can accurately model pedestrian behavior. However, they are difficult to implement since the realization of realistic scenarios is always elusive. Other studies focus on investigating the evacuation process via computer simulations by controlling the designed environments (Bode & Codling, 2013; Davidich, Geiss, Mayer, Pfaffinger, & Royer, 2013; Han, Liu, & Moore, 2017; Kinatered et al., 2014; Zhou. Dong, Zhao, Ioannou, & Wang, 2019). Nevertheless, the choice of exits in this direction also remains a great challenge, especially when dealing with complex environments (Zheng, Jia, Li, & Jiang, 2017; Zia & Ferscha, 2009). In most studies, the choice of exits is determined by several factors, including the distance from pedestrians to the exits, the width of exits, the degree of congestion, and the familiarity of pedestrians with the environment (Cao, Fu, & Song, 2018; Hao et al., 2014; Kinatered, Comunale, & Warren, 2018; Lo, Huang, Wang, Yuen, 2006; Wagoum et al., 2017). Such methods would usually design complex numerical operations for multi-exit selection and have achieved remarkable results according to environmental conditions. For example, Ren-Yong and Hai-Jun (2010) proposed a nested logit discrete model (NLDM) to investigate the choice of exits, considering factors such as visibility, intimacy, and physical conditions of the exits. Zheng et al. (2015) proposed an improved adaptive multi-factor model (AMFM), considering the factors of spatial distance, density, and exit width. As a representative of traditional multi-exit evacuation simulation methods, they designed a sophisticated mathematical formula to define the rules for pedestrians to choose exits. The hyperparameters are specified as the basis that might influence agents' choices. Our method, instead, does not involve the designing of specified moving rules. The only set of hyperparameters that need to be fine-tuned are the weights among different reward functions. To verify the effectiveness of our proposed approach, we select NLDM and
2.3 | Deep reinforcement learning and its applications in pedestrian simulation

Reinforcement learning (RL) has attracted wide attention, largely due to its powerful performance in playing games such as Go and Atari games (Mnih et al., 2013; Silver et al., 2017). The purpose of RL is to discover a function (i.e., a state-to-action mapping) in order to acquire the maximum long-term reward under the current state. Due to its great performance, RL has been widely adopted in a variety of fields, including complicated decision-making, robot simulation, and intellectual games.

In recent years, many studies have been conducted to combine deep neural networks (DNNs) and RL into so-called deep reinforcement learning to solve more complicated problems, where DQN, a value-based DRL method, has caused widespread concern. The results showed that it achieved excellent results that surpassed human-level performances in multiple Atari games. Scholars started to explore the possibilities of improving training speed and accuracy, such as the proposal of Double DQN (Van Hasselt, Guez, & Silver, 2016), Dueling DQN (DDQN; Wang, Schaul, et al., 2015), prioritized experience replay (PER; Schaul, Quan, Antonoglou, & Silver, 2015), Noisy DQN (Fortunato et al., 2017), and Categorical DQN (Bellemare, Dabney, & Mounos, 2017). Recently, Hessel et al. (2018) proposed a Rainbow DQN approach that integrates several independent improvements of the DQN algorithm. Compared with other algorithms, Rainbow DQN has achieved noticeable improvements in the reward and convergence speed. Given its great capabilities, Rainbow DQN has been widely applied in open car simulation (Güçkıran & Bolat, 2019), adaptive traffic signals (Nawar, Fares, & Al-Sammak, 2019), and predictive panoramic video streaming (Xiao, Wu, Shi, Zhou, & Chen, 2019).

Nowadays, DRL is extensively used for robot navigation and pedestrian simulation. In robot navigation, the robots are able to detect the environment with the support of raw cameras or other types of sensors. In this case, the DRL algorithm is responsible for generating an optimal action to navigate the robot to the destination without collisions with other obstacles (Kahn, Villaflor, Ding, Abbeel, & Levine, 2018; Zhang et al., 2017). For crowd simulation, virtual pedestrians are the agents in the environment (usually a micro-scene), aiming to reach their destinations. While in motion, each pedestrian should avoid static and dynamic obstacles and ensure the trajectory without noticeable oscillations. Numerous attempts have been made to use DRL in crowd simulation. Godoy, Karamouzas, Guy, and Gini (2016) proposed a novel distributed approach, Coordinated Navigation (C-Nav), to address the multi-agent navigation problem in complex environments. Long et al. (2018) proposed a decentralized collision avoidance policy based on DRL for multi-robot systems where the inputs are directly detected from the raw sensors. To smoothen the global trajectory of agents, Xu, Huang, Li, and Li (2020) presented a local motion simulation method that integrates ORCA and DRL, named ORCA-DRL. The model has shown great ability to generalize, smooth the global trajectory, and improve the learning speed, compared with other DRL methods. Nevertheless, the PPO algorithm applied in Xu et al. (2020) falls short of adapting to a dynamic number of agents. In addition, the state inputs based on the traditional raycasting method largely limit its performance. Despite the success of DRL in crowd simulation, explorations of the potential of DRL in multi-exit evacuation simulation by involving room environments remain rare.

In order to fill the gap, we propose a novel multi-exit evacuation simulation, MultiExit-DRL. In this prototype, ORCA is applied to avoid collision of pedestrians during the simulation, and the Rainbow DQN architecture is used to guide pedestrians in choosing the best direction. Unlike other studies in multi-exit evacuation simulation, this study does not explicitly define the best choice of exits for the pedestrian to evacuate at each frame. Instead, we defined eight directions in each frame for them to learn the best choice using Rainbow DQN. We further designed several indoor environments to illustrate the advantages of integrating DRL in multi-exit environments. The designed MultiExit-DRL model has demonstrated good results in comparison with the other two traditional models,
and therefore it can be applied in many local indoor environments to investigate the behavior of pedestrian flows. It can also be applied as an important component to evaluate the evacuation efficiency for different building designs.

3 | METHODS

3.1 | Methodology overview

We set the simulation environment as a two-dimensional space $\mathbb{R}^2$, which consists of $N_{\text{ped}}$ randomly distributed pedestrians and $N_{\text{exits}}$ exits with varying properties. We further define the exit set as $S_{\text{exits}}$. The goal of each pedestrian is to evacuate the room as quickly as possible through one of the available exits without colliding with obstacles or other pedestrians. Each pedestrian is represented by a disc of radius $r$ with maximum speed $v_{\text{max}}$. The obstacles are composed of line segments defined by a series of counter-clockwise points. Pedestrian $i$ in frame $t$ is characterized by the following parameters: position $p_t^i$, velocity $v_t^i$, speed $v_t^i$ (scalar), collision-free velocity (i.e., optimal velocity) $v_{\text{opt}}^i$, and direction $a_t^i$.

Each exit has two properties. For exit $j$, $p_{\text{exit}}^j$ denotes the position of exit $j$ and $w_{\text{exit}}^j$ denotes the width of exit $j$. The pedestrians in the room are able to observe the positions and velocities of other pedestrians without explicit communication. For each frame within a total of $m$ frames during a simulation cycle, pedestrians update their positions based on the $v_{\text{opt}}^i$ restrained from the kinetics rules (Figure 1). The simulation process continues until all pedestrians have successfully evacuated the room, or the number of frames reaches the horizon $T$. After that, we need to reset the environment and continue to simulate the process repeatedly until the number of frames reaches $m$. At each frame $t$, pedestrians interact with the environment and the resulting interaction data, including state, action, reward, and terminal information, are stored in the container with a capacity of $N_{\text{buffer}}$ (Figure 1a). We further define an integer value, named learning start ($L_s$), to encourage the pedestrian to explore the environment sufficiently. At each training step, we sample interaction data from the container with a defined batch size. DNN parameters are further updated by Rainbow DQN, where a crossing-entropy loss is implemented to minimize the Kullback–Leibler (KL) divergence between predicted state–action probabilities and target state–action

**FIGURE 1** Methodology overview: (a) general simulation structure; and (b) interacting process for all agents at frame $t = k$ (modified from Xu et al., 2020)
probabilities. At the end of each training step, the priorities are updated based on the loss weight. In the interaction process (Figure 1b), each agent obtains the state from the simulation environment at each frame and then inputs the state to the DNN, which returns an action based on the given state. The action is an integer indexed from 0 to 7, representing the eight directions. We further transfer the action to the velocity via the direction and $v_{\text{max}}$. Note that this velocity cannot guarantee collision avoidance. Therefore, we further input the velocity to the ORCA framework, which outputs the final optimal velocity $v_{\text{opt}}$. The position of each agent is updated based on the optimal velocity, and the environment is updated accordingly. Our method can be regarded as a hierarchical model, where the DNN is applied to obtain a coarse desired velocity, and ORCA further refines the velocity. ORCA, as a geometric method, can ensure collision avoidance. Specifically, for an agent, we assume that his or her current velocity is the maximum speed toward the target point. When calculating the optimal velocity for the next frame, we first calculate the VO area with surrounding obstacles according to its own shape and velocity. Then we judge whether the agent falls into the VO area with surrounding obstacles according to its own shape and velocity. Then we judge whether the agent falls into the VO area at the next frame. If s/he falls into the velocity obstacle area, the agent is expected to collide with other obstacles. Thus, the current velocity needs to be adjusted to deviate from the VO area. More details on the simulation implementation are presented in the following sections.

3.2 Deep reinforcement learning

3.2.1 State space, action space, and reward function

State space
We assume that pedestrians are fully aware of the environment. Similar to the state space in the Atari game, we directly abstract the images in the last three frames as the current state. The image is resized to 84 x 84. In our method, we use two coordinate systems, the environment coordinate system (ECS) and the screen coordinate system (SCS). The ECS aims to describe the interaction environment, while the SCS aims to visualize the evacuation process and convert the frame image to the state space. It should be understood that in practical environments, this problem is a typical GIS problem since it involves space and pedestrians as the moving objects in case of emergency (Tryfona, Prive, & Jensen, 2003; Xu & Güting, 2013; Zheni, Frihida, Ghezala, & Claramunt, 2009). Due to the fact that the inputs are grayscale images (one channel with value in the range [0, 255]), the difference between the walls and other moving agents is not distinguished. We set the grayscale images as follows:

1. The background of the images is set to a zero-value matrix with $h$ rows and $w$ columns, where $h$ and $w$ are also regarded as the height and width of the image.
2. The value of a pixel representing the current pedestrian is set to 255.
3. The value of a pixel representing static obstacles and other pedestrians is set to 100.

Action space
In our method, we discretize the action space with eight directions, indexed from 0 to 7. Each agent chooses a direction from the DNN and then translates the direction to a normalized vector $v_{\text{norm}}$. The best velocity is calculated as $v_{\text{norm}} \times v_{\text{max}}$.

Reward function
A total of four reward functions are included in this study, including the goal reward function $R_{\text{goal}}$, the collision reward function $R_{\text{collision}}$, the smooth reward function $R_{\text{smooth}}$, and penalty reward function $R_{\text{penalty}}$. The total reward $R$ is the aggregation of all four rewards:

$$R = w_1 R_{\text{goal}} + w_2 R_{\text{collision}} + w_3 R_{\text{smooth}} + R_{\text{penalty}}$$ (1)
where \( w_1, w_2, w_3 \) and \( R_{\text{penalty}} \) represent the weighting parameters.

\( R_{\text{goal}} \) defines the reward of one-step movement, described as:

\[
R_{\text{goal}} = \max_{j \in S_{\text{goal}}} \left( 1 - \text{Dist} \left( p_i^j, p_{\text{exit}}^j \right)^{w_4} \right) = \max_{j \in S_{\text{goal}}} \left( 1 - \text{Dist} \left( p_i^{j-1}, p_{\text{exit}}^j \right)^{w_4} \right)
\]  

(2)

where \( \text{Dist} \left( a, b \right) \) is the Euclidean distance between \( a \) and \( b \), and \( w_4 \) is a hyperparameter in the range \((0, 1)\). Equation (2) consists of two major parts. The first part is the reward based on the current position \( p_i^j \) for pedestrian \( i \). The second part is the reward based on the last position \( p_i^{j-1} \). \( R_{\text{goal}} \) represents the reward for the current step relative to the previous one.

\( R_{\text{collision}} \) defines the collision reward of pedestrian \( i \), which can be calculated by the optimal velocity \( \text{opt} v_i^t \) and the desired velocity from DNN:

\[
R_{\text{collision}} = \max_{j \in S_{\text{goal}}} \frac{\text{Dot} \left( \text{opt} v_i^j, \text{Norm} \left( p_{\text{exit}}^j - p_i^j \right) \right)}{v_i^{\max}}
\]

(3)

where represents the normalization function, and \( \text{Dot} \left( a, b \right) \) represents the dot product value, which defines the similarity between \( a \) and \( b \).

\( R_{\text{smooth}} \) is used to evaluate the smoothness of the pedestrian’s motion by comparing the current optimal velocity with the previous optimal velocity:

\[
R_{\text{smooth}} = \text{Dot} \left( \text{opt} v_i^j, \text{opt} v_i^{j-1} \right)
\]

(4)

### 3.2.2 Rainbow DQN

Reinforcement learning generally includes three components: the environment, the agents, and the reward. At each frame \( t \), an agent obtains the state \( s_t \) from the environment and conducts an action \( a_t \) based on \( s_t \). Then the simulation transits to the next state \( s_{t+1} \) and the agent receives a reward \( r_t \), an evaluation of \( r_t \). The interaction continues until the agent reaches the goal, or the iteration number reaches horizon \( T \). Theoretically, RL is a branch of the Markov process decision \( \langle S, A, P, r, \gamma \rangle \), where \( S \) is the state space, \( A \) is the action space, \( P \) is the transition relationship, \( r \) is the reward function, and \( \gamma \) is a discount factor that defines the foresight of the agents. An agent’s movement satisfies the Markov property, that is, the current state is only related to the historical trajectory. The objective of RL is to find a state–action pair mapping that maximizes the total expected return, \( R_t = \sum_{k=0}^{T} \gamma^k r_{t+k} \). In general, a DNN is required to map the state–action function, as the state space and action space tend to be multi-dimensional. In what follows, we briefly introduce the Rainbow DQN algorithm to optimize \( R_t \).

Rainbow DQN is an integration of a set of advanced DQN approaches. In our experiments, we utilized Double DQN, DDQN, PER, Multi-Step DQN, Categorical DQN, and Noisy DQN. In DQN, \( Q^\pi(s, a; \theta) \) represents the predicted return in state \( s \) with action \( a \) following policy \( \pi \) under the parameter setting \( \theta \) in the predicted DNN architecture. A bootstrap method (Mnih et al., 2013) is used to represent the target expected return:

\[
E \left[ R_t | s_t = s_1, a_1 = a_1 \right] \approx r_t + \gamma \max_{a_{t+1}} Q \left( s_{t+1}, a_{t+1}; \hat{\theta} \right)
\]

(5)

where \( \hat{\theta} \) represents the target DNN parameters, defined to reduce the correlation between predicted expected return and target expected return. To optimize the \( R_t \), a mean square error (MSE) loss is defined as

\[
L(\theta) = \left( r_t + \gamma \max_{a_{t+1}} Q \left( s_{t+1}, a_{t+1}; \hat{\theta} \right) - Q \left( s_t, a_t; \theta \right) \right)^2,
\]

(6)
where we only update \( \theta \) and \( \hat{\theta} \) will be automatically updated to \( \theta \) after a constant number of frames (\( F_f \)).

Double DQN is regarded as a decoupling approach to reduce the overestimation bias, where the target expected return is defined as:

\[
E \left[ R_t \mid s_t = s, a_t = a \right] \approx r_t + \gamma Q \left( s_{t+1}, \arg \max_{a_{t+1}} Q \left( s_{t+1}, a_{t+1}; \hat{\theta} \right) ; \theta \right) \tag{7}
\]

DDQN decomposes the last output value, \( Q(s, a; \theta) \), into two streams, the state value \( V(s) \) and the advantage value \( A(s, a) \), and combines the two streams by a special aggregator to give a new output value:

\[
Q(s, a; \theta) = V(s) + A(s, a) = \frac{\sum_a A(s, a')}{N_{\text{actions}}} \tag{8}
\]

where \( N_{\text{actions}} \) represents the total number of actions. Multi-Step DQN is adopted here to facilitate faster learning. We rewrite the \( n \)-step reward as:

\[
r_t^n = \sum_{k=0}^{n-1} \gamma^k r_{t+k+1} \tag{9}
\]

Thus, the target expected return based on Double DQN can also be rewritten as:

\[
E \left[ R_t \mid s_t = s, a_t = a \right] \approx r_t^n + \gamma^{(n)} Q \left( s_{t+1}, \arg \max_{a_{t+1}} Q \left( s_{t+1}, a_{t+1}; \hat{\theta} \right) ; \theta \right) \tag{10}
\]

To describe the expected value more accurately, Categorical DQN applies the distribution of values, \( Z_\theta(s, a) \), instead of a single value, where \( \theta \) represents the parameters of the related DNN. \( Z_\theta(s, a) \) denotes a discrete distribution, parameterized by \( N_{\text{atoms}} \cdot V_{\text{min}} \) and \( V_{\text{max}} \) respectively denote the minimum of maximum boundaries. For each atom in the distribution \( z_i \), defined as \( z_i = V_{\text{min}} + i \Delta z \), where \( i \) is the atom index in the range \([0, N_{\text{atoms}}]\) and \( \Delta z \) is the interval distance between two joint values, calculated as \( \Delta z = \frac{V_{\text{max}} - V_{\text{min}}}{N_{\text{atoms}} - 1} \), the predicted return expectation \( Q(s, a) = \sum_j z_j p_j(s, a; \theta) \), where \( p_j \) is the prediction probability of atom \( i \) in state-action. In the training process, the target probability \( \hat{p}_i \) at each atom \( i \) can be updated as:

\[
\hat{p}_i = \sum_{j=0}^{N_{\text{atoms}} - 1} \left[ 1 - \frac{z_i - V_{\text{min}}}{\Delta z} \right] p_j \left( s_{t+1}, \arg \max_{a_{t+1}} Q \left( s_{t+1}, a_{t+1}; \hat{\theta} \right) \right) \tag{11}
\]

where \([a, b)\) represents the bound of argument in the range \([a, b)\] and \( \hat{z}_j \) represents the estimated value for atom \( j \), equal to:

\[
\hat{z}_j = r_t + \gamma z_j \tag{12}
\]

The KL divergence, denoted by \( D_{\text{KL}}(\hat{P} \mid P) \), is applied to evaluate the difference between \( \hat{P} \) and \( P \), and the cross-entropy term is considered to be the loss function:

\[
L(\theta) = - \sum_i \hat{p}_i \log p_i (s, a; \theta) \tag{13}
\]
Finally, Noisy DQN is applied to balance the tradeoff between exploration and exploitation. A general linear layer with input $x$ and output $y$ can be represented by:

$$y = wx + b$$

(14)

where $w$ and $b$ denote the weights and bias, respectively. In Noisy DQN, $w$ and $b$ will be redefined by a Gaussian distribution. Therefore, the related noisy linear layer can be described as:

$$y = (\mu^w + \sigma^w \odot \epsilon^w) x + \mu^b + \sigma^b \odot \epsilon^b$$

(15)

where $\odot$ represents elementwise multiplication to increase the noise in the Gaussian distribution. $\mu^w$ and $\sigma^w$ ($\mu^b$ and $\sigma^b$) are the Gaussian distribution parameters for weights $w$ (bias $b$).

To speed up learning efficiency, we applied PER, a sampling-based optimization algorithm. For each data tuple $<s_t, a_t, r_t, d_t, s_{t+1}>$, where $d_t$ is a Boolean value to judge whether the agent reaches the destination at time $t$. The corresponding priority $p$ is positively related to $L(\theta)$:

$$p \propto L(\theta)$$

(16)

3.2.3 | Network architecture

As mentioned above, a state space is composed of the last three grayscale images with size $84 \times 84$, and the action space is a discrete space that includes eight different directions ($N_{actions} = 8$). Based on this setting, the input and output spaces of the DNN are respectively set to $3 \times 84 \times 84$ and $51 \times 8$. As shown in Figure 2, three two-dimensional convolutional layers are applied to the input $S_t$. After each convolution operation, batch normalization is further applied to prevent overfitting (Ioffe & Szegedy, 2015). The output from the convolutional layers is flattened and decomposed into two components. The first component consists of two fully connected (FC) layers with 512 rectifier units and $N_{atoms} \times N_{actions}$ rectifier units, respectively. The second component consists of two FC layers with 512 rectifier units and $N_{actions}$ rectifier units, respectively. Noisy terms are added to all the FC layers to encourage exploration at each interaction. Finally, we aggregate the outputs of the two components as the final output, including $N_{actions}$ values for each action in the action space. The activation functions in both convolutional and FC layers are the rectified linear unit nonlinearities (Nair & Hinton, 2010).

4 | EXPERIMENT ENVIRONMENT AND SCENARIOS

4.1 | Coding environment

The algorithm of this model is implemented using the Python programming language. PyTorch packages are applied to build a DNN for mapping the relationship between the state and action. In addition, OpenCV packages are used to collect and visualize data. The program runs on a computer with Ubuntu 18.04 in an environment that consists of an i7 CPU, 64 GB RAM, and two NVIDIA GTX 1080 Ti GPUs. The simulation process runs on the CPU while the DNN is trained on the GPUs. The hyperparameters setting used in this study can be found in Table 1.

4.2 | Scenarios

A virtual indoor environment with two exits named Exit$_a$ and Exit$_b$ respectively, is designed as the research environment. The room geometry is square with a side length of 100 and a wall width of 2.0. The radius of each pedestrian
A total of three scenarios are presented to evaluate the performance of our method. Those scenarios include: varying exit width ratio ($p_{ew}$) with a uniform pedestrian distribution; varying pedestrian distribution ratio ($p_{pd}$) with a uniform exit width; and varying exit opening times with a uniform exit width and a uniform pedestrian distribution. In the first scenario, the distribution of the pedestrians and the width of Exit $b$ ($w_b = 4.0 \times r$) are held constant while the width of Exit $l$ ($w_l$) is set to $4.0 \times r$, $6.0 \times r$, and $8.0 \times r$, respectively. Assuming $p_{ew}$ represents the ratio of the two exits, $w_b/w_l$, the three sub-scenarios include $p_{ew} = 1:1$, $p_{ew} = 1:1.5$, and $p_{ew} = 1:2$. This scenario aims to investigate the model's performance under different exit width ratios with uniform distribution of pedestrians. In the second scenario, both exits have the same width, $4.0 \times r$. A total of $m$ pedestrians are within the room but with different distributions. Assuming $p_{pd} = \frac{N_{exit,l}}{N_{exit,b}}$, where $N_{exit,l}$ ($N_{exit,b}$) represents the number of pedestrians whose closest exit is Exit $l$ (Exit $b$), three different distributions are designed, namely $p_{pd} = 1:1$, $p_{pd} = 1:2$, and $p_{pd} = 1:3$. This scenario investigates the model's performance in handling uneven congestion at the exits during the evacuation. In the third scenario, the distribution of pedestrians and the width of the two exits are held constant. However, Exit $l$ does not open at the initial frame while Exit $b$ remains open throughout the entire simulation process. In this scenario, the open time for Exit $l$ is set at the 15th, 30th, and 45th frames, respectively. This scenario creates a dynamic simulation environment as the pedestrians are unaware of when another exit is open. To compare the effectiveness and efficiency of our proposed method, different numbers of pedestrians ($m = 12, 24, 36$) are tested.

**FIGURE 2** Network architecture
in all three scenarios. We investigate the performance of methods from two perspectives: the total frames for evacuation; and the utilization efficiency of the two exits \( r_{\text{util}} \). Since \( N_l \) and \( N_b \) represent the number of pedestrians to evacuate from Exit\(_l\) and Exit\(_b\), respectively, \( r_{\text{util}} \) can be calculated as:

\[
\begin{align*}
    r_{\text{util}} &= \frac{\min \left( \frac{N_l}{w_l}, \frac{N_b}{w_b} \right)}{\max \left( \frac{N_l}{w_l}, \frac{N_b}{w_b} \right)} \\
    \text{with a range of } [0,1],
\end{align*}
\]

With a range of \([0, 1]\), \( r_{\text{util}} \) represents how efficiently Exit\(_l\) and Exit\(_b\) are utilized in general. The higher the value of \( r_{\text{util}} \), the more the efficiency of the two exits for the pedestrians to pass through during the evacuation. We only investigate \( r_{\text{util}} \) in the first two scenarios due to the delay imposed on opening Exit\(_l\) in the last scenario. The proposed MultiExit-DRL method is compared against AMFM (Zheng et al., 2015) and NLDM (Ren-Yong & Hai-Jun, 2010).

## 5 | RESULTS

To compare the performances of this model, we obtain screenshots of different frames during the evacuation process from all three scenarios (Figures 3–5). An interval of 10 frames is used for the first two scenarios, while an interval of 15 frames is used for the third scenario, given its longer simulation process. The total frames for pedestrians to evacuate the room and the utilization efficiency of the two exits are analyzed for the three designed scenarios.

### 5.1 | Model performances under different exit width ratio \( p_{\text{ew}} \)

We first evaluate the model performance under different \( p_{\text{ew}} \) with uniform distribution of pedestrians. Given different exit widths, pedestrians are expected to find appropriate exits to prevent unnecessary congestion, thus
leading to high exit utilization efficiency and low total frames (total time for all the pedestrians to evacuate). As expected, the total frames increase as the number of pedestrians increases for the same $p_{ew}$ (Table 2). It is reasonable that, with the same indoor environment, it takes a longer time for more pedestrians to evacuate the room. Compared with the other two methods, our proposed MultiExit-DRL method achieves the best performance in all conditions (Table 2). As the number of pedestrians increases (e.g., in the 36-pedestrian case), a larger performance gap is found between MultiExit-DRL and the other two methods, suggesting that MultiExit-DRL can efficiently allocate a larger number of pedestrians to exits. For example, in the 36-pedestrian case with $p_{ew} = 1:1$, the evacuation via AMFM and NLDM takes 115 and 106 frames, respectively, while the evacuation via MultiExit-DRL takes only 60 frames. Since we hold $w_b$ constant while incrementing the $w_l$, a faster evacuation is achieved by MultiExit-DRL as the wider exit allows more pedestrians to evacuate. This phenomenon proves that, after training with the DRL algorithm, pedestrians are able to recognize the wider exit ($\text{Exit}_l$ in this case) and evacuate more quickly.

The efficiency of exit utilization of this scenario is presented in Table 3. All the methods achieve decent performances with $p_{ew} = 1:1$ (i.e., $w_b = w_l$), suggesting that all three models can handle the situation where the two exits have the same width, creating the same attraction for all the pedestrians. However, the superiority of MultiExit-DRL is clear when the two exits become unbalanced. For example, in the 36-pedestrian case with $p_{ew} = 1:2$ (i.e., $w_l = 2w_b$), the values of $p_{ew}$ in AMFM and NLDM are 0.06 and 0.18, respectively, while the value of $p_{ew}$ in MultiExit-DRL is 1.00 (Table 3), suggesting that the efficiency of exit utilization in MultiExit-DRL is significantly better with an unbalanced exit width ratio compared with the other two methods. The superiority of MultiExit-DRL is well documented from the screenshots of frames during the evacuation process (Figure 3). When $p_{ew} = 1:1$, pedestrians

![FIGURE 3 Model performance for 36 pedestrians under different exit width ratios](image-url)
in AMFM and NLDM are clearly crowded at Exit₁, evidenced by the screenshots of the 20th, 30th, and 40th frames (Figure 3). This congestion is significantly due to the low efficiency of exit utilization with many frames of the AMFM and the NLDM method. In comparison, pedestrians in MultiExit-DRL are able to choose exits more appropriately, given the unbalanced $p_{pd}$, leading to the perfect efficiency of exit utilization with few frames.

5.2 | Model performances under different pedestrian distribution ratio ($p_{pd}$)

This scenario investigates the model’s performance in handling different initial distributions of pedestrians. Given the different distribution patterns (i.e., a different $p_{pd}$), pedestrians are expected to adjust their strategies during the evacuation to exit the room as fast as possible with high efficiency of exit utilization. As shown in Table 4, our proposed MultiExit-DRL method achieves the best performance in all of the designed conditions. Similar to the comparison in the first scenario, MultiExit-DRL significantly outperforms the other two methods as the number of pedestrians increases regardless of the variations of the initial distributions, thus demonstrating its great capability in handling large numbers of agents. In addition, the insensitivity to the different initial distributions in our method suggests that pedestrians have learned to adjust their strategies appropriately to prevent congestion at different initial locations. For instance, in the crowded 36-pedestrian case, pedestrians in MultiExit-DRL evacuate the room in 60 and 58 frames for $p_{pd} = 1:1$ and $p_{pd} = 1:3$, respectively (Table 4). Despite the fact that $p_{pd} = 1:1$ suggests an even distribution while $p_{pd} = 1:3$ suggests the number of pedestrians whose initial locations are closer to Exit₆ is three times as many as Exit₁, both rooms are evacuated within a similar number of frames (Table 4).
In terms of exit utilization, higher $r_{util}$ values are found for MultiExit-DRL in the crowded environments (cases with 24 and 36 pedestrians), where all $r_{util}$ values are above 0.8, suggesting great efficiency of exit utilization (Table 5). In the 12-pedestrian case, however, MultiExit-DRL presents low $r_{util}$ with uneven initial distributions ($p_{pd} = 1:2$ and $p_{pd} = 1:3$). This indicates that, in the current hyperparameter setting, pedestrians in MultiExit-DRL prioritize the closer exit in the uncrowded environment. Despite the low $r_{util}$, pedestrians in MultiExit-DRL can still evacuate the room faster than the other two methods, as evidenced by the low number of the total frames (Table 4). It is observed that the initial uneven distribution might cause congestion at the exit for certain methods, which is well documented by the screenshots of frames. When $p_{pd} = 1:2$ and $p_{pd} = 1:3$, pedestrians in AMFM clearly congest at Exit$_b$, as Exit$_b$ is the closest exit to the initial locations of most pedestrians. However, this congestion unavoidably results in low efficiency of exit utilization and longer evacuation time. In comparison, pedestrians in MultiExit-DRL can clearly adjust their evacuation strategies, leading to a balanced exit assignment. The 30th frame in all cases shows that, with our proposed MultiExit-DRL method, both exits are targeted with a balanced number of pedestrians, which largely increases the efficiency of exit utilization and reduces the evacuation time.

5.3 | Model performances under different opening times of Exit$_i$

The different opening times of a certain exit create a dynamic simulation environment where pedestrians are expected to recognize the newly opened exit and adjust their evacuation strategies accordingly. Similar to the
previous two scenarios, MultiExit-DRL requires the least frames for pedestrians to evacuate the room compared with AMFM and NLDM in all conditions (Table 6). As the number of pedestrians increases, the superiority of MultiExit-DRL becomes obvious. In addition, the earlier the Exit$_l$ opens, the faster the pedestrians in MultiExit-DRL evacuate, especially in a more crowded environment (e.g., the 24- and 36-pedestrian cases). Since Exit$_b$ is the only available exit before opening Exit$_l$, all pedestrians are moving in one direction towards Exit$_b$ (Figure 5). The discrepancy of pedestrians’ behaviors occurs when pedestrians start to be aware of the availability of Exit$_l$. In the case where Exit$_l$ opens at the 15th frame, about half of the pedestrians in MultiExit-DRL switch to the new exit, thus greatly speeding up the evacuation process. In comparison, pedestrians in AMFM and NLDM usually fail to respond in a timely manner (witness the AMFM pedestrians in the 45th frame when Exit$_l$ opens at the 15th frame).

To sum up, three different scenarios are designed to compare the performance of our proposed method, MultiExit-DRL, against AMFM and NLDM. The model’s performance is revealed by the total number of frames and the efficiency of exit utilization, $r_{util}$. The results indicate the remarkable superiority of MultiExit-DRL in terms of total frames, as pedestrians under the MultiExit-DRL method are able to evacuate the room in the least frames in all designed conditions. With the room becoming more crowded by adding additional pedestrians, the performance gap becomes more obvious, suggesting great generalization capability of the MultiExit-DRL method. As for the efficiency of exit utilization, MultiExit-DRL shows consistently high $r_{util}$ values in the first scenario with a varying $p_{ew}$, indicating that pedestrians under MultiExit-DRL can adjust their strategies according to different exit widths once the evacuation phase starts. MultiExit-DRL also exhibits high $r_{util}$ values in crowded environments in the second scenario with a varying $p_{ped}$. Although it presents low $r_{util}$ in the 12-pedestrian case, the evacuation is still completed within fewer frames compared with AMFM and NLDM. The underperformance of AMFM can be explained by its intrinsic design. As an improved adaptive multi-factor model, it utilizes an additional judgment to prevent pedestrians from frequently changing their targeted exit. Despite its great performance in uncrowded environments, pedestrians in AMFM fail to modify their strategies in a timely manner when more pedestrians are added to the environment. Therefore, congestion at one exit and idleness at the other are found in AMFM, especially in crowded environments. NLDM, as a nested logit discrete model, uses non-strict logical judgment,

### TABLE 2 Total frames for evacuation under different exit width ratios ($p_{ew}$) with 12, 24, and 36 pedestrians

| Methods   | 12 pedestrians | 24 pedestrians | 36 pedestrians |
|-----------|---------------|---------------|---------------|
|           | $p_{ew} = $   |               |               |
| AMFM      | 1:1 51 1:1.5 48 1:2 46 | 1:1 69 1:1.5 65 1:2 69 | 1:1 115 1:1.5 78 1:2 62 |
| NLDM      | 53 41 47      | 90 59 70      | 106 88 54     |
| MultiExit-DRL | 38 39 34  | 50 45 45  | 60 57 52  |

Note: Best performances among the three methods are highlighted in bold.

### TABLE 3 Utilization efficiency ($r_{util}$) under different exit width ratios ($p_{ew}$) with 12, 24, and 36 pedestrians

| Methods   | 12 pedestrians | 24 pedestrians | 36 pedestrians |
|-----------|---------------|---------------|---------------|
|           | $p_{ew} = $   |               |               |
| AMFM      | 1:1 0.71 1:1.5 0.25 1:2 0.18 | 1:1 0.71 1:1.5 0.90 1:2 0.29 | 1:1 0.89 1:1.5 0.36 1:2 0.06 |
| NLDM      | 1.00 0.93 0.67 | 1.00 0.75 0.29 | 0.89 0.50 0.18 |
| MultiExit-DRL | 1.00 0.93 0.70 | 0.85 0.93 0.70 | 0.80 0.83 1.00 |

Note: Best performances among the three methods are highlighted in bold.
potentially leading to frequent changes of targeted exits of its pedestrians, consequently leading to more frames. Unlike the other two methods, MultiExit-DRL provides a proper direction for each pedestrian under the current state via Rainbow DQN, allowing the pedestrians to choose the optimal directions of movement by given reward functions. It further allows pedestrians to rapidly explore other options instead of congesting at a certain exit.

To illustrate that our model can work with more pedestrians and in a $k$-door ($k > 2$) environment, we present simulation results for MultiExit-DRL in two more scenarios, three exits with 48 pedestrians (Figure 6) and four exits with 60 pedestrians (Figure 6).

### DISCUSSION

In this study, we presented a novel multi-exit evacuation simulation by integrating ORCA and DRL. In general, our method exhibits excellent performance in both the training and evacuation processes. Specifically, it outperforms other approaches due to its efficient learning speed, stability, ability to scale up, and efficient exit utilization.
Transcending popular ray-based state acquisition approaches that require intensive computational resources (Lee et al., 2018; Xu et al., 2020), we directly capture screenshots of the environment via OpenCV as the source of information, allowing the pedestrians to learn their surroundings quickly. Besides, by packing adjacent frames as input to the DNN, no external information (such as position and velocity) is required. Given the exponential growth in complexity of the ray-based methods when the number of pedestrians increases, the complexity of our method keeps to linear growth which makes it more efficient to perform simulations with large numbers of pedestrians. Stability is another merit of our approach. In the multi-exit simulation, pedestrians often hesitate when faced with multiple exits, given the limited designs of traditional mathematical models. Although research suggested that this problem can be mitigated by a proper probability assignment (Zheng et al., 2017), still the existing models fail to give stable results. In our method, pedestrians are granted a global perspective by the application of DRL for selecting the optimal action in the current state. Given the nature that DRL is committed to maximizing the long-term reward, pedestrians interact with the environment numerous times to learn the best action for the total return. Once the best state-to-action mapping function has been learned by the pedestrians, they take actions without hesitation, allowing the simulation process to be pretty stable. Our method is tested with a different number of pedestrians, and the results have shown excellent performance compared with other traditional methods. This scaling-up ability is due to the fact that the effectiveness of our approach largely depends on the reward function rather than the complexity of the environment. As the number of pedestrians increases, the state space expands accordingly, which in turn leads to longer training time. However, the increase in the number of pedestrians does not affect pedestrians taking optimal action to maximize the value of the reward function in the given state. Finally, the results showed excellent exit utilization as pedestrians in our approach hardly ever crowd at a certain exit. For traditional multi-exit selection methods, once a pedestrian chooses an exit at a certain frame, s/he moves towards the target exit, regardless of any changes, thus causing congestion that eventually reduces the evacuation efficiency. In our approach, however, instead of choosing a specific exit for a pedestrian in a given frame, an optimal direction is calculated based on DRL. This optimal direction prevents further crowding at a certain exit by allowing pedestrians to explore other opportunities. This behavior can be achieved via the implementation of an appropriate reward function via DRL but is difficult to express via traditional multi-exit selection simulation that utilizes mathematical models.

Despite the merits explained above, it worth noting that our model is designed under the following constraints. Limited by the discrete nature of Rainbow DQN, we adopted a discrete space with a total of eight possible directions. It turns out that, in a few cases, oscillation of movements occurred during the experiments, even in the absence of obstacles. The application of DRL algorithms with a continuous action space such as DDPG, PPO, or SAC might further improve the model performance. The total reward function in our method is composed of four sub-reward functions (the goal, collision, smooth, and penalty reward functions), each of which needs a hyperparameter to specify its importance. However, the balance among the four hyperparameters requires trial-and-error experiments, which are not sufficiently covered in our work. Besides, some assumptions are introduced in this article, for example that pedestrian size (the radius $r$) and maximum speed are set to the same value, and all pedestrians are fully aware of the environment. Therefore, our method is a medium-level behavior simulation model and is suitable for the scene with homogeneous pedestrians.

Future studies should focus on designing simpler and more reliable reward functions. It is acknowledged that the performance of DRL algorithms greatly relies on the number of training samples. Our approach uses a single-core data collection method that basically causes sub-optimal simulations under the same experimental configuration and reward function. This can be regarded as the result of incomplete exploration of agents. In the future, the potentially more advanced DRL methods, such as the APeX-DQN (Horgan et al., 2018), need to be explored. Moreover, as a simulation-oriented method, more pedestrian behaviors, such as panic and group behaviors, should be modeled in the future to satisfy the needs of real-world applications. In this study, we assume that pedestrians are fully aware of the environment so that we can transform the navigation problem into a Markov decision process. However, this assumption may not be ideal for more complex, large-scale scenarios. In the future, we plan to design a simulation method based on the hidden Markov decision process. Finally, the pedestrians in our
experimental design have the same size and velocity. However, in practice, we should not assume pedestrians to have uniform attributes. Our future research will extend this method to accommodate heterogeneous pedestrians.

7 | CONCLUSIONS

Multi-exit evacuation simulation is a key area that needs more attention due to its potential applications in public safety. Traditional multi-exit evacuation simulation methods largely rely on discrete multi-exit selection methods that are characterized by different factors such as distance, density, and exit width. These methods are, however, coupled with low exit utilization and congestion at the exits. In this article, we focused on multi-exit navigation in designed room micro-environments, of which all pedestrians are fully aware. Based on this assumption, we proposed a novel simulation model that integrates ORCA and DRL, referred to as the MultiExit-DRL, where local collision avoidance detection is achieved via ORCA, and movement direction is achieved via DRL. We further designed a DNN framework to facilitate state-to-action mapping. In the designed framework, successive screenshots (gray-scale images) are used as the raw state, and they have proven to be faster at data collection than ray-based state acquisition. The action space is further divided into eight isometric directions for pedestrians to vacate. Rainbow DQN, a DRL algorithm that integrates several advanced DQN methods, is applied to improve data utilization and algorithm stability. We compared our proposed MultiExit-DRL method with two traditional multi-exit evacuation simulation models, AMFM and NLDM, in three individual scenarios: varying pedestrian distributions with a uniform exit width; varying exit widths with a uniform pedestrian distribution; varying exit opening schedules with a uniform exit width and a uniform pedestrian distribution. The proposed MultiExit-DRL shows excellent learning efficiency in all of the designed experiments. It also shows excellent utilization of exits regardless of the number of pedestrians. Nevertheless, MultiExit-DRL has some limitations, such as the utilization of discrete action space and homogeneous pedestrian design. Further research should focus on solving these problems and investigate the model's performance in large areas with more complicated scenarios.

ACKNOWLEDGMENTS

This research was financially supported by the National Natural Science Foundation of China Grants (No. 41771410) and Key Project in Philosophy and Social Science of Ministry of Education of China (No. 19JZD023). The study was conducted at the Geoinformation and Big Data Research Lab at the University of South Carolina.

DATA AVAILABILITY STATEMENT

No third-party data were used in this study. The source code and videos for demonstration can be found on GitHub at https://github.com/XD1227/MultiExit-Rainbow.

ORCID

Xiao Huang https://orcid.org/0000-0002-4323-382X
Zhenlong Li https://orcid.org/0000-0002-8938-5466

REFERENCES

Alonso-Mora, J., Breitenmoser, A., Rufli, M., Beardsley, P., & Siegwart, R. (2013). Optimal reciprocal collision avoidance for multiple non-holonomic robots. In A. Martinoli, F. Mondada, N. Correll, G. Merroud, M. Egerstedt, M. Ani Hsieh, ... K. Stay (Eds.), Distributed autonomous robotic systems (Springer Tracts in Advanced Robotics, Vol. 83, pp. 203–216). Berlin, Germany: Springer.

Bareiss, D., & van den Berg, J. (2015). Generalized reciprocal collision avoidance. International Journal of Robotics Research, 34, 1501–1514. https://doi.org/10.1177/0278364915576234.

Bellemare, M. G., Dabney, W., & Munos, R. (2017). A distributional perspective on reinforcement learning. In Proceedings of the 34th International Conference on Machine Learning, Sydney, NSW, Australia (pp. 449–458), Red Hook, NY: Curran Associates.
Ben, X., Huang, X., Zhuang, Z., Yan, R., & Xu, S. (2013). Agent-based approach for crowded pedestrian evacuation simulation. *IET Intelligent Transport Systems*, 7, 55–67. https://doi.org/10.1049/iet-its.2011.0236.

Bertinetto, L., Valmadré, J., Henriques, J. F., Vedaldi, A., & Torr, P. H. (2016). Fully convolutional Siamese networks for object tracking. In G. Hua & H. Jégou (Eds.), *Computer vision—ECCV 2016 Workshops* (Lecture Notes in Computer Science, Vol. 9914, pp. 850–865). Cham, Switzerland: Springer.

Bode, N. W., & Codling, E. A. (2013). Human exit route choice in virtual crowd evacuations. *Animal Behaviour*, 86, 347–358. https://doi.org/10.1016/j.anbehav.2013.05.025.

Burnstedde, C., Klauck, K., Schadschneider, A., & Zittartz, J. (2001). Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physica A: Statistical Mechanics & Its Applications*, 295, 507–525. https://doi.org/10.1016/S0378-4371(01)00141-8.

Cao, K., Chen, Y., & Stuart, D. (2016). A fractional micro-macro model for crowds of pedestrians based on fractional mean field games. *IEEE/CAA Journal of Automatica Sinica*, 3, 261–270.

Cao, S., Fu, L., & Song, W. (2018). Exit selection and pedestrian movement in a room with two exits under fire emergency. *Applied Mathematics & Computation*, 332, 136–147. https://doi.org/10.1016/j.amc.2018.03.048.

Chen, P.-H., & Feng, F. (2009). A fast flow control algorithm for real-time emergency evacuation in large indoor areas. *Fire Safety Journal*, 44, 732–740. https://doi.org/10.1016/j.firesaf.2009.02.005.

Choi, J., Choi, J., & Kim, I. (2014). Development of BIM-based evacuation regulation checking system for high-rise and complex buildings. *Automation in Construction*, 46, 38–49. https://doi.org/10.1016/j.autcon.2013.12.005.

Chopard, B., & Droz, M. (1998). *Cellular automata*. Berlin, Germany: Springer.

Curtis, S., & Manocha, D. (2014). Pedestrian simulation using geometric reasoning in velocity space. In U. Weidmann, U. Kirsch, & M. Schreckenberg (Eds.), *Pedestrian and evacuation dynamics 2012* (pp. 875–890). Cham, Switzerland: Springer.

Davidich, M., Geiss, F., Mayer, H. G., Pfaffinger, A., & Royer, C. (2013). Waiting zones for realistic modelling of pedestrian dynamics: A case study using two major German railway stations as examples. *Transportation Research Part C: Emerging Technologies*, 37, 210–222. https://doi.org/10.1016/j.trc.2013.02.016.

Dijkstra, J., Jessurun, J., & Timmermans, H. J. (2002). A multi-agent cellular automaton model of pedestrian movement. In M. Schreckenberg & S. D. Sharma (Eds.), *Pedestrian and evacuation dynamics* (pp. 173–180). Berlin, Germany: Springer.

Fiorini, P., & Shiller, Z. (1998). Motion planning in dynamic environments using velocity obstacles. *International Journal of Robotics Research*, 17, 760–772. https://doi.org/10.1177/027836499801700706.

Fortunato, M., Azar, M. G., Piot, B., Menick, J., Osband, I., Graves, A., ... Pietquin, O. (2017). Noisy networks for exploration. Preprint, arXiv:1706.10295.

Frank, G. A., & Dorso, C. O. (2011). Room evacuation in the presence of an obstacle. *Physica A: Statistical Mechanics & Its Applications*, 390(11), 2135–2145. https://doi.org/10.1016/j.physa.2011.01.015.

Geraerts, R. (2010). Planning short paths with clearance using explicit corridors. In *Proceedings of the 2010 IEEE International Conference on Robotics and Automation*, Anchorage, AK (pp. 1997–2004). Piscataway, NJ: IEEE.

Godoy, J. E., Karamouzas, I., Guy, S. J., & Gini, M. (2016). Implicit coordination in crowded multi-agent navigation. In *Proceedings of the 13th AAAI Conference on Artificial Intelligence*, Snowbird, UT (pp. 2487–2493). Menlo Park, CA: AAAI.

Güçkıran, K., & Bolat, B. (2019). Autonomous car racing in simulation environment using deep reinforcement learning. In *Proceedings of the 2019 Innovations in Intelligent Systems and Applications Conference*, Izmir, Turkey (pp. 329–344). Piscataway, NJ: IEEE.

Guo, R.-Y., Huang, H.-J., & Wong, S. (2012). Route choice in pedestrian evacuation under conditions of good and zero visibility: Experimental and simulation results. *Transportation Research Part B: Methodological*, 46, 669–686. https://doi.org/10.1016/j.trb.2012.01.002.

Gupta, S., Davidson, J., Levine, S., Sukthankar, R., & Malik, J. (2017). Cognitive mapping and planning for visual navigation. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI (pp. 2616–2625). Piscataway, NJ: IEEE.

Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. Preprint, arXiv:1801.01290.

Haghani, M., & Sarvi, M. (2016). Human exit choice in crowded built environments: Investigating underlying behavioural differences between normal egress and emergency evacuations. *Fire Safety Journal*, 85, 1–9. https://doi.org/10.1016/j.firesaf.2016.07.003.

Haghani, M., Sarvi, M., Ejtemai, O., Burd, M., & Sobhani, A. (2015). Modeling pedestrian crowd exit choice through combining sources of stated preference data. *Transportation Research Record*, 2490, 84–93. https://doi.org/10.3141/2490-10.

Han, Y., Liu, H., & Moore, P. (2017). Extended route choice model based on available evacuation route set and its application in crowd evacuation simulation. *Simulation Modelling Practice & Theory*, 75, 1–16. https://doi.org/10.1016/j.simpat.2017.03.010.
Zhou, M., Dong, H., Zhao, Y., Ioannou, P. A., & Wang, F.-Y. (2019). Optimization of crowd evacuation with leaders in urban rail transit stations. *IEEE Transactions on Intelligent Transportation Systems*, 20, 4476–4487. https://doi.org/10.1109/TITS.2018.2886415.

Zia, K., & Ferscha, A. (2009). A simulation study of exit choice based on effective throughput of an exit area in a multi-exit evacuation situation. In *Proceedings of the 13th IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications*, Singapore (pp. 235–238). Piscataway, NJ: IEEE.

---

**How to cite this article:** Xu D, Huang X, Mango J, Li X, Li Z. Simulating multi-exit evacuation using deep reinforcement learning. *Transactions in GIS*. 2021;25:1542–1564. https://doi.org/10.1111/tgis.12738

---

**APPENDIX**

![Figure 6](image-url)  
*Figure 6*  MultiExit-DRL in two designed indoor micro-environments: (a) three exits with 48 pedestrians; and (b) four exits with 60 pedestrians.