Generating Emergency Evacuation Route Directions Based on Crowd Simulations with Reinforcement Learning

1st Ahmet Emin Ünal
Research and Development Center
adesso Turkey
İstanbul, Turkey
ahmet.unal@adesso.com.tr

2nd Cengiz Gezer
Research and Development Center
Panasonic Electric Works Turkey
İstanbul, Turkey
cengiz.gezer@tr.panasonic.com

3rd Burcu Kuleli Pak
Research and Development Center
adesso Turkey
Istanbul, Turkey
burcu.kuleli@adesso.com.tr

4th V. Çağrı Güngör
Department of Computer Engineering
Abdullah Gül University
Kayseri, Turkey
cagri.gungor@agu.edu.tr

2022 Innovations in Intelligent Systems and Applications Conference (ASYU) | 978-1-6654-8894-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/ASYU56188.2022.9925560

***Abstract*—In an emergency, it is vital to evacuate individuals from the dangerous environments. Emergency evacuation planning ensures that the evacuation is safe and optimal in terms of evacuation time for all of the people in evacuation. To this end, the computer-enabled evacuation simulation systems are used to generate optimal routes for the evacuees. In this paper, a dynamic emergency evacuation route generator has been proposed based on indoor plans of the building and the locations of the evacuees. To generate the optimal routes in real-time, a reinforcement learning algorithm (proximal policy optimization) is presented. Comparative performance results show that the proposed model is successful for evacuating the individuals from the building in different scenarios.**

***Index Terms*—emergency evacuation, crowd simulation, path planning, reinforcement learning, deep learning**

I. Introduction

Emergency evacuation is the movement of a group of people from a threat which can be considered as an emergency case. These threats are including (but not limited to) natural disasters, fire, military and terror attacks. Even the cyber-attacks are threats due to widespread use of on cyber-physical systems in buildings. These threats can happen in high office buildings, stadiums, schools, hospitals and even outdoors. Emergency evacuation planning ensures that the evacuation is safe and optimal in terms of evacuation time for all of the people in evacuation.

In [1], the significance of the evacuation planning can be better understood with real-world examples. In March 2018, a fire happened due to an electric arc at a supermarket in Russia and all 64 victims died from toxic smokes as the evacuation instructions did not lead them to safe regions, while the highest building in Osaka Japan managed to evacuate more than 1000 people unharmed during the great earthquake in 2011 thanks to well-organized emergency planning.

Recently, the emergency evacuation simulation tools are utilized for generating computer aided evacuation plans and analyzing the evacuation times for the buildings and based on simulating crowd dynamics and pedestrian motions. These tools are used for generating regulations to use in a possible evacuation case. The existing evacuation simulation tools are based on cellular automata, agent-based models, social force models, queuing models, particle swarm optimization models and fluid-dynamic models. In general, the existing path planning methods take significantly high amount of time as the complexity of the building and the numbers of the people increase [1]. Such negative impact cannot be tolerated for generating real-time evacuation guidance. Moreover, the social force models and cellular automata-based evacuation simulation tools fail on building a universal yet specific pedestrian model.

To address these challenges, in this paper, a dynamic emergency evacuation route generator has been proposed based on indoor plan of the building and the locations of the evacuees. To generate the optimal routes in real-time, a reinforcement learning algorithm (proximal policy optimization [2]) is presented. In the proposed reinforcement learning model, the evacuation problem is defined by choosing the environment as the building and the agents as the evacuees who has evacuation directions as actions. Comparative performance evaluations demonstrate that the agents show behaviors of choosing the shortest path. In addition, if the model predicts that a crowd may block the path, it changes its path accordingly. The agents also avoid the nodes with high danger value most of the time. Moreover, the agents create natural clusters and move in these clusters. These clusters of agents are also divided if the model decides on splitting the agents on different paths. Clustering people may have positive effects in case of an emergency. This is because the people will be able to help each other in a group,

TABLE I
COMPARISON OF BUILDING EVACUATION MODELS WITH ARTIFICIAL INTELLIGENCE AGENT BEHAVIOR

Model	Grid/ Structure	Perspective	Movement	Validation	Route choice
Legion	Continuous	Individual perspective	Inter-person distance, Conditional	C,FD,PE,3P	Conditional
EPT	Fine network	Individual perspective	User's choice, Conditional	FD	Shortest, conditional
Myriad II	Coarse network, Fine network, Continuous	Individual perspective	Density correlation, User's choice, Inter-person distance, Acquiring knowledge	PE, 3P	Various
MassMotion	Continuous	Individual and Global perspective	Conditional	C,FD,PE,OM	Shortest, conditional
MASSEgress	Continuous	Individual perspective	Conditional	PE,OM	Conditional – visual perception
Proposed Model	Coarse network	Global perspective	Conditional	C	Conditional – AI/RL

C: Validation against codes.
FD: Validation against fire drills or other people movement experiments/trials.
PE: Validation against literature on past experiments (flow rates, etc.).
OM: Validation against other models.
3P: Third party validation.
N: No validation work could be found regarding the model.
Note. Adapted from [7, Tab. 1].

where it is not possible in individual movements of agents. Importantly, the calculated actions as evacuation directions may be given to the mobile device of the individuals so that a fast and safe evacuation on a building would be fulfilled. Thus, the system would be giving real-time and personalized evacuation directions to each evacuee.

This study is organized as follows. In Section 2, we describe the related work in this field. In Section 3, we introduce the proposed method for generating individual evacuation instructions. In Section 4, we present performance evaluations of the proposed model, possible emergency cases and scenarios. Finally, the last section concludes the paper. With this study, a reinforcement learning solution to the emergency evacuation problem will be proposed as a contribution to the literature. The new application is aimed to decrease the evacuation time of all evacuees in a given building.

II. RELATED WORK

Recently, Gwynne et al. [3] have assessed twenty two different computer aided evacuation models classified into three approaches, namely optimization, simulation, and risk assessment. Atila et al. [4] have proposed an intelligent indoor evacuation model, which gives evacuation instructions to individuals in case of a building fire. It is shown that the proposed model shortens the uncertainty time in a case of an emergency by providing clear information and directions and will help to decrease the congestion. In addition, recent studies provide the detection of the obstacles or abnormal movements in crowds [5], [6] based on Internet of Things and image processing models. Recent advances in indoor position tracking also enables to pinpoint the location of a specific individual [8]. These recent studies motivate researchers to develop individual evacuation models for evacuees and getting feedback from both people and the building in an automatized manner.

To provide the people with the evacuation plan in case of an emergency, the determined regulations are published in the buildings. However, the building structure may change or paths may be blocked by obstacles or crowds in case of an emergency. An efficient evacuation system should take these dynamic situations into consideration so that the evacuation will be successful. Hence, the dynamic instructions for the evacuation should be accessible for the individuals in real-time in order to take the instructions and perform necessary actions. These instructions may be announced to the individuals, e.g., via speakers. However, these instructions are predicted to be different according to the position of the evacuee. Hence, a method for publishing these instructions in real-time for each individual, such as sharing the instructions via mobile devices (with a prior installed mobile application or SMS), may be more practical. The possible solution for this complex task of the generating a dynamic path for each of the numerous individuals in real-time is investigated in this paper in detail.

The crowd simulations assist the simulation for generating a real-case approach on an emergency case as the pedestrian movement and behavioral analysis of the individuals have to be analyzed [9]. In [1], the authors proposed a neural network-based path planning method for evacuations to construct real-time optimal paths for evacuation. To this end, reinforcement learning can also be used for generating the required path for optimal evacuation and the reinforcement learning method is favorable for problems where environment is known [10]. In this reinforcement learning model, the evacuation problem can be defined by choosing the environment as the building and the agents as the evacuees who has evacuation directions as actions. The models-based on artificial intelligence (AI) incorporate occupants performing actions and movements toward a specified goal, and these models attempt to simulate human intelligence in the evacuation instead of probabilistic, conditional (rule based), or implicit behaviors [7]. The existing building evacuation models based on AI are summarized in Table I and compared with the proposed model.

III. MATERIALS METHODS

Before starting to describe the implementations and the proposed methods, the problem, environment observations and the agent actions are defined in the following. The problem is providing the individuals with directions, which will

eventually generate the route for evacuation, thus the actions will be these directions at a given time. The problem can be summarized as giving short instructions for each individual in a timestamp. The observation from the environment will be the locations of individuals or crowds and the location of the source of dangers, which is the reason of evacuation.

Representing this problem as a reinforcement learning (RL) problem will enable us to integrate the neural networks for decision selection and give specific agent based directions to the individuals in an emergency case. If we select the building or an area, which the evacuation takes place as an environment for the reinforcement learning and select the individuals as the agent of the systems, we can create a multi-agent reinforcement learning approach on the problem. The trained model, which generates the optimal route with the help of reinforcement learning, can be used in the same system for possible emergency cases. It would be possible to give the directions to the individuals as we were giving the actions to the agents in the RL model.

Importantly, as the problem is an agent based solution, which concerns individualistic perspective, a multi agent reinforcement learning (MARL) approach is found to be more appropriate for this kind of problem. If we represent the problem in a game theory point of view, the task type is fully cooperative and the agent awareness is tracked or type-aware.

A. Graph representation of the problem

Based on a coarse network approach, the rooms, halls, elevators and staircases in the buildings are represented as the nodes of the building graph. We are generating this graph using the Building Information Model (BIM) of a certain structure. First, the spatial topology based on the bounding boxes of the structure areas are generated and divided into different bounding box segments for simplifying the graph structure. The simplified and limited graph representation would be the environment of the reinforcement learning (RL) problem. The reason for simplifying the representation of the graph, and thus, every node will have limited number of edges, is to provide the agents with discrete action spaces based on a node location. In this approach, each of the nodes in the graph will have four horizontal and two vertical edges at most for simplifying the network. The proposed model will be able to simulate and work on any type of building as long as it is represented as a graph as it is depicted.

To generate the environment for simulating crowds and human behaviors, we have used the Unity Engine. The visualization advantages to simulate crowd behaviors and the official Machine Learning Agents (ML-Agents) library of the Unity Engine, which enables the engine to work with a python application programming interface, are the main reasons for selecting such an environment. It is decided that the graph representation will be inside the Unity, while the neural network model will be trained in Python and afterwards it will be tested and executed in the engine again. Unity Machine Learning Agents toolkit (ML-Agents) is a machine learning library which Unity officially supports and develops. This open-source project enables simulations to serve as environments for training intelligent agents. First, the stable release date of the product is April 30, 2020. The toolkit is simple-to-use, and thus, is selected for creating the evacuation model in this study. The agent brains and the connection to the neural network model are coded in C, but the model is trained in Python via the connection to the ML-Agents library.

The user specific features can also be given to the agents for analyzing different behaviors of the individuals. These features may change according to age of the people, physical handicap of the people, who may have hard times in staircases, sick-abed people who should avoid narrow hallways, family clusters, or even the patience of the people, who waits in a congestion. Taking these features into consideration will enable the simulation to be more realistic. To have realistic scenarios, it is decided to simulate an office environment and to enable agents to walk around the building interior. The waypoints are gathered from the graph representation of the building. The waypoints in the simulation have been generated as the doors or separating lines of the room segments for providing the agents to follow the given directions.

The policy network is created for agents to decide which waypoints they will go towards. Policies have been specialized for the building. This means a model that is trained for a building structure will not have any use on another type of building structure. It also means a change in the building structure will raise a need for retraining the policy. Policy input will consist of current agent position, positions of the other agents, danger position, and future decisions of the other agent, which will then be mapped to policy output as the next waypoint step of the current agent.

B. Environment

The environment will be the graph representation of the buildings or open areas that the evacuation will take place. The assumptions are the rooms, where the people are present, are known and the system can reach and give directions to individuals for them to follow. The room or building layout does not change without any notice. Unknown change in the structure of the building may make the model non-operational. It is important to note that the selected paths should not be congested with obstacles as there should not be any unknown or not notified blockades. The agent's task is getting the most reward by avoiding danger and escaping from the building.

The undirected graph representation of the building should be prepared in initial setup of the application by utilizing the BIM model of a building. The graph is created by generating nodes for each room and by splitting rooms larger than the other rooms where there are more than one exits from the room to another in the same direction. The demo environment for training and testing the proposed algorithm is configured as shown in Fig. 1 as a one floor of a sample building, where the exit room is the target to reach. Also, the graph representation of the environment can be seen in Fig. 2. In this figure, the rooms are named from 0 to 5, where 0 corresponds to the exit, 5 corresponds to the dynamically created danger zone

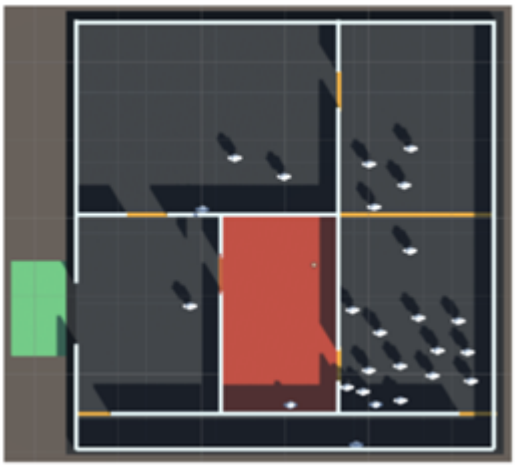

Fig. 1. The sample office, which contains some agents and the source of the danger, is visualized as the room with red floor. The split lines of the hall have been created for simplifying the correspondence to the graph representation.

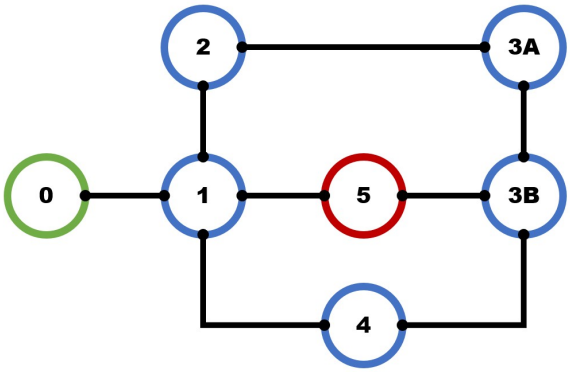

Fig. 2. The graph representation of the example environment.

in this time step, and the room 3 is splitted because of its size or complexity. In the environment the agents follows the path given by the model with a randomly assigned constant movement speed (between 1m/s and 5m/s).

C. Reinforcement Learning Parameters

The action space, observation space, and the rewards of the reinforcement learning model are defined under the following sections.

1) Action Space: The action space of an agent is limited in six directions and a "no movement" action which corresponds to a vector of magnitude seven. The agent can go north, south, west, or east (if vertical paths are defined, such as staircases, elevators, the agent can go up and/or down). At each collision with a new room on the simulation or when the agent arrives to its destination, the agent can decide on a new action. There are seven actions that can be taken by the agent. These actions can be listed as staying in the room, going to the north direction from the current room, going to the south direction from the current room, going to the west direction from the current room, going to the east direction from the current room, going upstairs from the current building floor, and going downstairs from the current level. If an agent chooses an action, which there is not a way in the graph, then the action is changed to "stay in the room" action for simplification purposes.

2) Observation Space: The observations are consisting of all of the environment as the problem can be defined as fully observable Markov Decision Process. Because this problem is a multi-agent problem, the future decisions of the other agents are added to the observation as well. The location of the agent, the crowdedness of the each room (which can be defined as the number of agents in a room at a given time), the future predicted crowdedness (which will be calculated by taking current output of the model into consideration as next location of the agents), and the location of the danger points need to defined and computed.

If the specified building has number of rooms defined as 'n', the current and future crowdedness of the rooms will be defined with a vector of n magnitude, which will have number of agents in every corresponding node. These two vectors are normalized according to the total number of agents in the simulation.

The rooms with danger constants will be fed to model as a vector of magnitude n, where the most dangerous node have a value of 1 while the safest node will have the value of 0. The danger value may spread to neighbor nodes with elapsing time for realistic danger predictions, but it has been decided that more research is needed for such implementation on the transmissions of these possible hazards. The position of the agent will be fed to the model as a one hot encoded vector with n magnitude. The agent is currently in has the value of 1 while the other nodes have the value of 0.

Overall, the state space has the dimension of $4{*}n$ as a vector, but for providing a simple memory implementation for the agents it has been decided to stacking state space 4 times and thus combining the last 3 previous state values with the current state vector. The stacked value count is not experimented and an optimal stack value may be found. This value will increase the input size significantly as the number of nodes increases in complex and large buildings, so it may be favorable to keep the stacked value as low as possible.

3) Rewards: In the aforementioned sample office environment, the model has been put to test and the following optimal reward values has been found and used in more complex tasks. These values may change depending on building layout structures, but the found values gave successful results on various sample structures. This value also referred as the safety value in this study.

Equation 1 defines the selected reward function. If an agent escapes, a positive constant reward of $r_t^{reached}$ is given to the agent. If an agent changes its room or decides on staying, a reward of negative r_t^{step} is given to the agent. If a person encounters a danger zone, a negative reward of r_t^{danger} is given to the agent, which is calculated as $r_t^{danger} = -\omega^{value} {*} c^{danger}$ where the ω^{value} is the estimated level of the

danger which may be between 0 and 1 and a danger constant c^{danger}. Also, a negative reward of $r_t^{\Delta time}$ is also given to the agent, which is calculated as $r_t^{\Delta time} = -\omega^{\delta time} * c^{time}$ at each decision step, where $\omega^{\delta time}$ is the time it takes for a_{t-1} to finish and c^{time} is a configured constant.

$$R_t = r_t^{reached} + r_t^{step} + r_t^{danger} + r_t^{\Delta time} \quad (1)$$

The reward $r_t^{reached}$ and r_t^{step} are configured as +100 and -1 respectively. The time constant c^{time} and danger constant c^{danger} is selected by experimenting on the aforementioned sample environment and found as 20 and 50, respectively. These values may change according to the building structure types. If all the agents escapes from the building to the designated goal or escape point, the environment is considered as solved and the episode finishes. If the number of steps reaches to maximum step, then the environment resets and the environment is considered as not solved and the episode is halted.

D. Policy

For providing agents to decide on one of the actions in the action space by observation and reward, a deep neural network structure is used. Current state-of-the-art projects on video games, 3D Locomotions and board games, such as Go, are real-world examples where deep neural networks models have been utilized. It has been found that the policy gradient methods shows considerably positive results on deep neural networks [2]. Thus, in this study, a policy gradient method, proximal policy optimization (PPO) is used for optimizing the neural network.

IV. Performance Results

The policy is trained in parallel with many clone of the same environments for increasing the speed of the training and the results are plotted as follows. The environment is built with numerous rooms, which will contain people in random position and in random movement. The threat source will happen in a random position and the evacuation time and the reward of agents which corresponds to the safety of agents evaluated for performance comparisons.

The model is trained in a sample office environment, which has a simple layout diagram as shown in Fig. 1. The layout transferred to Unity Engine and after creating the graph representation, we started to train the model with one agent and multi-agent (total number of forty agents). The mean and the standard deviation of the rewards of these experiments can be seen in the Fig. 3 and Fig. 4 for one agent configuration, and in Fig. 5 and Fig. 6 for multi agent configurations of the simulation. The performance results show that the proposed model is successful for evacuating the individuals from the building in different scenarios. For testing purposes, randomized positions of the individuals on the test building are used.

The simulation run on a PC with 16GB RAM, Intel i7 10th gen processor and NVIDIA RTX2060 GPU. The trained reinforcement learning model inference time for each action

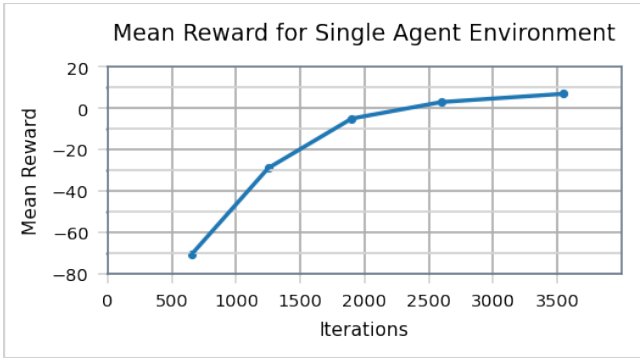

Fig. 3. Average of acquired rewards in relation to iteration steps of one agent in the environment.

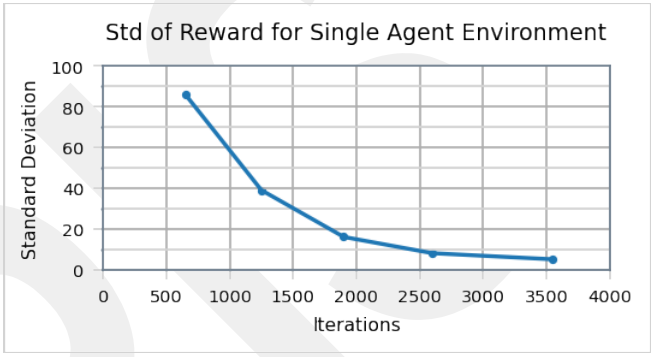

Fig. 4. Standard deviation of acquired rewards in relation to iteration steps of one agent in the environment.

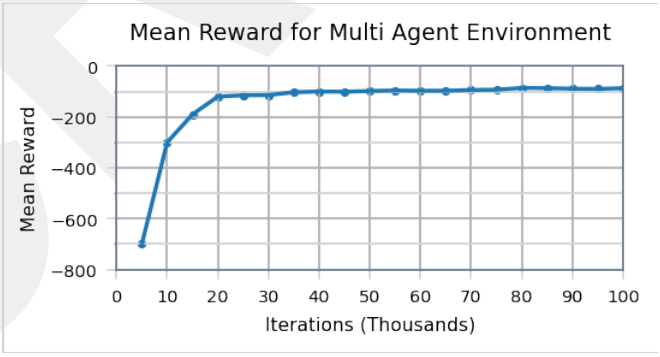

Fig. 5. Average of acquired rewards in relation to iteration steps in the multi-agent environment.

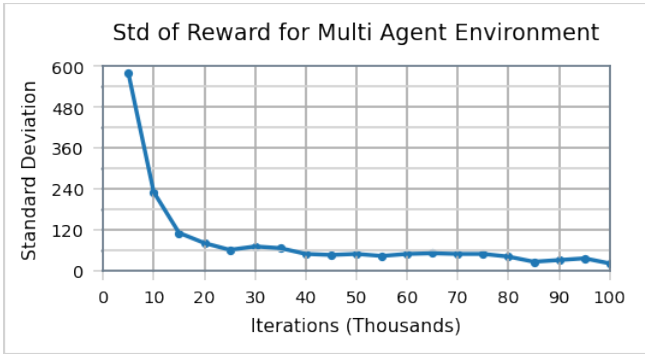

Fig. 6. Standard deviation of acquired rewards in relation to iteration steps in the multi-agent environment.

query of an agent is measured between 0.015 and 0.002 seconds in average for different environment configurations. The agent observations can be fed to the model in parallel on high-end cloud computing services to accelerate the inference. The model returns at least 66 output per second, in which the model can be considered as real-time capable.

The actions of the agents, that will be given as evacuation instructions in the simulation, will be calculated by evaluating the position of the agent in the simulation. While testing or running the application in a real world environment, the instructions will be given to the individuals in a way that every individual will be able to go to the door near to themselves. For example, if there are five people in a room and the trained model decides to give them the instructions of sending two of them to east direction and three of them to west directions, the instructions for sending east directions will be given to those people who are near to the east door of the room in the real world application. This is predicted to prevent possible congestions happening and help the individuals to follow the instructions to evacuate faster, similar to how it was observed in the simulation.

The proposed model is also compared with a baseline model, a modified shortest path algorithm (A*). The danger zones adds a constant cost to the heuristic calculation, which results on prioritization of the non-danger nodes. The shortest path is calculated each time step and followed by its respected agent. The movement speed of the agents are set to a constant value ($5m/s$) during comparison. The performance of the model is measured with complete evacuation time and the total safety value (reward calculation), and used for comparison. In a given building layout, these values are dependent to the number of evacuees in the building and the location of the danger zone. If the danger zone blocks the path of evacuees, the total safety value decreases dramatically. The acquired changes in the sample office environment can be seen in Fig. 7.

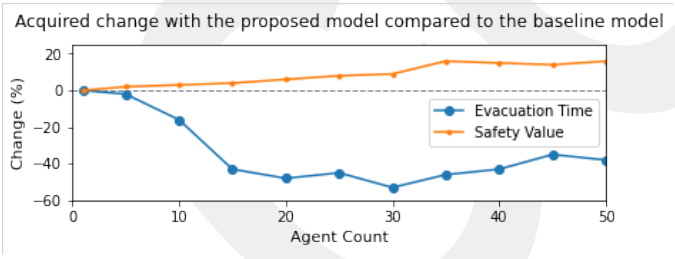

Fig. 7. The average acquired change in the complete evacuation time and the safety value in percentage when we used the proposed model in the same environment configuration instead of the baseline modified shortest path algorithm.

Conclusions

Recently, the emergency evacuation simulation tools are utilized for generating computer aided evacuation plans and analyzing the evacuation times for the buildings and based on simulating crowd dynamics and pedestrian motions. These tools are used for generating regulations to use in a possible evacuation case. In this paper, a dynamic emergency evacuation route generator has been proposed for buildings based on indoor plans of the building and the locations of the evacuees. To generate the optimal routes in real-time, a reinforcement learning algorithm (proximal policy optimization [2]) is presented. In the proposed reinforcement learning model, the evacuation problem is defined by choosing the environment as the building and the agents as the evacuees who has evacuation directions as actions.

Comparative performance evaluations demonstrate that the agents show behaviors of choosing the shortest path. In addition, if the model predicts that a crowd may block the path, it changes its path accordingly. The agents also avoid the nodes with high danger value most of the time. Importantly, the calculated actions as evacuation directions may be given to the mobile device of the individuals so that a fast and safe evacuation in a building would be fulfilled. Thus, the system would be giving real-time and personalized evacuation directions to each evacuee. Future work includes predicting where the center of the danger is based on human behavior and predicting the people who needs help by analyzing their movements and their responds to the given directions. Also, in the future, it may be possible to use the proposed model for finding the rooms, where the possible evacuation problems will happen, and thus, recommending possible locations for emergency exits.

References

[1] Y. Peng, S. Li and Z. Hu, "A self-learning dynamic path planning method for evacuation in large public buildings based on neural networks", Neurocomputing, vol. 365, pp. 71-85, 2019.

[2] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms", OpenAI, 2017.

[3] S. Gwynne, E. Galea, M. Owen, P. Lawrence and L. Filippidis, "A review of the methodologies used in the computer simulation of evacuation from the built environment", Building and Environment, vol. 34, no. 6, pp. 741-749, 1999.

[4] U. Atila, I. R. Karas, M. K. Turan, and A. A. Rahman, "Automatic generation of 3D networks in CityGML and design of an intelligent individual evacuation model for building fires within the scope of 3D GIS". Innovations in 3D Geo-Information Sciences, pp. 123-142, 2014.

[5] R. Mehran, A. Oyama and M. Shah, "Abnormal crowd behavior detection using social force model", 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009.

[6] H. Huang, C. Hsieh and C. Yeh, "An Indoor Obstacle Detection System Using Depth Information and Region Growth", Sensors, vol. 15, no. 10, pp. 27116-27141, 2015.

[7] E. D. Kuligowski, R. D. Peacock, B. L. Hoskins, "A Review of Building Evacuation Models, 2nd Edition", Technical Note (NIST TN), Gaithersburg: NIST Pubs, 2010.

[8] . Mainetti, L. Patrono and I. Sergi, "A survey on indoor positioning systems", 2014 22nd International Conference on Software, Telecommunications and Computer Networks (SoftCOM), 2014.

[9] S. Wong, Y. Wang, P. Tang and T. Tsai, "Optimized evacuation route based on crowd simulation", Computational Visual Media, vol. 3, no. 3, pp. 243-261, 2017.

[10] D. Cruz and W. Yu, "Path planning of multi-agent systems in unknown environment with neural kernel smoothing and reinforcement learning", Neurocomputing, vol. 233, pp. 34-42, 2017.

[11] H. Chu, J. Yu, J. Wen, M. Yi and Y. Chen, "Emergency Evacuation Simulation and Management Optimization in Urban Residential Communities", Sustainability, vol. 11, no. 3, p. 795, 2019.

[12] G. Mishra, S. Mazumdar and A. Pal, "Improved algorithms for the evacuation route planning problem", Journal of Combinatorial Optimization, vol. 36, no. 1, pp. 280-306, 2016.