Integrating Dynamic Environment Modelling with Reinforcement Learning for Fire Safety Design

Isabelle Fitkau[1]

[1]Department of Civil Systems Engineering, Institute of Civil Engineering, Technische Universität Berlin, Gustav-Meyer-Allee 25, 13355 Berlin, Germany

E-Mails: i.fitkau@tu-berlin.de

Abstract: Escape path planning belongs to the domain of fire safety design in buildings, ensuring the safety of individuals during emergencies. Applied Reinforcement Learning (RL) research has focused on optimizing training for simple evacuation scenarios. Research findings indicate that agents can learn to escape environments using path-planning methods. However, it has yet to be considered in depth how immediate changes to the environment affect the learning process. By using dynamic environment modeling, the agent can learn to modify floor plans based on the results of escape paths. This approach leads towards a performance-based validation tool, where the results of evacuation path planning can be used to, e.g., support regulatory compliance checks. Our work focuses on creating a reinforcement learning environment that simulates a floor plan to train the agent to identify escape paths and strategically optimize the layout by placing the exits for evacuation. The environment creation is based on a grid-based environment using a discrete action space. This allows for fast prototyping since learning in environments with fewer choices leads to less variability of outcomes and facilitates comparison with existing research, which the study addresses regarding reproducibility and comparability of the environments and the robustness of the algorithm's outcomes.

Keywords: reinforcement learning, fire safety design, dynamic environment modeling, performance-based validation

Erschienen in Tagungsband 35. Forum Bauinformatik 2024, Hamburg, Deutschland, DOI: 10.15480/882.13532

1 Introduction

Fire safety is a multifaceted domain within construction, impacting various other areas. For example, including openings can influence structural design, while fire safety systems like sprinklers can affect MEP design. Additionally, the layout of escape routes can significantly impact the overall floor plan. The collaboration between floor plan designers and those responsible for compliance checking is

not just beneficial but crucial for the safety of the building's occupants. Ensuring compliance with prescribed and safety goal-oriented fire safety requirements involves a lengthy process, as automated methods also demand extensive knowledge [1]. The building design must be continually updated to reflect changes made by designers from other fields to meet fire safety standards. Escape paths, crucial for ensuring the safety of occupants, represent a fire safety aspect of floor plan design that must comply with cross-domain fire safety requirements. These are closely linked to room layouts, door placements, component parameters, and systems such as fire extinguishers. The principle of reinforcement learning (RL), which utilizes machine learning to optimize a simulation environment through continuous adaptation and learning, has been explored in existing research [2] - [4]. This research primarily focuses on optimizing training specifications to determine escape paths in static environments that do not change dynamically.

However, the application of RL in modifying floor plans based on escape path knowledge has yet to be thoroughly explored. A key question is whether an RL agent can be trained to develop fire-safe floor plans or improve existing ones by simulating escape path scenarios and making corresponding adjustments to the layout. The goal of setting up an environment for training an RL agent to recognize escape paths and strategically position fire safety elements, such as doors, based on pathfinding outcomes by incorporating dynamic environment modeling into the agent's learning process has been addressed only in a preliminary form [5]. This work presents the initial approaches that facilitate the development of a performance-based validation tool. It employs a more complex yet still straightforward methodology, which, through its implementation, focuses on identifying efficient and feasible environmental variables. These variables can then be utilized to describe the optimal environment for the specified use cases. Utilizing a grid-based environment with a discrete action space fulfills the requirement for more efficient prototyping.

The paper opens by reviewing the latest research and categorizing its contributions. It then describes the background and methodology for developing the prototype with dynamic environment modeling and shares initial results. The paper ends with a discussion of future research directions.

2 Background

Understanding that learning occurs through interaction with the environment is essential to grasp the nature of learning. Reinforcement learning (RL) is a computational method centered on goal-directed learning through interaction and maximizing reward signals. This approach involves discovering which actions yield the highest rewards through trial and error, without direct instructions, and observing the long-term consequences of actions [6].

RL has been used to model human behavior and interactions during emergencies in escape scenarios. While RL environments do not necessarily depend on visual representation, appropriate visualization can enhance realism and facilitate the agent's tasks by tracking its actions. For instance, one study simulates realistic fire scenarios and tests pre-trained agents on a large, complex building model represented solely as a graph, demonstrating scalability without visuals. The results indicate that combining off-policy, model-free algorithms improves performance in real-world fire evacuation

emergencies [2]. Several studies advocate for explicit visualization methods. One research paper employs a grid-based approach to pathfinding, discussing how an agent decides on escape routes [7]. Another study examines evacuation routes in a single room with multiple exits and obstacles [4]. Additionally, a framework using deep reinforcement learning in a 3D environment simulates crowd evacuation, analyzing how height, mass, and exit width affect evacuation dynamics [3]. Another piece of research applies RL to enhance emergency evacuation drills, using a controlled 3D environment where an agent trained as an evacuation leader streamlines the evacuation process for more efficient management [8]. Others employ tools such as Unity to adapt to the real-world needs of case studies that require it, such as active shooter incidents. The subsequent analysis then considers the understanding, implementation, and investigation of the impact of building design parameters on evacuations [9].
Despite the advancements, existing research critiques these models' static nature and lack of real-world features [2] - [4]. Dynamic environment modeling for escape scenarios using continuous ranging state and action spaces shows promise but necessitates extensive test runs and prolonged training [5]. Further research is needed on dynamic (non-stationary) modeling approaches within environments that yield faster results, such as using discrete action spaces to influence the decision-making process. Our proposed research framework aims to fill these gaps by integrating dynamic modeling into RL environments for escape scenarios, addressing the limitations of current models.

3 Methodology

Planning escape routes for emergency evacuation scenarios focuses on pathfinding individuals affected by the situation. Human movement patterns, which form these escape routes, can be modeled using various approaches. Our prototype movement is modeled using discrete action spaces within a grid-based environment. This approach translates navigation through the floor plan into numerical values, such as 0 for moving left, 1 for moving right, and 2 for moving straight. The agent then moves through the floor plan by executing these actions, progressing from point A (the starting point) to point B (the emergency exit or terminated state), thereby learning the escape path within the environment.
Concurrently, the agent must learn to apply the insights gained from escape path learning to modify the floor plan. This process involves adapting to changes in the environment during the agent's learning or training episodes. Therefore, the environment model must support a suitable action space and dynamic changes to facilitate effective learning and adaptation. Dynamic environment modeling refers to the capacity to alter the state of the environment within an ongoing episode in response to actions taken by an agent or external factors. Depending on the task, most RL environments within modeling frameworks remain stationary throughout an episode (initialized at the start and reset at the end). In evacuation scenario planning, pathfinding maps human movement through escape routes. These movement patterns can be modeled using simulations that support dynamic environment modeling, essential for adapting the floor plan during training. As the agent learns and

interacts with the environment, this approach enhances the applicability of the learned escape paths under varying conditions.

In our work, we employed the technique of prototyping to generate initial results for our approach. This entailed a continuous loop of developing, testing, demonstrating/visualizing, and refining the RL environment. Prototyping is beneficial as it allows us to improve the model iteratively, ensuring it accurately reflects real-world conditions and effectively adapts escape paths. This method aligns with our approach by enabling hypothesis testing, repeated experimentation, and ongoing refinement based on visualized results. A summary of RL modeling frameworks was conducted to develop an effective workflow for the simultaneous learning of escape paths and design decisions. This entailed the utilization of Farama's *Miniworld* [10] for the modeling of the environment, implementing reliable and widely-used RL algorithms via *Stable Baselines 3* [11], and *wandb.ai* [12] for the logging and tracking of experiments. The model was trained using the Proximal Policy Optimization Algorithm (PPO) [13].

4 Results

4.1 The environment

The agent's objective is to navigate from a starting point, designated as A, to a goal location, defined as B. The agent is randomly positioned at various starting locations, as described in section 4.2. The agent learns to navigate to B, the terminated state, solely through the input provided by the environment setup, the action and observation space, and the reward function.

The prototype we are discussing is a simple setup of three rooms and a hallway on an 8x20m floor. Four walls surround the arena, which are fixed in size and cannot be escaped. Another five walls separate the floor, each separating two rooms. The hallway has a door leading to a staircase, representing the terminated state (the target) where the agent reaches safety (see Figure 1). The partition walls have a sensor that detects the agent's touch when he decides to move through it, automatically creating a door. This door, represented by a passage, is placed where the agent touched the partition wall, enabling the agent to pass from one room into the other (see Figure 2). During reset, all placed doors are removed.

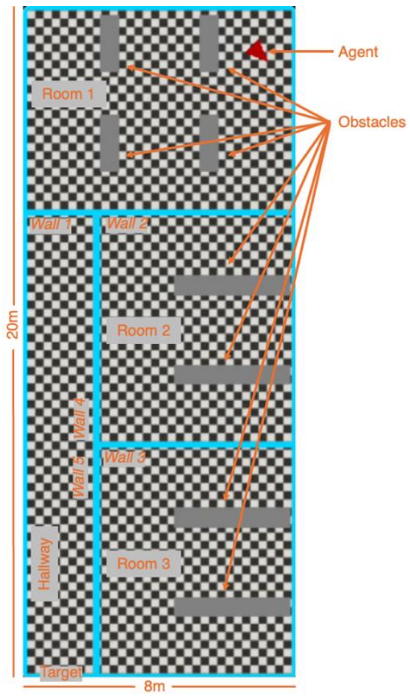

Figure 1: The environment set-up.

The duration of an episode is set to 3000 steps before resetting for a new episode. This timeframe was identified as pertinent for the agent to explore and locate the target in initial test runs. Tables 1 and 2 present the action and observation spaces. The rewards have been refined during several

prototyping steps and are meant to foster fast escape times. We tend to model a realistic environment that includes design aspects such as regulatory requirements (e.g., hallways have higher fire resistance requirements) without introducing too many biases simultaneously.

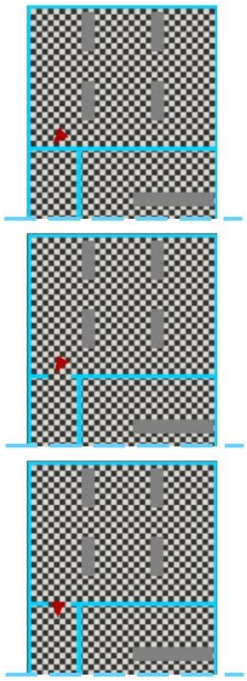

Figure 2: Dynamic environment modeling during an episode (example Room 1; applies to all walls once per episode).

The rewards are given as follows:

1. *reward += 1 / (distance_to_terminal + 1)* for reducing the distance to the target, else:
   reward -= 0.5 / (distance_to_terminal + 1)
2. *reward += 100* for creating a door
3. *reward += 0.5* for being in the hallway
4. *reward += 3000* for step-count < 1000,
   reward += 2000 for 1000 < step-count < 2000,
   reward += 1000 for 2000 < step-count < 3000
   for termination
5. *reward -= 500* for truncation

The prototype resulted in some assumptions (partly adjusted from [5]) and parameters being set as 1) fixed set up of floor plan walls, 2) no orientation/sight included for the agent – the agent neither has knowledge of the escape path nor a lidar sight or else implemented, 3) the door he positions through walking towards the exit has a fixed size and cannot be moved again during an episode and 4) the obstacles represent simplified furniture (e.g., office tables) and do not move or change sizes during training.

Table 1: Action Space.

turn_left	turn_right	move_forward	action_space
0	1	2	discrete(3)

Table 2: Observation Space.

agent position	direction of agent	target B position	distance to target B	encoded room name	observation_ space
x, y, z	angle	x, y, z	length	[1, 0, 0, 0]	Box(11)

4.2 The findings

Preliminary results indicate that when using the environment described in section 4.1, the agent can learn to navigate the environment for the trained number of steps and position doors within the floor plan based on its learned optimal escape path. The mean reward throughout the training can differ slightly for the mean length of an episode as the agent is randomly placed in a room by episodes reset, frequently changing the distance and location to the target.

Transfer learning during prototyping, resulting in the training results below (see Figure 3). *Run 0* has been conducted by placing the agent in four starting positions in Room 1 randomly. The resulting paths are shown in Figure 3a. After discovering and solving an environmental bug of too large of the

obstacles bounding boxes, we reran the training but overfitted to the former solution of *run 0*. Then, we conducted *run 1* using the 6mio-global step result model from *run 0*, representing half of the experiences and final learnings of *run 0*-agent (see Figure 4), which led to more exploratory and expected results (see *run 1*-paths of Figure 3b). Then, we transferred the final learning of *run 1* and added three starting positions A in Room 2, resulting in seven starting positions A for *run 2*. For the last *run 3*, we added three starting positions A in Room 3 to the existing ones, leaving us with ten starting positions A and the final learning of run 2, which resulted in the learning of paths shown in Figure 3b. Figure 4 presents the transferred learning from one run to another. Except for the described reusing of run 0 for run 1, all the other transfers used the learning results at the end of each training run, e.g., final learning at 9mio global steps of *run 1* for the start of *run 2*.

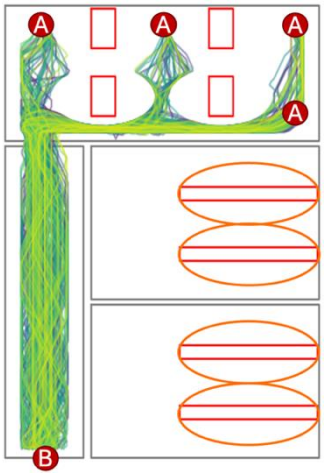

Figure 3a: Prelimary paths shown for *run 0*. Bounding boxes of obstacles represented with orange circles.

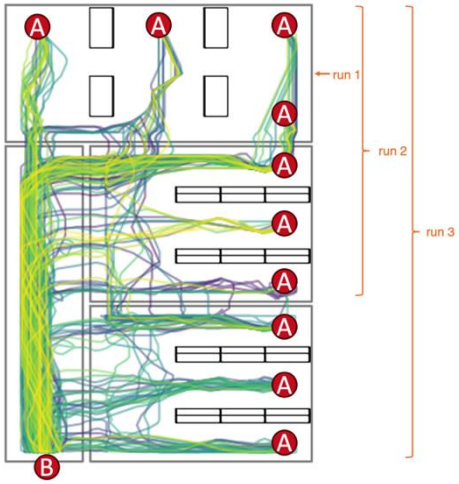

Figure 3b: Visualization of paths and sequence of runs after resolving the environmental bug of bounding boxes.

Figure 3: A typical sequence of prototyping. The paths shown in 3a (*run 0*) resulted from the fact that the obstacles bounding boxes prevented the agent from moving in certain areas (orange circles). After resolving the environmental bug, 3b shows the transfer learning results (*run 1*, then *run 2*, then *run 3*).

5 Discussion and Conclusion

This paper presents a novel approach to integrating dynamic environment modeling within a reinforcement learning (RL) framework. The objective is to optimize escape paths in evacuation scenarios. The agent modifies its environment dynamically by placing doors and learning optimal escape routes through interaction and adaptation. The discrete state and action space, operating within the defined range of motion, yielded usable, rapid results and facilitated additional refinement steps within the prototype method.

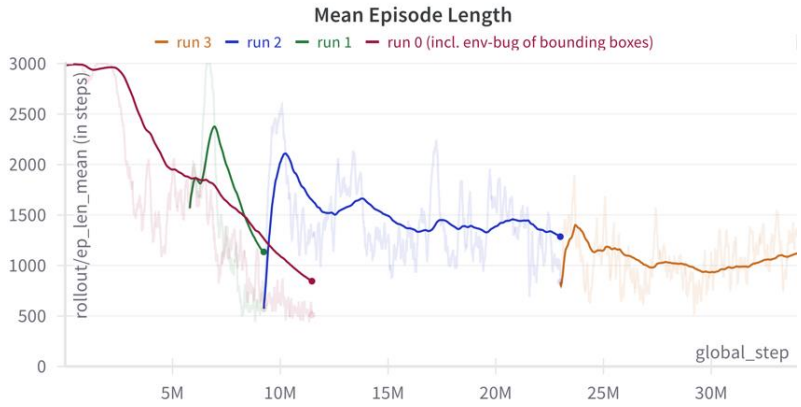

Figure 4: The agents mean episode length. *run 0* equals the training results shown in Figure 3a. To avoid overfitting: training *run 1* starts using the 6mio-global step result model from *run 0*.

Nevertheless, several limitations require further attention in future research. The static nature of the floor plan and the fixed size of components like doors restrict the adaptability of our model. Implementing scalable solutions for floor plan creation and additional building elements will assist in creating more realistic scenarios. Implementing scalable solutions for floor plan creation and additional building elements will assist in creating more realistic scenarios. The preliminary results indicate that incorporating orientation awareness and lidar sight could enhance the agent's human-like characteristics, thereby improving the efficacy of the training process. Furthermore, investigating the integration of semantic knowledge (e.g., hot-encoded room names) of learning outcomes would enhance the model's realism. Fine-tuning the algorithm's hyperparameters is paramount to achieving stable learning. Implementing advanced strategies to avoid local optima could enhance the current prototype’s complexity and deceptive rewards [14]. The PPO algorithm [13] proved beneficial; however, the dynamic environment setup requires an in-depth sensitivity analysis of variables and parameters.

This novel approach to escape path planning in a dynamic RL environment demonstrates promising results, indicating both feasibility and potential benefits for performance-based validation, such as developing authority compliance checks. Future work will focus on enhancing the environment's scalability and flexibility and fine-tuning algorithms to reflect realistic conditions better. This research contributes to advancing adaptive RL systems, which can inform the design of safer floor plans for fire safety.

References

[1] I. Fitkau and T. Hartmann, “An ontology-based approach of automatic compliance checking for structural fire safety requirements,” *Adv. Eng. Inform.*, vol. 59, p. 102314, Jan. 2024, doi: 10.1016/j.aei.2023.102314.

[2] J. Sharma, P.-A. Andersen, O.-C. Granmo, and M. Goodwin, “Deep Q-Learning With Q-Matrix Transfer Learning for Novel Fire Evacuation Environment,” *IEEE Trans. Syst. Man Cybern. Syst.*, vol. 51, no. 12, pp. 7363–7381, Dec. 2021, doi: 10.1109/TSMC.2020.2967936.

[3] D. Zhang *et al.*, “Deep reinforcement learning and 3D physical environments applied to crowd evacuation in congested scenarios,” *Int. J. Digit. Earth*, vol. 16, no. 1, pp. 691–714, Oct. 2023, doi: 10.1080/17538947.2023.2182376.

[4] Y. Zhang, Z. Chai, and G. Lykotrafitis, “Deep reinforcement learning with a particle dynamics environment applied to emergency evacuation of a room with obstacles,” *Phys. Stat. Mech. Its Appl.*, vol. 571, p. 125845, Jun. 2021, doi: 10.1016/j.physa.2021.125845.

[5] I. Fitkau and T. Hartmann, “Applying Reinforcement Learning for Design Decisions on Fire Safety Requirements in a Fire Evacuation Environment,” in *Proceedings of the 31st EG-ICE International Workshop*, Vigo, Spain, Jul. 2024. Accessed: Aug. 25, 2024. [Online]. Available: https://3dgeoinfoeg-ice.webs.uvigo.es/proceedings

[6] R. S. Sutton and A. G. Barto, *Reinforcement learning: an introduction*, Second edition. in Adaptive computation and machine learning series. Cambridge, Massachusetts: The MIT Press, 2018.

[7] P. Wongsai and W. Pawgasame, “A Reinforcement Learning for Criminal’s Escape Path Prediction,” in *2018 5th Asian Conference on Defense Technology (ACDT)*, Hanoi: IEEE, Oct. 2018, pp. 26–30. doi: 10.1109/ACDT.2018.8593191.

[8] N. Sinpan, P. Sasithong, S. Chaudhary, S. Poomrittigul, N. Leelawat, and L. Wuttisittikulkij, “Simulative Investigations of Crowd Evacuation by Incorporating Reinforcement Learning Scheme,” in *Proceedings of the 6th International Conference on Algorithms, Computing and Systems*, Larissa Greece: ACM, Sep. 2022, pp. 1–5. doi: 10.1145/3564982.3564983.

[9] R. Liu, W. Wu, B. Becerik-Gerber, and G. M. Lucas, “Enhancing Building Safety Design for Active Shooter Incidents: Exploration of Building Exit Parameters using Reinforcement Learning-Based Simulations,” in *Proceedings of the 31st EG-ICE International Workshop*, Vigo, Spain, Jul. 2024. Accessed: Aug. 25, 2024. [Online]. Available: https://3dgeoinfoeg-ice.webs.uvigo.es/proceedings

[10] M. Chevalier-Boisvert *et al.*, “Minigrid & Miniworld: Modular & Customizable Reinforcement Learning Environments for Goal-Oriented Tasks,” Jun. 23, 2023, *arXiv*: arXiv:2306.13831. Accessed: Jul. 07, 2024. [Online]. Available: http://arxiv.org/abs/2306.13831

[11] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, “Stable-Baselines3: Reliable Reinforcement Learning Implementations,” *J. Mach. Learn. Res.*, vol. 22, no. 268, pp. 1–8, 2021.

[12] L. Biewald, “Experiment Tracking with Weights and Biases.” 2020. [Online]. Available: https://www.wandb.com/

[13] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal Policy Optimization Algorithms,” 2017, *arXiv*. doi: 10.48550/ARXIV.1707.06347.

[14] Q. Wu, S. Zhao, F. Zhou, H. Zhang, Y. Huang, and K.-K. Ma, “Cognitive Escape Reinforcement Learning for Complex Decision Making,” Mar. 23, 2023. doi: 10.21203/rs.3.rs-2661516/v1.