CATM
Center for Advanced Transportation Mobility

Real-time Deep Reinforcement Learning for Evacuation under Emergencies

January 2024

Yujing Zhou, Yupeng Yang, Dahai Liu, Yongxin Liu, Sirish Namilae, Houbing Song
Embry-Riddle Aeronautical University

US DEPARTMENT OF TRANSPORTATION GRANT

CATM

DISCLAIMER

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated under the sponsorship of the Department of Transportation, University Transportation Centers Program, in the interest of information exchange. The U.S. Government assumes no liability for the contents or use thereof.

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.

4. Title and Subtitle Real-time Deep Reinforcement Learning for Evacuation under Emergencies	5. Report Date: January 2024 6. Source Organization Code: ERAU Cost Center 61687
7. Author(s) Yujing Zhou, Yupeng Yang, Dahai Liu, Yongxin Liu, Sirish Namilae, Houbing Song	8. Source Organization Report No. CATM-2024-R1-ERAU
9. Performing Organization Name and Address Center for Advanced Transportation Mobility Transportation Institute 1601 E. Market Street Greensboro, NC 27411	10. Work Unit No. 11. Contract or Grant No. 69A3551747125
12. Sponsoring Agency Name and Address University Transportation Centers Program (RDT-30) Office of the Secretary of Transportation–Research U.S. Department of Transportation 1200 New Jersey Avenue, SE Washington, DC 20590-0001	13. Type of Report and Period Covered Final Report: 10/01/2021-12/31/2023 14. Sponsoring Agency Code: USDOT/OST-R/CATM

15. Supplementary Notes:

16. Abstract

This study adopted an A3C algorithm to simulate the evacuation process under different situations (e.g., multiple agents and different environmental conditions) and results were compared with Deep Q-Networks (DQN), to demonstrate the efficiency and effectiveness of A3C algorithm use in evacuation models. Results indicated that under static environments, A3C demonstrated superior adaptability and quicker response times. Furthermore, with an increasing number of agents, A3C showed better scalability and robustness when managing complex interactions and provided quick evacuations. These outcomes highlight A3C's advantage over traditional RL models under varying and challenging conditions. The report concludes with a discussion of the practical implications and benefits of these models. It emphasizes their potential in enhancing real-world evacuation planning and safety protocols.

17. Key Words Reinforcement Learning, Multi-agent collaboration, emergency, airport evacuation	18. Distribution Statement Unrestricted; Document is available to the public through the National Technical Information Service; Springfield, VT.

19. Security Classif. (of this report) unclassified	20. Security Classif. (of this page) unclassified	21. No. of Pages 78	22. Price …

Executive Summary

Emergencies can happen at anytime and anywhere in our daily lives. “Emergency” is a broad term that summarizes several different situations, which are generally serious, unexpected, and require immediate actions. This study focused on aviation emergency evacuation situations, which have the characteristics of high level of time pressure and uncertainty. Therefore, a real-time decision-making assistance system is needed and will be largely helpful when an emergency exists at the airport or on an aircraft when an evacuation occurs. The system and model built and described in this paper are based on the Asynchronous Advantage Actor Critic (A3C) algorithm, which is one of the newest algorithms under Deep Reinforcement Learning Algorithms. It provides quicker and more efficient evacuation routes for the agents in the environment compared to traditional evacuation simulation models, thus saving time for passengers. This study adopted an A3C algorithm as the tool to simulate the evacuation process under different situations (multiple agents and different environmental conditions) and results were compared with Deep Q-Networks (DQN) to demonstrate the efficiency and effectiveness of the A3C algorithm use in evacuation models. Results indicated that under static environments, A3C demonstrated superior adaptability and quicker response times. It performed 43.86% faster than DQN in terms of the average time taken for agent evacuation and converged quicker at around 100 episodes compared to 250 episodes for DQN algorithm. In scenarios with moving threats, the A3C algorithm also outperformed DQN in terms of dynamic pathfinding efficiency and maintaining agent safety. Furthermore, with an

increasing number of agents, A3C showed better scalability and robustness in managing complex interactions and providing quick evacuations for multiple agents. These outcomes highlight A3C's advantage over traditional models, especially in terms of adaptability, efficiency, and scalability under varying and challenging conditions. The report concludes with a discussion of the practical implications and benefits of these models. It emphasizes their potential in enhancing real-world evacuation planning and safety protocols. The practical implications and benefits of these models are provided at the end of the report.

Table of Contents

Abstract

Emergencies can happen at anytime and anywhere in our daily lives. “Emergency” is a broad term that summarizes several different situations, which are generally serious, unexpected, and require immediate actions. This study focused on aviation emergency evacuation situations, which have the characteristic of high level of time pressure and uncertainty. Therefore, a real-time decision-making assistance system is needed and will be largely helpful when an emergency exists at the airport or on an aircraft when evacuation occurs. The system and model built in this paper are based on the Asynchronous Advantage Actor Critic (A3C) algorithm, which is one of the newest algorithms under Deep Reinforcement Learning Algorithms. It provides quicker and more efficient evacuation routes for the agents in the environment compared to traditional evacuation simulation models, thus saving time for passengers. This study adopted an A3C algorithm as the tool to simulate the evacuation process under different situations (e.g., multiple agents and different environmental conditions) and results were compared with Deep Q-Networks (DQN) to demonstrate the efficiency and effectiveness of A3C algorithm use in evacuation models. Results indicated that under static environments, A3C demonstrated superior adaptability and quicker response times. It performed 43.86% faster than DQN in terms of the average time taken for agent evacuation and converged quicker at around 100 episodes compared to 250 episodes for DQN algorithm. In scenarios with moving threats, the A3C algorithm also outperformed DQN in terms of dynamic pathfinding efficiency and maintaining agent safety. Furthermore, with an increasing number of agents, A3C showed better scalability and robustness in managing

complex interactions and providing quick evacuations for multiple agents. These outcomes highlight A3C's advantage over traditional models, especially in terms of adaptability, efficiency, and scalability under varying and challenging conditions. The report concludes with a discussion of the practical implications and benefits of these models. It emphasizes their potential for enhancing real-world evacuation planning and safety protocols. The practical implications and benefits of these models are provided at the end of the report.

Section 1: Introduction

Every second counts in the face of an emergency. Emergencies often threaten property, the environment, and, most importantly, human lives. Among various types of emergency situations, those occurring in transportation sectors, such as aviation, require immediate attention and care. Given that the population involved in different modes of transportation is always high and in constant motion, any emergency during normal operations could significantly disrupt schedules and endanger thousands of lives. To deal with emergency situations, researchers have conducted several studies that are predictive and proactive. In the aviation industry, the emergency is a sensitive and concentrated area of interest where strict rules and regulations have been set and the most effective and efficient reaction to the emergencies that occur under high mental workload and time pressures is required. Emergency situations demand swift, precise, and efficient responses to minimize damage and prevent future occurrences. Being both reactive and proactive is essential for effectively managing emergencies that occur in aircraft and airports. Therefore, a decision-making system that could provide the most time-saving route for emergency evacuation at the airport is necessary.

Safety and compliance are of paramount importance in the aviation industry today. According to the Federal Aviation Administration (FAA), more than 45,000 flights are handled daily, with approximately 2.9 million passengers flying in and out of the United States (FAA, 2022). Given such a large number of daily flights, passenger safety becomes a top priority for airport designers and airlines. According to airline on-time statistics and delay causes (Figure 1) reported by the Bureau of Transportation Statistics

(BTS) in 2021, the five broad categories most affecting normal airline and airport operations are: (1) Air carrier delays, (2) Extreme weather conditions, (3) The National Aviation System (NAS), (4) Late-arriving aircraft, and (5) Security concerns. Each of these factors could cause significant delays and elevate risk during operations.

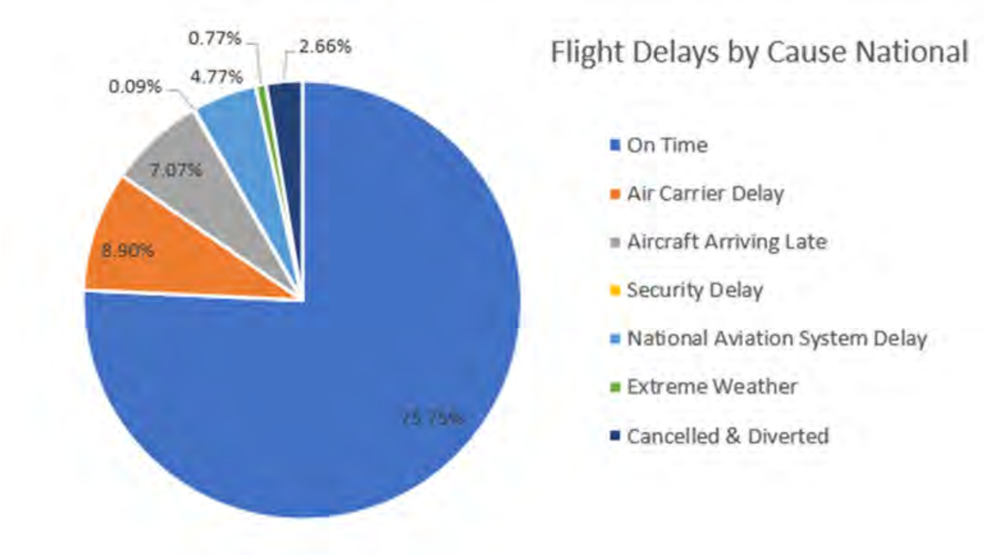

Figure 1. Flight Delays by Cause National

This paper is structured as follows: Section 2 provides a literature review of recent research on simulating and dealing with emergency situations in various environments. This section also analyzes two types of emergencies in the aviation industry, examining the effects of different emergencies on aircraft and airport environments, as well as regulations and evacuation plans. It also reviews previous studies related to airport evacuation and analyzes the use and implementation of algorithms to optimize evacuation routing and planning. The algorithms and models discussed include, but are not limited to, the Asynchronous Advantage Actor Critic (A3C) algorithm, Multi-Agent Deep Reinforcement Learning (MADRL), velocity obstacle (OV), and social force (SF). Section 3 describes methods and the model establishment and system architecture, including theoretical functions and simulation environments. Section 4 presents the final evacuation simulation results and an analysis of the outcomes. Finally, Section 5 provides a comprehensive conclusion of the entire model and discusses future improvements and directions.

Section 2: Literature Review

Emergency Situations

“Emergency” is a broad term that summarizes several different unexpected situations. Emergencies could cause severe consequences and should be mitigated immediately when they occur. According to the Oxford English Dictionary, an emergency is “a serious, unexpected, and often dangerous situation requiring immediate action,” which illustrates the following characteristics of an emergency: unpredictable, dangerous, and needs immediate attention and action (Alexander, 2013). An emergency could disrupt the operations easily because of its unpredictable characteristics, which will lead to severe consequences if the emergency is not managed properly and in a timely manner. Therefore, mitigations from both proactive and reactive aspects are important to protect lives and properties from emergency situations. Depending on different types of emergency situations, the time required for information processing and reaction could be different; thus, the mitigation and evacuation plan could also vary.

Emergency situation examples include floods, hurricanes, fire hazards, traffic accidents, blizzards, hail, etc. Each different situation has a different scale and severity, which will have various consequences and damage. While most emergency situations could be mitigated to some extent by following the correct steps of normal operations, reactive actions, including evacuation plans and real-time decision-making systems, still play an important role in correctly responding to emergency situations. For example, when a fire suddenly occurs at the cinema or there is an emergency landing at an airport, it could be largely mitigated or solved by the emergency response and evacuation plans, which will not cause a huge interruption to normal operations. However, there are emergency situations that cannot be spotted and reacted to immediately, such as natural disasters and terrorist attacks. When massive emergency situations happen, the normal emergency response plan will not be fast enough to respond to protect lives and properties (Alexander, 2013). Therefore, the real-time decision-making system and human reaction to emergency situations are essential and should be immediately involved for a better consequence.

Risk assessment is always a good way to help better understand an emergency situation and, thus, to design and react efficiently and safely. To evaluate the safety of evacuation when a large steel gym collapses due to a localized fire, Zhang et al. (2016) proposed a steel-temperature rise model which considered both the effect of smoke thermal radiation and convention and the effect of flame thermal radiation on steel components. Since the collapse of a steel structure gymnasium has a more significant impact on evacuation than the smoke hazard, a method was developed to quantitatively assess the casualties caused by the collapsed structure. Moreover, a quantitative risk assessment of evacuation safety was performed comparing the Available Safe Egress Time (ASET) and Required Safe Egress Time (RSET). The result of the experiment showed that the modified temperature rise model of steel components accurately predicted the temperature rise in the fire. Furthermore, by taking into account the distance from the farthest point to the safety exit, different moving speeds, the width of the evacuation exit, and the density of people, the experiments demonstrated that the model could predict the movement time during evacuation in steel-structured gymnasiums accurately.

Aviation Emergency

Emergency situations in aviation can be classified into two categories: aircraft emergency and airport emergency. Aircraft emergencies can be complex, dangerous, and may be caused by several different reasons. Aircraft emergencies, including instrument failure, autopilot failure, landing gear failure, engine failure, etc., can happen at any time while in flight. To deal with these unwanted emergency situations, the Federal Aviation Administration (FAA) and manufacturers have published emergency response plans and checklists for each different model of aircraft in order to help pilots deal with emergencies and make better decisions under time pressure. For example, when there is an engine failure during a flight, pilots have three choices of emergency landing to deal with the situation: precautionary landings, forced landings, and ditching. Precautionary landing is the safest and most likely survival landing, which has only a 0.06% fatality rate. However, when it comes to forced landing and ditching, which can cause big pressure on pilots and do not allow enough time and reaction, the fatality rate increases to as high as 10% and 20% (Rossier, n.d.). The fatality rate indicated the danger of engine failure but also demonstrated the importance of precaution and proactive plans and solutions.

Compared with aircraft emergency, airport emergency situations are less frequent but pose an equal or even greater threat to lives and properties. According to the Advisory Circular (AC) 150/5200-31C published by the U.S. Department of Transportation (DOT) and FAA, an airport emergency is “any occasion or instance, natural or man-made, that warrants action to save lives and protects property and public health” (DOT&FAA, 2009). To be more specific, the FAA and DOT classified different types of airport emergencies, including but not limited to (Villamizar, 2022):

- airport and aircraft malfunctions that impede safe flight,
- suspicious packages,
- bomb threats,
- sabotage of aviation-related equipment,
- structural fires and non-structural fires,
- natural disasters, etc.

To deal with these different types and severities of airport emergency situations, the FAA and DOT require each certificate holder of an airport to have the ability to introduce and maintain an airport emergency plan (AEP). The AEP should cover responses to each different situation and maximize the ability to protect property and public health (DOT&FAA, 2004). A qualified AEP typically follows four phases (i.e., mitigation, preparedness, response, and recovery), which are suggested by the Federal Emergency Management Agency (FEMA) for emergency management.

There are various reasons and causes that could lead to emergency situations for both aircrafts and airports, while most of the emergency situations need immediate evacuation and other types of measures to mitigate the consequences. For example, the right engine of a Boeing 767 aircraft from American Airlines caught on fire during the takeoff phase in 2016. Although Chicago O'Hare International Airport had an emergency evacuation due to this emergency situation, 20 people still received different levels of injuries while 161 passengers were all evacuated through the slides (Hradecky, 2016). Another evacuation that happened at Denver International Airport was caused by a left engine fire on a Bombardier CRJ-700 from United Airlines in 2017. In the evacuation, 59 passengers plus four crewmembers evacuated and none of them were injured according to the report (Hradecky, 2017). In 2017 at Daytona Beach International Airport, an Airbus A320 performed an emergency landing when the front windshield was cracked because of hail. The 132 passengers, including crewmembers, were safely evacuated because of the efficient evacuation plan and design (Hradecky, 2017). In all of these three emergency cases, the evacuation plan and rule played important role. In 2017, the FAA established the rule for evacuation plans and routes on aircraft that limits the time required by all the passengers evacuating from the aircraft to 90 seconds, which is called the “90 s evacuation rule.” Because of the rule, the evacuation of different types and designs of aircraft could be limited in a time frame and performed efficiently. Thus, for emergency situations, the aviation industry needs more measures to help mitigate the impact of emergency situations. Accordingly, the evacuation plan and emergency response system can also be divided into aircraft evacuation and airport evacuation, which will be explained more detailly in the following sections.

Airport Evacuation Simulation

To achieve the goal of safety and efficiency, evacuation plans and routes play important roles in airlines and airports. In FAR part 25, the FAA established the “90 s evacuation rule,” which, as stated previously, limits the time required by all the passengers evacuating from the aircraft to 90 seconds. However, unlike aircraft evacuation, airport evacuation does not have a rule or regulation that sets the time frame for evacuation. Therefore, people need to pay more attention while designing the evacuation plan since every different airport has different characteristics and construction. In the Code of Federal Regulations (CFR) title 14 part 139.325, the FAA requires each certified airport to develop and maintain an airport emergency plan (AEP), which is designed “to minimize the possibility and extent of personal injury and property damage on the airport in an emergency” (CFR, 2022). In this case, time is not a variable nor a criterion to evaluate the effectiveness of an evacuation anymore. One of the most significant difficulties in airport evacuation and AEP is that it is impossible to apply a generic plan to all of the different airports designs and layouts. To design and evaluate an effective airport evacuation, researchers will need to explore and build the model based on different airports because of the different layouts and locations, which is a time-consuming and complex task.

Chen et al. (2019) conducted an agent-based simulation for airport evacuation to determine the optimal number of exit doors and the best evacuation path for passengers when an emergency happens. In the study, they mainly focused on how to improve the efficiency of evacuation under emergency situations at the airport and used a local airport as the base for their model. The results also evaluated whether the evacuation strategies used right now are adequate considering the continuously growing number of passengers. The agent-based simulation is achieved by using a simulation software named AnyLogic, which is able to simulate the routes and evacuation process for airports while changing the variables, including exit doors, evacuating path, etc. (Chen et al., 2019).

The result from the number of changing variables simulations illustrated the relationship between the passenger volume and the total evacuation time. Naturally, passengers will need more evacuation time when the passenger volume is higher. Additionally, the number and location of exit doors are also two essential factors. In the simulation, passengers tend to evacuate through doors that are closer and easier to access, which could potentially increase the time of evacuation because of congestion. Thus, Chen et al. (2019) recommend that each airport administration design its own exit plan, including the path, the number, and the location of emergency exits in the airport. The simulation and method can be used to estimate the evacuation time for different airports based on construction and design. Although the simulation is helpful for optimizing evacuation plans and routes for airports, it still needs to be improved by considering agents or people that are moving at a relatively low speed, for example, the elderly and children. Moreover, the location of an emergency or threat in the airport can also be considered a variable that could make evacuation more efficient by changing the routes or making multiple plans.

The most important thing in airport operations is safety. Since full-scale evacuation practice is too expensive and time-consuming, computer model techniques are widely used for pedestrian evacuation. Cheng et al. (2014) developed an agent-based model in order to simulate the passenger evacuation process. In the model, the airport, which should be viewed as a complex system, was divided into landside and airside, while pedestrians were divided into passengers and wavers. Three emergency exits were available on both sides of the airport. Three security levels of passengers were defined based on their location and were directed to evacuate through different exits.

As shown in Figure 2, passenger activities were sorted into processing activities that were mandatory for passengers aboard the airplane and discretionary activities, which were defined as any other passenger activities conducted during the non-processing time. Processing activities include check-in, security, customs, and boarding, whereas discretionary activities include walking, shopping, eating, etc. Two levels of behavioral responses were considered in the model. The global behavioral level outlined the general evacuation process: respond to signal, move toward exits, wait before exits, and complete evacuation. Moreover, the local behavioral level considered different response times of passengers due to different ages and travel purposes and considered that group travel passengers would adjust their speed to match the slowest group member. Furthermore, it was assumed that there was no panic behavior during the evacuation.

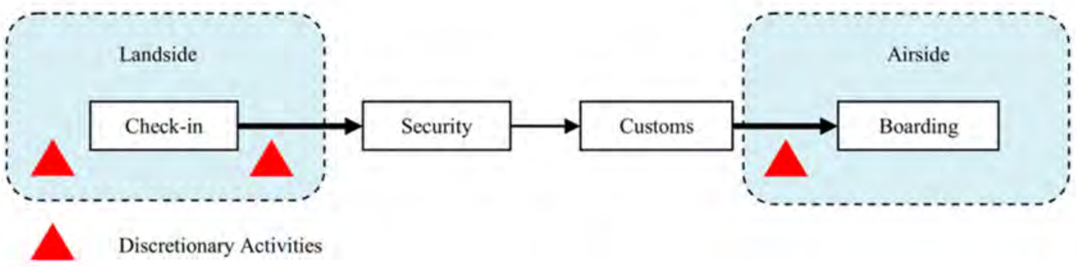

Figure 2. The Departure Process of Passengers in Airport (Cheng et al. 2014)

The simulation experiment results demonstrated the evacuation time of passengers with group dynamics is longer than that of passengers who travel on their own since group passengers would spend more time making decisions, moving, and waiting for other members. The experiment also verified that the agent-based model could be utilized to analyze pedestrian group dynamics in a complex environment. However, the model developed in this study is not designed for extreme situations, and airport staff was not taken into consideration during experiments.

Due to different physical or mental conditions, people with disabilities would be most affected when emergencies happen, and the built environment would also affect them far more than the general population. However, previous studies have not adequately considered persons with disabilities when constructing evacuation models or designing simulated environments. Thus, since it is critical to incorporate disability considerations into emergency evacuation plans, preparations, and other activities, Christensen and Sasaki (2008) developed an agent-based simulation that can classify built environments based on environmental characteristics and simulate complex populations based on individual variable criteria. Their BUMMPEE (Bottom-up Modeling of Mass Pedestrian flows – implication for the Effective Egress) model allows simulated behaviors to more appropriately represent the diversity and prevalence of individuals with disabilities and their interaction with the built environment, thereby determining the effectiveness of the built environment in adapting to the needs of persons with disabilities during evacuation.

To develop their model, Christensen and Sasaki (2008) assessed the impact of current and proposed Americans with Disabilities Act Accessibility Guidelines

(ADAAG) on the built environment and identified five groups of disabilities and six criteria that differ in relation to the particular forms of disability, and four environmental characteristics that had a significant impact on people with disabilities. The five disability groups included physical, mental, go-outside-home, sensory, and self-care disability. Moreover, the six criteria were specified as individual speed, individual size, individual ability to negotiate the terrain, individual perception, individual psychological profile, and individual assistance. The BUMMPEE model addressed these criteria by including seven different populations: motorized wheelchair users, non-motorized wheelchair users, the visually impaired, the hearing impaired, the stamina impaired, individuals without disabilities familiar with the environment, and individuals without a physical or sensory disability but less familiar with the environment. Each population was separately defined by the difference in speed, size, and ability to negotiate the terrain. Furthermore, the four environmental characteristics were classified as exit character, route character, obstacle character, and planned systems.

After setting the basic criteria, population, and environmental characteristics, the BUMMPEE model was written in C++ with the use of a common graphical interface structure. To validate the reasonableness of the model, a physical evacuation of the same environment and population was conducted at the Utah State University Human Services Research Center (HSRC) on September 14, 2005, to compare model results with actual results. However, due to differences in the data available for the physical and simulated models, only two measurements (maximum evacuation time and the number of evacuees evacuating at each exit) were used for comparison. The results validated the consistency and similarity of the BUMMPEE model with real-world scenarios by showing that the average total evacuation time of the evacuation simulations was only 33 seconds less than the physical evacuation.

Application of Machine Learning in Evacuation

Machine learning, a study "of computer algorithms that can improve automatically through experience and data" (Mitchell, 1997), has been continuously developed and implemented by researchers in a number of different areas, including medical, image recognition, computer vision, product recommendation, etc. In the aviation industry, machine learning was also used as a powerful tool that helps provide coherent and safe trips for more than thousands of flights daily.

Compared to private vehicle traveling, public transportation systems like the metro, bus, and airport have more complexity and the possibility of emergencies. Because of their unpredictable variables, including passengers who are changing all the time, the probability and consequences of emergency situations in the public transportation system are normally severe and need immediate mitigation or evacuation. To accomplish this task, Ma et al. (2022) achieved passenger flow forecasts using the machine learning algorithm under emergency situations in the metro. They pointed out that because of passenger congestion during the peak period of transportation, it is necessary to develop a passenger flow forecast system by applying the Long Short-Term Memory (LTSM) model, which is one of the applications of machine learning algorithms.

In the research, Ma et al. (2022) utilized transfer learning and the LTSM network to predict the passenger flow of the metro transportation system in both normal and emergency situations. They pointed out that although Gao et al. (2019) have analyzed the passenger flow change through a mathematical modeling method, it is still difficult for this method to accomplish the task with the huge amount of passenger flow under emergency situations. Thus, compared to traditional methods, the LTSM network and transfer learning together are more suitable for measuring a large amount of passenger flow under emergency situations.

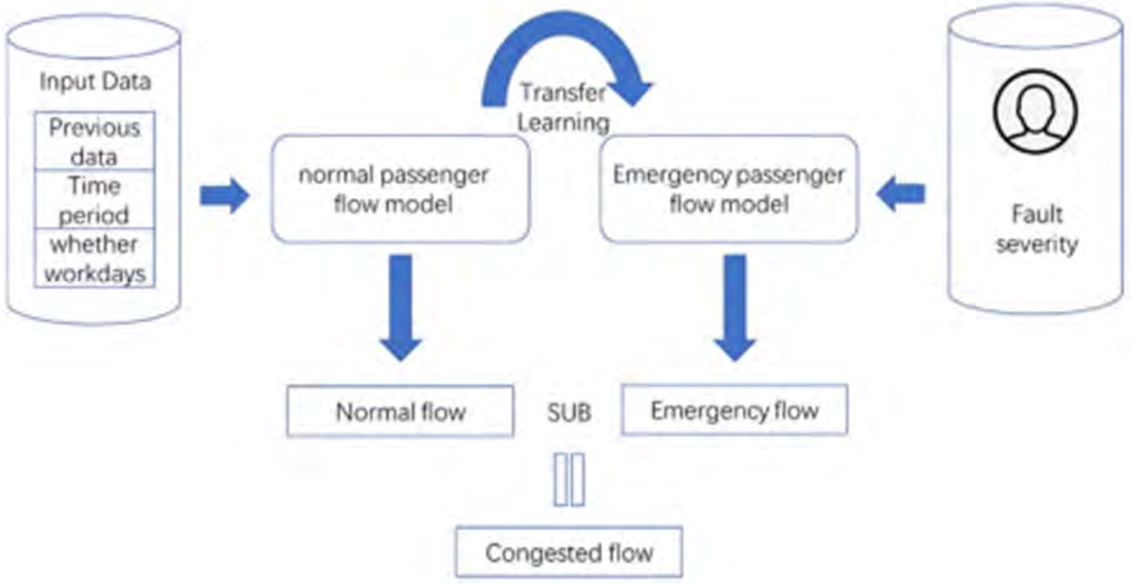

Figure 3. The Transfer Learning Procedure Between Normal and Emergency Passenger Flow (Ma et al., 2022)

In the research, Ma et al. (2022) first divided the model into two parts: the normal passenger flow model and the emergency passenger flow model. They completed the first part, which is the normal passenger flow model first, then a transfer learning model was applied to convert the normal passenger flow model into an emergency passenger flow model in order to predict and simulate the passenger flow under emergency situations (Figure 3). The result shows that machine learning algorithms are extremely helpful in reflecting the capacity of public transportation like airports and metro. Research demonstrated that machine learning algorithms have the ability to catch the sensitive change in passenger flow and deal with either big or small passenger flow under emergency situations.

Gota et al. (2020) leveraged machine learning algorithms and addressed airport emergency situations from another aspect. In their research, implementation of machine learning was applied to the threat objects detection system, which represents the "proactive" level of an airport emergency plan. Gota et al. (2020) used the convolutional neural network and specialized libraries. The algorithm and network are able to recognize threats using X-ray images. The result indicates the usage of machine learning in threat object detection, like bomb threats in an airport, which detects the threat early and shortens the evacuation time, thus improving the evacuation process and protecting lives and properties.

Another study conducted by Zarraonandia et al. (2009) presents a virtual environment that could simulate different emergency situations in the airport and find each different emergency cause and consequence. The virtual environment was built using DimensioneX. The results show that the virtual environment can be used as an evaluation and learning tool in order to help supervisors and researchers design a better airport emergency plan. Therefore, we can choose to test the emergency situation decision-making system and the multi-agent collaborative evacuation model in the virtual environment and continuously improve the process and result of the system.

In emergency search and rescue, the rational use of robots can reduce the loss of human resources and property, reduce possible injuries to search and rescue personnel during the process, and even rescue survivors faster. With a fully automated system as the ultimate goal, Vaidyanath et al. (2020) developed a system consisting of a swarm of unmanned aerial vehicles (UAVs) and a virtual "spokesperson" in the experimental virtual reality simulation environment to determine when the system should rely on UAVs and speakers, and when a human operator should be prompted to intervene.

In the experiment, the virtual reality environment depicted a wildfire ravaging a small town, while the system (UAVs and spokesperson) and the human operator needed to cooperate to find and guide survivors to a safe escape from the fire. The UAVs capable of issuing pre-recorded warning messages would need to contact and convince a civilian group to leave their homes before the fire reaches the scene and then guide the group to a safe evacuation. Virtual humans (spokesperson) can assist with missions and negotiate with groups through a UAV, while human operators can be interrupted for help when the UAV feels unable to convince the group in time. The spokesperson was implemented in the Wizard of Oz (Woz) setting to collect real-world interaction data between human operators and the system. Vaidyanath et al. (2020) utilized reinforcement learning to automatically learn a strategy to be followed by the UAV after finding survivors, convincing them to leave, and guiding them out.

Vaidyanath et al. (2020) modeled the Reinforcement Learning (RL) problem as a Markov Decision Process (MDP) and experimented with both the Monte Carlo simulations and the Temporal Difference Learning methods. The study focused on the on-policy Monte Carlo algorithm since it worked the best. Three possible levels of communication with the survivor groups were set: UAV warning, spokesperson persuasion, and human operator convincement. There were three types of survivor groups that would behave differently when dealing with the UAVs: the stubborn couple, the old couple, and the babysitter with a child. Moreover, there were seven policy actions: warn, allow-spokesperson-to-negotiate, interrupt-operator, query-for-guidance-info, UAV-guide, Vehicle-guide, and wait. Furthermore, five state variables were set: operator-business-level (0-3), group-status (0-5), fire-approach-time (0-4), preferred-guidance-type (1-3), and negotiate-status (0-2). As a result, a total of 1,440 states and 10,080 state-action pairs were developed.

By setting the fire approach time to values 2, 3, and 4, three different policies that worked for all survivor groups were learned and each policy was trained for 1 million episodes and tested for 10,000 episodes. The final result demonstrated the success rate of saving survivors exceeded 95% when the fire-approach-time variable was set to 4, the average success rate of saving survivors was 90% when the variable was set to 3, and when the variable was set to 2, the rescue survivors had an average success rate of 74%.

There are many machine learning applications in the evacuation process, one of the most popular methods is the Reinforcement Learning model, due to its strength in dealing with uncertain environments. In the next section, a specific machine learning algorithm, Reinforcement Learning is reviewed.

Reinforcement Learning (RL)

As one of the unique and useful machine training methods, Reinforcement Learning (RL) has the characteristic that does not require a supervisor or complete model (Sutton & Andrew, 2018). By utilizing its unique features, the learning agent in a reinforcement learning model is able to make its own decision while trying to achieve the goal and maximizing the total reward. The agents normally can train themselves by performing countless choices and attempts, which improves its result from simple right or wrong to a set of complex steps and tactics in order to reach the best result.

When emergencies occur, the most important and difficult task is to evacuate the crowd safely and efficiently. During the evacuation process, incorrect evacuation methods and routes will undoubtedly cause more casualties, but the uncertainty of crowd movement can also make the evacuation mission more difficult to succeed. However, the behavior logic and trajectory of the crowd can be predicted, and there is no doubt that this crowd behavior simulation could be very helpful for evacuation in times of emergency.

While traditional methods of crowd evacuation simulation lack consideration of variable scenarios and dynamic movements, in order to improve the visual realism of crowd simulation, Yao et al. (2019) developed a reinforcement learning-based data-driven crowd evacuation (RL-DCE) framework. Firstly, the DCE model was established. To quantify the cohesiveness of crowds, the model extracted dynamic characteristics, including position and velocity, from video. A cohesiveness-based K-means (C-K-means) algorithm was built and thus grouped the crowd to predict their trajectories. Second, a hierarchical path planning mechanism was developed. Two layers were proposed in the mechanism. By utilizing reinforcement learning algorithms, the top-layer would train the control policy using the obtained trajectories to handle the dynamic environment, while the bottom-layer was responsible for collision avoidance by using the reciprocal velocity obstacles (RVO) model.

After comparing path computation based on group and the individual, Yao et al. (2019) validated the efficiency of the group-based path calculation. Moreover, by comparing the trajectory obtained in the simulation and the trajectory in the video, the result showed that the path planning method is able to make the simulated trajectory very close to the trajectory in the video. Furthermore, Yao et al. (2019) also analyzed path control in terms of weight parameter adjustment and found that the path planning method is able to handle path changes in a dynamic environment. Besides, after 180 iterations, the reinforcement learning-based path calculation curve tends to stabilize, which demonstrates that the method is convergent. Also, by comparing with other methods, Yao et al. (2019) validated that the RL-DCE improved the visual realism of crowd evacuation simulation while adapting to the dynamic environment.

How to evacuate people more efficiently and safely in a crisis situation is always the most important thing. Zhang and Guo (2014) developed a novel distributed multi-robot system to guide the evacuation of people in emergency situations. By incorporating a cooperative exit-seeking algorithm into the system, the robots can work together by estimating gradients online and tracking gradient descent while moving in sequence. To better simulate human behaviors, two human panic behavior models were considered in the system, and both static and dynamic routing evacuation experiments were simulated in a mall-like environment. The result demonstrated that in the case of 130 evacuees, the usage of a multi-robot system could reduce evacuation time by approximately 50%, while more than 40% of time reduction was proved in the case of 250 evacuees.

As the immersed tube tunnel becomes more and more popular, fire evacuation has become more and more critical due to the deep depth and the closed and narrow design of the tunnels. Since the previous study failed to consider personal behavior in actual situations during the evacuation, Tian and Jiang (2018) established the largest section immersed tube tunnel experimental base in the world and conducted a large-scale fire evacuation test to the high-temperature hazard range of fire and calculate the realistic evacuation speed. Based on all their experiment, Tian and Jiang (2018) successfully built an evacuation model and calculated the minimum safe escape time by establishing a mathematical algorithm.

All experiments were carried out in the Hong Kong-Zhuhai-Macau experimental base while considering the limited evacuation area and multiple evacuation directions of the tunnel. In the experiment, only the effect of high temperature was considered and the scale of the fire was 50MW. The results stated that the temperature would be above 60 °C (unbearable temperature for breathing) in the range of about 25m of fire upstream and downstream from the height of 1.5m. By utilizing reinforcement learning, the authors transform the tunnel fire evacuation problem into an O-D (origin-destination) path selection problem. The experiment result demonstrated that as the iteration number increases, the number of people on each path to an exit tends to a definite value so that the final result converges. For the future experiment, people in the non-hot area influenced by smoke as well as people traveling in groups would also be taken into account.

Another study conducted by Kim and Pineau (2015) illustrated the possibility and usage of RL in adaptive path planning in human environments. Their framework of adaptive path planning consists of three modules, which are the feature extraction module, the inverse reinforcement learning module (IRL), and the path planning module (Kim & Pineau, 2015). By combining three modules together, the system is able to navigate and choose the most efficient path in a changing environment, full of moving targets (Figure 4). The research is quite useful when applied to an airport emergency environment, where people are panicking and trying to find the closest and safest path to evacuate.

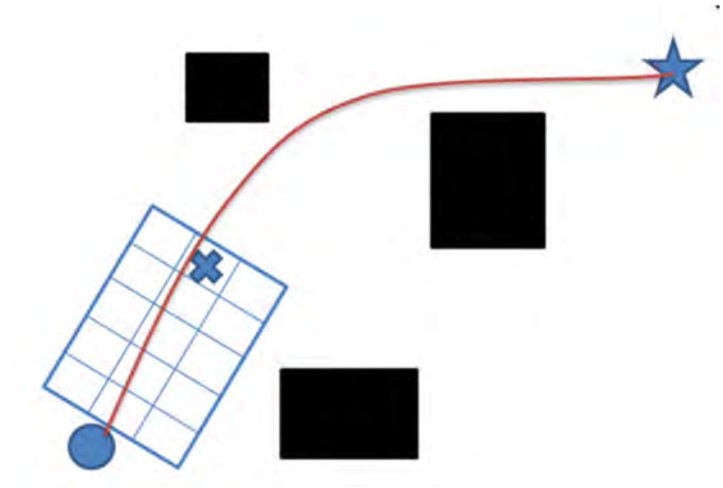

Figure 4. Illustration of Robot Adaptive Path Planning Using Inverse Reinforcement Learning (Kim & Pineau, 2015)

The experiments were tested in three different scenarios, including an uncontrolled dynamic environment (Kim & Pineau, 2015). According to their result, the module which utilized inverse reinforcement learning used “a set of demonstration trajectories generated by an expert to learn the expert’s behavior” when it was faced with several features that have different states. In the uncontrolled dynamic experiment, the framework illustrated its ability to adapt to the changing environment, which suits our goal of making the system work and optimizing the evacuation in a real-world airport.

Asynchronous Advantage Actor Critic (A3C)

Asynchronous Advantage Actor Critic (A3C) is one of the most efficient algorithms of RL. The agents learn from repeated action, value, and feedback from the policy through interaction with the environment and optimize and converge to the best result. The A3C algorithm could utilize the advantage of asynchronous training and improve the agent's training speed. In the environment, multiple agents can explore and take actions independently while maintaining collaboration with each other. In other words, A3C distinguishes itself from traditional reinforcement learning algorithms by employing a parallelized training process, where multiple agents independently explore different copies of the environment simultaneously (Figure 5). This approach allows for diverse experience sampling, which reduces correlation in the training data and improves learning efficiency. Unlike many RL algorithms that update the global network synchronously, which can lead to bottlenecks and inefficient learning, A3C updates the global parameters asynchronously as soon as each agent completes its batch of experiences. This enables faster learning and adaptation, as the global network benefits from the cumulative knowledge of all agents without waiting for any single agent's episode to conclude.

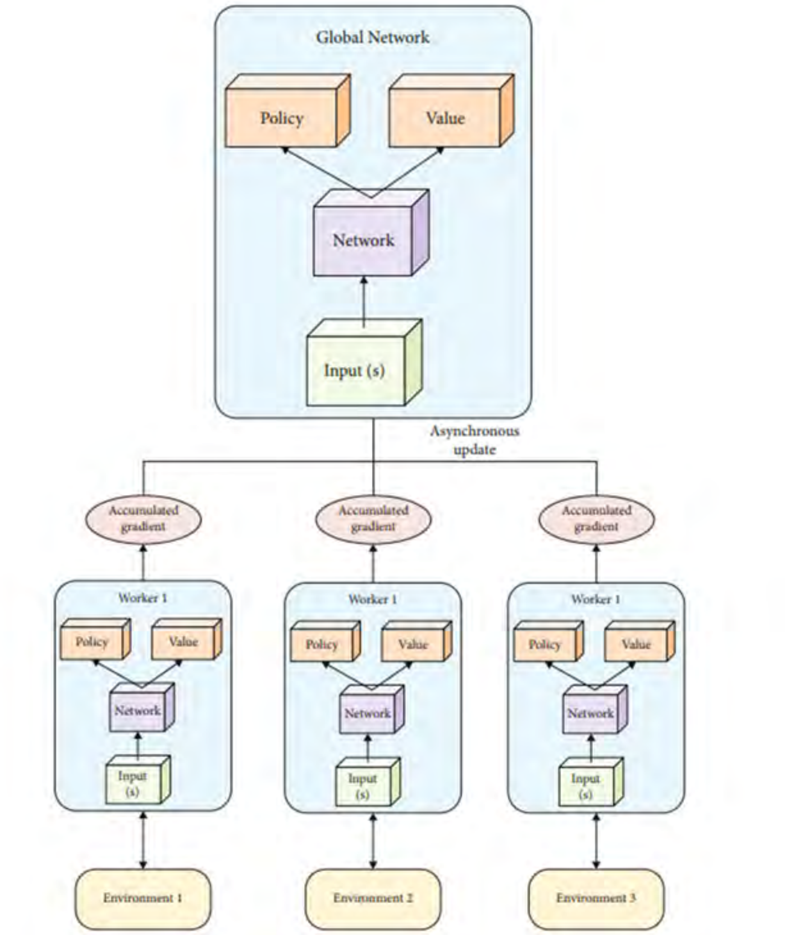

Figure 5. The A3C Algorithm Framework (Lee & Yoo, 2023)

Research done by Ding et al. (2021) illustrated the advantages of using A3C in optimizing the routing path for data transmission evaluation. In their research, Ding et al. (2021) implemented the delay tolerant networks (DTN) and applied Deep Q-learning Network (DQN) and A3C under different scenarios. The results show that A3C can get close to the top value with fewer episodes compared to the DQN algorithm, which indicates the efficiency and improvements of the A3C algorithm in node and link equilibrium. By using A3C, the evacuation model in a different situation could reach its optimal value and converge in a relatively short amount of time.

When emergencies occur, panic usually happens among people and everyone's behaviors and thoughts are usually different, which will lead to different evacuation intentions and actions. Therefore, studying human psychology and behavior is very important for the design and research of evacuation models. Since previous studies have not or rarely paid attention to psychology and behavior, Vorst (2010) suggested that psychological parameters and human behavior should be added to the emergency evacuation model, so as to make the evacuation model more comprehensive and realistic.

Vorst (2010) used John Leach’s Dynamic Disaster Model which divided a disaster into three phases including the pre-impact phase, impact phase, and post-impact phase. Moreover, a total of five stages were developed: the threat stage, warning stage in the pre-impact phase, recoil stage, rescue stage, and post-traumatic stage in the post-impact phase. The model also suggested that specific human behaviors that are considered psychological responses to disasters are consistent throughout each stage and phase and do not vary greatly from disaster to disaster.

Furthermore, Vorst (2010) also examined the evacuation rate before and after a hurricane to explain the importance of taking human factors into account when developing evacuation models. It was found that before the disaster, 30% of all residents refuse to evacuate, and when the specific emergency situations changed to be more urgent (during the disaster), 5% of all residents refuse to evacuate. These responses caused 25% longer evacuation times. Additionally, families’ or parties’ intention to stay together would affect evacuation efficiency as well. Moreover, it was also mentioned that women would feel more stressed than men, which would lead to a 20% longer evacuation time. Based on the research, Vorst (2010) proposed various parameters that can be used in evacuation models and concluded that considering psychological parameters and human factors would make evacuation models more comprehensive and realistic and make predictions more accurate.

Social-Force (SF) Models

Social-Force (SF) models for pedestrian dynamics were first introduced by Dirk Helbing and Péter Molnár in 1995. In their model, the motion and decisions of pedestrians can be described by social forces, an analogy from kinetic molecular motions, thus providing valuable information for planning and designing public area construction and evacuation (Helbing & Molnar, 1995). These 'social forces' are conceptualized as a set of invisible factors that influence pedestrian behavior, similar to the forces acting between molecules that govern their movement and interaction. Specifically, social forces in the context of pedestrian dynamics include the desire to reach a destination, the need to maintain a personal space bubble to avoid collisions with others, the tendency to align with the flow of surrounding people, and the impact of environmental elements such as barriers. Each of these forces plays a crucial role in shaping how individuals navigate through and interact with their surroundings, making the SF model a powerful tool for understanding and predicting pedestrian movement patterns in complex environments. Helbing and Molnar explained that the “forces” in the model were not simply exerted by the environment in which pedestrians were located; instead, they were the measure of internal motivations for each individual to make a decision or perform a specific motion.

Generally speaking, human behavior is hard to predict and irregular (i.e., “chaotic”), especially when there are multiple individuals who exist in the same environment. Their motivations and motions are unpredictable and sometimes even interact with each other to make the situation even more complex. However, when sticking to a relatively simple stochastic situation, the motion of individuals and behavioral models will be developed if “one restricts oneself to the description of behavioral probabilities that can be found in a huge population of individuals” (Helbing & Molnar, 1995), which refers to the gas-kinetic pedestrian model (Helbing, 1993).

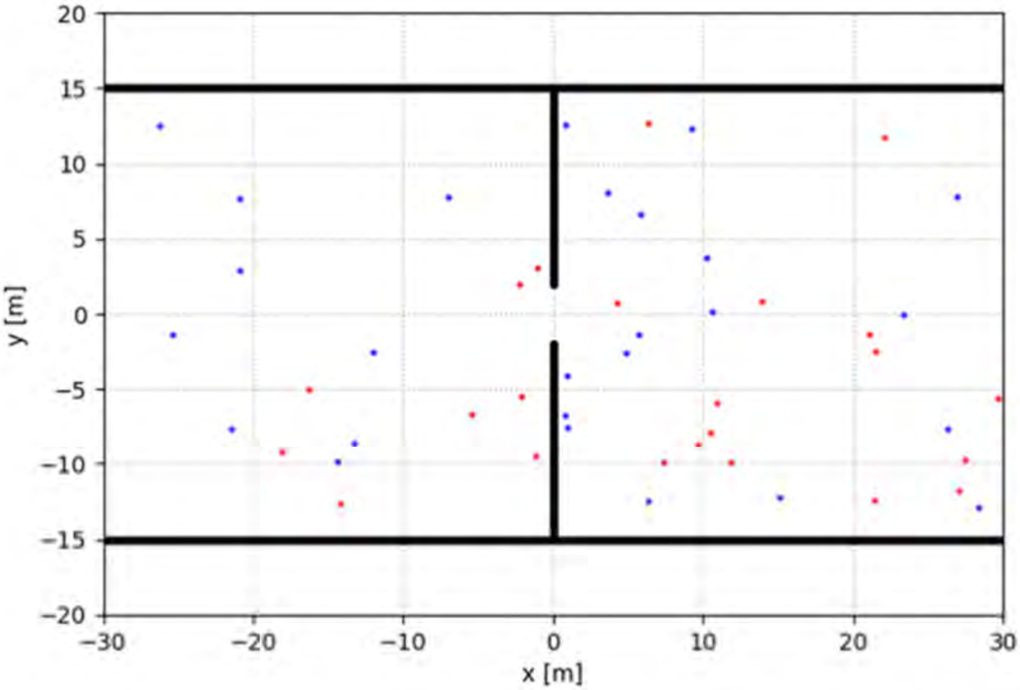

Figure 6. Illustration of the pedestrian model

Figure 6 shows how the system and model simulate the action and decision-making process of pedestrians. In the illustration, pedestrians are divided into two different groups, with each color representing a motion direction. One pedestrian from the left has crossed the narrow door, which makes the other pedestrians in the same group tend to move forward with him/her in the same direction. This behavior can be explained by social forces: the pedestrian who crossed influences the others through a social force that encourages group members to follow, enhancing their tendency to move in the same direction. Consequently, pedestrians in the opposite direction then have to wait, a behavior driven by social forces that represent the psychological and physical need to avoid collisions and maintain personal space. This waiting behavior is a manifestation of repulsive social forces exerted by the moving group, which temporarily increases the social pressure on the opposing pedestrians to pause their movement and yield space. The diameter of each circle represents the speed of each individual, indicating how social forces not only influence direction and decision-making but also the speed at which pedestrians feel comfortable moving in response to the surrounding social dynamics.

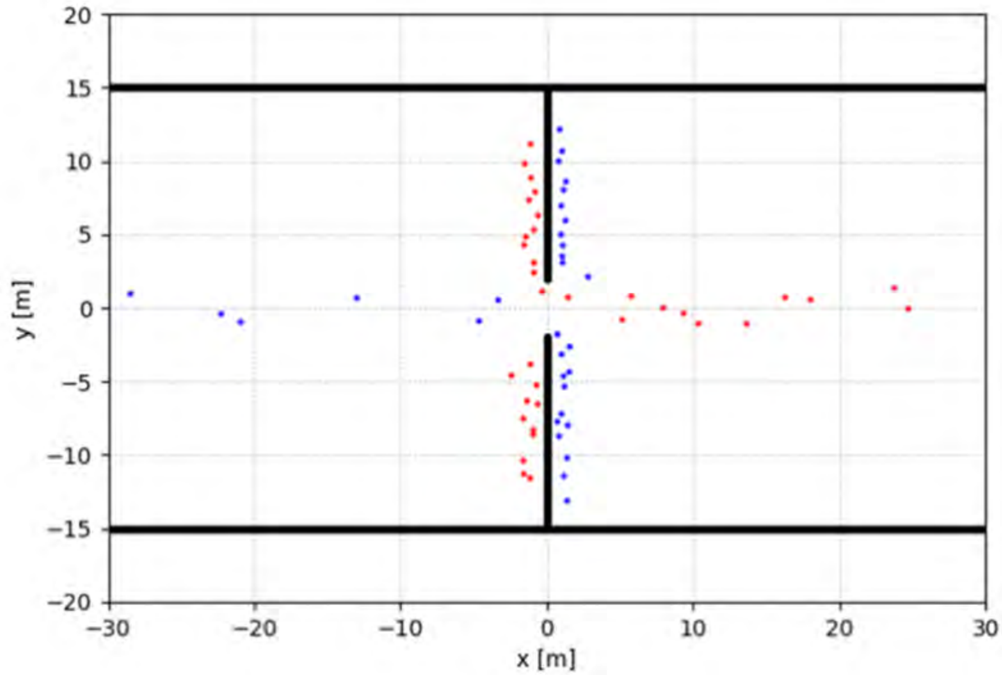

Figure 7. Observation of pedestrian density and lanes formation

Figure 7 shows how the social force model operates and the factors that affect individuals' decisions. In the figure, the flow of pedestrians can be observed with a uniform moving direction when the density of individuals is above a certain point. The computational results were calculated and based on "N=4 (or 5) lanes on a walkway that is 10 m wide and 50 m long" (Helbing & Molnar, 1995). The pedestrians still move in two opposite directions while there is no command or rule forcing them.

Zhang et al. implemented the SF model with deep reinforcement learning (DRL) algorithm and applied them to emergency evacuation in a room with obstacles (Zhang et al., 2021). They found that although the SF model is fairly successful in simulating emergency evacuation situations, the optimal evacuation result is still questionable in complex environments that have obstacles. Therefore, Zhang et al. (2021) developed the

DRL algorithm with the social-force model, which helped train the agents to find the evacuation exits fast and efficiently.

In the research, Zhang et al. (2021) first create a model of an empty room where there is no obstacle in the environment. Thus, the self-driven force model pointed to the nearest exit directly to test its correctness. According to the model, the median time that it took to evacuate for the two methods used in the research is "not significantly different" (Zhang et al., 2021). Then the model was changed to one obstacle and one exit in the room, where their method and the SF model provided similar results. In a follow-up study, they also investigate evacuation for a room that has multiple exits. The results indicate the ability of the agents that are trained to successfully evacuate from the nearest exit, which illustrated the efficiency of training in the SF model through a network that was shared among different agents while every agent kept its own motivation and action. In summary, the Q-learning approach and SF model they used was able to deal with emergency situations in a room that has multiple exits with multiple obstacles. The SF model is able to solve the problem of finding the most efficient and nearest exit for fast and safe evacuation under emergency situations in a complex environment.

According to Helbing and Molnar's research, it is possible to view and predict individuals' behavior using a set of equations of motion. When a pedestrian faces situations that happened or is normally confronted with, his/her action would be based on his/her experience and automatically respond with the best reaction in his/her opinion. Thus, the systematic temporal "changes *dwa/dt* of the preferred velocity *wa(t)* of pedestrian a are described by a velocity quantity *Fa(t)* that can be interpreted as a social force" (Helbing & Molnar, 1995). Although the force represents the effect from other individuals and the borders, the force is not actually applied to the individual’s body. Instead, the social force represents the motivation of the pedestrian to take action. In summary, a pedestrian would take action when he/she is subjected to external forces (from other individuals or borders) in a SF model. Moreover, this theory has been mathematically founded and adapted into research (Helbing, 1993).

Multi-Agent Reinforcement Learning (MARL)

According to the research done by Papoudakis et al. in 2020, a multi-agent reinforcement learning (MARL) system can be viewed as a “society of agents,” where agents can interact with each other and cooperate to achieve a common goal (Papoudakis et al., 2020). Compared to traditional single-agent methods that are used, a multi-agent system has the advantage that could shorten the time of simulation and calculation while staying accurate. Because of the multiple numbers of agents, a complex task or exploration can be solved by jointly interacting agents with coordinated behaviors in a short period of time (Gosavi, 2004). However, developing a multi-agent system does not mean simply adding multiple numbers of agents into the same environment and making them work. Since the complexity of the entire system will increase because of the increased interaction and action of agents, the interaction between agents is more important than the number of agents.

In a cooperative environment of multiple agents, the policies and instructions for each different agent will become difficult to perform or optimize sometimes because of the curse of dimensionality in the environment. On the other hand, the delay between the correlated actions and rewards is unignorable and significant in the cooperative multi-agent environment. Because of the determined character of maximizing the gain for each different agent, there will be the agent who could not receive their deserved reward and been affected by other agents' favorable actions. Thus, to address the problems of the curse of dimensionality and the unbalanced rewards between multiple agents, the agents should be trained from a macro aspect, which helps the agents have a bigger view of the whole environment and system, thus improving the results and solutions for the multi-agent model.

Researchers have combined the advantage of reinforcement learning and multi-agent system and applied MARL in various fields. Martinez-Gil et al. (2011) utilized the MARL technique in learning the behaviors and navigational patterns of humans to simulate virtual pedestrians' movements. They managed to reveal the basic movements and real characteristics of pedestrians, which indicates the validity of the MARL algorithm and model in simulating multiple agents' movements. In the paper, they prepared two RL algorithms based on Vector Quantization (VQ) for Q-Learning (VQQL) algorithm (Martinez-Gil et al., 2011). The results showed that both approaches are able to obtain adequate vector quantization for each different agent in the environment. The MARL technique can be used to solve a large amount of agents' movement problems like evacuation under emergency situations. With the help of reinforcement learning, the system is able to scale movement path and speed control, which is extremely helpful when emergency situations occur. At the end of the report, Martinez-Gil et al. (2011) also suggested that the two reinforcement learning algorithms used in the research could be integrated for the future work.

Emergencies that occur in small, enclosed spaces with high population densities, such as classrooms and movie theaters, are often the most likely to cause casualties. Thus, evacuation designs and plans are becoming increasingly important. Since it is hard for static evacuation methods to demonstrate the difference in safety for all alternative designs, and putting people in real emergencies is time-consuming, dangerous, and expensive, Liu et al. (2016) developed an agent-based simulation model via NetLogo to investigate the correlation between evacuation efficiency and classroom layout. This model, utilizing the principles of multi-agent systems (MAS), simulates the complex interactions between individual agents representing people during an evacuation. By establishing a set of behavioral rules for these agents based on real-world experience, the agent-based modeling (ABM) techniques employed in the study can successfully reflect the dynamic characteristics of human evacuation behaviors. Each agent operates autonomously yet interacts with other agents and the environment in a manner that captures the collective behavior of crowds in emergency situations, showcasing the power of MAS in understanding and optimizing evacuation processes.

There are four major parts in the simulated environment of Netlogo: turtles (agents), patches (grids that constitute a 2-D world), links (the social and physical connection between agents), and observers (supervisors). Liu et al. used two major types of general classroom layouts, while two different types of evacuation scenarios (a self-organized scene and a premeditated scene) were designed and thus led to four categories. The self-organized agents only are set as selfish and independent and only consider running out quickly, so the population density and distance to the nearest exit would most influence their decision-making process. In contrast, the premeditated agents who have conducted fire drills would follow the instructions and evacuate while avoiding collision with other agents. Speed limits were also introduced to get more realistic results. Agents were randomly divided into three groups that had a speed of 0.6 m/s, 1m/s, and 1.2 m/s. The results showed that a classroom layout with two exits would shorten evacuation time while the agents under premeditated behavior rules could evacuate both quick and safe (Liu et al., 2016).

Summary

The review of existing literature on emergency situations, aviation emergencies, evacuation processes, and technological approaches such as machine learning, RL including the A3C algorithm, SF models, and MARL algorithms highlights the significant progress made in this field. However, there is still a noticeable gap in the practical application and effectiveness of advanced computational models, specifically on utilizing the A3C algorithm in real-world evacuation scenarios. Most studies have concentrated on theoretical or simulated environments with limited exploration into complex and real-life emergency situations. These situations normally involve dynamic threats and a high density of individuals. This study aims to bridge this gap by not only applying the A3C algorithm in realistic evacuation simulations but also by comparing its efficacy with established models like DQN. The findings from this research enhance our understanding of the practicality and adaptability of RL algorithms in emergency evacuations, which is crucial in aviation emergencies where timely and efficient evacuation can significantly impact safety and survival. By advancing the knowledge in this field, this study contributes to the development of more effective evacuation strategies, potentially saving lives and optimizing emergency responses in high-risk situations.

Section 3: Methodology

The model and environment for this study were built using two types of algorithms: A3C and DQN. This section provides a detailed description and explanation of these two algorithms, including the models and code structure and how they were applied to the model and environment.

A3C Algorithm

The model's primary goal was to achieve efficient and optimal decision-making for evacuation during emergency situations using the A3C algorithm. As stated in the literature review section, A3C is a powerful reinforcement learning algorithm that addresses the challenges associated with traditional reinforcement learning methods. One of the main benefits of A3C is its ability to use multiple actors to explore different parts of the environment in parallel simultaneously. This speeds up the training and learning process and reduces the redundancies between agents and training environments (see Figure 8).

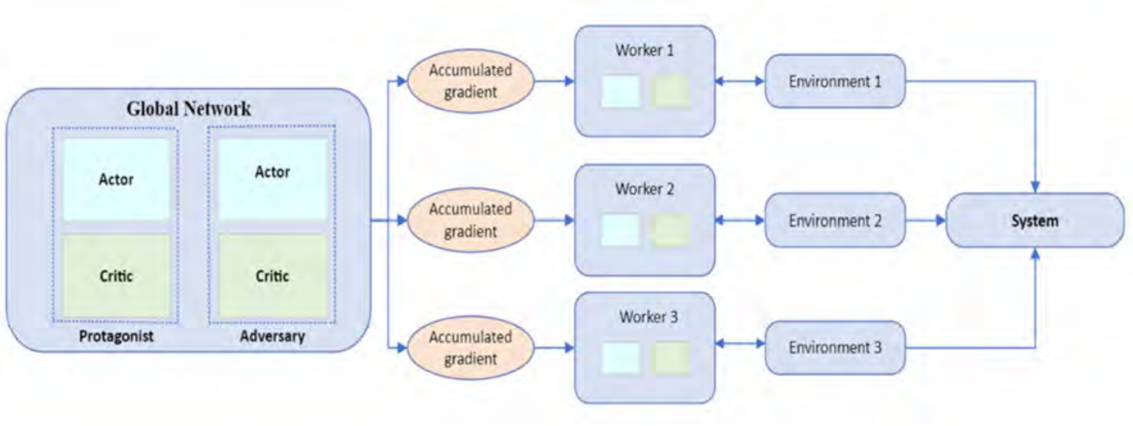

Figure 8. Asynchronous Advantage Actor-Critic Network

The A3C algorithm-based model consists of two components that achieve the Actor-Critic structure: the actor and the critic. The actor represents the policy function for the agents in the environment, defining the probability of each step for each agent based on the current state of the environment. One essential aspect of A3C is instructing and guiding the agents' actions for the best outcome. On the other hand, the critic plays an important role in estimating the value function of the current policy for the agents. It acts like a critic who evaluates the quality of the actor's actions. The critic will compute the advantage score and update the policy for the actor to obtain a better policy and rewards in the following episodes. By combining the advantages and functions of both actor and critic, A3C can utilize both the benefits of policy-based and value-based approaches to achieve the best learning efficiency. The policy-based approach directly optimizes the policy as it directly enhances the policy function that maps states to actions, while the value-based approach is more stable and has lower variance in their estimates compared to policy-based methods (Sutton and Barto, 2018). Therefore, A3C combines the strengths of both methods and makes it more effective in a wide range of RL problems, including those with complex action spaces.

The following pseudocode (Figure 9) illustrates the logic for the A3C algorithm and how each actor-learner thread works (Mnih, 2016). The θ represents the policy in this algorithm, which was strictly followed by the agent for taking actions and making decisions. While θv models the value function V, which evaluates the effectiveness of the action that agents took. Both θ and θv are implemented using convolutional neural networks (CNNs) with shared parameters, except for the output layer. In order to output the probabilities of actions for the agents to take, a softmax function was given to θ. For θv, a linear layer was used for outputting a scalar value for evaluation of the actions.

Algorithm 1 Parallelized Actor-Critic with Advantage Estimates

```
// Define global shared vectors $\theta$ (policy) and $\theta_v$ (value), and a global counter $T = 0$
// Define thread-specific vectors $\theta'$ (policy) and $\theta'_v$ (value)
Initialize thread counter: $t \leftarrow 1$
repeat
    Reset gradients: $d\theta \leftarrow 0$, $d\theta_v \leftarrow 0$
    Align thread-specific parameters: $\theta' \leftarrow \theta$, $\theta'_v \leftarrow \theta_v$
    Set thread iteration start: $t_{\text{start}} \leftarrow t$
    Obtain current state: $s_t$
    repeat
        Execute action $a_t$ as per policy $\pi(a_t|s_t; \theta')$
        Receive reward $r_t$ and observe new state $s_{t+1}$
        Increment thread counter: $t \leftarrow t + 1$
        Increment global counter: $T \leftarrow T + 1$
    until $s_t$ is terminal or $t - t_{\text{start}} = t_{\max}$
    Compute $R$ for state $s_t$: $R = 0$ if $s_t$ is terminal, or $R = V(s_t, \theta'_v)$ otherwise // Bootstrapping for non-terminal states
    for each step $i$ from $t - 1$ to $t_{\text{start}}$ do
        Update $R$: $R \leftarrow r_i + \gamma R$
        Calculate gradient for $\theta'$: $d\theta \leftarrow d\theta + \nabla_{\theta'} \log \pi(a_i|s_i; \theta')(R - V(s_i; \theta'_v))$
        Calculate gradient for $\theta'_v$: $d\theta_v \leftarrow d\theta_v + \partial(R - V(s_i; \theta'_v))^2 / \partial \theta'_v$
    end for
    Apply asynchronous updates to $\theta$ and $\theta_v$ using $d\theta$ and $d\theta_v$
until $T > T_{\max}$
```

Figure 9. Pseudocode for each actor-learner thread in A3C algorithm

Starting from initialization, the Global shared counter T was initialized to 0, while the thread step counter was initialized to 1. The parameter vectors in this algorithm are also set as two types: global shared parameter vectors θ and θv, and the thread-specific parameter vectors θ' and $\theta' v$. At each step, the algorithm will update the thread-specific parameter vectors θ' and $\theta' v$ to match the global parameter vectors θ and θv. The system then interacts with the environment, where the action a_t is chosen based on the current state St and the policy π $(a_t \mid St ; \theta')$. It observes the reward rt and the new state $St+1$. This step will repeat until a terminal state is reached or a maximum number of steps, which is $t - t_{start}$, are taken. When the algorithm encounters a terminal state, indicating the end of an episode, the expected future return R is set to 0, as there are no further rewards to be obtained beyond this point. If the final state is non-terminal, the return will be generated using the value function $R = V\,(st,\, \theta' v)$. The return R and accumulated gradients with respect to θ' and $\theta' v$ are calculated for each step from $t\text{-}1$ to t_{start}. The return is then calculated by summing up the rewards from the current state to the end of the episode, with each reward discounted by a factor of γ raised to the power of the number of time steps away from the current state:

$$R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots + \gamma^{T-t} r_T$$

Where r_t is the reward at time t and γ is the discount factor. The γ was set between 0 and 1 and determines the importance of future rewards, where a value of 0 means the agent will only consider the current reward and a value close to 1 indicates the agents will value the future rewards more strongly. Thus, in this project, the γ was set to 0.99, which is a relatively large gamma value in reinforcement learning applications, as the future rewards are considered almost as important as the immediate rewards in the case of successful evacuations in the airport. The discount factor closest to 1 will encourage the agents to consider long-term rewards and make them tend to choose actions based on better long-term outcomes. Also, for a stochastic dynamic environment like the airport, a higher discount factor could help stabilizing the learning process and reduce the variance in the updates. Although the training and learning process could be slow because of the large value of discount factor, 0.99 is still a good gamma value after the hyperparameter tunning process.

The A3C algorithm is an algorithm that uses the Policy Gradient framework, which normally uses the Temporal Differences-error (TD-error) method as its critic and evaluates how good the actor is. The following formula represents the empirical estimate of the policy gradient for RL problems:

$$\nabla_{\theta} R_{\theta} \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} \left(\sum_{t'=t}^{T_n} \gamma^{t'-t} r_{t'}^{n} - b \right) \nabla_{\theta} \mathrm{lo}\,\mathrm{g}\, \pi_{\theta} \left(a_t^n | s_t^n \right)$$

As a policy gradient method, A3C improves the stability of the method and convergence properties of policy gradient methods. The advantage function, used in A3C, measures the difference between the expected return of taking an action in a particular state and the expected return of behaving according to the current policy in that state. The advantage function can be written as follows:

$$Advantage\ (A_t) = R_t - V(s_t)$$

In this simplified formula, the R_t represents the cumulative discounted rewards from time to time and the $V(s_t)$ represents the critic part, which mainly provides estimation of the state value for the state s_t. The advantage essentially tells how much better or worse the action is by comparing it to the expectation of the critic part. To be more specific, the advantage formula can be detailed as follows:

$$A(s_t, a_t) = r_t + \gamma V \ \ (s_{t+1}) - V \ \ (s_t)$$

Here, the r_t is the reward at time t. As mentioned before, estimated by the critic, *V(st)* is the value of state *s_t* and γ is the discount factor. By using this advantage formula, the algorithm is capable of calculating the difference between cumulative discounted rewards and the expected rewards given by critic, and thus can evaluate how much better the action truly is.

As the advantage formula gauges the expected return disparity between the action and current policy, the update formula is also essential for calculating the expected reward gradient in respect to the policy parameters. The A3C policy optimization formula can be expressed as follow:

$$\nabla_\theta J(\theta) \approx \nabla_\theta \log \pi_\theta \, (a_t|s_t) A(s_t, a_t)$$

$\pi_\theta(a_t|s_t)$ is the probability of taking action in state s_t under policy θ. The update formula is essentially computing the gradient of the expected reward with respect to the policy parameters, scaled by the advantage.

In practice, the A3C algorithm also introduces an entropy regularization term to encourage exploration during the training process. By encouraging the agents to choose something without a curtain decision or knowledge (exploration), the agents could actually learn more from the environment and the policy by interacting more. This is known as the exploration-exploitation trade-off (Figure 10), which is a fundamental dilemma in reinforcement learning.

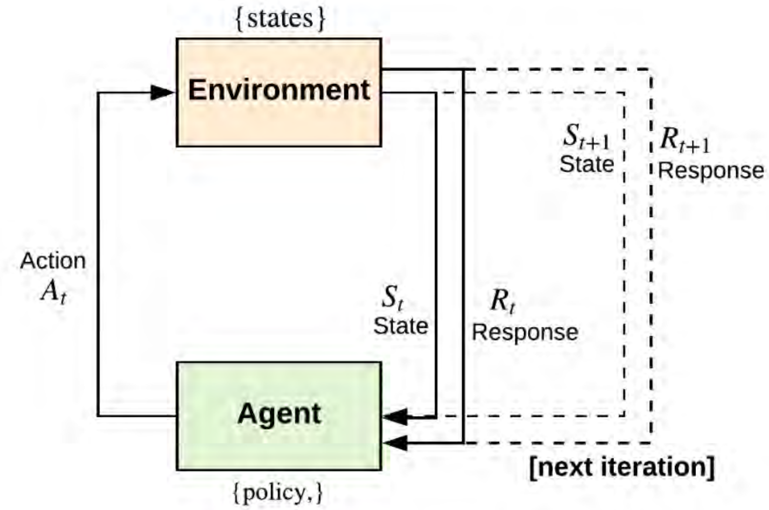

Figure 10. The Exploration-exploitation Trade-off

After adding the consideration of the exploration-exploitation trade-off, the total loss for the A3C algorithm typically consists of three parts: the policy gradient loss, value function loss, and entropy regularization. The full objective function that A3C tries to optimize can be formulated as:

$$L(\theta) = -\log \pi_\theta\,(a_t|s_t)A(s_t, a_t) + \lambda\left(V(s_t) - \hat{V}(s_t)\right)^2 - \beta\sum_a \pi_\theta(a|s_t)\log \pi_\theta\,(a|s_t)$$

The first term, $-\log \pi_\theta\,(a_t|s_t)A(s_t, a_t)$, represents the policy gradient loss. This term encourages the probability of actions that lead to higher-than-expected returns to increase and the probability of actions that lead to lower-than-expected returns to decrease. Here, $A(s_t, a_t)$ is the advantage function, which measures how much better an action is compared to the average action at state S_t. The second term, $\lambda\left(V(s_t) - \hat{V}(s_t)\right)^2$, is the value function loss, which is a mean-squared error that penalizes the agent for incorrect value estimates. The agent's current estimate of the state value $V(s_t)$ is compared to the target value $\hat{V}(s_t)$, which is computed using the bootstrapped returns. This value function loss ensures that the value function is a good predictor of future rewards, which is critical for calculating accurate advantage estimates. Last but not least, the entropy regularization $-\beta \sum_{a} \pi_{\theta}(a|s_t) \log \pi_{\theta} (a|s_t)$ encourages exploratory behavior by dissuading the policy from becoming overly deterministic, thus maintaining a healthy balance between exploration and exploitation. As the training process minimizes these three terms of the loss function, it concurrently refines the parameters of both the policy and the value function. This refinement guides the agent towards a more effective strategy over successive iterations.

DQN Algorithm

To compare efficiency in terms of time and learning curves by measuring the average reward, the Q-learning algorithm was employed as a foundational component of the Deep Q-Network (DQN) to simulate evacuation procedures in an airport environment. DQN served primarily as a standard baseline for this research because it is commonly used in many types of simulations, including evacuation scenarios, within reinforcement learning studies. As one of the earliest deep learning-based RL algorithms, DQN successfully demonstrated that neural networks could approximate Q-values for high-dimensional state spaces. This capability makes it well-suited for the complex simulations of this research and establishes a solid performance benchmark for comparison with the A3C algorithm.

Q-learning is an RL algorithm that seeks to learn the value of an action in a particular state, aiming to maximize the total reward. The pseudocode provided (Figure 11) offers a clear and basic illustration for the implementation of the Q-learning process (Yang, 2020).

Algorithm 2 Q-learning Algorithm

```
Initialize $Q(s, a)$ as zero;
for each episode do
    Initialize state $s$
    for each step of the episode do
        Choose action $a$ from state $s$ using $Q$-table
        Take action $a$, observe $r$, $s'$
        $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$
        $s \leftarrow s'$
    end for
end for
until $s$ is terminal
```

Figure 11: Pseudocode of the Q-learning algorithm as the foundational component of DQN

The Q-values are initialized arbitrarily, providing a starting point for the algorithm to begin learning. These values represent the expected utility of taking a given action in a given state and are updated iteratively as the agent interacts with the environment. This initialization provides a baseline from which the algorithm can begin the learning process. Then the learning occurs over a series of episodes. Each episode begins with the initialization of the states (s), representing the starting conditions of the evacuation simulation, such as the initial position of agents within the airport. Within each episode, actions are chosen using an ε-greedy policy, which involves selecting the action with the highest Q-value (exploitation) with probability 1−ϵ or a random action (exploration) with probability ϵ:

$$With\ probability\ 1 - \varepsilon: Choose\ a = argmax\_a\ Q(s, a)$$

$$With\ probability\ \varepsilon: Choose\ a\ randomly$$

After executing an action, the Q-value is updated based on the Bellman equation, which incorporates the immediate reward and the discounted maximum future reward:

$$Q(s, a) \leftarrow Q(s, a) + \alpha\,[r + \gamma \max(over\ a')\, Q(s', a') – Q(s, a)]$$

In the DQN algorithm, $Q(s, a)$ represents the current estimated value of taking action a in state s, encapsulating the expected utility of this decision. The learning rate, denoted as α, plays a critical role in balancing new incoming information with existing knowledge, thereby controlling the rate at which the algorithm adapts. The immediate reward received from taking action a in state s is represented by r, serving as a direct feedback mechanism for the action's effectiveness. The discount factor, γ, quantifies the importance of future rewards, enabling the algorithm to prioritize short-term gains against long-term benefits. Upon taking action a, the agent transitions to a new state s', with $\max(over\ a')\ Q(s', a')$ indicating the best possible future reward attainable from this new state, guiding strategic decisions towards the most beneficial outcomes. This update process is repeated for each step within an episode, continuing until a terminal state is reached. This marks the conclusion of one simulation run, with the goal of optimizing evacuation strategies in the face of dynamic challenges and threats.

The DQN algorithm improves the classical Q-learning framework by incorporating a deep neural network to estimate the Q-value function. This is crucial for handling the high-dimensional state spaces found in complex environments, such as airports. The algorithm strategically uses replay memory to store transitions experienced during the simulation, which are later randomly sampled to update the neural network. This method diversifies the learning experience, preventing overfitting to recent transitions and smoothing the training process over numerous episodes. As a result, the DQN algorithm can incrementally refine its Q-values and gradually identify optimal evacuation paths. The algorithm demonstrates an ability to learn and improve evacuation strategies iteratively by confronting and negotiating various challenges, such as dynamic threats and the coordination of multiple agents.

The utilization of the DQN algorithm, supported by an advanced Q-learning framework, enables effective simulation and enhancement of airport evacuation procedures. The algorithm's adept handling of complex scenarios through continuous learning and adaptation illustrates its potential in optimizing evacuation paths, ultimately contributing to safer and more efficient emergency responses. In this study, DQN is not only a powerful tool but also a critical benchmark for evaluating the A3C algorithm. By comparing the performance and learning dynamics of DQN and A3C within the same simulation environment, we can clearly delineate the strengths and limitations of each approach. This comparative analysis enhances our comprehension of the applicability of RL algorithms to real-world challenges, such as airport evacuations. It positions DQN as a fundamental model against which the innovations and improvements of A3C can be measured, highlighting areas where A3C may offer advantages in terms of learning efficiency, scalability, and adaptability to complex, dynamic environments.

Environment

The environment utilized in this project was developed and customized solely using the Gym toolkit (Brockman et al., 2016). Developed by OpenAI, Gym was designed as a toolkit for creating and comparing various types of RL algorithms. As the agents operate under the policies of different algorithms within the Gym environment, they generate varying average rewards and levels of efficiency, culminating in unique learning results. The environments available through Gym span from traditional control tasks and games, such as Atari, to robotic simulations. In this particular project, a unique Gym environment was developed from scratch, simulating a real airport environment, without dependence on pre-existing environments. The customized environment permits agents to function across any algorithm, evading limitations to a specific algorithm. The customized environment created by Gym is ideal for the project's objective of fitting agents and enabling them to find the optimal evacuation path under various algorithms in different airports. Gym's capabilities allow the transformation of any real-life airport into a virtual environment, where agents can determine the best evacuation route. In other words, the biggest advantage of using Gym environment is that it, combined with the self-learning feature of machine learning, generates the optimal route for any actual airport, thereby saving time and lives.

This project's customized environment consists of several key components: environment space, agent, observation, reward, and episode. The environment space is where agents move and take actions, defined by a state space and an action space. The state space is the system's current state, as indicated by the agent location, which will be clearly defined in the code. The action space encompasses all nine potential actions an agent can take within the environment: the four cardinal directions (North, South, East, West), the four intercardinal or ordinal directions (Northeast, Southeast, Southwest, Northwest), and remain static. The environment's comprehensiveness and detail are enhanced by these two spaces. The agent is a crucial element within the environment, interacting by taking actions within specific states. When the agent takes action, it receives observations and rewards from the environment, which helps it make better decisions in future episodes based on its policy and learning algorithms. The environment provides agents with observations and rewards, which are information about the current state of the system and scalar feedback based on the agent's actions. By utilizing these fundamental elements, agents can continue to take actions and maximize their cumulative reward over time.

The customized environment was created by transforming a pre-existing airport map. The map was based on a real-world local airport to ensure a more accurate and comprehensive result for emergency evacuation (see Figure 12). Adopting the real-world airport map into the reinforcement learning environment offers advantages such as increased realism, complexity, and detail. The use of clear and concise language with a logical flow of information ensures that the text is comprehensible and objective. The realistic and complex environment of real-world airports, as opposed to the pre-made environment in Gym, can aid in the development and training of more robust and capable algorithms such as A3C and DQN. These algorithms are better equipped to handle a variety of different scenarios. Additionally, the environment built around the local airport can serve as a benchmark for various algorithms and provide a shared foundation for comparing their performance in the same environment.

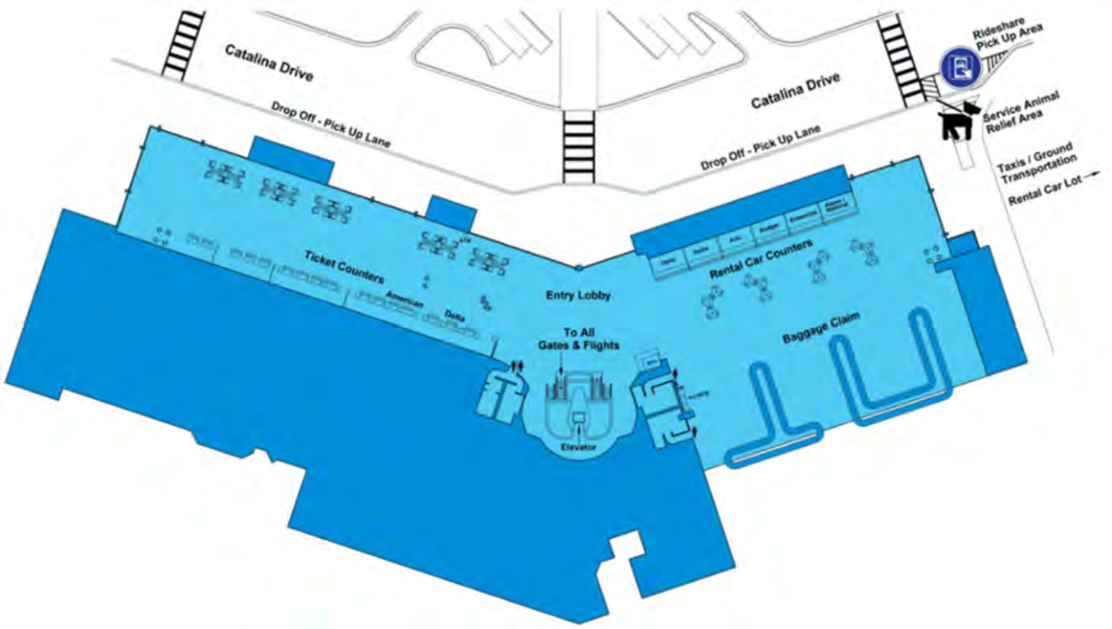

Figure 12. A Local Airport Ground Floor Map

The normal airport operating process consists of four phases: check-in, security, waiting area, and boarding (Figure 13). These phases typically occur at different locations within the airport, which may have multiple floors. This project focuses on the first floor of the airport, where the check-in and security phases take place. Compared to the first two phases, there is less probability of emergency situations occurring in the waiting and boarding area. This is because passengers have already passed the security check and are carrying baggage and items that strictly follow the rules and restrictions of the airport and the Transportation Security Administration (TSA). Emergencies, such as bomb threats and fires, tend to occur more frequently during the first two phases: check-in and security on the ground floor of the airport.

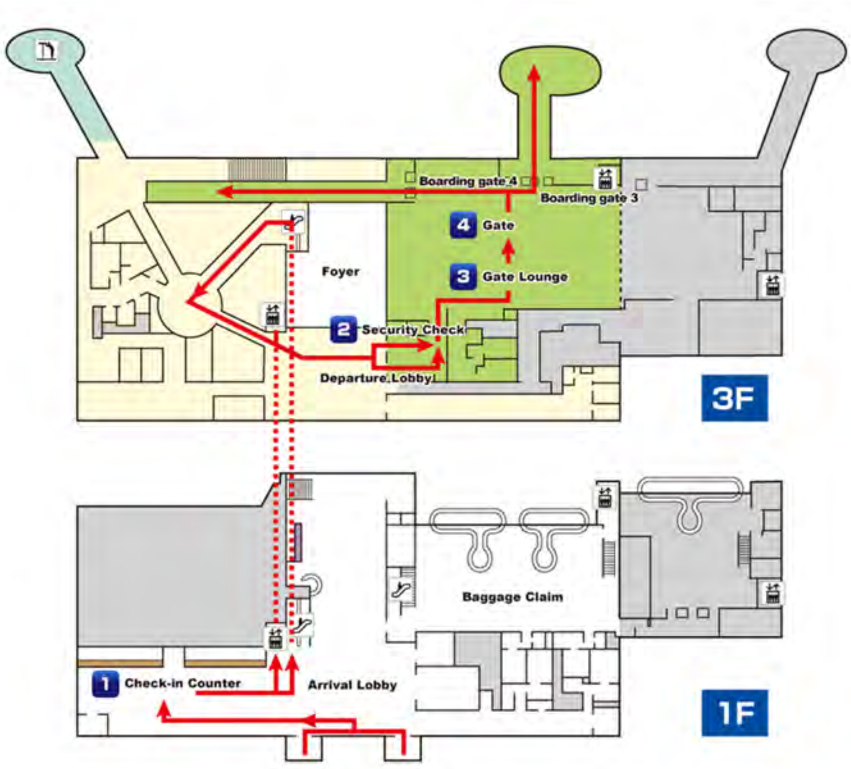

Figure 13. The Four Phases of Airport Operation

To correctly simulate the local airport map, we define ‘self.walls’ in the customized environment to draw the outline as well as all of the obstacles in the airport. The line ‘self.walls = np.full((self.size, self.size), False)’ is used to create a 2D array (or matrix) with the shape defined by ‘(self.size, self.size)’. And each element of this array is initially set to False to simulate the existence of the wall and obstacles in the airport. When agents in the model take action, each of them will normally take one step in each direction it choses. For walking distance, the step length on average is 2.2 feet for a woman and 2.5 feet for a man (StepsApp, 2023). When there are emergency situations, passengers tend to run and move faster than normal walking, then the distance of traveling will become bigger. Therefore, the size value for the map was set to 50 pixels, which makes the environment come to 2,500 (50x50) pixels large. This causes the model to have a large and reasonable environment for the agent to explore and learn by taking steps. With the consideration of step length, obstacles in the airport, and the compatibility of Gym customized environment, the complete environment was built as shown in Figure 14.

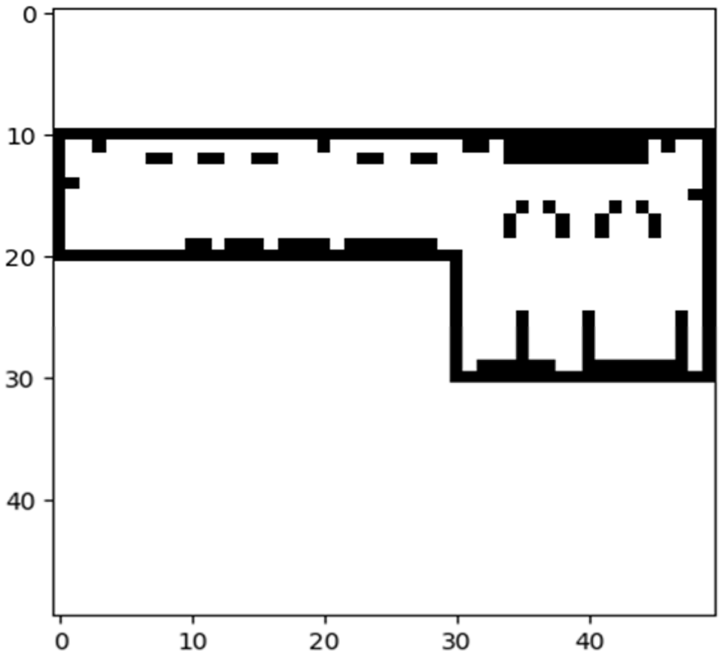

Figure 14. The Customized Environment

To simulate the local airport better, seven (7) exits in total were designed and put into the environment at the location where the real ones are. They are marked as Exit 1 to Exit 7 from left to right in the environment and highlighted using green color. Because of the compatibility of Gym, instead of building the map with angles, the airport ground floor was divided into two parts, left and right, and put them together as a rectangle. This makes the environment exclude other affecting factors and extra rooms for entering the second floor.

In this environment (see Figure 15), the ticket counters are kept to provide a realistic simulation of the airport, as well as the rental car counters and baggage claim.

With the consideration of the first two phases, check-in and security, we put the threat near the baggage claim area, representing a potential fire accident or bomb threat. Therefore, there will be four situations based on the environment settings and number of agents, which are: single-agent static threat situation, single-agent moving threat situation, multi-agent static threat situation, and multi-agent moving threat situation. Each of the situations was test and used as the benchmark platform for both A3C and DQN algorithms in the project.

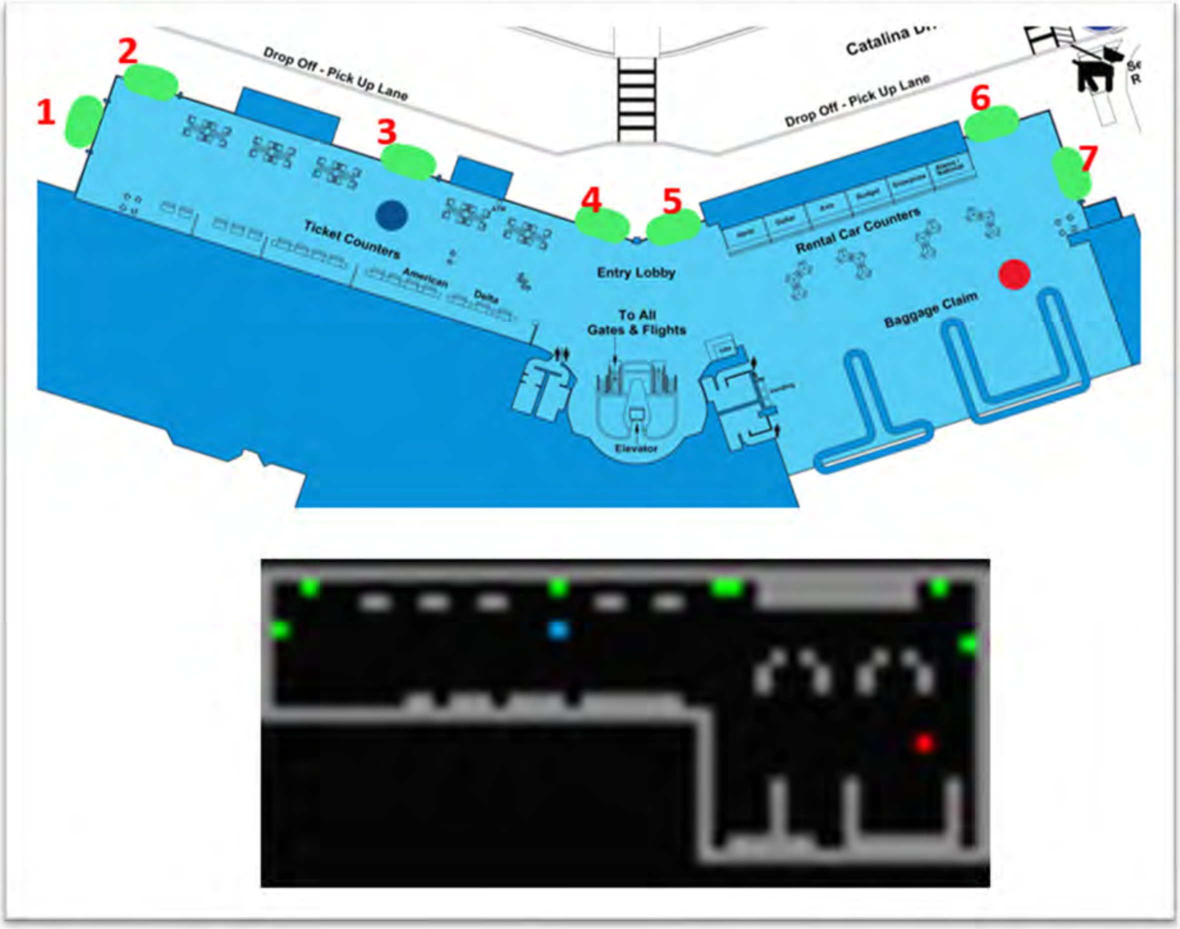

Figure 15. The Customized Environment by Adopting the Real-world Airport Map

Static Threat Environment

In the static threat scenario, a single agent is tasked with finding the shortest, fastest, and safest route to escape from the airport under the assumption of a static threat. Without the obstruction of other agents (passengers), this individual is free to explore the environment with each step taken. The experience gained will be collected and learned from through this exploration-exploitation process using the algorithms. The scenario involves a bomb threat at the airport, which poses a danger to thousands of lives. To make the situation more realistic and dangerous, the threat is placed near the baggage claim area. The bomb threat is represented by a red dot that remains static while the agents explore the optimal path to escape from the airport. Figure 16 shows the complete environment, including the single agent and the bomb threat.

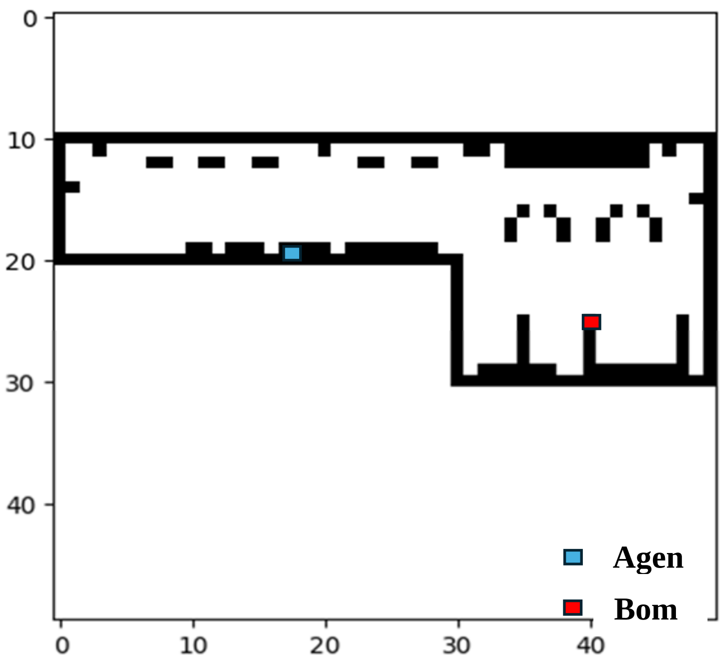

Figure 16. Single Agent Static Threat Environment

Moving Threat Environment

In the moving threat scenario, the limitation of one agent in the environment remains unchanged. However, the threat has been replaced with a more dangerous and realistic object: fire. Emergency situations, such as fire hazards, differ in scale and severity from other hazards and emergencies. As previously stated, while proper adherence to normal operations can mitigate most emergency situations to some extent, reactive measures, such as evacuation plans and real-time decision-making systems, still play a crucial role in effectively responding to emergencies. Therefore, in the event of a sudden fire at the airport, emergency response and evacuation plans can greatly mitigate or resolve the situation without causing significant disruption to normal operations. The threat in this case was simulated using a fire spreading model coded by Martin in 2023 using MATLAB, following a specific pattern to mimic real-world scenarios (Martin, 2023). The original model simulated a forest fire using a spreading model. It created a random forest on an n x n grid, with a probability p of igniting a tree that had not yet burned in its Moore neighborhood. This added more randomness to the model during the spreading process (see Figure 17).

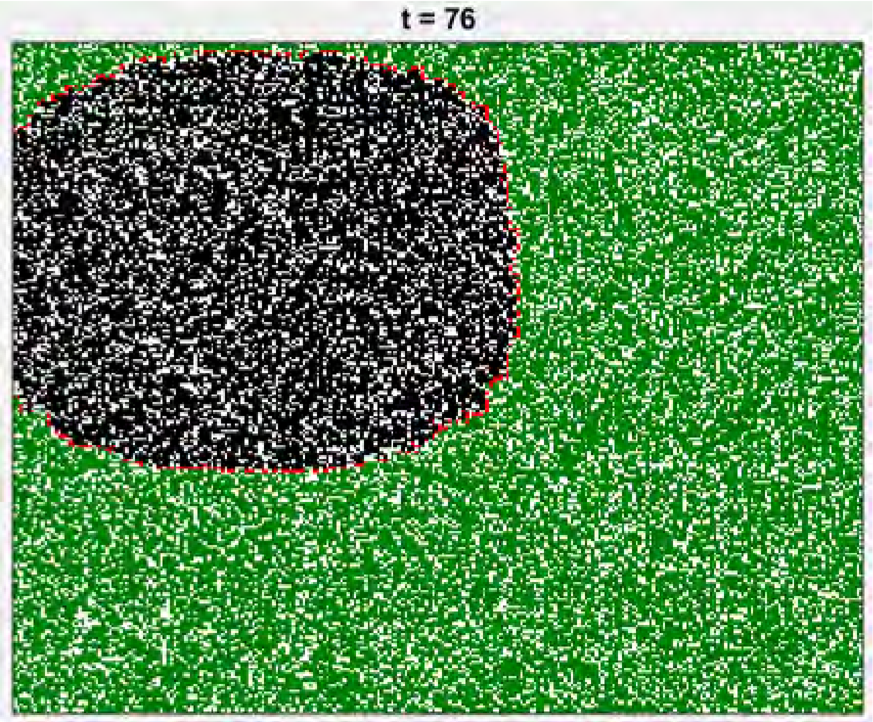

Figure 17. Simulation of a Forest Fire with the Spreading Model in MATLAB (Martin, 2023)

To improve the accuracy of this model as a representation of fire spreading patterns in airports, the probability of igniting a tree was modified to the probability of spreading to its closest cell and enlarging itself. This means that the fire will spread regardless of whether there is a 'tree' in its Moore neighborhood or not. As a result, the fire spread model in this project becomes more realistic and quantifiable with each step the agent takes and learns (Figure 18).

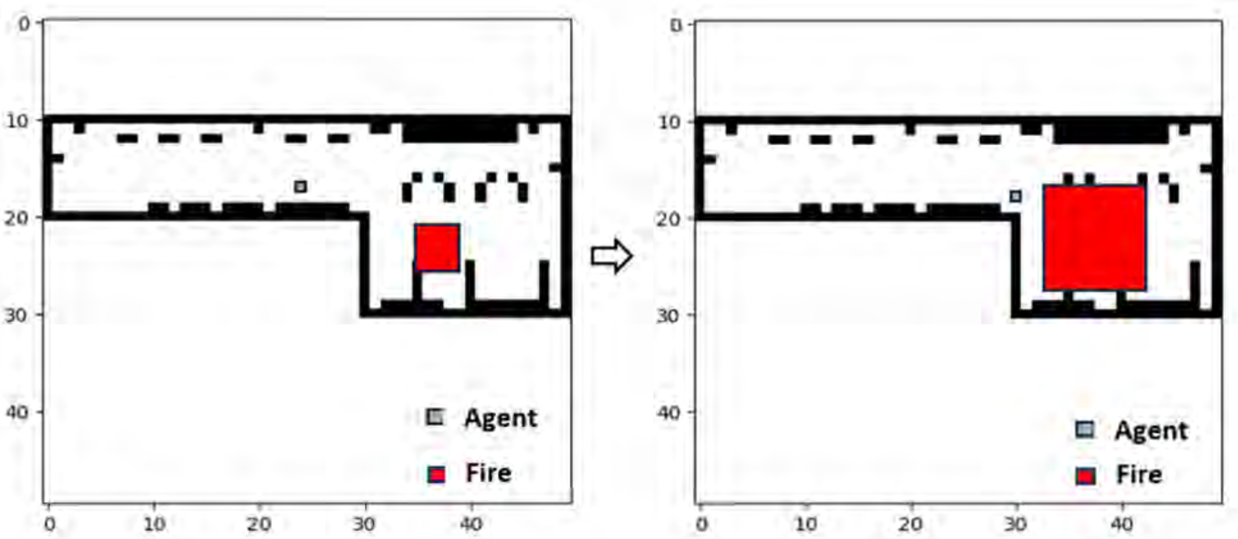

Figure 18. Single-Agent Fire Environment Changes with Episodes

The model was run on a computer equipped with an AMD Ryzen 5600X 3D CPU, an NVIDIA GeForce RTX 3060 Ti GPU, and 16GB of RAM. This setup provided the necessary computational resources for the demands of the RL algorithms used in the study. Each model and simulation was conducted over 1,000 episodes, with each episode representing a complete run of the environment from start to finish. The finish could be a successful escape from the airport, a collision with the static or moving threats, or even a crash between other agents for a multi-agent situation. To ensure statistical significance and account for variability, each model was tested across at least 50 trials. Throughout these episodes, key data points were collected, including metrics such as cumulative rewards per episode, the frequency with which specific exits were taken, and the duration of each episode from start to finish. The cumulative rewards per episode serve as the basis for a comprehensive analysis of the models' performance in terms of each algorithm's learning process. To facilitate a direct comparison between the two models under investigation, identical conditions were maintained across trials, and the same sets of metrics were used for evaluation. The comparison of learning curve and efficiency measured in seconds focused not only on the efficiency and effectiveness of the models in achieving the set objectives but also on their learning behavior over time and adaptability to the environment.

Section 4: Results

To evaluate and summarize the findings of the evacuation under different settings, the results are classified and compared across three distinct scenarios: a single-agent static threat environment, a single-agent moving threat environment, and a multi-agent environment. In each scenario, evacuation simulations were conducted using two different algorithms, A3C and DQN, to compare their efficiencies. This comparison involved measuring the time taken for each successful evacuation in seconds and analyzing the learning process by evaluating the cumulative rewards per episode. These metrics helped assess the performance of the two algorithms under identical environmental conditions and settings.

Static Threat Environment

Efficiency was measured in terms of the average time in seconds taken for the single agent to evacuate from the static threat environment. Figure 19 shows a distinct advantage of A3C over DQN in the static threat environment. In the graph, the x-axis represents the number of episodes and the y-axis shows the average evacuation time in seconds. As shown in Figure 19, A3C (in blue) consistently demonstrates a faster evacuation time across episodes compared to DQN (in red), showcasing its distinct advantage in this aspect. Specifically, the average evacuation time for A3C is approximately 22.32 seconds and, for DQN, it is about 39.75 seconds. This shows that A3C is approximately 43.86% faster than DQN in terms of the average time taken for agent evacuation. This percentage reflects the significant efficiency advantage of A3C over DQN in this scenario. The better efficiency of A3C compared to DQN can be attributed to the asynchronous learning process. In this case, the agent was learning and updating the strategies every time it finished using the DQN algorithm and policy, while A3C has multiple threads to learn and update its strategies simultaneously, which is significantly faster than DQN in terms of the speed of exploring the environment and learning the policy.

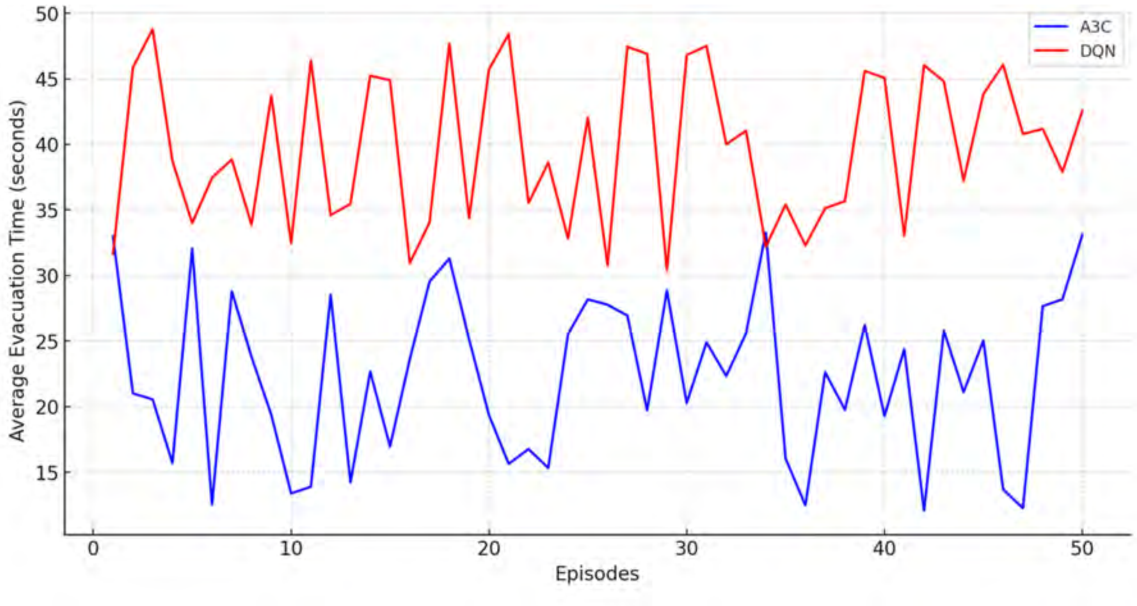

Figure 19. Average Time Taken for Agent Evacuation Static Threat Environment

Moreover, from the learning curve aspect, A3C and DQN do show a different number of episodes required for the agents to reach a stable evacuation strategy. The learning process of both A3C and DQN algorithms was analyzed across various environments, including scenarios with single agent static threats, single agent moving threats, and multi-agent contexts. This comprehensive analysis was designed to assess how each algorithm adapts and optimizes its strategy in response to different types of challenges.

For the single agent static threat environment, the result illustrates that both A3C and DQN displayed a similar pattern in the learning curve. The static nature of the threat allowed both algorithms to quickly learn and devise effective strategies. However, A3C has maintained a slight edge in terms of quicker strategy optimization, as indicated by its steeper learning curve in the initial episodes. The stable evacuation strategy was reached at around 100 to 200 episodes for A3C algorithm, while for DQN, the stable evacuation strategy was reached at around 200 to 300 episodes in this single agent static threat environment (Figure 20). This may be due to DQN's inherent learning mechanism, which might require more iterations to effectively capture and respond to the environmental parameters in this scenario.

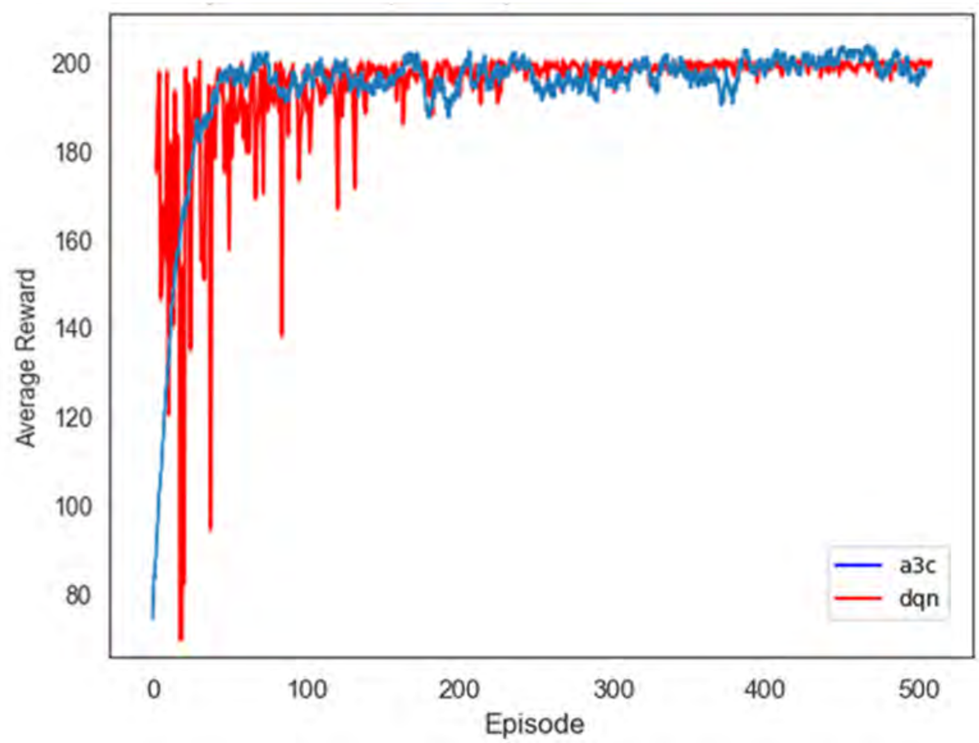

Figure 20. Average Reward Comparison in Single Agent Static Environment

Moving Threat Environment

In the single agent moving threat environment, the learning curves show a more notable difference in the efficiency and how each algorithm deals with the dynamic threats in the airport. A3C's response was swift and showed a rapid decrease in evacuation time as the agent started to learn and navigate in order to avoid the moving threat efficiently. While DQN took longer time to adapt and form a stable evacuation strategy compared to the swift learning process of A3C as seen in the gradual slope of the learning curve (Figure 21).

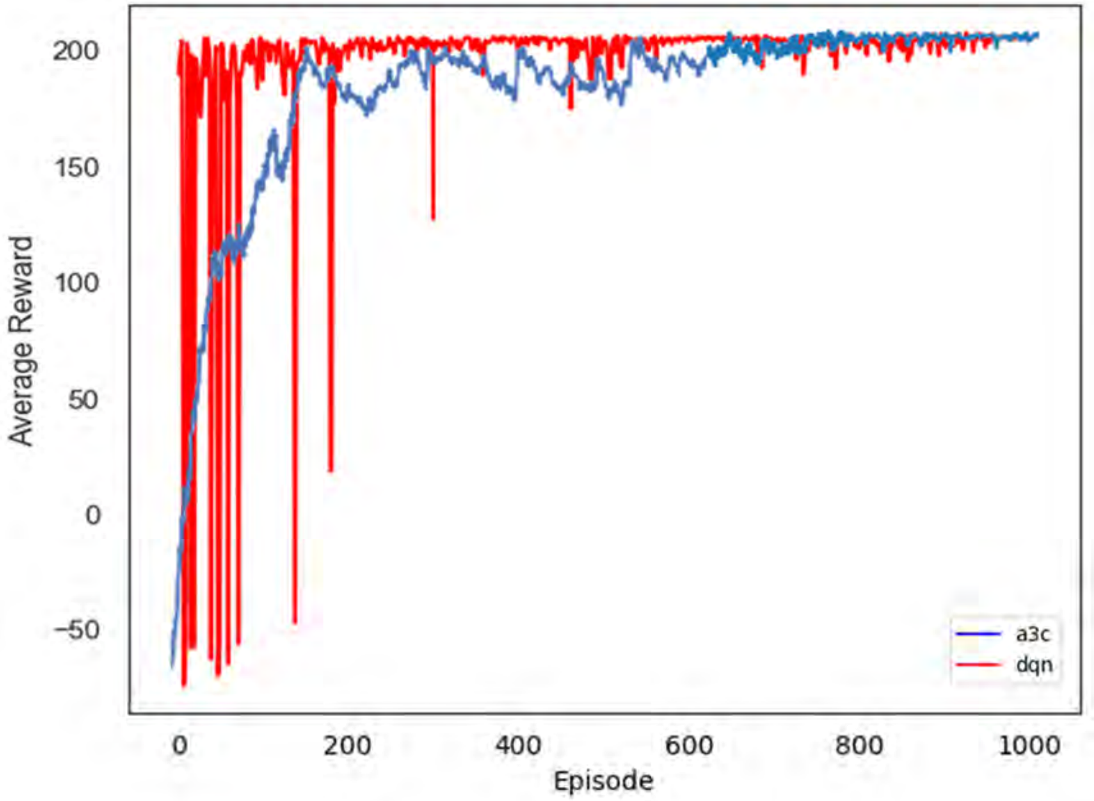

Figure 21. Average Reward Comparison in Single Agent Moving Environment

Compared to the learning curve of static threat environment setting, the results generated in moving environment are more volatile, while the learning curves in static environments exhibited a relatively smooth progression towards stability. In the moving threat environment, both A3C and DQN displayed learning curves with significant fluctuations. And this was characterized by periodic increases and decreases in the algorithms' learning curve across episodes, indicating a continuous process of adaptation and re-adaptation to the changing positions and patterns of the moving threat as the fire is enlarge itself and spreads in a specific pattern. This indicates the capability of the continuous learning process of A3C, which could quickly adapt to the ever-changing environment that includes the threat that needs to be avoided. Moreover, the fluctuations in the learning curve align well with the concept of A3C algorithm, which enables rapid response and adaption to the surrounding environment and the moving threat but can lead to some temporary performance dips as the new strategy is being adapted and implemented.

Multi-agent Environment

The multi-agent environment presented a more complex challenge. A3C's asynchronous characteristic provides more advantages in this scenario, which allows multiple agents to learn simultaneously and synergistically. This led to a more efficient learning process, as reflected in the steeper learning curve (Figure 22). DQN, while effective in learning cooperative strategies, exhibited a slower convergence, indicated its limitations in handling simultaneous learning tasks like the multi-agent evacuation mission.

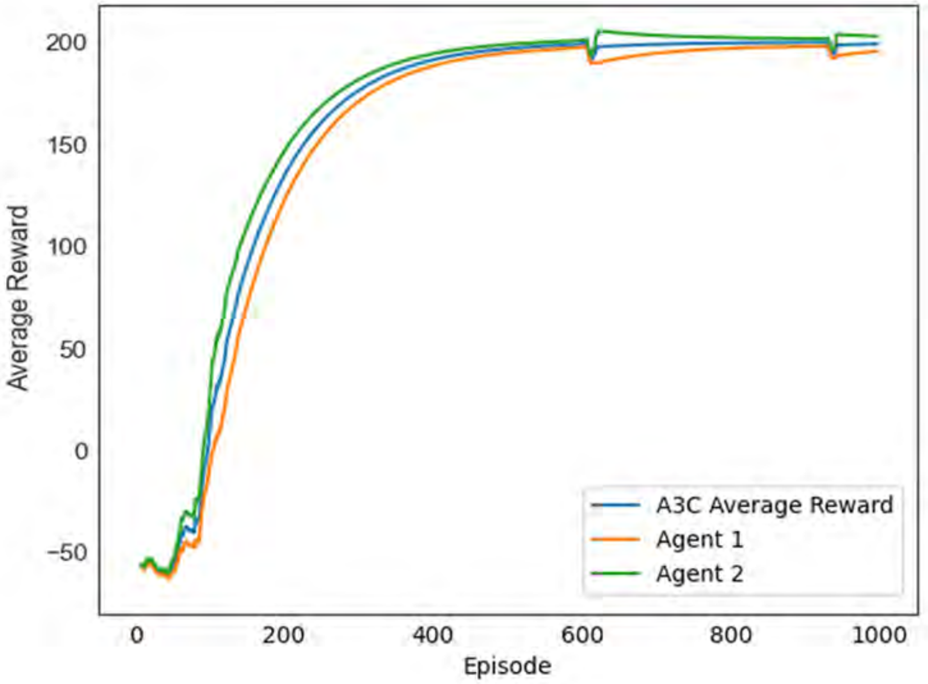

Figure 22. Average Reward of A3C in Multi-Agent Environment

Compared to DQN, A3C's strength in a multi-agent environment comes from its asynchronous learning mechanism. This unique feature allows multiple agents to explore and learn from the environment independently while contributing collectively to the overall learning process. This approach not only accelerates learning but also forms faster development of strategies among multiple agents, where the experiences of one agent can indirectly influence and enhance the learning of others (Figure 23). The experiments demonstrate that effective cooperative strategies are achieved more rapidly and efficiently, as shown by the steeper learning curve.

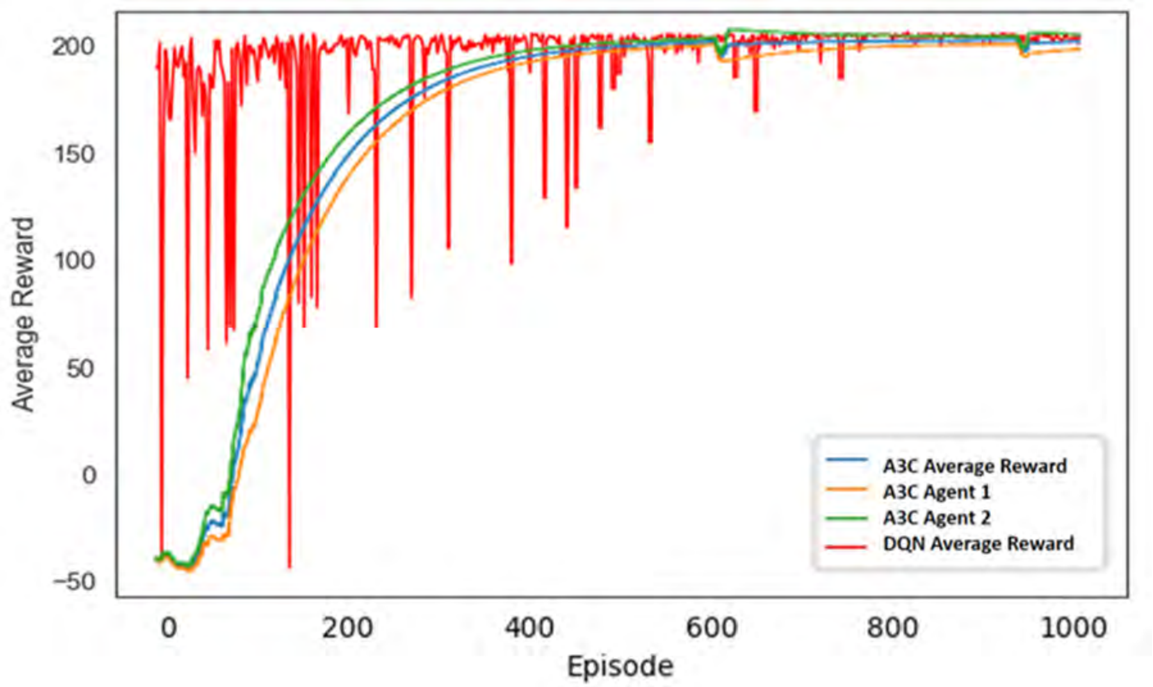

Figure 23. Average Reward Comparison in the Multi-Agent Environment

Although DQN is capable of formulating cooperative strategies, it faces challenges in the multi-agent context. Its learning mechanism, which is more linear and singular in approach, struggles with the simultaneous learning tasks required in this environment. The convergence towards effective cooperative behavior is slower, indicating a limitation in its ability to quickly integrate and adapt to the diverse experiences of multiple agents.

The results indicate that A3C's asynchronous and concurrent learning capabilities offer significant advantages in dynamic and complex multi-agent environments where rapid adaptation and coordination among agents are crucial. This is particularly relevant in scenarios such as collaborative robotics and multi-agent simulations where efficient and cooperative strategy development is key. A3C’s adaptability in multi-agent environments also illustrated its potential for environments that are not only dynamic but also unpredictable, like the airport which has multiple agents (passengers) and moving threats. Its ability to quickly accept new information and adjust the strategies accordingly makes it a robust choice for scenarios where environmental conditions or agent roles may change rapidly.

In summary, a significant difference was observed in the learning curves of the A3C and DQN algorithms when comparing them in static threat environments, moving threat environments, or multi-agent environments. Across all environments, A3C consistently demonstrated a more robust and faster learning process compared to DQN for finding optimal evacuation paths and reducing evacuation time. These results are attributed to A3C's ability to handle asynchronous updates and adapt to dynamic environments more efficiently. However, while DQN demonstrates proficiency in less complex scenarios, such as the single-agent static threat environment, it still lags behind in more complex environments. Therefore, it requires rapid adaptation and multi-agent coordination, similar to the A3C algorithm.

Section 5: Conclusion and Discussion

In this study, we have successfully bridged the gap identified in the literature review regarding the practical application of advanced computational models, specifically of the A3C algorithm, in simulating real-world evacuation scenarios. Our comparative analysis of the A3C and DQN models in simulating airport evacuations revealed that A3C outperforms DQN in several key areas: 1. Efficiency in Evacuation in terms of speed and correctness, 2. Adaptability to Dynamic Environments, and 3. Robustness in Learning and Strategy Adaptation. Notably, A3C demonstrated superior adaptability and efficiency in managing dynamic threats and high-density environments, which are characteristic of real-life aviation emergencies. A3C's proficiency in adapting to changing conditions was particularly effective in scenarios with moving threats, highlighting its practical advantage in unpredictable situations. Moreover, the A3C model facilitated faster evacuations and managed high-density crowds more effectively, reducing bottlenecks and ensuring smoother evacuation flows. It also exhibited a robust learning curve and adapted its strategy efficiently in response to environmental changes which was less efficient in the DQN model.

The findings from our comparative analysis have profound implications for transportation safety and policymaking. This study suggests that A3C is generally more versatile and efficient in various environmental settings, especially in scenarios that require quick adaptation and multi-agent learning. A3C's adaptability in dynamic environments, such as those with moving threats, and its effectiveness in multi-agent scenarios, highlight its potential for complex real-world applications where diverse challenges are present. The study emphasizes the significance of choosing an appropriate algorithm for specific environmental challenges. For instance, the evacuation mission at the airport is a complex and constantly changing scenario. A3C is a particularly strong algorithm for dynamic real-world applications that require quick decision-making ability and adaptability due to its emerging speed, accuracy, and efficiency.

By illustrating A3C's effectiveness in managing dynamic threats and facilitating smooth evacuation flows, our study advocates for the integration of advanced computational models into the emergency response strategies of airports and other critical infrastructure. For policymakers and emergency planners, adopting A3C-based simulations can lead to more informed decision-making, enabling the design of evacuation protocols that are both safer and more efficient. Furthermore, these results can guide the development of policies that prioritize investments in technology-driven safety improvements, ensuring that transportation systems are resilient against a wide range of emergency scenarios. Utilizing the rapid adaptability of machine learning algorithms like A3C, the modern transportation system could dynamically optimize the shortest and safest evacuation routes in real time when emergency situations happen. The optimization could not only be applied to airport evacuation, but also expands to natural disasters or urban crises in an area. A3C’s ability to adapt quickly to the changing environment such as traffic congestion, road closures, or hazardous conditions can allow the agents (e.g., passengers, pedestrians, vehicles etc.) to reroute efficiently for a safer and less congested path. Also, the algorithms including A3C and DQN as used in this research can be used for simulation and training in the transportation industry to save time and money when determining the best evacuation method under different emergency scenarios. These simulations can assist with training personnel in the industry, thus helping the company improve their evacuation regulations and plans and ensure a more effective response during actual emergencies. Moreover, A3C can also be implemented in intelligent traffic systems to manage and control traffic flow during emergencies. Its capability to process multiple agents and data simultaneously enables quick and real-time adjustments for traffic signals, lane assignments, etc. so that the intelligent traffic system can facilitate smoother evacuation or emergency response.

However, there are still limitations to the algorithms and research. One limitation is the computational resources required for complex and multi-agent scenarios. The computational demands of the A3C algorithm present a significant challenge, especially when scaling to extremely large or complex simulations. This may limit its applicability in resource-constrained environments or necessitate substantial investments in computational infrastructure. Additionally, our simulations may not fully capture the unpredictability of human behavior in emergency situations, potentially affecting the real-world accuracy of our findings. The reliance on specific simulation parameters also raises questions about the generalizability of the results across different types of emergencies or transportation settings.

For the future directions, combining the SF model with evacuation algorithms and models could lead to further improvements. The SF model takes into account individuals' preemptive path adjustments in response to anticipated social interactions, not just physical constraints. This integration would result in more accurate simulations of crowd behavior, optimizing evacuation strategies and enhancing safety in complex airport environments. This could include average physical dimensions for men, women, and children, as well as the moving speed of different age groups. By incorporating these features, the model will be closer to reality and better equipped to handle emergency situations. Last but not least, to make the model more accurate and efficient for emergency situations at airports, it may be beneficial to consider adding dimensions for human width and evacuee speed. Incorporating more complex and realistic elements into the simulation, such as varying human behavior patterns, complex architectural features, and unpredictable environmental conditions, would more closely mimic real-world scenarios. With further work, the model could have a broader scope that meets the requirements of complex real-time decision-making.

References

- Alexander, D. E. (2013). Emergency, Definition of. In K. B. Penuel, M. Statler & R. Hagen (Eds.), *Encyclopedia of crisis management* (324-325). Thousand Oaks: SAGE Publications, Inc.
- Bureau of Transportation Statistics. (2021) On-time performance: Causes of delay. U.S. Department of Transportation. https://www.transtats.bts.gov/ot delay/ot_delaycause1
- Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). OpenAI Gym. arXiv:1606.01540. https://doi.org/10.5281/zenodo.16014
- Chen, J., Liu, D., Namilae, S., Sang-A, L., Thropp, J. E., & Seong, Y. (2019). Effects of Exit Doors and Number of Passengers on Airport Evacuation Effeciency Using

Agent Based Simulation. *International Journal of Aviation, Aeronautics and Aerospace*, *6*(5), 3. https://doi.org/10.15394/ijaaa.2019.1418

- Cheng, L., Reddy, V., Fookes, C., & Yarlagadda, P, K, D, V. (2014). Impact of Passenger Group Dynamics on an Airport Evacuation Process Using an Agent-Based Model. *2014 International Conference on Computational Science and Computational Intelligence*, 161-167. doi: 10.1109/CSCI.2014.111.
- Christensen, Keith and Sasaki, Yuya.(2008).Agent-Based Emergency Evacuation Simulation with Individuals with Disabilities in the Population. http://jasss.soc.surrey.ac.uk/11/3/9.html
- *Did you know about the step length?* (n.d.). https://steps.app/en/blog/stepcounting/did-you-know-about-the-step-length#:~:text=Did%20you%20ever%20think%20about,2.5%20feet%20for%20a%20man
- Ding, G., Li, X., Shi, X., & Yu, P. (2021). Data Transmission Evaluation and Allocation Mechanism of the Optimal Routing Path: An Asynchronous Advantage Actor-Critic (A3C) Approach. *Wireless Communications and Mobile Computing*, *2021*, 1–21. https://doi.org/10.1155/2021/6685722
- Gao, H.; Xu, J.; Li, S.; Xu, L. Forecast of passenger flow under the interruption of urban rail transit operation. In *Lecture Notes in Electrical Engineering: Proceedings of the 4th International Conference on Electrical and Information Technologies for Rail Transportation (EITRT)*; Springer: New York, NY, USA, 2019; pp. 283–291.
- Gosavi, A. A. (2004). Reinforcement Learning Algorithm Based on Policy Iteration for Average Reward: Empirical Results with Yield Management and Convergence

Analysis. *Machine Learning 55*, 5–29. https://doi.org/10.1023/B:MACH.0000019802.64038.6c.

- Gota, Puscasiu, A., Fanca, A., Valean, H., & Miclea, L. (2020). Threat Objects Detection in Airport using Machine Learning. *2020 21th International Carpathian Control Conference (ICCC)*, 1–6. https://doi.org/10.1109/ICCC49264.2020.9257293
- Helbing, D., & Molnar, P. (1995). Social force model for pedestrian dynamics. *Physical review E, 51*(5), 4282.
- Helbing, D., Physica A 196, 546 (1993).
- Hradecky, S. (2016, October 28). Accident: American B763 at Chicago on Oct 28th 2016, rejected takeoff, fire at right hand wing due to uncontained engine failure. The Aviation Herald. Retrieved from http://avherald.com/h?article=49ffa115&opt=0
- Hradecky, S. (2017, July 3). Incident: Skywest CRJ7 at Denver on Jul 2nd 2017, engine fire on landing. The Aviation Herald. Retrieved from http://avherald.com/h?article =4ab23fdb&opt=0
- Hradecky, S. (2017, July 11). Incident: Delta A320 near Daytona Beach on Jul 10th 2017, hail strike. The Aviation Herald. Retrieved from http://avherald.com/h?article =4ab7c100&opt=0
- *How to measure steps* | Cleveland Heights Parks & Recreation, OH. (n.d.). https://www.chparks.com/411/How-To-Measure-Steps#:~:text=An%20average%20person%20has%20a,steps%20has%20many%20health%20benefits

- Kim, & Pineau, J. (2015). Socially Adaptive Path Planning in Human Environments Using Inverse Reinforcement Learning. *International Journal of Social Robotics*, *8*(1), 51–66. https://doi.org/10.1007/s12369-015-0310-2
- Lee, J., & Yoo, H. (2023). Deep Reinforcement Learning Processor Design for Mobile Applications. In *Deep Reinforcement Learning Processor Design for Mobile Applications* (pp. 1–93). https://doi.org/10.1007/978-3-031-36793-9_1
- Liu, R., Jiang, D., & Shi, L. (2016). Agent-based simulation of alternative classroom evacuation scenarios. *Frontiers of Architectural Research*, *5*(1), 111-125.
- Martinez-Gil, F., Lozano, M., & Fernández, F. (2011). Multi-agent reinforcement learning for simulating pedestrian navigation. *In International Workshop on Adaptive and Learning Agents*, 54-69
- Martin Thomas (2023). Fire Spread Model (https://www.mathworks.com/matlabcentral/fileexchange/74499-fire-spread-model), MATLAB Central File Exchange. Retrieved December 1, 2023.
- Sutton, Richard S., and Andrew G. Barto. *Reinforcement Learning: An Introduction.* 2nd ed., The MIT Press, 2018.
- Ma, J., Zeng, X., Xue, X., & Deng, R. (2022). Metro Emergency Passenger Flow Prediction on Transfer Learning and LSTM Model. *Applied Sciences*, *12*(3), 1644–. https://doi.org/10.3390/app12031644
- Mitchell, Tom (1997). Machine Learning. New York: McGraw Hill. ISBN 0-07-042807-7. OCLC 36417892

- Miyoshi, T., Nakayasu, H., Ueno, Y., & Patterson, P. (2012). An emergency aircraft evacuation simulation considering passenger emotions. *Computers & Industrial Engineering*, *62*, 746-754. http://doi.org/10.1016/j.cie.2011.11.012
- Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., ... & Kavukcuoglu, K. (2016, June). Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning* (pp. 1928-1937). PMLR.
- Papoudakis, G., Christianos, F., Schäfer, L., & Albrecht, S. V. (2020). Comparative Evaluation of Multi-Agent Deep Reinforcement Learning Algorithms.
- Rossier, R. (n.d.). *EMERGENCY LANDINGS*. AOPA. https://www.aopa.org/training-and-safety/students/flighttestprep/skills/emergency-landings
- Sandoval, E. (2017, July 10). Cracked windshield forces Haiti-bound Delta jet to land in Daytona. ClickOrlando.com. Retrieved from https://www.clickorlando.com/news/cracked-windshield-forces-haiti-bound-delta-flight-to-land-in-daytona
- Sutton, Richard S., and Andrew G. Barto. (2018). Reinforcement learning: An introduction. MIT press.
- Skanda Vaidyanath, Kallirroi Georgila and David Traum. (2020). Using Reinforcement Learning to Manage Communications Between Humans and Artificial Agents in an Evacuation Scenario. https://www.aaai.org/ocs/index.php/FLAIRS/FLAIRS20/paper/view/18469/17622.
- Tian, K., & Jiang, S. (2018). Reinforcement learning for safe evacuation time of fire in Hong Kong-Zhuhai-Macau immersed tube tunnel. *Systems Science & Control Engineering*, *6*(2), 45-56.

- U.S. Department of Transportation, Federal Aviation Administration, Advisory Circular (AC) 150/5200-31C, Airport Emergency Plan, June 19, 2009, p. 1.
- U.S. Department of Transportation, Federal Aviation Administration, Federal Aviation Rule (FAR) 139.325, Airport Emergency Plan, June 09, 2004, p.22.
- Villamizar, H. (2022, June 22). *How Airport Emergency Response Works.* Airways Magazine. https://airwaysmag.com/airport-emergency-response/
- Vorst, H.C.M. (2010). Evacuation models and disaster psychology. *Procedia Engineering*, *3*, 15–21. https://doi.org/10.1016/j.proeng.2010.07.004
- Wang, Q., Liu, H., Gao, K., & Zhang, L. (2019). Improved multi-agent reinforcement learning for path planning-based crowd simulation. *IEEE Access*, *7*, 73841-73855.
- Yao, Z., Zhang, G., Lu, D., & Liu, H. (2019). Data-driven crowd evacuation: A reinforcement learning method. *Neurocomputing (Amsterdam)*, *366*, 314–327. https://doi.org/10.1016/j.neucom.2019.08.021
- Yang, Y., Zhang, K., Liu, D., & Song, H. (2020). Autonomous UAV Navigation in Dynamic Environments with Double Deep Q-Networks, *2020 AIAA/IEEE 39th Digital Avionics Systems Conference (DASC), San Antonio, TX, USA, 2020*, pp. 1-7, https://doi.org/10.1109/DASC50938.2020.9256455.
- Zarraonandia, T., Vargas, M. R. R., Díaz, P., & Aedo, I. (2009, July). A virtual environment for learning aiport emergency management protocols. In *International Conference on Human-Computer Interaction* (pp. 228-235). Springer, Berlin, Heidelberg.

- Zhang, Chai, Z., & Lykotrafitis, G. (2021). Deep reinforcement learning with a particle dynamics environment applied to emergency evacuation of a room with obstacles. *Physica A*, *571*, 125845–. https://doi.org/10.1016/j.physa.2021.125845
- Zhang, S., & Guo, Y. (2015). Distributed multi-robot evacuation incorporating human behavior. *Asian Journal of Control*, *17*(1), 34-44.
- Zhang, G., Zhu, G., Yuan, G., & Wang, Y. (2016). Quantitative risk assessment methods of evacuation safety for collapse of large steel structure gymnasium caused by localized fire. *Safety Science*, 87, 234-242.
- Zhaoyuan Gu, Zhengzhong Jia and Howie Choset(2019, December). Adversary A3C For Robust Reinforcement Learning, arXiv:1912.00330.