Optimized Evacuations for Active Shooter Incidents to Improve Safety Outcomes

A Dissertation Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

Joseph Lavalle-Rivera

August 2025

This work is dedicated to my wife, Jen, and children, Konner, Parker, and Zoey, who had to put up with a move to another new place and deal with my periods of stress, doubt, and regret. Its been a long time with this cloud hanging over my head and I am sure you all have felt its effects in one way or another. I could not have started, struggled with, and now completed this effort without your support. I cannot express how thankful and appreciative that I am to have had your support from the beginning.

Acknowledgments

This work was made possible thanks to the help of my advisor, Subhadeep Chakraborty, and my committee members Russell Zaretzki, Anahita Khojandi, and Amir Sadovnik. I would also like to thank the members of the COSMOS lab in the Mechanical, Aeronautical, and Biomedical Engineering department, specifically Anirudh Ramesh who has been working through this problem set with me since 2020. I appreciate and am thankful for the administrative support and guidance of Allie Burns, Rebecca Christ, and Rachel Roberts of the Bredesen Center staff. Lastly, and most importantly, I would like to thank my wife, Jennifer Lavalle-Rivera, for her support and encouragement throughout this long process.

This work has been supported by National Science Foundation Cyber-Physical Systems (NSF CPS) program under Award Number 1932505 and U.S. Department of Homeland Security Soft Target Engineering to Neutralize the Threat Reality (SENTRY) program under Grant Award 22STESE00001-03-02.

Abstract

Active shooter incidents (ASIs) represent a uniquely complex and time-critical challenge for emergency evacuation planning, where traditional protocols such as "Run, Hide, Fight" often fall short due to the dynamic nature of the threat and the cognitive overload experienced by evacuees. This dissertation presents a comprehensive framework for optimizing evacuee routing during ASIs using both model-based and model-free reinforcement learning (RL) techniques. The problem is formulated as a Non-Homogeneous Semi-Markov Decision Process (NHSMDP), capturing the temporal and spatial dynamics of both evacuees and the shooter within a graph-based representation of building layouts.

Three major contributions are made: (1) the development of Naive ASTERS, a model-based RL algorithm that incorporates shooter location, node safety, and exit proximity to generate optimal routing policies; (2) the extension to C-CASTERS, which introduces capacity constraints and iteratively generates multi-route evacuation plans to reduce bottlenecks and crowding; and (3) the integration of model-free approaches—Q-Learning and N-Step Temporal Difference Learning—trained in a discrete event simulation (DES) and evaluated in a high-fidelity, physics-based Unreal Engine 5 (UE5) environment. Across thousands of simulated scenarios, N-Step Temporal Difference Learning consistently outperformed other methods in survival rate, casualty reduction, and adaptability to dynamic threats.

This work advances the state of the art in emergency evacuation modeling by demonstrating the viability of offline-trained RL policies in real-time, physics-based environments. It also lays the groundwork for future integration of virtual reality training, multi-agent coordination, and real-time decision support systems, with the ultimate goal of enhancing public safety and survivability during active shooter events.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

There have been approximately 150 mass shootings in the US from August of 1982 to present. The number of events is approximate due to the ongoing discussion within law enforcement and other interested communities as to what characterizes an event as a mass shooting.

Mother Jones, an independent news outlet, keeps an updated database of events starting in August of 1982, continuing through to the present, and was last updated on June 21, 2024 [24]. The Mother Jones database provides details on each of the events like location, time, weapon used, previous mental illness purported, and sources for the data. Examining the database indicates the total of 150 events, but also a generalized increase in the number of events since they began being tracked in 1982. The US went from a single event in 1982 to 12 separate mass shootings in each of 2022 and 2023. The cumulative count of mass shooting in Fig. 1.1 demonstrates that the increasing growth in the number of events began in 2010. While there is unanimous agreement that mass shootings are occurring too frequently, the reasons and sources behind this increase in events are not widely agreed upon [26, 79, 69]. The discussion continues to revolve around the availability of firearms, the lack of security in and around our schools, and the mental health epidemic that seems to be worsening at a worse rate than the mass shootings. Further, politicians, lobbyists, and activists continue to clash and disagree as to what policies are required to address and reduce the frequency of

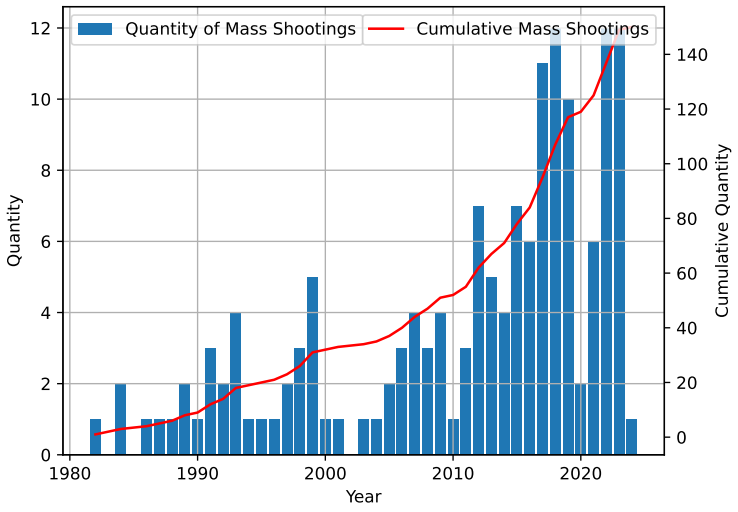


Figure 1.1: Mother Jones Mass Shooting Database

mass shootings [61, 65, 84, 76]. While current proposals include more in-depth backgrounds checks prior to firearm purchase, federal registry of private firearm ownership, complete banning of private firearm ownership, better mental health evaluations and assistance, and red flag laws, they all attempt to accomplish the noblest and hardest feat in this arena: prevent the mass shooting from occurring in the first place. While the efforts to do this are most definitely worth it, they are often not enough and lack effective execution in reality. On the other hand, a more achievable goal to pursue is the safe evacuation of the victims out of the situation all together.

Law enforcement and trainers for programs like ALICE [75] have been long been proponents of the "Run, Hide, Fight" protocol. While operating under this protocol, an evacuee would run for an exit if they thought they could make it, hide if they thought they couldn't make it or know where the exit was, or defend themselves by confronting the shooter if they had no other options. The primary problems with this protocol are the stress and confusion inherent within the mind of the evacuee and the lack of credible information available to the evacuee. Asking an evacuee to make a logical or thoughtful decision during a life or death active shooter situation is unrealistic and unproductive. Further, there has been research on the effectiveness of this protocol and how it is implemented [48, 98, 45, 22, 53, 21]. Even with increased emphasis on these tragic incidents, there have not been advancements in the protocols or instructions to evacuees. New protocols have been adopted, but they have primarily changed the wording used, "Move, Evade, Defend" or "Avoid, Deny, Defend", without actually changing the approach or goals of "Run, Hide, Fight". While the policies that may provide assistance are in the purview of the legislative bodies, providing assistance to the evacuees is an open area of research and one that is ripe to provide positive outcomes in these unnecessary situations.

The Active Shooter Tracking and Evacuation Routing for Survival (ASTERS) project was awarded an NSF grant in 2019 under the Cyber-Physical Systems division. There are four thrusts within the ASTER project: computer vision and multimodal fusion to identify and track the shooter, route optimization and planning, communication of vital situational information, and engagement within the end-user community i.e. schools and their administration. The goal of this project is to develop an end-to-end self-contained

system that identifies and tracks a shooter's movements throughout a building, provides the shooter's location to an optimized routing algorithm that identifies the safest actions for the evacuees to take, and communicates those actions to the evacuees as well as the shooter's position to law enforcement. In this work, we focus on the safety of the evacuees and ways to improve their probability of survival by researching and developing novel methods of optimizing the their routing.

We attempt to bridge the current gap of actionable instructions for evacuees by approaching the task of optimizing the routing using model-based and model-free reinforcement learning approaches. We modeled building layouts as graph networks with the decision at each node being to either move to an adjacent node or stay in the current node. We settled on modeling the dynamics of the situation as a Non-Homogeneous Semi-Markov Decision Process (NHSMDP): semi-homogeneous semi-Markov due to the transition probabilities dynamically changing based upon the shooter's and evacuees' movements and a Markov decision process because the escape of a building during an active shooter situation is not a single decision, but is a series of sequential decisions. Taking this into account, we focused on the following research questions:

1. Can a NHSMDP sufficiently define and model the actions of a shooter and the evacuees in an active shooter situation so as to provide optimized and safe instructions to the evacuees for their extraction from the building?

2. What improvements and adjustments can be made to incorporate realistic capacity constraints for large groups of evacuees and the requirement for multiple routes?

Our work in model-based and model-free routing has resulted in significant improvements in casualties and escapes over the "Run, Hide, Fight" protocols currently trained when simulated in both a discrete-event simulation (DES) and Unreal Engine 5 (UE5) and established a methodology to develop and explore routing options within these situations to improve the safety outcomes for the evacuees.

1.1 State of the Research Thrusts

1.1.1 Active Shooter Survival

As indicated previously, the quantity of mass shootings in the US is growing at an increasing rate and, with no prescribed governmental policies to prevent events on the table, there seems to be no end in sight. The predominance of the most current research on active shooter survival covers building characteristics and associated assets that increase the likelihood of escape or difficulty for shooter movement. Unfortunately, there is very limited research on active shooter evacuations and even these often concentrate on the behavioral aspects of evacuation and not the routing for safety.

Taking into account the occurrence of active shooter situations and there effects has become a budding topic in building design and layout. When discussing building design for active shooter evacuation scenarios, the approaches can vary widely, but have the common theme of expediting movement through and out of the building. The following research had a common theme: they didn't examine the routing methods and instead used social behavior mechanics with shortest path algorithms to route their evacuees. Arteaga and Park used agent-based modeling and simulation to evaluate the width of doorways, hallways, and exits and there effects on evacuee movement in the building[4]. Similar to what we discovered through our capacity constrained evacuation routing, they found that the bottlenecks caused by the width of doorways, regardless of room, exit, or inter-hallway, had the greatest impact on the speed at which a building could be evacuated. While it is common sense for this to be the case, it is encouraging that we both came to the same conclusion from two disparate approaches. Liu et al. developed a simulation of an autonomous shooter agent in an office setting and examined the effects that the quantity and locations of exits had on favorable outcomes in an active shooter event[57]. They quantitatively came to the unsurprising conclusion that the more exits a building had, the better the outcomes were for the evacuees. Further, they also found that the accessibility, which they viewed as number of difference possible routes, to the exits nearest to the shooter played a significant role in the outcome. Zhu et al. used a VR environment and over 150 participants from schools and office buildings to analyze the effects building countermeasures had on the evacuating

behavior of the VR participants[101]. They created an office building and a school in a VR environment and then created replicas of them with additional countermeasures the originals did not have like additional fencing and barriers, frosted windows, staggered doorways, etc. With the participants operating in the "Run, Hide, Fight" protocol, they found that the participants hid more often in the buildings with the countermeasures than the buildings without with the researchers coming to the conclusion that the perception of safety provided by the countermeasures led to more cautious evacuee behavior.

While the literature for active shooter evacuations is limited, there a few groups of researchers who are broaching the the topic from different perspectives than our group. Gunn et al. approached providing optimal guidance to evacuees by creating a network of nodes, using a divide-and-conquer approach to split evacuees into groups, and assign them into routes via a max flow model that moves them to either exits or safe rooms[30]. They compared the results of their guided simulations with unguided simulations and, while they didn't consider the number of casualties, they did find that their optimized guidance directed evacuees to lesser used routes and passages reducing the crowding effect than the unguided simulations. Arteaga et al. examined the effect that trained evacuation leaders had on the successful evacuation of a building and compared it to leaderless evacuations[5]. The researchers concluded through simulating various factors (occupancy, fire rates, firearm range, evacuee decisions, number and location of leaders) that having trained evacuation leaders (agents with better knowledge of the building and procedures) contributed to fewer casualties across the board. These findings also support the active shooter training that takes place in schools and offices to ensure preparedness should something terrible occur. Finally, Lu et al. studied the effects of providing the location of gun fire to participants acting as evacuees in a VR environment[64]. The researchers established that providing the location of gun fire to the participants increased their probability of survival. The examined the participants behavior of looking for the shooter and how the provided information shortened the time required to locate the shooter, but did not make a determination on how the participants used the information to select the route they chose. While not guidance specific, this study lends credence to the aspect of providing information to evacuees and how it affects their decision making.

1.1.2 Optimized Evacuation Routing

Optimized evacuation routing, regardless of the reason necessitating the evacuation, is an active and ever evolving area of research within multiple communities interested in the topic. On the whole, the evacuation literature is primarily focused on either building evacuations due to fire or city evacuations due to natural disaster. After categorizing the current research into one these two areas, the actual methods used to develop and analyze the best routes vary greatly as the possible approaches are quite broad, as are the characteristics of what identifies the best route.

Building evacuations in the event of a fire has been and continues to be a very thoroughly researched area of interest. The lens that researchers view the problem of building evacuation has as much to do with their solution approach as the characteristics and requirements of the problem. Reinforcement learning is a popular approach, however, their are different thrusts by which researchers are attacking this problem. One of these varied methods and perspectives includes an approach that attempts to mimic nature like the ant colony optimization (ACO)[99]. Using the idea of ants exploring around their nest for food, researchers mimic this behavior by modeling multi-agent random searches that introduce a "pheromone factor" that affects the behavior of the other agents by making them more or less likely to take the same path based upon the strength of the "pheromone". This approach would seemingly help many agents to identify the best path from a source node to a target node, however, it also requires the agent discovering the optimal path to return to the source so others can follow which is not very effective for evacuation purposes. Zhang et al. developed the Improved Adaptive Ant Colony Optimization algorithm that took risk and energy consumption into account as well as path length to determine optimal the optimal route and found it outperformed the Pathfinder algorithm, an industry standard. A more straight forward, classic reinforcement learning approach used a modified SARSA (state-action-reward-state-action)[37]. Huang et al. introduced a Radar-assisted SARSA that made use of an exit radar and a fire radar to provide additional information to learning agents to improve the avoidance of danger. Their method improves on the shortest distance

paths typically provided for larger buildings, and at the same time, mimics our use of cameras to provide improved situational awareness to evacuees for better decision making.

Another popular approach to building evacuation problems is a graphical network approach. Simplifying the blueprints or layouts of a building into nodes and arcs provides faster computation for many algorithms and also brings some classical operations research methods into the discussion for route finding. For instance, Lotero et al. modeled a building evacuation as a minimum cost flow problem on a graphical network[60]. They combined a Ford-Fulkerson algorithm with fire-induced risks based upon MSHA safety standards to identify optimal routes from an underground coal mine fire that included toxic vapors that also needed to be avoided. They were able to use this method to identify optimal routes avoiding hazardous elements and also inform development and locations of ventilation designs. Another network-based approach made use of the ripple spreading algorithm to identify possible paths of evacuation from a building or, in other cases, from cities[96]. Xiaobing et al. developed a capacity constrained ripple spreading algorithm (CCRSA) that updated remaining capacity on each arc at each time instant while adding ripple wait times if capacity was insufficient. Using this method allowed them to identify the shortest time path taking into account wait times along the way. They were able to improve upon standard path planning algorithms in most trials while also improving upon computation requirements.

Yet another differing novelty to building evacuation is the application of human factors, both physical and behavioral. Pisirir et al. used the Social Forces Model to model the behavior of evacuees as a group and within crowds[78]. They also included anthropometric data (expressions of the physical characteristics and capabilities of the human body) that informed the movements of the members of groups and its affect on the groups ability to evacuate effectively. They found that adding the fidelity of real human measurements and capabilities greatly improved the results of simulation when they were substituted for more generalized movement speeds or agility and encouraged future researchers to refrain from using generalized movement information. Other approaches attempt to observe and more accurately model human behavior and choices through the possibilities of multiple route evacuations[83]. In essence, Sheu examined and analyzed the dichotomy between individual risk perception, survival psychology, and group behavior through the lens of a

family evacuating a home. He then extrapolated that to evacuations of larger buildings with on the fly clustered groups and applied similar behavioral trends to them. Sheu was able to identify several insights into the dynamics of group behavior and showed that agents interested the welfare of a group can improve the safety outcomes of an entire group. We hope to incorporate group dynamics of speed and ability within the final thrust of our research.

Beyond the evacuation of a building, whether for fire or not, routing is required for serious situations like natural disaster (flooding, earthquakes, wildfires, etc.) and the methods for determining the best routes are as varied as those with building evacuations. Khalili et al. conducted a thorough literature review of disaster mass evacuations and traffic considerations[42]. Interestingly, given the breadth of disciplines involved in evacuation research, they found that the largest gaps were data availability and accuracy, model validations, stakeholder engagement, and integration of uncertainty and dynamic factors. The final gap, integration of uncertainty and dynamic factors, is something we concentrated upon as active shooter situations are loaded with uncertainty and are very dynamic with movement from the shooter and the evacuees. That being said, they did acknowledge the various approaches to studying mas evacuations with three of the large concentrations being graphical network based algorithms, algorithms that provide increased information to agents, and reinforcement algorithms that combine selfish routing with hazard planning, safe routes, and timeing decisions.

Graphical networks are a popular approach to evacuations, building or otherwise, due to the simplification and discretization of state spaces. There are many ways to simply spaces, but when dealing with roads, graphical networks are a convenient means of capturing the potential routes be which evacuees can move. Maulana and Sari examined the routing of vehicles in the event of rivers overflowing[68]. They applied Sample Additive Weighting (SAW) and Network Analysis Fastest Route in conjunction with Geographic Information Systems (GIS) to determine the optimal routes when rivers over flow into flood plains. Their methodology was able to determine the best evacuation center and the safest route to reach it among upwards of 50 possible routes. Mizumura et al. analyzed the benefit of the shortest route when evacuation a flood plain within the vicinity of flooded rivers[72]. While it may seem logical that the shortest route out of a danger area is optimal, it often is not

due to any number of factors including geography, capacity on the route, or even the lack of information about the route and the final destination. Through multi-agent simulations, they were able to determine the greatest good for a local population occurred when the vast majority of evacuees took the shortest route increasing the evacuation safety and also reducing the necessary evacuation time. They were not concerned with crowding at critical nodes, but crowding is something we are very cognisant of and have taken great care to address.

Similar to one of our thrusts of research, providing additional information to evacuees to improve their situational awareness is a significant subset of current evacuation research. Naito et al. improved upon previous studies that examined the effects of providing additional information (closed roads) on agents and their ability to make optimal choices by also providing congestion amounts on road segments[74]. They found that agents provided with the additional information about their environment chose more optimal routes and decreased their evacuation time. Further, decreased evacuation time also prevented selfish agents from causing longer evacuations due to non-optimal decisions. Along the same line of additional information, Dong et al. applied a hybrid of ACO genetic algorithm (GA) to the avoidance of flooded areas during an evacuation[19]. They applied the ACO with the "pheromone" factor to the backtracking and combined with the GA and improved upon the performance of the simple ACO to determine the optimal route. Further, they were able to use this hybrid algorithm to avoid flood influence and provide a guarantee for the emergency transfer of evacuees. As indicated earlier, one of our primary goals is to provide more and better information to both the evacuees and the law enforcement on the scene; it is promising to see the effects that information can have on the results.

Just as with the building evacuations, reinforcement learning provides a very logical approach to modeling and simulating disaster evacuations. Li et al. introduced the Reinforce Routing (RR) model that incorporates dynamic nature of flood conditions, road closures, and real time changes inherent in urban disaster logistics[50]. They incorporate these additional factors into a safety score which is then part of the reward structure for the RL formulation. Their algorithm outperformed most shortest-path based algorithms and provided and promising avenue to enhancing evacuation routing within cities. Similar to

their algorithm, we incorporated other aspects of evacuation into our reward functions that provided better results than the typical reward structure of most RL formulations. Islam et al. examined the problem from another point of view: cooperation instead of selfish routing[38]. They viewed the selfish routing of individuals as an extremely inefficient means of routing and proposed the Evacuation Planning Game that allowed agents to choose their route and when they wanted to depart. This is especially of interest to our group as evacuating from an active shooter doesn't necessarily mean you want to run immediately. Our problem requires a lot of considerations that other evacuation problems don't need to examine, but can still lend ideas of ways to handle it.

1.1.3 Capacity Constrained Routing

Capacity constrained routing should be a rich area of research based upon growing populations and limited space on the roads, however, there have been few real impactful advances within the last few years. Interestingly, due to the explosion of wireless devices (phones, TVs, appliances, smart home items etc.) internet protocols and routing of wireless internet traffic have become the driving force behind capacity constrained routing as researchers attempt to optimize the Quality of Service (QoS) by closely managing end-users traffic routing. Physical vehicle routing is still an area of research, but much of it is focused on the vehicle capacity routing problem which is similar to the classic knapsack problem.

As I said, managing wireless traffic to prevent packet loss, optimize bandwidth, and maximize QoS metrics seems to be the research area most concerned with capacity constrained routing. While not directly applicable to our work, the goals are similar in that they try to prevent crowding and congestion between the end users and their information destination. Dalal et al. developed an ad hoc mobility model to address network congestion and load balancing in a cloud computing environment and improve secure IoT communications between devices[17]. Their proposed method of management discovered multiple alternate paths between nodes, created a safety or stability rank for each path based upon the power of both end nodes, and then routed traffic intelligently keeping quantity of packets under a threshold. They were able to improve network lifespan through load balancing and reduced congestion while also reducing packet loss and packet delays. There

are some direct translations such as the ranking of paths based upon safety or stability; ranking possible paths to exits based upon a risk or safety factor could easily be applied to evacuation routing. Chen et al. attempted address mobility and resource limitation of the nodes that affect performance in an mobile ad hoc network[12]. In contrast to our work where the capacity constraints of the edges are the limiting factors, the researchers' approach has the capacity constraints on the nodes with the interesting that nodes can actually move around in the network changing its adjacent nodes in real time. Their approach to support QoS was to create a topological change adaptive ad hoc on-demand multipath distance vector (TA-AOMDV) protocol that adapts to the high speed movement of nodes within the network. Similarly to Dalal, they identified stable paths taking into account the nodes constrained resources (bandwidth, queue length, residual energy etc.) and the stability of the links of the path. They incorporated periodic probabilistic estimates to forecast link stability, much like probabilities of a shooter's location, in order to prematurely adapt to the rapid changes in the network topology. They were able to significantly improve the QoS over current protocols when the node speed increase higher than 30 m/s. Krolikowski et al. examined joint routing and scheduling in large deterministic networks using cycle specified queueing and forwarding[46]. They used segmented routing, not finding complete paths but sections that could be modularly combined to create paths between nodes, and transmission queues to maximize traffic acceptance for network planning and flow admission. The segmented/modular routing approach in conjunction with dynamic programming allowed them to improve upon benchmark performance while reducing the time required to calculate optimal pathing.

As indicated previously, capacity constrained routing for physical travel is as prevalent as one would think, however, there are some examples that can be helpful and informative for our work. Zhong et al. attempted to develop a risk-averse optimization of disaster relief facility locations and vehicle routing under a stochastic demand[100]. The researcher introduced an interesting concept of conditional value at risk with regret (CVaR-R) which is in essence the regret of the worst case scenarios. They used the CVaR-R as a risk measure to consider both reliability and unreliability of the stochastic demand variability. They framed the problem as a bi-objective mixed integer nonlinear programming model and solved it with

a hybrid genetic algorithm. They also incorporated the Nash bargaining method to capture the decision maker's preference of the two objectives. Their risk averse approach identified the trade-off inherent in the bi-objective framing of the problem and demonstrated the effect the decision maker can have based upon their preferences. Casteigts et al. examined the development of shortest paths in temporal networks which are networks where the edges and the nodes they connect change over time[10]. Interestingly, the idea of edges "opening and closing" through time is actually similar to our work, except we look at the shooter's reach as the factor closing the edges while they are looking at it as time that does this. The researchers added constraints on dwell time in nodes as well as maximum capacities on edges moving the problem of path shortest path calculation from polynomial time to NP Hard. While their work isn't overly applicable to ours, it is still an interesting heuristic approach to a capacitated routing problem. Finally, Froger et al. looked at EV routing with capacity constraints on the charging stations (the nodes of the network)[27]. At this time, many of the EV routing problems look to optimize the EV's route while assuming that charging stations along the route are free and available. The researchers simulated capacity constraints at the charging stations as well as other methods of charging along the route. They used a continuous-time model and an algorithmic framework that iterates between a route planner using a local search algorithm and a solution assembler using a branch-and-cut algorithm to assemble segments of a path into a workable solution that meets the charging constraints. They established a means of route development that accommodated capacity constraints along a path due to charging requirements and were able to still improve on the benchmark results of the EV routing problem.

In this we address the cross section of these areas of research and propose approaches that are not currently found in literature. We push forward to apply multiple reinforcement learning techniques, model-based and model-free, to the simple and capacitated routing optimization for safety while also addressing gaps in the research literature within the realm of active shooter situation survival. Our work expands beyond current trained survival practices and demonstrates alternate methods that ultimately improve the safety outcomes of evacuees

1.2 Research Objectives

The purpose of this work is to expand upon the current approaches for active shooter evacuations and demonstrate that our approach of modeling the evacuations as a NHSMDP, the buildings as graphical networks, and using classical reinforcement learning methods with modifications to optimize the routing can provide better safety outcomes for the evacuees than other current policies and procedures. Our results show that our methods improve upon the current procedures used in evacuation routing and, due to the limited available active shooter specific approaches and the stakes involved for evacuees in these situations, any and all improvements can have an out-sized impact on the safety and survival of the victims and move the state of this vital research area forward. Furthermore, our use of multiple methods to optimize routing provides options for implementation that range from more conservative and risk averse preferring to evacuate only when it is absolutely safe to more aggressive exit seeking routing that may allow increased time in line of sight in order to exit the building quickly. Being able to tailor the modeling and output to the needs of the end user is vitally important to ensure buy in to the viability of the results as well as to meet the needs of that end user.

Our novel contributions are as follows:

1. We demonstrate the effectiveness of modeling active shooter evacuations as NHSMDPs, modeling the buildings as graphical layouts to complement the NHSMDPs, and optimizing the routing using a model-based value iteration approach to improve the safety outcomes for the evacuees. The Naive ASTERS approach provides the value of being in a particular node at a particular time instead of concentrating on what action should be taken allowing us to model and evaluate the ability of evacuees to find safe locations and hide which is something that is lacking in the vast majority of other approaches to this problem area. Naive ASTERS is able to outperform the Natural Response algorithm which is developed using law enforcement guided advice.

2. Because the layout and dimensions within a building greatly affect the ability to quickly and safely evacuate a building, we examined the expansion of our route optimization model to include methods to account for limited capacity in nodes and on edges,

provide multiple iteratively optimal routes, and reduce crowding at bottleneck nodes. C-CASTERS took the occupation of every node, the shooter's location, and the running time of the event as inputs and created sets of optimal routes from each node accounting for and reserving the evacuees required capacity along their evacuation route and improving upon the performance of the Naive ASTERS as well as the expert advice grounded Natural Response.

3. In order to more realistically model these evacuations, we moved from the discrete event simulation we wrote to the physics-based UE5 simulation to include better fidelity of crowding mechanics and introduce a generalized concept of computer vision identifying and tracking the shooter throughout the building. Further, while the model-based approaches of Naive ASTERS and C-CASTERS perform very well and improve the safety outcomes of the evacuees, they are relatively computationally expensive and take time to calculate the optimal routes when taking into account real-time information. Model-free approaches allow us to train evacuee agents in advance and establish optimal policies to act upon given real-time updates. W evaluate both Q Learning and N-Step Temporal Difference learning approaches and demonstrate that they provide excellent performance when compared to the Naive ASTERS and the C-CASTERS and do so in real-time due to the agent training occurring in advance.

1.2.1 Model-Based Optimized Evacuation Routing

The approach we elected to use to determine the optimal routing for evacuees in an active shooter situation is one of the earliest examples of reinforcement learning: Richard Bellman's value iteration[9]. While it was developed in 1957, it is a viable method for optimizing policies for a NHSMDP and has the benefit of quantifying the value of being within a given state instead of simply what action should be taken in each state.

The majority of the current research on evacuation methods seek to minimize the evacuation time or distance traveled. We were unable to find any routing methods in recent research that had any other objective than escape. A key element in the modeling of an active shooter situation is that, while escaping the building is typically the safest option,

temporarily hiding or avoiding the shooter until a safer escape route opens up can provide a higher probability of survival to the evacuee. These other methods will provide the quickest route or the shortest route to an exit, but usually don't take into account the shooter's position or his ability to move within the building which can and should drastically change the behavior of the routing algorithm.

For our first set of experiments, we examined the performance of our Naive ASTERS value iteration method against the performance of the Natural response algorithm based upon the expert guidance of law enforcement in our discrete event simulation developed in python using the Simpy package. We developed our ASTERS value iteration modification to incorporate not only time and distance to the nearest exit, but also the relative safety of each node or its hardness, the last known location of the shooter, the probability of encountering the shooter, and the probability of being within harming distance of the shooter. The probability of encountering the shooter and the probability of being within harming distance of the shooter change continuously as both the shooter and evacuees move around in the building which is the reason for the non-stationarity and the need to model this problem with a NHSMDP. Our state space for the ASTERS routing algorithm and the simulation is $S^e = \{N^e, N^s, t\}$, where N^e are the nodes that the evacuees can occupy, N^s are the nodes that the shooter can occupy, and $t \in T$ is quantized time elapsed since the shooter was last spotted by a camera i.e. the last known position of the shooter and how long ago that occurred. The actions were dependent on the node location and its adjacent nodes; most importantly, unlike other approaches to evacuation situations, each node also had the optional action of remaining in place. We conducted simulations using two buildings layouts with multiple parameter combinations that changed the spawn locations of the shooter, his initial target node, the initial distribution of the evacuees, the speed the evacuees traveled at, and the number of cameras and their locations.

The ability of the ASTERS algorithm to incorporate a safety factor for each node and a "stay put" action provided a great boost to the performance within our simulations over the Natural Response algorithm as the algorithm could instruct evacuees to stay in the current node as they waited for the shooter to move far enough away to provide a safe exit route. Our results (discuss in depth in Chapter 2) demonstrate significant improvement by decreasing

the number of casualties and the time in the shooter's line of sight while also increasing the number of evacuees who exited safely.

1.2.2 Model-Based and Capacity Constrained Optimized Multi-Route Evacuation

A gap in our first set of experiments was the small number of evacuees (one per node) inside the simulation that needed routes to escape. We demonstrated that we can provide life saving instructions when each node only has a single evacuee, however, our initial approach would have provided the same route to each evacuee in the node so adding additional evacuees would have only had a multiplicative affect upon the metrics we captured. We know that, in any kind of evacuation scenario, the crowding and times to exit increase as you add more and more people. We acknowledge that we must modify our Naive ASTERS value iteration approach to be able to account for many evacuees and, most importantly, multiple routes.

Interestingly, the literature on capacity constrained routing, whether for evacuations or travel in general, had a common theme: the majority used a single objective function focused on minimizing the time required for all travelers to reach their destination. As we found in our first experiments, an active shooter situation requires more nuance to the routing and cannot simply be the shortest route to the exit. Our objective remained the same, determine the safest route from each node, however, after that route was at full capacity, we needed to determine what the next most optimal route was taking into account remaining capacity available. We accomplished this by adjusting the transition probabilities for each edge that was at full capacity.

Our approach to deal with limited capacity along evacuation routes was to iteratively identify the optimal routes from each node, then find the second most optimal route, and continue doing so until all evacuees had routes for their survival. To accomplish this, the Capacity-Constrained ASTERS (C-CASTERS) algorithm took the shooter's position and the evacuee occupancy of each node as its input. It then conducts value iteration to determine the values of each state in the state space and determine the optimal route from each node while also keeping track of the max number of evacuees that can use that route based upon

current available capacity (if a route has a hallway that 20 evacuees can traverse and a doorway that only allows two people to go through it at a time, then the max number of evacuees on that optimal route would be 2). The algorithm then assigns that route to the max number of evacuees and reserves their required capacity along the route and through time. The transition probabilities for each transition in time that has 0 available capacity is then reduced to 0% and the algorithm recalculates the value iteration resulting in a new optimal route from the same node. The algorithm does this until all evacuees have an assigned route.

In our second set of experiments, we implemented C-CASTERS, Naive ASTERS, and the Natural Response algorithms on the single level school and a new dorm floor with many network cycles resulting in a lot of route options. We increased the number of evacuees up 510 for the school and 256 for the dorm floor and simulated multiple shooter spawn locations, target rooms, evacuee distributions, and location update times. Our results (discussed in more detail in Chapter 3) showed two things: C-CASTERS outperformed both the Naive ASTERS and the Natural response in casualties, escapes, and time in line of sight and, more importantly, greatly reduced the crowding of evacuees in the key bottlenecking nodes within both graphs.

1.2.3 Model-Based and Model-free Routing in Physics-Based VR Environment

Our final set of experiments will look to improve upon the realism of our previous work by transition from the discrete event simulation into Unreal Engine 5 which will enable physics-based movements, generalized crowding behavior, and a simple implementation of the computer vision aspect of the project. For the simulation environment, we created a replica of the dorm floor in UE5 that is to scale and is impressively polished. We have established the same node and edge locations within the UE5 background so our routing will still be the same in essence giving a list of nodes as actions, however, UE5 will actually control how the evacuees move from one node to the next using its physics-based spatial characteristics. UE5 will control movement speed of the evacuees as they move individually

and in groups to increase the realism. Finally, we have implemented cameras within the UE5 environment that will naively identify the shooter (the shooter will be green and the evacuees blue) and track his position throughout the building providing his location to the routing algorithms to update their guidance instructions. The real computer vision model being worked on at Iowa State University is able to identify the gun in the shooters hands and track both the shooter and the gun, so our implementation is a simplified version to capture the effects and possibilities presented by having the cameras included. Finally, we will run the three current algorithms, C-CASTERS, Naive ASTERS, and Natural Response, but also we will explore and contrast the performance of mode-free approaches like Q-Learning and N-Step Temporal Difference Learning when dealing with a physics-based capacity constrained simulation environment. We plan to demonstrate the viability of model free approaches in order to cut down on the calculation required to develop routes, simplify the means of communicating routes, and consider its affect on the crowding possibilities.

1.3 Dissertation Outline

This dissertation is a manuscript-style dissertation in which each of the core chapters was either previously published in a peer-reviewed journal or is currently in the submission process.

Chapter 2 is a version of our paper "The Effectiveness of Naive Optimization of the Egress Path for an Active-Shooter Scenario" which was published in the Journal Helyion in in 2022. This work introduced our premise that modeling the evacuation of agents in an active shooter situation can adequately be accomplished by formulating it as a Non-Homogeneous Semi-Markov Decision Process, modeling the buildings as graphical networks, and optimizing the routing using model-based reinforcement learning methods. We simulated an temporal evacuation in a single level acyclic school and a multi-level cyclic hospital in a discrete event simulation we built in python. Our ASTERS routing algorithm outperformed a routing algorithm based upon external expert advice (Natural Response) provided by a law enforcement associated individual who had a plethora of experience responding to and investigating active shooter situations.

Chapter 3 is a version of our paper "An Optimized Evacuation Plan for an Active-Shooter Situation Constrained by Network Capacity" which is currently in the process of submission to an academic journal. This work introduced our method of dealing with capacity constrained network nodes and edges. We expanded upon the value iteration approach of the previous paper by providing individualized routes to each evacuee, reserving capacity along their route, recalculating the optimal routes using value iteration once and edge along a route was at max capacity, and continuing to iteratively do this until all evacuees were provided an individual route. We again simulated an evacuation in our discrete event simulation using the same single level acyclic school and a new university cyclic dorm floor that increased the complexity due to smaller hallways and doorways restricting movement, and passageways connecting hallways which allowed the shooter more freedom of movement and increased unpredictability. Our new iterative algorithm, C-CASTERS, outperformed both our previous Naive ASTERS and the Natural Response algorithms by reducing the number casualties, the time in the shooter's line of sight, and even improving upon the crowding effects at key bottle neck nodes.

Chapter 4 is a version of our paper "To Be Named Extension into 3-Dimensional, Physics-Based Simulation Space Using Model-Based and Mode-Free Reinforcement Learning Techniques" which is currently in the process of submission to an academic journal. In this work we created the cyclic dorm floor network in Unreal Engine 5 (UE5) to run the simulation with physics-based movements and crowding to improve and validate the result found in the previous work. Further, we incorporated a conceptual aspect of the computer vision thrust of the ASTERS program by including cameras that could identify and track the shooter throughout the simulation and provide that real time information to the routing algorithms. We are awaiting simulation results and will present those results at a later date.

The final chapter concludes this work by reiterating the main contributions and associated results for each of the primary chapters within the context of the current state of the research. We present our plans for future simulations and experiments to continue to refine our approach and methods. Lastly, we identify future work that will expand upon the framework we have laid out that we believe would be continue to fill in the gaps surrounding the evacuation of individuals during an active shooter situation.

Chapter 2

Model-Based Reinforcement Learning Optimized Evacuation Routing

A version of this chapter was originally published in the journal Helyion:

Lavalle-Rivera, Joseph, Aniirudh Ramesh, Laura M. Harris, and Subhadeep Chakraborty. "The effectiveness of naive optimization of the egress path for an active-shooter scenario." Heliyon 9, no. 2 (2023).

Joseph Lavalle-Rivera, Aniirudh Ramesh, and Subhadeep Chakraborty conceptualized the study and determined the simulation development and setup. Joseph Lavalle-Rivera ran all simulations and was primary author of the manuscript. Joseph Lavalle-Rivera, Aniirudh Ramesh, and Subhadeep Chakraborty determined the appropriate methods post process data analysis and evaluation. Aniirudh Ramesh and Subhadeep Chakraborty proofread and edited the final manuscript.

No revisions to this chapter have been made since the original publication.

2.1 Abstract

There have been 130 mass shootings in the United States from 1982 to June, 2022 according to the Mother Jones database of active shooter events. In these critical scenarios, making the right decisions while evacuating can be the difference between life and death. However, emergency evacuation is intensely stressful, which along with lack of verifiable real-time information may lead to costly incorrect decisions. In this paper, we demonstrate the effectiveness of a non-homogeneous semi-Markov-Decision-Process (NHSMDP) based naive algorithm that relies on prior knowledge about the layout of a building and uses recurring updates of the shooter's location (based on automatic processing of images from a camera network) to provide an optimized egress plan for evacuees. While emergency evacuations due to fire and natural disasters are well researched, the novelty of this work is in the response to a threat that moves either purposefully or randomly through the building and in incorporating the ability for an evacuee to wait for danger to pass before beginning egress and during the process of evacuation. This ability to include sojourn times in the optimized scheme is due to the NHSMDP formulation and is a notable augmentation to the current state-of-the-art. We show that following this algorithm can reduce casualties by 56% and the time spent by

evacuees in the shooter's line of sight by 52% compared to an intuitive natural response guided by expert advice.

2.2 Introduction

Emergency evacuations are a topic of great interest to researchers from many different backgrounds. Whether it is traffic evacuation in the event of disasters like earthquakes and hurricanes, analysis of human behavior during evacuation proceedings, or the movement of crowds within the context of an evacuation, the approaches to manage and optimize these evacuations are widely varied and thoroughly researched. The impetus for the quantity of the research conducted is directly related to the importance of ensuring safe and effective egress for all evacuees in each of these situations as they are usually characterized by great danger. Improvements in speed, effectiveness, and efficiency in all evacuation situations result in fewer casualties and the pursuit of situational improvements is sufficient rationale for such optimization. The interested reader can find a comprehensive literature review and mapping of the research conducted in the area of emergency evacuations in the survey paper by Liu et al. [55]. They identified three main knowledge groups within the literature: traffic evacuation, group behavior, and crowd evacuations. The methods, approaches, and rationale within these knowledge groups varies greatly as is expected in such an open problem. For example, Lin et al. focused on human behavior within crowds during building evacuations [52]. Others have focused on specific groups, like Duleb et al., who studied vulnerable populations that require additional preparation in order to effectively evacuate in a given situation [20]. Some researchers like Abdulhalim et al. have identified very specific and unique populations,like deaf and hard of hearing children, that require special consideration when determining evacuation proceedings [2]. Mirahadi and McCabe developed a fire evacuation model based upon Dijkstra's shortest path algorithm [71], but they, like most of the available research on evacuation planning and execution, still expect to route all personnel to safety as quickly as possible, but do not consider the option of "waiting out" the danger. It is quite reasonable to assume that in most usual hazards, such as fire, earthquake, etc. fastest evacuation is the safest evacuation.

However, the focus of this paper is on evacuation during an active shooting incident, an unfortunately relevant and recurring emergency that necessitates us to diverge from traditional solutions, since unlike natural disasters, the threat from shooters is localized around the shooter, very mobile and transitory. Therefore, hiding before and during evacuation (by temporarily darting in to safer rooms) are options worthy of exploring. While the literature on egress is vast and a good source of inspiration, the unique nature of active shooting events poses unique algorithmic challenges that are discussed in this paper.

While emergency evacuations are extremely stressful events, they are even more so during an active shooter incident and the confusion and panic along with lack of verifiable information can lead to sub-optimal, short-sighted decisions. The Mother Jones Mass Shootings Database classifies mass shootings as events where a shooter with firearms opens fire on a large number of people, killing at least four victims. According to a study, public mass shootings have occurred 128 times between 1982 to 2022 in the United States [23]. Multiple studies suggest that the United States has had the highest number of mass shootings in the world [85, 70]. More than half of these shootings occur either in schools or workplaces [24]. In these critical scenarios, evacuees have to make key decisions such as whether to run or hide, or when and where to hide, or which direction to run, or whether to try and stop the shooter, etc. Getting these critical decisions right can mean the difference between life and death. The Department of Homeland Security advises evacuees in these scenarios to exercise the "Run-Hide-Fight" or "Avoid-Deny-Defend" protocols as a general strategy. The protocol advises evacuees to run and escape if possible as the first preference. If an exit is not accessible, the evacuees are asked to hide, stay silent, and barricade the rooms they are in. If neither "Run" nor "Hide" is an option, the evacuees are advised to aggressively fight the shooter as the absolute last resort. However, without specific information about the shooter's location, and without guidance while escaping the building, this generalized "Run-Hide-Fight" protocol may not be effective, or worse, it can be misleading. Without guidance in the escape plan, the evacuees may run toward the shooter rather than away, or come out of their barricaded room at the wrong time. In fact, this happened during the Parkland School Shooting in 2018, where a flood of people ran straight towards the direction of the shooter [14]. Further, these situations are, in most cases, so short in duration that

when the first responders arrive, often within three to five minutes, the vast majority of the damage is already done, making the initial actions taken by people involved that much more critical.

A guidance system that tracks the dynamic location of the shooter in the building and communicates the safest egress plan to the evacuees based on this information has the potential to save lives during these unfortunate events. In this paper we exclusively focus on the routing strategies of the evacuees to maximize safety under a variety of conditions and situations, with the assumption that in any implementation, this algorithm will be paired with a camera-network based automatic shooter-identification and tracking system. We make no specific claims or impose constraints on the shooter-identification system, rather treat the camera locations and identification rates as independent parameters which affect the effectiveness and viability of our planning routine.

The interactions between the shooter and evacuees can be modeled in various ways - it may be conceived of as a non-cooperative game between competing players [95], as a pursuit-evasion problem [93] or as a graph search problem such as in the game of cops and robbers [77]. In this paper, we choose to decouple the shooter's decisions from the evacuees' activities, in the sense that the shooter pursues his own hidden agenda that we do not control nor have any knowledge of. This assumption is partly driven by the need to frame the egress route suggestion problem in an optimization framework, but also from practical considerations, since such motives are most often impossible to know or model apriori.

In this framework, we assume that the shooter's movement dynamics is described by a semi-Markov chain (SMC) with a discrete state space [28]. An SMC is a generalization of a traditional Markov chain with abstract sojourn time distributions. The evacuees' movements on the other hand, are modeled as a non-homogeneous semi-Markov system (NHSMS). This grants the model the capacity to incorporate both arbitrary waiting times at each state, but also an evolving transition probability which is based on the shooter's movements. For both shooter and evacuees, the discrete states of the underlying MDP are mapped into rooms, sections of hallways, stairways, and exits while the routes from one node to another are represented by edges. Since most analytical results in SMCs such as probabilities of occupancy, first passage times and duration of dwelling in a particular state are all obtained as

distribution over an arbitrary interval of time, we take advantage of the short duration of the event (at most 5 minutes) to create an expanded state space by incorporating discretized time into the state definition. A finite horizon value iteration with non-homogeneous transition probabilities is then used to compute the optimal egress policy for all possible locations of the shooter and evacuees in the graph.

to evaluate our capacitated algorithmFor the remainder of this paper, we first give a summary of current related literature and other works in section II, describe the specifics of our NHSMS formulation in section III, define the simulation environment used to evaluate the performance of both our ASTERS routing algorithm and a "Naturalistic Response" algorithm in section IV, discuss and compare the results from simulation in section V, and, finally, identify future adaptions to the routing algorithm in section VI .

2.3 Literature Survey

Emergency evacuations can be caused by a multitude of reasons including, but not limited to: earthquakes, hurricanes, floods, fires, active-shooter scenarios, etc. Given that time is critical in these evacuations, it is imperative that the evacuees choose an optimal route of escape as quickly as possible. Optimization of the routes taken by the evacuees, contingent upon their safety and a few constraints, can potentially save lives by maximizing the number of evacuees escaping safety.

Optimization of egress paths is a complex task as it not only is constrained by the physical geometry of the building, but also the capacity of different paths, dynamic-moving threats, etc. It should also account for human behaviors and factors such as herding mentality of people, panic, opinion propagation, misinformation, leader-follower dynamics, etc.

Hossam Abdelgawad and Baher Abdulhai completed a thorough review of the emergency evacuation literature in regards to transportation and, in particular, framing the problem as a network design problem [1]. They highlighted the benefits and shortfalls of using simulation as a method for testing and optimizing proposed solutions to emergency evacuations while also compiling proposed policies designed to increase the efficiency of said evacuations.

Thomas Kisko and Richard Francis together published their work on EVACNET+ [44] in 1985 in which they described an algorithm to determine optimal building evacuation plans. This was paired with a user-friendly interface that lets users model the building by representing the halls, rooms, etc. as nodes and passageways and corridors between them as edges. Then, the program outputs the optimal escape plan for that building floor plan to the user. This is done by using a capacitated network flow trans-shipment algorithm, which is a graph-theoretic approach to solving the network flow optimization problem. This algorithm is most suited for emergency evacuations caused due to fires.

Qingsong Lu and Shashi Shekar worked developing a capacity-constraint based evacuation algorithm [62]. They modeled capacity as a time series and used a capacity-constrained heuristic algorithm to determine the evacuation plan. They proposed an algorithm called Multiple-Route Capacity Constrained Planner (MRCCP), which produces near-optimal results but is significantly computationally less complex.

Farid Mirahadi and Brenda Y. McCabe focused on the evacuation of buildings in the event of a fire and proposed a real-time safest path based upon Dijkstra's shortest path algorithm [71]. They developed a risk factor for building compartments based upon proximity to the fire and potential blockages and produced routes that conformed to the Active Dynamic Signage System. The building is monitored in real time and the planned routes are adjusted as the event develops and their algorithm was shown to enhance the safety of evacuees.

Luh et al. in their work used a network-flow based novel stochastic dynamic programming to maximize the flow of people through the exits, given many other constraints[66]. They present a simulation of their algorithm which demonstrates rapid evacuation that avoids bottlenecking. AR Srinivasan discusses how decisions and opinions of evacuees can influence the entire group's behaviour during these emergency evacuation scenarios[88].

Yin et al. approached the evacuation of dense, urban areas using agent-based modeling. They used phone location data to develop baseline evacuation plans based upon typical population densities [97]. They then used their offline database to accelerate the real-time adjustment of plans drastically decreasing the computation time required for real time development and slightly improving the performance.

Haghpanah et al. explored the evacuations of hospital, taking into account the different mobility disabilities of patients and requirements for distancing due to infectious diseases [31]. They introduced a non-ICU patient classification framework that identified the various mobility factors characterized by the patients conditions. They showed that guidance from a larger nursing team can greatly improve evacuation time while also demonstrating that a COVID-19 only exit much less effective.

Chou et al. approached building fire disasters from the aspect of routing the firefighters to appropriate locations to best assist evacuees [13]. The authors combined current firefighting equipment and procedures as well as Bluetooth technology, mobile network locations, communication systems, and signage to best direct the movement of evacuees and firefighters alike in a given indoor building fire. The proposed approach improved upon the number of casualties and the resources required to address the situation. This is potentially helpful in future work in active shooting evacuation research that will include routing of first responders to the threats location when they arrive upon the scene.

Balboa et al. used information provided by hazard detection systems, such smoke detectors identifying a fire's location, to calculate optimal routes that were the "fastest" (shortest according to Dijkstra's) and "safest" (maximized distance between the evacuation route and the hazard) within real-time simulations to guide evacuees [8]. They tested their approach in a multi-enclosure building with actual participants using a dynamic signage approach for guidance and demonstrated the effectiveness of real-time guidance for the evacuation of a building. They identified improvements in evacuee movement speed and required evacuation time lending credence to their hypothesis of the increase in performance based on intelligent dynamic signage. The use of dynamic signage and its effectiveness is promising as any real-time routing for an active shooter situation would need signage that does more than simply point in a direction.

Abusalama et al. examined a capacity constrained approach to routing and evacuation planning [3]. The authors developed a Dynamic Real-Time Capacity Constrained Routing algorithm that modeled capacities based upon time series to address the Emergency Route Planning problem. They demonstrated an improvement on both computation required and

time for evacuation. As indicated in most of this literature, this work does not account for the mobile threat to the evacuees, nor the ability for an evacuee to hide from said threat.

Despite the literature for emergency evacuation planning in general being vast and well documented, works on egress solutions for active-shooter scenarios are very scant. This may be partially attributed to the complexity of the problem. Such events last very few minutes and are highly stochastic. Moreover, due to the dynamic nature of the shooter, the evacuation plan has to be dynamic as well. Gunn et al modelled an optimal egress plan for active shooter scenarios [30]. They developed a novel divide-and-conquer approach to divide evacuees into groups and used dynamic stochastic programming for finding the optimal paths for these groups. However, their work did not take into account several important factors like dynamic shooter movement, the shooter's line of sight, human factors, capacity constraints, etc. To our knowledge, no work has been done so far that solves an egress plan for active-shooter scenario that dynamically updates as the location of the shooter changes.

2.4 Methodology

We model the shooter's movement by a semi-Markov chain $\{X_t\}_{t\geq 1}$ with state space $S^s = N_1^s, N_2^s, ..., N_n^s$ corresponding to nodes in the building layout. Here n is the total number of nodes and the superscript s signifies that this state space corresponds to the shooter's movement. Later on, we will define $S^e = N_1^e, N_2^e, ..., N_n^e$ for the evacuee's state space which physically overlaps with the shooter's space, but conceptually part of a separate NHSMDP setup.

The successive states/nodes chosen by the shooter to visit are defined by the transition probability matrix and the sojourn time in each node, conditioned on the current and the next state to be transitioned into. Thus, during the transition times, the process is equivalent to an embedded Markov process. Let transition probabilities $p_{N_i^s \to N_j^s}(t)$ be the probability of the shooter who entered state N_i^s during the last transition at time t to transition to state N_j^s in the next transition. The transition probabilities should satisfy the same equations of a Markovian process, that is, $p_{N_i^s \to N_j^s} \geq 0, \forall N_i^s, N_j^s \in S$ and $\Sigma_{j=1}^n p_{N_i^s \to N_j^s} = 1, \forall N_i^s \in S$.

When the shooter enters node N_i^s at time t, we assume that this node determines the next transition to node N_j^s, which occurs according to the transition probabilities. However, before making the transition from state N_i^s to state N_j^s and after the next state N_j^s is selected, the chain holds in state N_i^s for time t_{ij}. The sojourn time t_{ij} is a positive random variable with density function $h_{ij}()$, which is called the function of sojourn time to transition from state N_i^s to state N_j^s. Thus, $P(t_{ij} = m) = h_{ij}(m)$, for $m = 1, 2, ..,$ and $N_i^s, N_j^s \in S$. We assume that the mean values of the distributions of sojourn times are finite.

In the context of the shooter's movement, various types of motivations and personalities can be modeled with judicious choice of the sojourn time distribution and the transition probabilities. Consultation with topical experts has suggested that there are three primary personality profiles that characterize a shooter [75] -

- either a shooter tries to maximize damage and thus seek out the most densely populated areas in the building,
- or a shooter is following a personal motive and seeks out a specific target,
- or a shooter is carrying out an unplanned act of violence.

Since it is almost impossible to predict beforehand the motive prompting a particular type of shooter, and moreover, the cost of an erroneous determination can be catastrophic, in this paper we assume a memory-less uniform probability discrete choice model, which can be explained with the help of a simplified example, shown in Fig. 2.1.

We start with the assumption that we know the location, i.e the state of the shooter at $t = 0$. The memory-less uniform probability discrete choice model dictates that at

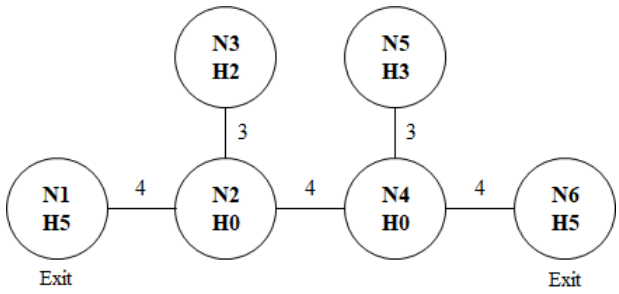


Figure 2.1: Toy Problem Used to Demonstrate the Dynamics of Transitions

any instant, the shooter can choose to stay in the current node for the duration of the discretized decision cycle, or start moving to any of the adjacent nodes based on the building layout. Without knowing or trying to predict the shooters actions, we assumed the worst-case scenario for planning purposes that the shooter could go in any direction with equal probability. Again, using the toy problem setup, a shooter in N_4^s at $t = 0$ would choose to move to the adjacent nodes with probabilities $P(N_2^s) = P(N_4^s) = P(N_5^s) = P(N_6^s) = 0.25$. With this assumption of memoryless uniform choice probability, the cascading probabilities of the shooter's positions at the various nodes and edges can be computed. Table 2.1 below demonstrates this. It may be noted that at $t = 1$ $P(N_4^s) = 0.25$ whereas the probability the shooter is on an edge moving to the node of choice is 0.25 as well, i.e. $P(E_{2\to4}^s) = P(E_{4\to5}^s) = P(E_{4\to6}^s) = 0.25$.

Based on the travel time between nodes, the probabilities of reaching nodes N_2^s, N_6^s and N_5^s spike at $4, 4$ and 3 seconds respectively. This sequence of unbiased discrete choice between the accessible nodes, travel time and sojourn time at the nodes is cascaded until we can compute the probabilities of the shooter being in any of the accessible rooms in the building at any subsequent times. This propagation of probability continues throughout the 300 seconds which is set as an upper bound of the duration of these events based on expert advice.

A limitation of this shooter-movement model is that, we assume that the shooter does not turn back or change his decision while halfway to a target node, i.e. each sequential target choice is made only after each node is reached, but not in the links connecting these nodes. Moreover, the memory-less choice dynamics may be improved, by more realistically modeling the sojourn times to follow a specific distribution. However, this model does provide a robust baseline for evaluating the performance of the algorithm for routing the evacuees away from the shooter. The evacuation optimization algorithm is discussed next.

2.4.1 Evacuation Algorithm

The algorithm used to determine the routing of the evacuees follows a practical implementation of the non-homogeneous semi-Markov decision process (NHSMDP) optimized with finite-horizon value iterations. A Markov Decision Process formulation is defined by the

Table 2.1: Probable location of shooter at time=t.

Time in Seconds	0	1	2	3	4	5	6	7
N_1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
N_2	0.00	0.00	0.00	0.00	0.25	0.13	0.05	0.02
N_3	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.06
N_4	1.00	0.25	0.06	0.02	0.00	0.00	0.13	0.13
N_5	0.00	0.00	0.00	0.25	0.19	0.11	0.06	0.03
N_6	0.00	0.00	0.00	0.00	0.25	0.19	0.11	0.06
$E_{1\to2}$	0.00	0.00	0.00	0.00	0.00	0.06	0.09	0.11
$E_{2\to3}$	0.00	0.00	0.00	0.00	0.00	0.06	0.09	0.04
$E_{2\to4}$	0.00	0.25	0.31	0.33	0.09	0.09	0.10	0.14
$E_{4\to5}$	0.00	0.25	0.31	0.08	0.15	0.22	0.15	0.12
$E_{4\to6}$	0.00	0.25	0.31	0.33	0.08	0.15	0.22	0.31

4-tuple (S, A, P_a, R_a) where S is a set of possible states, A is a set of possible actions that can be taken from these states, $P_a(s, s') = P(s_{t+1} = s'|s_t = s, a_t = a)$ is the probability that action a in state s at time t will lead to state s' at time $t + 1$, and $R_a(s, s')$ is the expected immediate reward received after transitioning from state s to state s' due to action a. In contrast to standard MDPs, the NHSMDP differs in two main aspects, the non-stationary transition probabilities and the idea of sojourn times at various nodes. These differences and their treatments are explained next section in the context of an egress plan.

State

In the context of an egress plan, the state is a tuple defined by $S^e = \{N^e, t\}$, where N^e are the nodes that the evacuees can occupy, and $t \in T$ is quantized time, elapsed since the shooter was last spotted by a camera. By folding time into the state definition allows us to include plans for waiting in a state, but exponentially increases the size of the state space. However, statistics and expert consultations suggest that most active shooter situations culminate within 5 minutes or 300 seconds. Discretizing time at 10 seconds intervals (any higher resolution would be irrelevant, since the slow speed of human movement dynamics would not allow any appreciable changes at finer time scales) yields $|T| = 30$ time steps. To put that in context, for a large building with 100 nodes (say), for each shooter position, the optimization space would be $|N^e| \times |A^e| \times |T| = 100 \times 100 \times 30 = 3000$, a level of complexity

that can be handled by modern computers, especially since the optimization is completed off-line and stored as a policy.

Action

In our problem, the evacuees' actions $A = \{M^e_{\mathcal{S},\mathcal{T}}\}$ represent the decision to move from a certain source node ($N^e_{\mathcal{S}}$) toward a certain target node ($N^e_{\mathcal{T}}$) at each discretized time step until they reach safety. Based on the layout of the building, the action can either be to stay in their current node until the next decision cycle, and then choose another action, or they can begin moving towards any adjacent node in the graph network starting from their current node location.

Transition Probability

The probabilities associated with successfully executing the chosen action are calculated based on the shooter's movement model. It is assumed that the evacuees' probability of transitioning to their target node within a certain time is entirely dependent on them not encountering the shooter en route and is portrayed in Figure 2.2. This may be calculated by inputting elements from Table 2.1 into Equation 2.1 for each step in the evacuees path from source to the target node.

$$P(N^e_{\mathcal{S}}, N^e_{\mathcal{T}}) = 1 - max(P(N^s_i), P(E^s_{i,j})), \quad \forall N_i \bigcup E_{i,j} \tag{2.1}$$

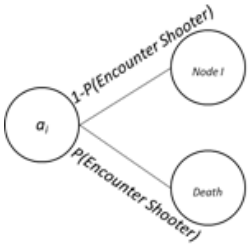


Figure 2.2: Probabilities Associated with an Action

Unlike traditional MDPs, the transition probability is an evolving function of time depending on the shooter's last known location and his possible movement choices. This non-stationary transition matrix makes this an NHSMDP.

In the context of the toy problem, Nodes N_1 and N_6 are exits and the values on the edges are travel times between nodes. Each node also has a hardness attribute that will be addressed in the reward structure discussion. We conjecture that an evacuee, without taking into account willingness to comply, the uncertainty associated with guiding young children, etc. can successfully stay in their current node or move to another node with probability 1.00 unless there is an interaction with the shooter.

Here interaction is defined as either being in the same node as the shooter or being in his line of sight defined by the layout geometry, for example, the line of sight for a hallway node would include all nodes along the length of that hallway, while the line of sight from inside a room does not extend beyond the room itself. Table 2.2 demonstrates this modification which in essence changes the meaning of the probability to be the likelihood that the evacuee will be in the line of sight of the shooter.

Reward Structure

The final initialization of the NHSMDP is the reward structure that determines the benefit or detriment that goes along with executing a given action from a specified state. Our reward structure is based upon the hardness of the node and the time to reach the nearest exit from that node. We consider the hardness of the node as a metric of how much protection the node can offer against a potential shooting, i.e. an open hallway node has a hardness score of

Table 2.2: Probability of Harm by Shooter at time=t.

Time in Seconds	0	1	2	3	4	5	6	7
N_1	0.00	0.00	0.00	0.00	0.25	0.13	0.09	0.11
N_2	1.00	0.25	0.31	0.33	0.25	0.13	0.13	0.14
N_3	0.00	0.00	0.00	0.00	0.00	0.06	0.09	0.06
N_4	1.00	0.25	0.31	0.33	0.25	0.22	0.22	0.31
N_5	0.00	0.25	0.31	0.25	0.19	0.22	0.15	0.12
N_6	1.00	0.25	0.31	0.33	0.25	0.19	0.22	0.31

0 whereas a corner room with no windows will have a higher hardness score. An exit would have the largest hardness score since it provides the chance to escape away from the building. In a practical implementation, the hardness would be based upon the type of door, presence of windows, and the perceived building materials used in the construction of the room. The times to the nearest exit for each node were calculated based upon Dijkstra's shortest path algorithm [18] using travel time as the weight. Incorporation of this distance-from-exit metric in the reward incentivizes the algorithm to prioritize routes that lead towards one of the exits rather than move the evacuees deeper into the building in search of safer rooms. The reward equation follows:

$$R(N_{\mathcal{S}}^{e}, N_{\mathcal{T}}^{e}) = \begin{cases} \alpha \times \frac{h(N_{\mathcal{T}}) - h(N_{\mathcal{S}})}{h_{max} - h_{min}} \\ \qquad +(1-\alpha) \times \frac{\tau(N_{\mathcal{S}}) - \tau(N_{\mathcal{T}})}{\tau_{max} - \tau_{min}} \\ 10 \qquad \text{if action leads to an exit} \\ \text{-10} \qquad \text{if action results in an interaction} \end{cases}$$

where $h_{\mathcal{S}}$ and $h_{\mathcal{T}}$ represent the hardness of the source and target nodes respectively, while τ represents the shortest time to reach the nearest exit. α is a tunable factor that can be engineered to more heavily weigh either the hardness metric or the distance-from-exit metric in the reward function. A small parameter sensitivity study was performed to find an optimal value of $\alpha = 0.75$.

The overarching idea behind this reward function design is that if the evacuee moves from a "softer" node to a "harder" one, they gain a positive reward by virtue of moving to a safer node and vice-versa. The opposite is true of the time to exit for each node. An action that leads to an exit receives a reward of 10 and an action that leads to an encounter with the shooter receives a reward of -10. The rewards for reaching an exit node or the shooter-occupied node are constant and each represent a terminal state.

Value Iteration

With the framework of our problem set, we determined the optimal actions in each state using value iteration [89]. For each location of the shooter, the finite horizon optimization

problem was over the defined NHSMDP. The values for the states are calculated according to the equation

$$V^{N_s}(N^e_{\mathcal{S}}, t) = max_a P(N^e_{\mathcal{S}}, N^e_{\mathcal{T}}) \times R(N^e_{\mathcal{S}}, N^e_{\mathcal{T}}) + P(N^e_{\mathcal{S}}, N^e_{\mathcal{T}}) \times (\gamma V(N^e_{\mathcal{T}}, t + \tau(\mathcal{S} \to \mathcal{T})))$$

where N_s is the current known node of the shooter, $N^e_{\mathcal{S}}$ is the current (Source) node of the evacuee, $N^e_{\mathcal{T}}$ is the destination node, $P(N^e_{\mathcal{S}}, N^e_{\mathcal{T}})$ is the probability of the evacuee reaching the target node at time $t + \tau(\mathcal{S} \to \mathcal{T})$ starting from the source node at time t, where $\tau(\mathcal{S} \to \mathcal{T})$ is the travel time between the source and target nodes. $R(N^e_{\mathcal{S}}, N^e_{\mathcal{T}})$ is the reward for moving from source to target, γ is the discount factor (0.75 in our case). The expected reward for every action for every state is compared to the current value of the state. If it is better, the value is updated. If it is not better, the value remains the same and the algorithm moves onto the next action. This process is iterated through each state-action pair until the largest change is smaller than $\epsilon = 0.0001$ or a maximum number of iterations (1000 in our case) is reached. Once the value iteration has converged, the value iteration process is iterated through one more time to record the optimal action in each state. The optimal action becomes the routing path determined by the current state of shooter node, evacuee node, and current time. An example routing plan for an evacuee located in N_5 when the shooter is located in N_4 would be $(N_5, N_5, N_5, N_5, N_5, N_5, N_4, N_6)$. This plan demonstrates how the evacuee should wait in N_5 for 6 cycles that is 60 seconds before moving to N_4. This process is completed for each of the possible starting shooter locations. The result is a look-up table with an entry for every combination of shooter position, evacuee position, and time elapsed that will direct the evacuees to safety in the simulation environment.

2.4.2 A Broader Perspective on the Choice of NHSMDP as a Modeling Framework

Due to the specific nature of the challenges related to an active shooter scenario, two aspects of the optimization objective made it uniquely different from most methods found in evacuation and egress literature:

1 A complex mobile threat that is dangerous within a certain vicinity of the shooter and consequently is transient; and

2 The need to maintain the ability of an evacuee to wait, hide, and proceed with evacuation.

An optimization structure built around NHSMDP addresses both of these challenges. From a language theoretic point of view, a semi-Markov chain and a deterministic finite automata (DFA) are identical, which can be informally argued as follows: Let the list of each node visited by the evacuee be expressed as a string w and the collection of all such possible strings denoted by the language $\mathcal{L} = \{w\}$. The string w can always be broken into three strings $w = xyz$; such that, $xy^kz \ \forall k \geq 0$ is also in $\mathcal{L}$. The k repetitions of y here implies that the evacuee stay hidden in Node y for k decision epochs. This is the basis of the Pumping Lemma [35] that proves that $\mathcal{L}$ is a regular language and hence can be expressed by a deterministic finite state automata. It is crucial to note that this is only valid as long as there is no restriction on the number of epochs an evacuee can stay in one node. If additional restrictions are applied on the path taken, in some cases, the language $\mathcal{L}$ might cease to be a regular language and the NHSMDP formulation would not be appropriate.

Moreover, since we are not merely interested in expressing possible routes, but maximize the probabilistic reward, we have to account for the change in probability as a cascading function of the shooter's movement in time. The simple act of quantifying the product space of time and nodes as the state space of our NHSMDP ($S^e = \{N^e \times t\}$) gives the formulation sufficient expressive power to allow estimation of cumulative gains from waiting in the same node for multiple epochs and then venturing out towards safety. For example, in Figure 2.1, given a state tuple of S=(N_4^e,N_1^s,0), our algorithm would guide the evacuee to N_5^e as moving

to the exit keeps them in the shooter's line of sight for longer. If the shooter then moves to N_3^s, our evacuee could then potentially make their move to the exit. Most routing research currently identifies shortest or fastest routes, not necessarily the safest.

The nature of the problem along with the short duration which makes it feasible to work with a time expanded state space creates an unique opportunity to use a model based optimization. In comparison to other potential methods in machine learning such as Deep Reinforcement Learning, this provides a logical explainable methodology with guaranteed optimization results.

2.5 Experimental Design

2.5.1 Simulation Environment

In order to evaluate the performance of our algorithm, we used the package "Simpy" [73], a process-based, discrete event simulation framework within Python, to create an environment where a shooter could act within a building environment and allow multiple evacuees to respond according to the routing produced by the routing algorithm.

Once the simulation was completed, performance metrics such as number of casualties, escapes, the amount of time spent within the shooter's line of sight, and the number of times an evacuee's routing plan was changed, were recorded. The time in the shooter's line of sight was evaluated by comparing each evacuee's current position to the shooter's position with respect to the specific layout of the building at every point of simulation time.

The deterministic metric for casualties was an extension of the line-of-sight evaluation. An evacuee was considered to be a casualty if the evacuee's node at any time was less than 4 edges away (based on Djakstra's shortest path algorithm) from the shooter's node while being in the shooter's node's line of sight. The length of 4 is based upon the generalized accuracy and lethality of a hand gun according to law enforcement professionals. An evacuee who reaches an exit node before being caught by the shooter was considered to have escaped. Each of these metrics was collected for every second of time up to the 300 second threshold or until every evacuee had either escaped or been caught.

2.5.2 Shooter Actions

The actions of the shooter were partially predetermined and randomized based upon the layout of the simulated building. At the beginning of the simulation, the shooter was initialized, which means the shooter spawned in a predetermined node with an attribute of a target node (both the spawn node and the target node are set by the parameters of the simulation and will be discussed later). Upon spawning, the shooter was programmed to determine the shortest path to its target node and to begin moving along that path (since most active shooter events are pre-planned, it is likely the shooter would know the fastest way to get to the desired location) [75]. Whenever the shooter reached the first way-point node along his route, it was removed from the planned path. This continued until the shooter reached the target node. Upon reaching the target node, the shooter was programmed to stay in the node for 5 seconds, after which, a new target node was randomly selected from the remaining unvisited rooms in the building, with closer rooms having a higher probability of being selected, and the shooter would begin moving in that direction. This process of the shooter randomly moving from node to node continued until either all evacuees had escaped/died or the 300 second threshold had been reached.

2.5.3 Evacuee Actions

The actions of the evacuees are based on the optimum route determined based upon their location with respect to the shooter's. For the purposes of determining the efficacy of the algorithm without considerations for compliance or other human-factors, we assumed that the evacuees would all perform the routing instructions without hesitation. In the same fashion as the shooter, the evacuees would be provided a complete path to an exit. That path may require the evacuee to start moving immediately, hold in a room for one or multiple decision cycles before moving, based upon the proximity of the shooter. Also, like the shooter, the evacuees do not make new decisions during the travel to the target node. Upon reaching the next node, the evacuee compares their current position to that of the shooter and either continues on the previous planned route, or changes the route because the state of the system has changed.

2.5.4 "Naturalistic Response"

We are going to refer to a set of simple rules that encode the egress logic behind the "run-hide-fight protocol" as the Naturalistic Response (NR) and use NR as the basis for comparison with the performance of the optimized routing algorithm. For a fair comparison, the NR algorithm would use the exact same information about the shooter's position as well as knowledge of the building layout and the ability to determine the shortest paths to all exits. The NR algorithm can be described as a set of rules as follows:

- At every decision update point, if an evacuee is in a hallway, he/she will either start running toward an exit or towards the nearest room based upon the proximity of the shooter.
- If the evacuee is in a room and if the shooter is within a certain proximity threshold path length of the evacuee's node, they will hide in the room.
- When the shooter is farther away than the threshold path length, the evacuee will begin moving toward the nearest exit.

For our simulation, we examined and compared the casualties and LOS times by varying the proximity thresholds from 1 to 8, named NR_1, NR_2, etc. for ease of reference. For example, for plan NR_8, an evacuee will hide in a room as long as the shooter is within a path length of 8 nodes and begin moving as soon as the shooter is farther away.

2.5.5 Graph Environments

We simulated the evacuation procedures in two different buildings: a single-level, three-wing school (Fig. 2.3a) and a two-level hospital (Fig. 2.3b). The algorithm, because of its reliance on the estimation of probabilities of evacuees encountering the shooter, needs to know some layout specific parameters, such as the distance between nodes, relative safety of nodes which we call hardness, and the line of sight from a given node along with the travel time between nodes. To get a realistic estimate of travel times between nodes, in the absence of real data, we used simulations created in Unity and Unreal Engine to determine the time to go from one node to another. Because these gaming engines utilize physics-based models of movement -

(a) Single-Level School Layout **(b)** Two-Level Hospital Layout

Figure 2.3: Two Different Graphical Layouts for Simulation Comparison

walking, running, and even model physical conflicts caused by obstructions, proximity and bottlenecks, we believe that the simulated travel times are reasonable estimates of actual movements. We performed simulations between each node pair and averaged travel time over 10 iterations.

The safety of a node, or hardness, was a subjective value that is more meaningful in the context of real-world implementation. For the purpose of our algorithm, all nodes in hallways or stairways were determined to have a hardness of *0* which implied that these spaces offered no protection whether from cover or concealment. An exit was determined to have the highest hardness as it is projected as offering the most safety in the form of escape. For out two graphs, both exits were given hardness values of *8* with interior rooms given hardness values in between the minimum and the maximum. The values for the rooms were subjective, but meant to reflect the level of relative protection offered, such as, a room with a single entrance and a windowless metal door that can be barricaded had a higher value than a room with multiple entrances and windows in the door. Further, in order to ensure the subjectivity of the values didn't overly affect the routing decisions, the reward function normalized the values on a scale of *0-1* as discussed in section III.A.4. The line-of-sight array was simply a list if nodes that could be seen from a given node and was manually determined based upon the graph network and building layout.

We experimented with the movement speed of the evacuees relative to the speed of the shooter. We simulated different types of evacuees by allowing the evacuees to move at 50%

(young children), 75% (middle school to high school), and 100% (college and adult) of the shooters speed. We also experimented with the distribution of the evacuees: either the evacuees were uniformly distributed only in the rooms, or they occupied both rooms and hallways. The final parameter we examined, which was common to both buildings, was the rate at which the evacuees were updated about the shooter's position. The position updates based upon time were tested at values of 30, 20, 10, and 1 second intervals with the 1 second update interval simulating full and continuous knowledge of the shooter's location.

The single-level school can be seen in Figure 2.3a. The hallways are indicated by nodes 1 through 18, the rooms are indicated by nodes 19 to 50, N_{51} is the cafeteria, and the exits are indicated by nodes 52, 53, 54, 55. The two-level hospital can also be seen in the Figure 2.3b. Similarly to the school, nodes 1 through 31 are the hallways, nodes 32 through 70 are the rooms, nodes 71 through 77 are the stairways, and nodes 78 through 82 are the exits. The remaining parameters, which contained multiple values and were distinct and specific to the school and hospital, were the spawn locations of the shooter, the shooter's initial target room, and camera locations. For the sake of clarity, the details of each of these parameters, will be discussed in more detail in the results section. The shooter's initial target rooms (N_{26}, N_{36}, N_{48}, and N_{51} in the school and N_{37}, N_{41}, N_{46}, N_{52}, N_{61}, and N_{69} in the hospital) were identified to ensure that the shooter was moving throughout the entire building and providing a variety of situations in which the evacuees would have to react. Accounting for all of the parameter combinations, the school and hospital were used to examine 3,456 and 9,028 different cases respectively.

2.6 Results

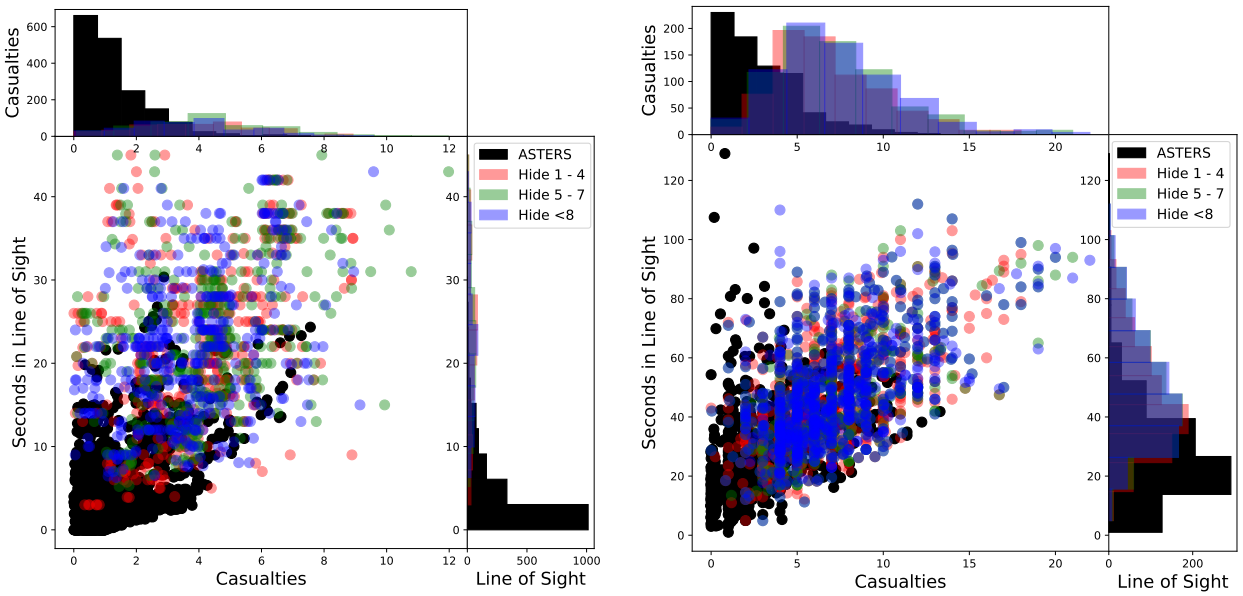
2.6.1 Overall Performance

The optimized routing algorithm was compared to each of the Naturalistic Response plans (corresponding to the different proximity thresholds ranging from a path-length of 1 up to 8) on both the single-level school and the two-level hospital layouts. The two graphs depicting

all of the simulation results combined for the school and the hospital are in Figures 2.4a and 2.4b.

The horizontal axes in these graphs are the number of casualties and the vertical axes are the time spent within the shooter's line of sight. As the legend in the figure suggests, the NR algorithms are parameterized by the proximity threshold and are grouped because they either performed exactly the same or with negligible difference. These graphs show the average of 50 runs in each of the $3,456$ scenarios in the school and $9,028$ scenarios in the hospital based on the combination of factors discussed earlier. The density histograms show that the concentration of the optimized algorithm's performance is significantly closer to 0 casualties as well as minimum time spent in the shooter's line-of-sight while the NR algorithms tend to be much more spread along the respective axes towards higher casualties and higher exposure to danger.

The school, which is much less complex architecturally than the hospital, shows a very clear advantage to the optimized performance in the sense that both casualties and LOS time



(a) School Simulation Results:

(b) Hospital Simulation Results:

Figure 2.4: Overall Simulation Results for School and Hospital: Each dot is color coded to represent the optimized algorithm and the naturalistic logic. The optimized routing algorithm data in each of the histograms are densely scattered around 0 casualties and 0 seconds spent in line of sight of the shooter. The distributions for casualties and LOS are separately seen along the two axes

are minimized. The more complex hospital demonstrates a more interesting phenomenon in the sense that even though casualty is minimized for our optimized plan, it does lead to isolated cases of evacuees getting long exposure in the line of site of the shooter. This discrepancy actually highlights the power of the optimized algorithm since it can distinguish between the fact that spending more time far away from the shooter even though in the line of sight can provide viable escape paths with low chance of death and actually urges the evacuees to utilize these paths. In real life, such fine tuned escapes may not be possible due to the uncertainty and delay in execution of the proposed plan, but this exercise does provide a successful algorithmic baseline which may be fine tuned based on human factor considerations.

While these graphs give a general idea the of performance, more quantitative measures can be observed from Table 2.3 which categorizes the tested simulated conditions to demonstrate different performance levels based upon the particular common parameter conditions present. For each set of conditions, the simulation was run 50 times for each of the 9 algorithms - (optimized and NR_1 through NR_8) The values reported in the table are the averages of the 50 runs. In this table, for clarity we only compared the performance of the optimized algorithm to the best performing of the eight NR algorithms in each parameter scenario.

Table 2.3 summarizes the results - the optimized algorithm outperformed the best NR algorithm in each of the parameter scenarios in both the school and the hospital. Across all combinations of parameters, following the optimized algorithm resulted in 56% fewer deaths and 52% less time spent in the shooter's line of sight in aggregate. The averaging over 50 iterations with randomly varying movement pattern for the shooter enforces robustness of the results produced. Moreover, since the random generator is seeded identically, the exact same (random) conditions are tested for all the algorithms, thus guaranteeing the consistency across the tests. Going beyond the gross statistics, we also examined our algorithm's performance individually across the different scenarios.

Table 2.3: Tabular Simulation Results Optimized VS the best NR responses: This table shows a pairwise comparison of the optimized algorithm with the best performing NR for each combination of parameters for both the school and hospital assets. The ASTERS routing algorithm outperforms the best NR in each and every parameter combination across both assets in both casualties and time spent in line of sight.

Shooter Spawn	Evacuee Distribution	Evacuee Speed	Hospital				School			
			ASTERS		Best NR		ASTERS		Best NR	
			Casualty	LoS	Casualty	LoS	Casualty	LoS	Casualty	LoS
Exit	Rooms	0.5	2.7	20.8	5.6	34.2	0.7	4.2	2.6	18.4
		0.75	1.7	16.2	5.4	33.7	0.3	2.8	2.9	17.7
		1.0	0.6	11.8	5.7	34.0	0.3	1.3	3.0	18.1
	Rooms and Halls	0.5	4.4	35.6	8.8	53.2	1.0	10.1	4.4	28.8
		0.75	2.6	35.6	8.8	53.2	1.0	10.1	4.4	28.8
		1.0	1.1	18.6	8.1	52.1	0.6	5.8	5.1	28.3
Hall	Rooms	0.5	2.7	20.6	5.0	33.9	1.0	3.8	1.5	14.5
		0.75	1.1	15.7	5.0	35.7	0.7	1.9	1.3	15.5
		1.0	0.8	11.3	4.5	35.7	0.4	1.4	1.0	14.5
	Rooms and Halls	0.5	5.7	39.1	8.6	58.8	2.6	12.2	3.4	25.6
		0.75	3.0	30.1	7.9	61.0	1.9	9.4	3.2	27.3
		1.0	2.3	22.5	7.7	60.4	1.6	8.0	2.6	26.2
Room	Rooms	0.5	3.9	21.9	6.3	33.7	2.0	5.9	2.5	18.9
		0.75	2.2	16.0	5.8	34.1	1.3	2.2	3.1	15.8
		1.0	1.9	10.8	4.8	34.4	1.2	1.5	2.7	16.0
	Rooms and Halls	0.5	6.3	37.9	9.6	52.1	2.6	8.8	3.3	25.5
		0.75	3.7	27.9	8.3	53.0	1.4	3.4	4.3	21.1
		1.0	3.0	21.3	7.0	52.8	1.3	2.3	3.7	21.3
Stair	Rooms	0.5	3.8	30.0	5.1	38.5				
		0.75	2.8	28.1	4.4	36.3				
		1.0	2.6	24.8	4.3	36.4				
	Rooms and Halls	0.5	5.3	44.7	8.0	58.6				
		0.75	3.6	36.8	7.0	55.6				
		1.0	3.2	33.9	6.4	55.7				

2.6.2 Shooter's Start Location

The location where the shooter was first spotted played an interesting role in the effectiveness of the algorithms' ability to route individuals to safety. There were nine shooter start locations for the school to provide a variety of situations for the evacuees to respond to. For the simulation, we assumed the spawning of the shooter to be initialization of the active shooter event. The school shooter spawned in three hallway nodes (3, 6, and 16), two room nodes (26 and 48), the cafeteria (51), and three of the four exits (52, 53, and 54). The hospital shooter spawned in six hallway nodes (2, 9, 15, 18, 21, 25, and 30), six room nodes (37, 41, 46, 52, 61, and 69), three stairway nodes (72, 74, and 77), and all of the exits (78, 79, 80, 81, and 82). These spawn locations for both the school and hospital were identified to ensure a variety of situations were examined.

As it can be seen in Figures 2.5c and 2.5d, the time in line of sight stays fairly constant across the board regardless of where the shooter spawns. An encouraging result of the simulation is the number of deaths in both buildings is decreased by at least 25% when the shooter spawns in an exit node (this means the shooter is identified before they enter the building). The shooter spawning in Hall and Room were far more deadly for the evacuees, but not significantly different from each other. The hospital, in particular, showed similar trends based upon the shooter spawn locations, however, since it is a more complex building with longer hallways and sight-lines, multiple floors, and more nodes, the contrast in performance based upon the spawn locations was not as evident.

What does this physically mean in the real world? When we are able to identify a shooter before they enter the building, we are more often able to direct evacuees to safety. However, when the shooter cannot be identified until they are in a room or hall, the deaths that occurred were always greater than or equal to 1 indicating that it may be too late to save everyone if the shooter is already in the building. When planning for security purposes, it may be necessary to apply more resources to the visual and physical identification of the shooter before they enter a building. These resources could take the form of a school resource officer, metal detector, or, as we are assuming for the purpose of this paper, computer vision assisted cameras.

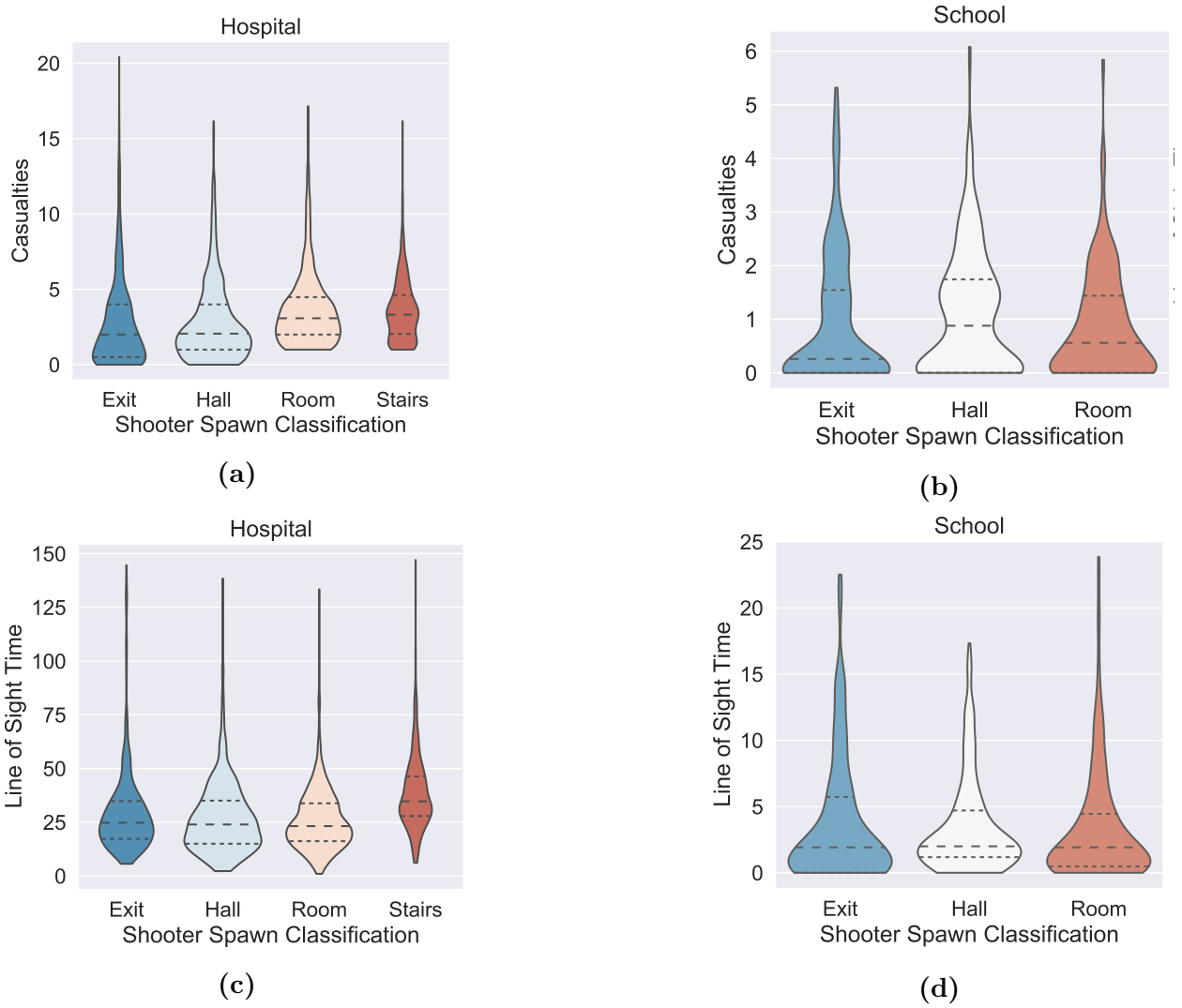


Figure 2.5: Overall The violin plots for casualties and time spent in line of sight distinguished by Shooter Spawn Location: a) Spawning the shooter inside rooms in the hospital incur increased casualties as a number of evacuees immediately get caught by the shooter. Stair spawns allow the shooter to move to either floor making evacuation decisions difficult; both result in higher mean casualties across all runs, b) the simpler school asset presents fewer layout challenges, but the hallway spawns incurred a higher mean casualty rate as well as more occurrences of higher casualties, c) the stairways in the hospital provide a significant disadvantage due the uncertainty of which floor the shooter will be on increasing the mean and upper bounds of the time spent in line of sight, and d) again, the simpler layout of the school did not demonstrate significant differences in the time spent in line of sight which really highlights the affect the stairways had.

2.6.3 Evacuee Distribution Effects

For both the school and the hospital, we tested two evacuee distributions: uniformly distributed only in the rooms and in both rooms and halls. As discussed earlier, the distributions had different meanings for each of the building layouts, but, in essence, are the same in the way they are implemented. Both the optimized algorithm and the best NR performed better when only the rooms were occupied at event initiation as seen in Figures 2.6a and 2.6b resulting in a 40% reduction in casualties across both buildings for the optimized plan and 43% reduction for the best NR. Evacuees present in both rooms and halls led to increased deaths in both buildings and also resulted in the some of the worst performances for the optimized algorithm among all the parameter combinations.

Thinking of this from a policy perspective, it is of course not practical to prevent people from using hallways in order to deter a shooter in one of these situations. That being said, there are potentially directives that can at least reduce the number of people in hallways and therefore increase their ability to maximize safety. An interesting idea was used on several college campuses during the 2020-2021 academic year to try and socially distance their students as much as possible. These universities staggered class times throughout the day in order to prevent the mass movement of students through hallways and walkways when all classes begin and end at the same time. By staggering the class ending times, they managed to reduce the number of people moving around on campus at any given point. A similar structure could work inside of a building as well to reduce the number of people in a hallway and at the same time avoiding the significantly slower moving times that occur when every class ends at the same time.

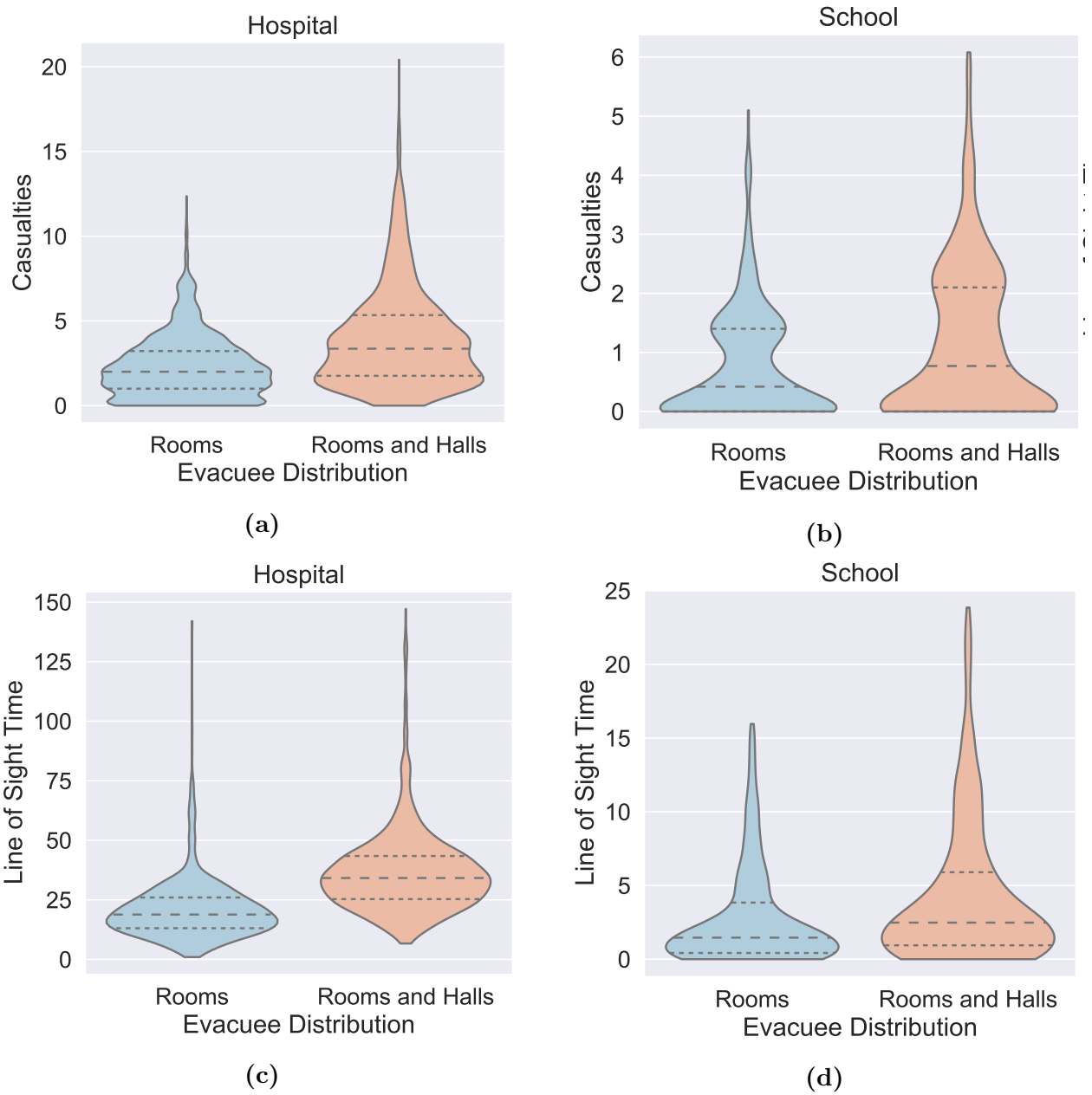


Figure 2.6: Violin plots for casualties and time spent in line of sight for both the hospital and school partitioned by the distribution of the evacuees. While the shape distributions of the hospital and school casualties are different, they each share the common aspect that evacuees distributed in the rooms only at the start of an active shooter incident incurred fewer mean casualties and lower ceilings of maximum casualties. Similar trend was observed for the LOS as well.

2.6.4 Evacuee Speed

The movement speed of the evacuees, both in the real world and the simulation, plays an enormous role in the performance the algorithm. There are vast differences between the movement speed of an adult and that of a group of young children. Further, there is a difference in the movement speed of someone who is calm, in no hurry, and someone is frantic in a dangerous situation. The simulation allowed the evacuees to homogeneously have different movement speed, meaning that all of the evacuees had the same speed, but it changed in conjunction with specific parameter combinations. The speeds the evacuees were able to move at were related to the speed of the shooter. A movement speed of 0.75 indicated that they moved at 75% of the shooter's speed.

The algorithm used the same speed as the shooter as a planning factor for the evacuees routing. So, evacuees with less speed than the shooter still used the routing meant for faster evacuees. The results of these speed differences can be seen in Figure 2.7. The deaths and time in line of sight can be seen in the violin plots, further broken down by evacuee distribution and building type. As it can be seen in the graphs, in each of the different scenarios between the hospital, school, and evacuee distributions, as the evacuees' speed increased, the amount of time they spent in the shooter's line-of-sight decreased making their ability to stay safe easier. The downward trend was the same for deaths in each of the scenarios as well. The difference in deaths and time in line of sight between the fastest and slowest movement speeds was 53% and 41% respectively, which is a significant difference. Looking forward, taking the speed of an evacuee into account and developing a plan specific to each group can only provide better results in the future.

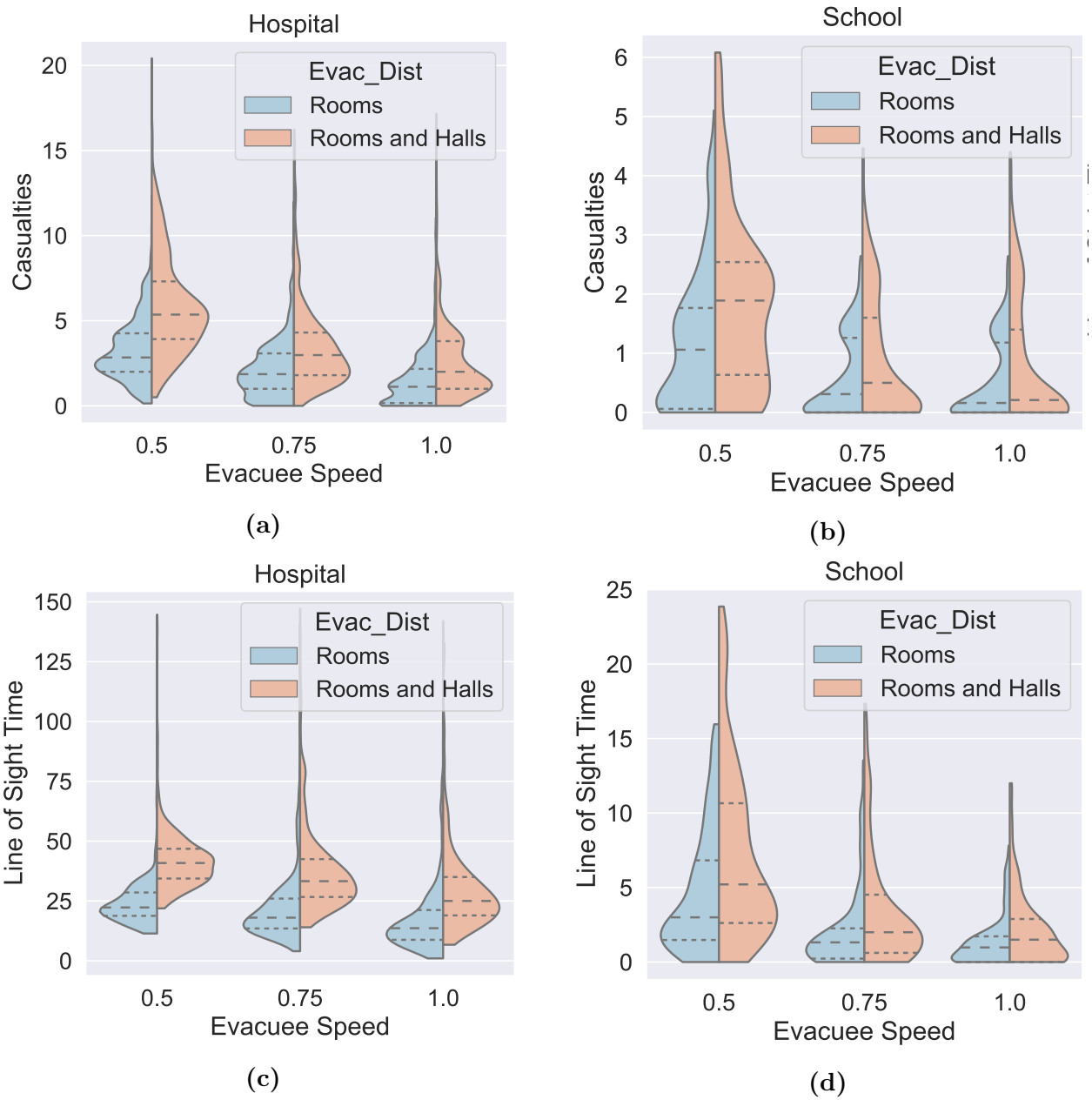


Figure 2.7: Violin plots for casualties and time spent in line of sight for both the hospital and school partitioned by the evacuees' movement speed and also by evacuee distribution. There are two common themes in every chart: 1) the slower the evacuees' speed was, the larger the mean casualties and mean time spent in line of sight and 2) considering each violin as a pairwise comparison within each evacuee speed, the room only distribution has fewer mean casualties and spends less time in line of sight in every pairing.

2.6.5 Cameras and Position Updates

As discussed earlier, this algorithm is assumed to complement a camera-based shooter-identification and tracking system. It is quite evident that the algorithm's effectiveness in providing optimal and timely guidance to the evacuees will depend on the frequency and quality of the updates it receives from the monitoring system. In order to investigate the routing algorithm's dependence on the performance of the camera network, we next look at a related problem - if resources are limited, and full coverage of the entire building with a dense network of cameras is not possible, what would be the optimal locations to place cameras so that the performance of the routing algorithm is still maximally preserved.

To solve the camera placement problem, we used the characteristics of the graph networks which were derived from the floor plans of the building we were instrumenting. The goal was to logically add the cameras to the model without redundant line of sight. We used the network adjacency matrix and calculated the betweeness centrality, a measure of how many shortest paths the node appears in, of every node. The choice of the betweeness centrality as the metric for camera placement is related to the underlying unbiased movement model adopted for the shooter's random walk. Such random walk would result in an eventual steady state probability distribution of the shooter's possible locations over the nodes of the graph. This distribution again is related to the "connectivity" of a node, i.e. the more number of ways a node is accessible, the large is the chance of the shooter to pass through that node. This in turn makes it a logical choice for placing a camera.

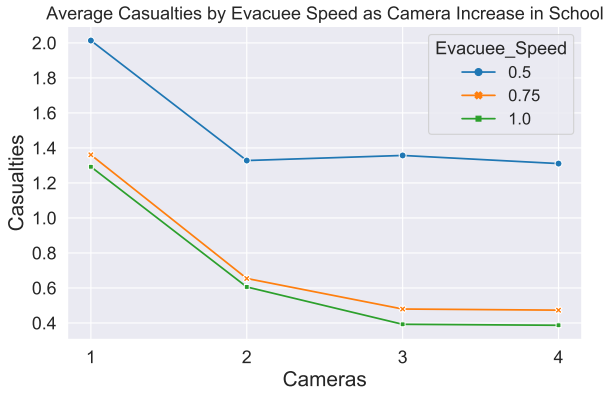
Once the betweeness centrality is calculated for the entire building graph, we then rank-ordered the nodes from greatest to least betweeness centrality measure and reduced the list by removing all of the nodes that had a measure of 0, meaning that the shortest paths they appeared in were the ones either beginning or ending with themselves. We then iteratively added a camera to the node with the largest betweeness centrality and then removed any node from the ordered list that was in the added camera's line of sight until the list was empty. Each of the camera locations for the school and hospital will be covered in the following sections.

Camera Performance for School

Using the methods previously discussed, we established four possible locations for the cameras in the school: *Nodes* N_6, N_8, N_{14}, *and* N_{51}, in that order. For each set of parameters, a camera was added iteratively and run 50 times. The results of iteratively adding the cameras into the simulation demonstrate the increasing effectiveness of the optimized algorithm as the amount of information on the shooter's position increases. Figures 2.8a and 2.8b show the output from these runs. The average deaths, broken down by evacuee speed, show the decrease in the deaths as the number of cameras in the simulation increase. On a side note, it also demonstrates the effect the difference in evacuee speeds has on the performance as well. The other figure shows the Earth Mover's Distance (EMD) for the distributions of casualties with addition of cameras. EMD is a distance measurement that quantifies how different two distributions are with values close to 0 being extremely similar and larger values being more dissimilar. As it can be seen, there is a significant difference between one and two cameras, a little less than half of the difference between two and three cameras, and almost no difference adding the fourth camera. Both the average deaths and the EMD mirror the same conclusions: performance continues to improve as cameras are added to the simulation providing more and more regular updates to the shooter's position. However, when having to take into account possible budgetary concerns for schools, the fourth camera in *Node 51*, the cafeteria, does not provide a significant jump in performance and could be potentially removed in the name of cost savings.

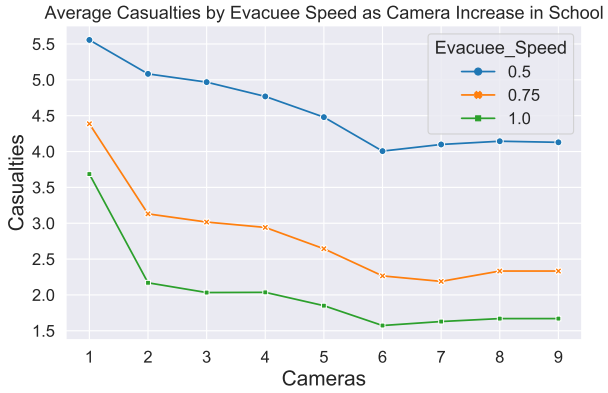
Camera Performance for Hospital

The hospital, due to its more complex structure and quantity of nodes, had nine cameras in the following locations: *Nodes* N_{22}, N_6, N_{13}, N_{76}, N_{28}, N_{71}, N_{73}, N_{62}, *and* N_{50}, in that order. The performance for the algorithms as the cameras are added iteratively can be seen in Figures 2.8c and 2.8d. The average casualties in the hospital steadily drop as the cameras are added providing the additional information on the shooter's position. Again, it also demonstrates the vast difference that evacuee speed plays in the performance of the algorithm. Deaths significantly drop when adding the second camera and then steadily

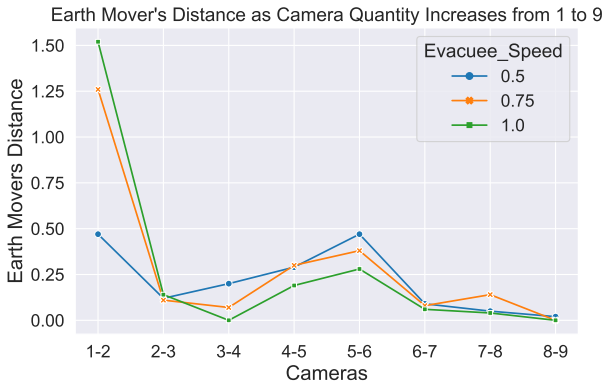


(a) Average casualties - Cameras in School

(b) EMD - Cameras in School



(c) Average casualties - Cameras in Hospital



(d) EMD - Cameras in Hospital

Figure 2.8: Affect of cameras in the school and hospital building layouts: *a:* The average casualties decrease as more cameras are added to the schools hallways indicating that increased awareness of the shooter's location provides increased safety protocols for the evacuees. *b:* The earth mover's distance (EMD) shows the amount of change when increasing the number of cameras, thus the large decrease in average casualties when increasing cameras from one to two, but much smaller decreases when increasing cameras from there. *c:* The average casualties steadily declines as more cameras are introduced, and potentially hitting a saturation point where more cameras provide negligible benefit at 7 cameras. *d:* while the EMD from one to two cameras is large, the addition of cameras after the second shows a little benefit, but not as drastic as one to two, which is explained by the two floor layout of the hospital and increasing to two cameras puts one on each floor. Beyond the 7^{th} camera, the effect of adding additional cameras almost falls to 0

decline when adding cameras 3, 4, 5, and 6. However, much like with camera 4 in the school, adding additional cameras provides significantly less benefit past camera 6 (diminishing returns). The EMD for the hospital camera distributions demonstrates the same trend with a big difference between one and two cameras, and very little difference in the changes between cameras three and six. The additional cameras past six again show very little difference in the distributions and hence very little benefit to the performance.

2.7 Conclusion and Future Work

In this work we have shown that the route planning in the event of an active shooter situation can be achieved using a naive NHSMDP approach. The reward structure that balances the benefit of being in a safer location against the benefit of moving closer to an exit while taking into account the location of the shooter allows the optimized routing algorithm to identify when it is safer to wait in a room or begin moving to an exit which is unique across all the other route planning algorithms. Across all combinations of parameters that we examined, the optimized routing algorithm outperformed the Natural Response plans by 56% fewer casualties and 52% less time spent in the shooter's line of sight in aggregate.

It is evident that the process of egress and optimization represented here is an abstraction of the true situation. Even within the bounds of our simplifying assumptions, there are several improvements that will potentially make this method a practical viable safety tool. Some of the current limitations of this work comes from lack of

- consideration of psychology, human factors and heterogeneity among the evacuating group,
- consideration of capacity and bottle-necking at doors and corridors, and
- ethical considerations

The most important limitation of this study is in the treatment of the evacuees as a perfectly compliant, logical, efficient homogeneous group of people. Obviously, none of these attributes are true in general, and especially for people under stress. Literature shows that

human factors, such as social pressure, leader follower, route familiarity and influence from trusted sources all play significant parts in the evacuees' choices and behavior [88, 87, 41]. Knowing the stress and danger involved in an active shooter situation, we want to further develop the algorithm to account for hesitancy, or even refusal, of the evacuees to obey the routing plans. Multiple factors will play into that work to include fear of the danger posed, trust/mistrust of a computer directed routing plan, etc. Heterogeneous populations with different abilities and capabilities will need to be accounted for to increase the applicability of the algorithm across multiple scenarios and complexities. Routing people of different ages and capacities, modeled as different speeds and instruction following accuracy, at the same time with different speed optimized routes will provide better feedback on how we can best assist schools with widely varying ages in these situations.

The capacity of a node or an edge is also an extremely important factor, especially for larger buildings like schools, office buildings, hospitals, etc. where a large number of people may spill into a hallway at the same time causing congestion. The algorithm should be able to account for that situation, develop plans for everyone in the building based upon available capacity, and direct some evacuees to hide in their current node for a period of time in order to prevent bottle-necking in the hallways and exits, where people are most vulnerable due to crowding and lack of cover.

In its current abstracted form, the merit of this work is in its effectiveness in developing an algorithm that successfully guides evacuees to safety in response to a mobile threat. Building upon this framework, the algorithm will be augmented in the future with more realistic considerations. However, one of the most difficult challenges in the realization of such algorithmic solutions is not technical, but ethical. We realized that significant moral dilemmas will need to be overcome when prioritizing evacuation routes of some groups over others, when capacity constraints demand such coordination. The amount of tolerable risk posed by the shooter and balancing that with the chance of escaping needs to be carefully considered. Taking into account these issues will continue to improve our performance as well as add realistic dimensions to the routing plan. However, similar to the famous trolley problem, the ability to make such nuanced decisions beforehand and codifying them in an algorithmic logic will no doubt need discussions with experts in various fields of psychology,

education and planning. It will not be possible to reach fair and equitable algorithmic solutions without some difficult moral and ethical debates.

Chapter 3

Multi-Route, Optimized Capacity-Constrained Evacuation Routing

A version of this chapter is currently being submitted to academic journals.

Joseph Lavalle-Rivera, Aniirudh Ramesh, and Subhadeep Chakraborty conceptualized the study and determined the simulation development and setup. Joseph Lavalle-Rivera ran all simulations and was primary author of the manuscript. Joseph Lavalle-Rivera, Aniirudh Ramesh, and Subhadeep Chakraborty determined the appropriate methods to post process the data for analysis and evaluation. Aniirudh Ramesh and Subhadeep Chakraborty proofread and edited the final manuscript.

This chapter is identical in content to the version being submitted to journals.

3.1 Abstract

A total of more than 3400 public shootings have occurred in the United States between 2016 and 2022. Among these, 25.1% of them took place in an educational institution, 29.4% at the workplace including office buildings, 19.6% in retail store locations and 13.4% in restaurants and bars. During these critical scenarios, making the right decisions while evacuating can make the difference between life and death. However, emergency evacuation is intensely stressful, which along with the lack of verifiable real-time information may lead to fatal incorrect decisions. To tackle this problem, we developed a multi-route routing optimization algorithm that determines multiple optimal safe routes for each evacuee while accounting for available capacity along the route, thus reducing the threat of crowding and bottlenecking. Overall, our algorithm reduces the total casualties by 34.16% and 53.3%, compared to our previous routing algorithm without capacity constraints and an expert advised routing strategy respectively. Further, our approach to reduce crowding resulted in an approximate 50% reduction in occupancy in key bottlenecking nodes compared to both of the other evacuation algorithms.

3.2 Introduction

The topic of emergency evacuations boasts a rich literature; Liu et al. [55] provide a comprehensive review with studies exploring a variety of approaches from agent-based models

[49, 31] and game theory [95, 93, 77] to statistical physics [87, 40]. Research topics range from human behavior (including vulnerable populations) effects [2, 20], to modeling, simulation, optimization and guidance oriented studies [97, 71, 52, 8, 13, 92, 34, 6, 88, 86], and the use of immersive virtual reality-based experiments to study evacuation decision-making [33].

Capacity-constrained evacuation routing optimization is more complex as it adds a temporal dimension to the graph environment [3, 59]. Optimization of egress paths in active-shooter scenarios is an even more complex task, not only due to the NP-hard nature of the optimization problem, but also because it must account for the dynamic and unpredictable nature of the threat as well as a multitude of psychological factors involved, such as social pressure, leader-follower and the "familiar route is a safe route" mentality. Gunn et al [30] is one of the very few works specifically on active-shooter evacuation where they divided evacuees into groups and used dynamic stochastic programming to find optimal routes for these groups. However, their work did not consider either the dynamic shooter movement, nor the capacity-constraint of the network. Lu et al. [63] explored fast quasi-optimal capacity-constrained routing solutions by modeling capacity as a time series and used a heuristic algorithm called Multi-Route Capacity Constrained Planner (MRCCP). However, they did not consider the threat to be dynamic.

The active-shooter evacuation scenario presents different challenges that most other evacuations do not. Most evacuations require the evacuees to exit the area as quickly as possible, while an active shooter evacuation may require the evacuees to choose between hiding in relative safety or evacuating to an exit.

Our previous work [47] focused on establishing the viability of using Reinforcement Learning for routing optimization in active shooter scenarios. In that work, we showed that the optimized egress routes resulted in 56% fewer casualties in a multi-parameter study. However, that study was focused on low occupancy cases that did not consider overcrowding and bottle-necking issues.

In this work, we have developed a novel algorithm that computes and ranks multiple routes, accounts for the maximum and available capacities within rooms, hallways, and through doorways, and allows evacuees to avoid being in the shooter's line of sight by hiding or evading the shooter in order to maximize safety to address the issues of capacity

constraints. We developed a discrete-event simulation using two topologically different building layouts to evaluate the effectiveness of our capacitated algorithm and compare it with other algorithms. Further, we have identified a method of ordering the routes that reduces the crowding and bottle-necking likelihood throughout the duration of an active shooter event.

For the remainder of this paper, we summarize the current literature and other works in section II, describe the specifics of our methodology in section III, discuss the experimental design, the simulation environment, and the different algorithms we evaluated in section IV, discuss and compare the results from our simulations in section V, and, finally, identify future plans to improve the algorithm in section VI.

3.3 Literature Survey

Capacitated transportation networks have been a robust area of research within many different academic communities like operations research, supply chain optimization, theoretical computer science, etc. The approaches to solve these complex models are vast and varied within the current literature. The reasons that motivate planned and optimized movement are also varied, ranging from decreasing delays in general traffic flow to quick evacuation during natural disasters. Most of these events require potentially different approaches to satisfy domain-specific requirements. For example, modeling and optimizing the throughput across a capacitated transportation network to improve traffic flow can necessitate a very different approach to that of an emergency caused by a natural disaster.

Emergency evacuations inside built environments are an interesting example of the capacitated transportation networks formulation. A specific application within this subset is the evacuation during active shooter events. Despite the large quantity of prior work in egress literature, a few key components of an active shooter evacuation makes it distinct from others - for one, the threat or danger in this situation is very mobile and moves in real time through the graph, which, along with the typical short duration of these events, may make it safer for evacuees to temporarily shelter in place or hide instead of trying to exit.

Moreover, the danger posed by an active shooter is often spread within his line of sight and not solely around his physical proximity based upon the weapon in use.

Hong et al. modeled the evacuation of a building as a Multi-Objective Dynamic Route Network Planning (MODRNP) problem [34]. Their approach uses iterative partition strategies to progressively identify shortest paths using Dijkstra's algorithm from the multiple sources to the multiple sinks. Li et al. use two methods, a maximum flow model (MFM) and a minimum cost maximum flow model (MC-MFM), to identify evacuation paths from a single source to either a single sink or multiple sinks[51]. Their results show that their models are practical and feasible, but don't necessarily avoid danger areas and only look to evacuate as efficiently as possible. Kaveh Shahabi and John Wilson examined road networks during wild fire evacuations and used their algorithm, Capacity-Aware Shortest Path Evacuation Router (CAS-PER), to identify shortest paths based upon available capacity[80]. Their emphasis was placed on the ability to adjust routes based upon road closures, however, they continued to identify routes as a static problem with an adjusted graph network. Liu et al. established routes for building evacuation for multiple sources and multiple sinks by iteratively reserving capacity along shortest paths until all evacuees had exited[54].

Using centrality measures of a graph to assist in the development of evacuation routes is an interesting and growing area of interest. Marin Lujak and Stefano Giordani used two metrics, evacuation betweeness-centrality and evacuation centrality, to identify and present what they refer to as agile routes[67]. They classify agile routes as ones that are similar in length, but often also include similar alternate routes from the same source to the same sink that avoid certain parts of the network viewed as dangerous. In parallel work, Tamakloe et al. used a vehicle evacuation algorithm based upon a link-based centrality measure to identify "agile paths"[90]. They used real traffic data from Seoul, Korea to demonstrate that their approach using centrality measure outperformed simple shortest path algorithms during high volume evacuation periods. The characteristic of being able to avoid specific parts of a network is of interest, however, in an active shooter situation, such danger zones are dynamic and staying in place may provide better safety.

Another alternate approach is that of iterative path assignment heuristics. Qingsong Lue and Shashi Shekhar demonstrated that their heuristic, Multiple Route Capacity Constrained

Planner (MRCCP), could achieve near optimal results by iteratively reserving capacity along the shortest routes[63]. They primarily tested the planner on smaller networks to ensure near optimal results. To go a step further, Kim et al. produced another heuristic called the Intelligent Load Reduction heuristic that scales to larger networks that capacity constrained planners would struggle with[43]. They are able to improve on computation time compared to the CCRPs by focusing on bottlenecks within the network and reducing the flow around them to account for the clogging of edges in the network. Neither of these methods account for a mobile danger as is the case in our work.

Tsai et al. examined a capacitated vehicle routing problem by comparing the use of a Deep Neural Network (DNN) based solution and that of a Sweep algorithm[92]. Using hurricane evacuation data from New Orleans, they hoped to show the DNN provided a vast improvement in the efficiency of routing when taking into account social distancing within the emergency vehicles. While efficiency did improve initially, the benefit provided by the DNN diminished greatly as the capacity of the emergency vehicle approached the average number of people per household being evacuated. The social distancing requirement increased the complexity of the routing and evacuation such that the use of smaller vehicles, which make up the vast majority of emergency transportation, almost completely removed the improvements of the DNN.

Acknowledging the risk or danger associated with a specific route or series of routes during evacuations is another aspect of pedestrian emergency evacuations that is well researched with many different approaches. Liu et al. examined evacuations aboard cruise ships that took into account route capacity and route safety[59]. Interestingly, they did not avoid riskier paths, but instead gave evacuees on the riskier paths a higher priority to take the capacity along their route. The acceptance of route risk and optimization of speed along the route could potentially be applied to an active shooter situation as there is often a high risk-high reward case when trying to exit the building. Lee et al. developed a means of evaluating evacuation actions using AnyLogic models during active shooter events[49]. They were able to examine the interactions of evacuation time, police response time, shots fired, etc. and evaluate the effectiveness that behavioral training, new technology, or architectural design of buildings had on the positive outcomes for the evacuees.

Arteaga et al. examined and scrutinized the effect that trained evacuation leaders had on positive evacuee outcomes during active shooter situations[7]. They showed that the presence of trained evacuation leaders always improved the outcomes for the evacuees, regardless of the number of leaders present or their distribution. The leader-follower aspect of evacuation is of extreme interest to our application and believe it may greatly improve our modeling and planning in the future.

In this paper, we propose a capacity-constrained framework based on Reinforcement Learning and a network optimization approach. Our approach of considering the active shooter as a dynamic threat with a dynamically changing risk metric and determining and distributing multiple evacuation routes that are ranked and allocated based upon danger metrics is novel and does not exist in the current literature to the best of our knowledge.

3.4 Methodology

In [47], the structure of a Non-Homogeneous Semi-Markov Decision Process (NHSMDP) was established for optimizing the evacuation of evacuees from one and two floor buildings in the event of an active shooter situation. This paper established the use of a NHSDMP structure as a suitable framework for modeling the dynamics and interactions of the shooters and evacuees[29]. Our previous work focused on the safe routing and movement of the evacuees throughout the buildings to avoid the threats posed by the shooter. We used a finite horizon reinforcement learning framework with time-expanded states to identify the optimal paths for the evacuees to avoid the shooter's line-of-sight. A wide range of simulation parameters including distribution of evacuees in the building, their movement speed relative to the shooter, location of first identification of the shooter, position update methods and frequencies were tested in which the NHSMDP method resulted in an overall 56% reduction in casualties across two virtual buildings.

One of the limitations of this work was the absence of capacity constraints along movement paths. As identified in the literature review above, capacity constraints in emergency evacuations are important to avoid bottle-necking and crowding. Specifically in the context of active shooter scenarios, due to the dynamic nature of the active shooter

threat, any bottle-necking near exits and narrow passageways would expose vulnerable people to serious harm. To address this important topic, using the same base framework as in [47], we introduce capacity constraints throughout the graph representations of the buildings on both nodes and edges. We deploy a Reinforcement Leaning- based algorithm on this graph that finds the optimal routes for the evacuees with a constraint on the maximum available route capacity and then iteratively finds the next optimal route until all evacuees have identified routes for safety.

3.4.1 NHSMDP States

The state in our NHSMDP structure is a three-tuple defined by $S = (node_e, node_s, t_i)$ for $0 \leq i \leq 300$ where $node_e$ and $node_s$ are the locations of the evacuee and the last detected location of the shooter respectively within the building, represented as a graph. The routes provided to the evacuees are computed based upon $node_e$ and a probabilistic risk measure of the likelihood of the evacuee being harmed by being in the shooter's line of sight, derived from $node_s$. Finally, the time at any instant, t_i, is the time from the start of an active shooter event ranging from t_0, the start of the event, to t_{300}, the end of an event measured in seconds. The finite time horizon was capped to 300 seconds based on data from the Mother Jones active shooter database, which says the majority of active shooter events conclude within 5 minutes[25]. The total number of states depends on the graph environment and the number of discretized intervals within the 5 minute interval.

We examined the performance of multiple routing algorithms in two separate building environments that will be discussed in more detail later: an acyclic, single-level school, and a school building with two parallel hallways and multiple passages connecting them that is representative of a graph structure containing cycles. In the two environments we simulated, the Acyclic School had a state space of $907,500$ different states ($55 \times 55 \times 300$) and the Cyclic School had $1,470,000$ states ($70 \times 70 \times 300$). Movement from one state to another is associated with a sojourn time based upon the distance between the nodes. For instance, if $node_i$ is 5 seconds away from $node_j$, then leaving the state-tuple $(node_i, node_s, 0)$ would result in potentially reaching the state-tuple $(node_j, node_s, 5)$.

3.4.2 NHSMDP Actions and Transition Probabilities

The possible actions that can be taken by an evacuee is based on the state that they are in and the available capacity at that time instant. Without considering capacity constraints, the possible actions in any given node are just the adjacent nodes to the agents current node. Consider a scenario where an evacuee is in $node_1$ in Figure 3.1. For an evacuee in $node_1$, the possible actions would be movement to $node_2$, $node_{19}$, $node_{24}$, or $node_{52}$ or they could choose to stay in their current node, $node_1$.

However, when considering capacity constraints, all of these options may not be available when choosing an action. For instance, if $node_1$ and $node_{19}$ are separated by a door, (as shown in Fig. 4(a)), the throughput is dictated by the flowrate through that door. Studies have shown that about two people per second and four people per second are the closest approximations of throughput of a single-door and double door doorway, respectively [15]. In the given state-tuple $(node_1, node_s, t_0)$, if there are two other evacuees moving through the doorway represented by $edge_{1-19}$ at t_0, then the available actions for the evacuee at t_0 are reduced down to $node_1$, $node_2$, $node_{24}$, or $node_{52}$. As the occupancy of nodes increases, especially around the exits, crowding and waiting can become a critical factor in the performance of any evacuation algorithm in such a severe situation. Hence, the challenge is to design an algorithm that minimizes crowding thereby improving throughput.

The probability to transition from one state-tuple to another is also directly related to the probability of being within range of the shooter's weapon during the entirety of the

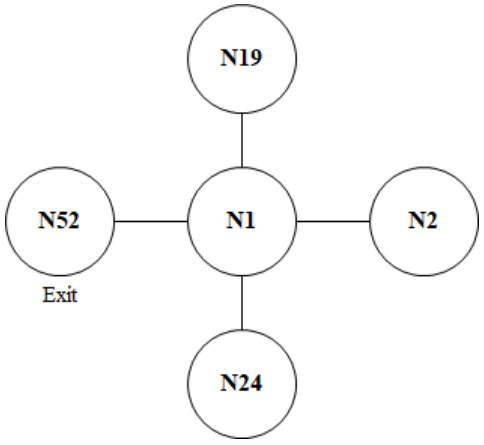


Figure 3.1: Possible Actions in $node_1$

transition and therefore does not remain constant as the shooter moves around the graph environment. When an evacuee takes an action, there are two possibilities: they reach the desired state-tuple or they come within range of the shooter's weapon as shown in Figure 3.2.

The actual probability of encountering the shooter or being within range of his weapon is dependent upon his last known location and the passage of time. We explain this using an example seen in Figure 3.3.

Each node in this example has a name, $N1$ for example, and a hardness, $H5$, that will be further explained in Section III C. Each edge has a value that indicates the sojourn time in between nodes. For this example, let us assume that we know the shooter is in $node_4$ at time t_0. Given the sojourn times and the available actions for the shooter, we compute the probability of the shooter's location over time as seen in Table 3.1. We know where the shooter is at t_0 and the available actions are movement to $node_2$, $node_5$, or $node_6$ or remain in $node_4$. Without trying to presume any information about the shooter's intent, we assume an unbiased random walk model, i.e. he can go in any direction with equal probability. With that taken into account, at t_1, the shooter has an equal probability of being on $edge_{2-4}$, $edge_{4-5}$, $edge_{4-6}$, or in $node_4$ if he stayed in place. We also assume that once he has made a decision, i.e. selected a target node, he does not make a new decision until he reaches his target. His possible location is propagated throughout the graph and through time as seen in Table 3.1.

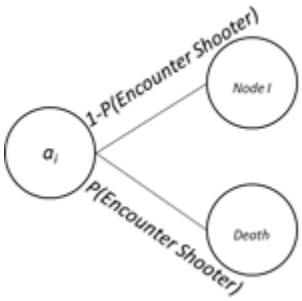


Figure 3.2: Transition Probability for a Given Action a_i

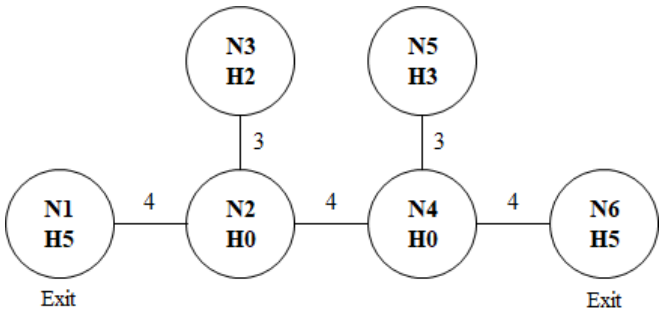


Figure 3.3: Probability Transition Example

Once the shooter's probability of visiting a specific node at a given time t_i is rolled out over the planning time horizon, a risk metric informed by an estimate of the shooter's line-of-sight is calculated. The line of site from each node is pre-computed from the geometric layout of the building. Experts have suggested that the range of specific weapons that the shooter is carrying is less important than being in the line of sight of said weapon i.e. the threat from a rifle accurate out to 200m is the same as the threat from a handgun accurate out to 25m if the victim is in an enclosed hallway at a distance of 50m. The most important aspect in such a scenario is the need to stay out of line of sight. We approached this aspect of harm by smearing the maximum probability of a given node across all the nodes that are in its line of sight at any give time t_i.

As can be seen in Table 3.2, the shooter has line of sight on nodes $node_2$, $node_4$, and $node_6$. These nodes have the probability of encountering the shooter, physically or within the line of sight and range of the weapon, smeared up to the greatest value among the nodes that are currently within his line of sight. The process continues as location probability propagated through time as above. The harm array in Table 3.2 shows the difference between physically encountering the shooter as in Table 3.1 and simply being within his range.

3.4.3 NHSMDP Reward Structure

The $reward_{max}$ element in the reward function is the large reward received when an evacuee reaches a terminal state - a positive $reward_{max}$ by escaping, or a negative $reward_{max}$ by becoming a casualty due to proximity to the shooter's location. While the remainder of the reward function provides those interim rewards to help develop the actions and thus the

Table 3.1: Probable location of shooter at time=t.

Time in Seconds	0	1	2	3	4	5	6	7
N_1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
N_2	0.00	0.00	0.00	0.00	0.25	0.13	0.05	0.02
N_3	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.06
N_4	1.00	0.25	0.06	0.02	0.00	0.00	0.13	0.13
N_5	0.00	0.00	0.00	0.25	0.19	0.11	0.06	0.03
N_6	0.00	0.00	0.00	0.00	0.25	0.19	0.11	0.06
$E_{1\to2}$	0.00	0.00	0.00	0.00	0.00	0.06	0.09	0.11
$E_{2\to3}$	0.00	0.00	0.00	0.00	0.00	0.06	0.09	0.04
$E_{2\to4}$	0.00	0.25	0.31	0.33	0.09	0.09	0.10	0.14
$E_{4\to5}$	0.00	0.25	0.31	0.08	0.15	0.22	0.15	0.12
$E_{4\to6}$	0.00	0.25	0.31	0.33	0.08	0.15	0.22	0.31

Table 3.2: Probability of Harm by Shooter at time=t.

Time in Seconds	0	1	2	3	4	5	6	7
N_1	0.00	0.00	0.00	0.00	0.25	0.13	0.09	0.11
N_2	1.00	0.25	0.31	0.33	0.25	0.13	0.13	0.14
N_3	0.00	0.00	0.00	0.00	0.00	0.06	0.09	0.06
N_4	1.00	0.25	0.31	0.33	0.25	0.22	0.22	0.31
N_5	0.00	0.25	0.31	0.25	0.19	0.22	0.15	0.12
N_6	1.00	0.25	0.31	0.33	0.25	0.19	0.22	0.31

routing, the $reward_{max}$ really provides the push and pull that drives the evacuees away from the shooter and towards an exit. The $hardness_i$ element represents that relative safety of $node_i$ - rooms with lockable doors have larger hardness index while nodes in exposed hallways have very small values. Using the previous example of moving from $node_1$ to $node_{19}$, the $hardness_{19}$ - $hardness_1 \geq 0$ would result in a positive reward of 1. Allowing the agent to receive the positive reward, even when there is no increase in hardness is vital as it allows an evacuee to remain hidden in a safer room without penalty. The $exittime_i$ is the time to travel to the nearest exit from $node_i$. Moving from $node_2$ to $node_1$ brings an evacuee closer to the exit at $node_{52}$ and hence would result in a positive reward of 1. If none of these cases occur, the reward for the evacuee is -1 because they have not improved their situation by either going to, or remaining in, a safer node or by moving closer to an exit. We analyzed the affect that different values of $reward_{max}$ had on the performance of our evacuees through experimentation in the discrete event simulator. The results of the parameter exploration will be discussed further in the results section.

Algorithm 1: Reward Structure for Actions

```
Moving from node_i to node_j;
if node_j = death then
    reward = −reward_max ;
else if node_j = exit then
    reward = reward_max ;
else if hardness_j - hardness_i ≥ 0 then
    reward = 1 ;
else if exittime_j − exittime_i < 0 then
    reward = 1 ;
else
    reward = −1 ;
end
```

3.4.4 Evacuation Algorithm - Capacitated Value Iteration

Finite horizon value iteration on the NHSMDP approach results in a single optimal policy for the evacuees to follow during the active shooter situation. However, the optimal policy, or route in our case, has capacity constraints that limit the number of evacuees who are able

to enact that policy at a given time. As the routes approach the limits of capacity creating bottlenecks, additional routes need to be considered.

The Capacity-Constrained ASTERS (C-CASTERS) algorithm described in this paper is an extension of the NHSMDP based optimization, in which the possible actions available in each state are dynamically reduced as the available capacity for the action reaches zero.

Algorithm 2: C-CASTERS Pseudo Code

Input : Occupancy: $E = \sum e_i \; \forall \; node_i \in N$
Shooter's Location: $node_s$
Node Order: $OrNo$ = list of nodes $\in N$
ordered by proximity to an exit and
then by proximity to $node_s$

Output: A set of optimized routes $\forall \; node_i \in N$ that accounts for all evacuees, E, in N

```
while E > 0 do
    for route_i ∈ route_max do
        for node_j ∈ OrNo do
            Conduct Value Iteration;
            Obtain Optimal Policy with maxsend;
            for t ← 1 to 30 do
                Reserve maxsend capacity along route through time;
                Update transition probabilities affected by maxsend;
            end
        end
        Subtract maxsend from e_j;
    end
    Assign route_i to evacuees;
    Subtract e_j from E;
end
```

There are three inputs into the C-CASTERS algorithm: the current occupancy of the graphical environment, the shooter's known location, and an ordering of all of the nodes in the graphical environment, $OrNo$. Each node has a number of evacuees occupying it at the start of the algorithm, e_i, such that $0 \leq e_i \leq e_{max}$ where e_{max} is the maximum occupancy of $node_i$ at any time. $E = \sum e_i$ is the total number of evacuees at a given time, which changes as evacuees escape or encounter the shooter. The second input is the shooter's last known location identified by an assumed monitoring system. The final input is the ordered list of nodes, $OrNo$, that establishes priority in two stages, first based on their proximity to an exit

and then further ordered based on proximity to the shooter with nodes closer to the shooter's position given priority. By ordering the nodes closest to the exits first, we are able to clear those nodes of evacuees and prevent mass crowding that would slow the overall evacuation process. Further, the secondary ordering based upon proximity to the shooter ensures that those closest to the shooter receive their routes with their capacity reserved before those that are farther away. In simulation, the crowding results of this ordering outperforms other approaches that do not take these factors into account.

The C-CASTERS algorithm initializes the transition probabilities based upon the shooter's known location and begins with the first set of routes, $route_1$. Starting at the first node in the ordered list of nodes with $e_i \geq 0$, the algorithm conducts the value iteration and determines the optimal policy, in this case the optimal route, for $node_i$. The algorithm also determines the maximum amount of evacuees that can be sent along that optimal route, $maxsend$, which is determined by the smallest bottle neck along the optimal route. For instance in Figure 3.1, given that the available capacities on $edge_{2,1}$ and $edge_{1,52}$ are 20 and 4, respectively the $maxsend$ along the optimal exit path $[2, 1, 52]$ from node $N2$ would be 4. Below is an example table showing how the capacity along the route $[2, 1, 52]$ changes over time.

For reference, the maximum capacity of $node_1$, $node_2$, $node_{52}$, $edge_{2\to1}$, and $edge_{1\to52}$ are 20, 20, 999, 20, and 4 respectively. The $maxsend$ value of 4 is reserved along the route through time changing the available capacity (Table 3.3). The remaining available capacity can be used for other policies as they are determined. For instance, according to Table 3.3, there is no more additional capacity on $edge_{1\to52}$ at time $t = 5$ so any other evacuees in $node_1$ at $t = 5$ would not be able to move to $node_{52}$ in the first second. The time to go through the door and move to $node_{52}$ from $node_1$ is 3 seconds, but the capacity on that edge is used for the first second of the 3 second travel time. As indicated earlier, the rate of travel through the doorway is two people per second, but if the travel takes longer than one second, then the edge needs to be opened for other evacuees even though the other travelers are still there. The assumption is that when moving through a doorway, the doorway is only blocked for the first second of travel and is then open for the remainder of the evacuee's travel time. The example below demonstrates how multiple routes from a single node provides option

Table 3.3: Capacity of Nodes and Edges Accounting for *maxsend*

Time in Seconds	0	1	2	3	4	5	6	7
N_1	20	20	20	20	16	20	20	20
N_2	16	20	20	20	20	20	20	20
N_{52}	999	999	999	999	999	999	995	999
$E_{2\to1}$	20	16	16	16	20	20	20	20
$E_{1\to52}$	4	4	4	4	4	0	4	4

for staging the evacuation. For example, the third route computed from Node 19 suggests waiting for 3 seconds before proceeding to exit. These 3 seconds are vital for suppressing the crowding near the exit.

- Route 1: [19, 1, 52]
- Route 2: [19, 19, 1, 52]
- Route 3: [19, 19, 19, 1, 52]
- Route 4: [19, 19, 19, 19, 1, 52]
- Route 5: [19, 19, 19, 19, 19, 1, 52]

After reserving the capacity along the route, the algorithm would then move onto the next node in the ordered list and repeat the value iteration, optimal policy determination, and capacity reservation until each node with $e_i \geq 0$ has an optimal policy/route for the evacuees. These initial routes are assigned to the *maxsend* number of evacuees and E, the sum of all evacuees without an optimal policy, is updated by subtracting those evacuees who have now been assigned a route. The algorithm restarts with a second round of routes for each *node* that has $e_i \geq 0$ and continues until every nodes $e_i = 0$. Once all evacuees have been assigned capacity constrained routes, the simulation will resume with the new routes for the evacuees to follow.

3.5 Experimental Design

In order to conduct experiments and test the performance of our algorithm against that of other approaches we used a discrete event simulation package in python that provided us the flexibility to investigate all the important effects of capacity on the efficiency of the evacuation protocol. The experimental design section follows where the simulation environment and the actions of the agents and shooter are described along with an examination of the key parameters, and a comparison with alternate evacuation algorithms.

3.5.1 Simulation Environment

The simulation environment utilized was a discrete event simulation package in Python called SimPy. SimPy allows us to create many agents, both shooter and evacuees, with asynchronous updates, i.e. each take turns choosing actions and moving through the graph. The capacity constraints in nodes and along edges are accounted for by not allowing an agent to complete its chosen action if capacity occupation has been reached at any point along the chosen route.

Each node and edge within the graphical environment is treated as a resource. Each of these resources has a finite capacity that cannot be overwhelmed due to the token taking mechanism that prevents too many users of a given resource. Each resource has a number of resource tokens, based upon their maximum capacity, which are requested and given to evacuees in a FIFO order. For example, an evacuee whose plan indicates movement from $node_1$ to $node_2$ would check to see if there is available capacity on $edge_{1\to 2}$. If there was available capacity, they would turn in their resource token for $node_1$ in exchange for an $edge_{1\to 2}$ token which they would hold until they reached $node_2$ where they would then turn in their $edge_{1\to 2}$ token and request their $node_2$ token. If $edge_{1\to 2}$ was already at full capacity, then they would remain in $node_1$ until there was available capacity on $edge_{1\to 2}$ and follow the same procedure above.

The simulations are run for 300 seconds of simulation time with the python package checking each agents actions at every second and ensuring that everything occurs in the

appropriate order. Movements for each evacuee and the shooter are tracked for the entirety of the simulation.

3.5.2 Shooter Actions

We assume that the shooter's plan or motivation is unknown prior to or during an occurrence. In the context of simulation, the shooter's actions are determined by first selecting a random target node, followed by the Dijkstra's shortest path from his current location to the target node. The shooter operates outside of the capacity constraints as anyone trying to share an edge or node with the shooter is automatically considered a casualty. The estimate for casualties is assumed to be deterministic based upon the proximity to the shooter and being within his line of sight. This is not realistic, but is an effective metric for consistent evaluation.

When the shooter reaches his target node, he waits for five seconds, chooses a new, never visited target node from the available remaining rooms, and begins moving. This process continues until all evacuees have either escaped, become casualties, or if the 300 seconds have passed. For each set of parameter combinations, we perturbed the initial conditions and used random seeds to run 20 simulations.

3.5.3 Evacuee Actions

The specific actions available to evacuees were discussed earlier in the value iteration preparation of the algorithm. We assume that the evacuees completely trust the algorithms and obey the instructions without hesitation or regard for their safety. This is an unrealistic but necessary assumption to avoid the variability associated with indecision, hesitation, or disobedience when comparing the performance of all the algorithms. Each evacuee is assigned a route at specific times, for example a route from $node_2$ could be [2, 1, 19, 19, 19, 19, 19, 19, 19, 19, 1, 52]. The evacuees follow the instructions until they receive a new route. In essence, the evacuee agents do not have any action choices as the algorithm acts as a single controller for all evacuee actions.

3.5.4 Graph Environments

We elected to use two different graphical environments for this work: a single level school that is acyclic and a topologically different school layout that is characterized by the presence of cycles and loops in the graph structure.

Acyclic School

The Acyclic School graphical environment consists of 55 total nodes as seen in Figure 3.4a. The node types are broken down as follows: Nodes 1 through 18 are hallway nodes, Nodes 19 through 51 are rooms nodes, and Nodes 52 through 55 are exit nodes. This graphical environment is useful as it does not provide cycles or loops where an agent can play cat and mouse with the shooter. It is a very straight forward layout that provides an excellent baseline understanding of the capabilities of the algorithms.

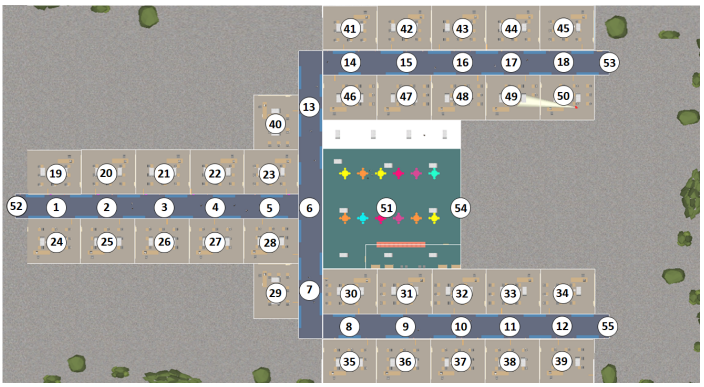
Cyclic School

The Cyclic School Floor is a graphical environment with two exits and 70 total nodes as seen in Figure 3.4b. The node type breakdown is as follows: Nodes 1 through 29 are considered hallway nodes, Nodes 30 through 68 are considered rooms nodes, and Nodes 69 and 70 are the exits. This graphical environment plays a key role in our analysis for two specific reasons. The first reason for its inclusion is the lines of sight down the hallway presenting dangerous possibilities when the shooter is in the hallway. The second, and more important reason, is that there are several connectors between the long hallways that allow both the shooter and evacuees to quickly move from one hallway to the other presenting the evacuees with a plethora of options, but also making the shooter's movements much more unpredictable and dangerous.

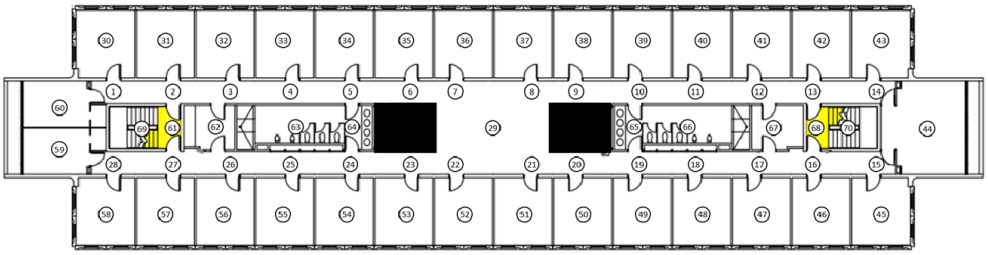
3.5.5 Parameters and Simulation Inputs

Evacuee Distribution

How the evacuees are distributed at the onset of the event was the first of two important parameters studied. The evacuees were either distributed at the start of the simulation



(a) Acyclic School



(b) Cyclic School Floor Plan

Figure 3.4: Two Different Graphical Layouts for Simulation Comparison

in only the rooms or uniformly distributed in both rooms and hallways. These two cases correspond to the typical bimodal distribution for movement within a school-day:

- **Rooms and Hallways:** During recess or in between classes, there are students in both rooms and hallways milling about,
- **Rooms Only:** During class times all the students are typically in classrooms.

Each spawn location, specific to the type of simulation was initialized with ten evacuees. For example, the Acyclic School graphical environment had 51 total non-exit nodes, Nodes 1 through 18 being hallway nodes and 19 through 51 rooms nodes. So, for the **rooms Only** case, there would be 330 evacuees with ten each initialized in Nodes 19 through 51. For the **Rooms and Hallways** simulation, we considered 510 evacuees with ten each in Nodes 1 through 51.

Shooter Spawn Location and Target Node

The other primary parameter of interest and effect was the location at which the shooter was initialized, representing the first sighting of the shooter in a real-life scenario. We chose the specific starting locations in each graph to capture a variety of situations to ensure that we adequately covered all situational possibilities. For the Acyclic School, we chose nine separate locations for the shooter to spawn: $node_2$, $node_6$, and $node_{11}$ for hallway nodes, $node_{20}$, $node_{38}$, and $node_{51}$ for room nodes, and $node_{52}$, $node_{54}$, and $node_55$ for exit nodes. For the Cyclic School, we chose eight spawn locations: $node_2$, $node_{16}$, and $node_{29}$ for hallway nodes, $node_{30}$, $node_{37}$, $node_{44}$, and $node_{59}$ for room nodes, and $node_{70}$ for the exit node.

Our assumption is that the primary target for the shooter will be a room within the building. With this in mind, each target node that the shooter chooses is randomly chosen from all the available unvisited rooms within the graphical environment. The proximity of the rooms to the shooter's current location increases the probability that they will be randomly chosen when a new target node is required. For instance, if the shooter's target node was 19 and he reached it, then the room nodes in the western hallway of the Acyclic School would have a higher probability of being chosen than those in the northern or southern

wing of the school. The randomness of the chosen targets, both initially and throughout the simulation help to ensure that the movements of the shooter do not become predictable.

3.5.6 "Naive ASTERS Evacuation algorithm"

The first algorithm that will be compared to the C-CASTERS algorithm is the naive ASTERS evacuation algorithm [47]. The naive ASTERS algorithm was based on the assumption that there is always enough flow capacity through each path point that crowding and bottle necking is never an issue. The naive ASTERS algorithm conducts the value iteration in the NHSMDP framework and determines a single optimal route from each node and shooter location combination. However, due to limitations in movement capacity through doorways, in the version of the problem under consideration, only two out of the ten evacuees would be able to execute that route immediately while the other eight would need to wait their turns (a few seconds) to execute it. This waiting adds up and cumulatively results in crowding at certain bottle-neck nodes.

3.5.7 "Naturalistic Response"

The Natural Response algorithm is based upon rules created by domain experts and the general guidance of the "Run, Hide, Fight" protocol. The Natural Response can be described in a series of steps:

- Check the distance between evacuee position and shooter's last known position
- If the distance is $\geq D_{threshold}$ path lengths, use Dijkstra's to move to nearest exit away from the shooter
- If the distance is $< D_{threshold}$, either move to nearest room or remain in room

The Natural Response algorithm demonstrates the actions of an evacuee in a real world scenario in a logical, simplified manner: hide until the shooter is far enough away and then move to the exit. $D_{threshold}$ was selected to be 7 based on the best performance achieved in a parameter sensitivity study. While this may keep evacuees safe, it also prevents many of

the evacuee from escaping at the same time. Also, just like the naive ASTERS, this scheme suffers from overcrowding, since everyone in a given node will attempt to follow the same route and leading to bottlenecks near exits. This will be further examined in the results section.

3.6 Results

In this section, we will systematically compare the overall performance of C-CASTERS versus the two other algorithms, and then discuss parameters of interest and their effects on the performance of each of the algorithms. For the simulation results, 20 randomized runs of 300 seconds were performed with each possible combination of parameter values to generate statistically significant data. For each of the results sections, we conducted simulations in both the Cyclic School and Acyclic School graphical environments, varied the starting locations with evacuees initially distributed in either rooms and halls or rooms only, and varied the initial spawn location for the shooter within the graphical environment with a randomized initial target node.

3.6.1 Effect of Reward Shaping

The reward function in our model is defined by several parameters, including $reward_{max}$, which represents the reward for escaping a dangerous situation, and $penalty_{max}$ representing encountering the shooter. The balance between *reward* and *penalty* dictates the trade-off between moving closer to an exit and moving to a safer node, often necessitating conflicting actions from the agents. In scenarios where capacity does not limit movement, agents prioritize short-term safety by avoiding the negative consequence associated with $penalty_{max}$. However, over time, the incentive of $reward_{max}$ at an exit provides agents with the impetus to leave their rooms and move towards the exit, leading to efficient outcomes in systems that are not capacity-constrained. In contrast, when crowding and wait times become concerns, agents need to prioritize more long-term caution in making efficient decisions. Given that such high-stakes events often unfold rapidly, we posited that avoiding *penalty*, i.e., remaining away from the immediate threat, is as critical as achieving *reward* by reaching

an exit. Therefore, we set the rewards and penalty values equal in absolute magnitude, that is, $penalty_{max} = -reward_{max}$, and focused our efforts on exploring the implications of this balance.

In the figures in the following sections, we examined the effect of varying values for the magnitude of $reward_{max}$ and $penalty_{max}$ on the number of casualties observed in the discrete-event simulation. Our goal was not only to establish the best $reward_{max}$ value for each of the possible combinations of parameters, but also to attempt to identify and examine common themes that occur naturally in the simulation results.

Cyclic School Parametric Reward Shaping: Figure 3.5 displays the cumulative averages of casualties of the 20 runs over time. In this specific simulation, the evacuees were initially distributed in the rooms and halls in the Cyclic School Floor. As can be seen in Figure 3.5, in the initial 20 seconds, the number of casualties is completely independent of the evacuation strategy, since it can be argued that the initial loss of lives when a shooter starts firing cannot be prevented by optimized escape plans. However, after the first 20 seconds, an optimal risk-taking behavior (moderate values of $reward_{max}$) can be distinguished from the "too safe" and "too risky" plans. Interestingly, both the extreme low values, $\{6, 8\}$ and

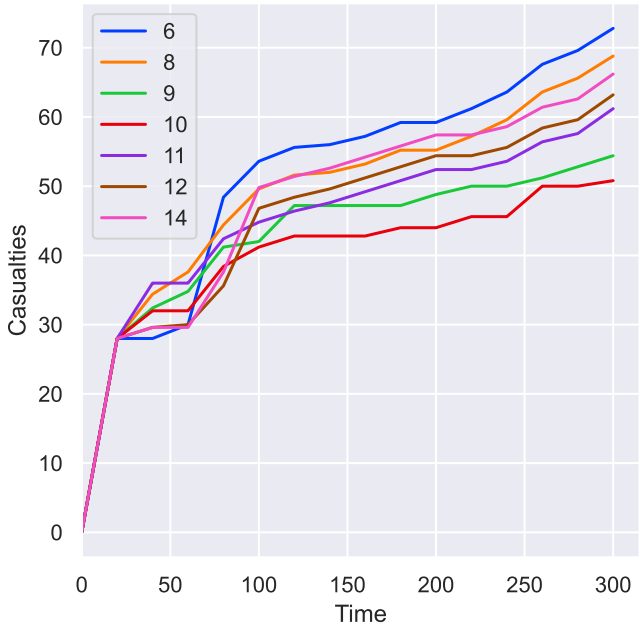


Figure 3.5: Escape Reward Value Affect Over Time - Cyclic School Floor: Rooms and Halls Distribution

the extreme high values $\{12, 14\}$, provide the worst total casualty rates. Evacuees who are too risk averse, corresponding to low $reward_{max}$ values of $\{6, 8\}$, move less, even when it is likely safe to do so, which can result in greater total casualties. Aggressive risk-taking plans, corresponding to $reward_{max} \in \{12, 14\}$, incentivize escape attempts even when it is less safe to do so. This also results in greater number of casualties. It is interesting to note that the high-risk behavior results in more casualties earlier in the simulation, but then also leads to more escapes as the time moves beyond the first minute. This is in some way similar to the outcome achieved in the currently practiced "run-hide-fight" protocol, where "running" towards safety is prioritized whenever possible. This research suggests that a more balanced outlook may be safer with medium values of $reward_{max} \in \{9, 11\}$) achieving the optimal balance between staying safely away from the shooter while also attempting to move towards the exit when the conditions are more ideal.

Figure 3.6 displays the results when the evacuees are initially distributed in the relative safety of the rooms, in contrast to hallways, when the shooter is first located. Beyond an overall lowering of the number of casualties due to the initial protection offered by the rooms, the trends for this situation is very similar to that discussed before - an optimum level of risk

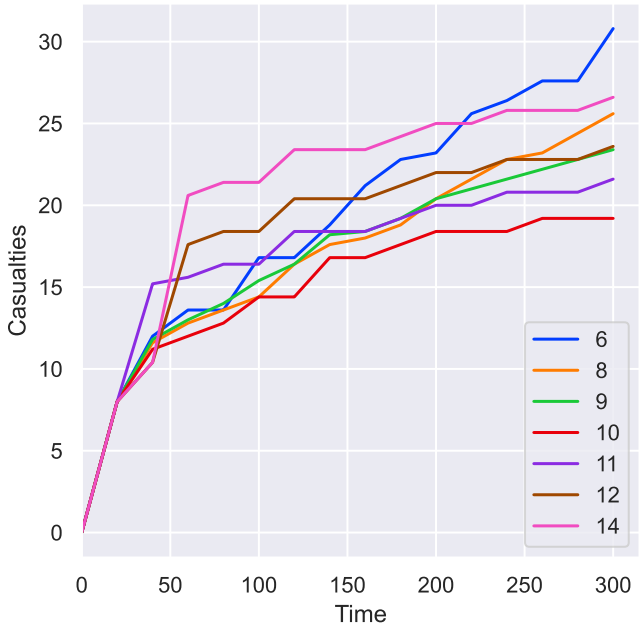


Figure 3.6: Escape Reward Value Affect Over Time - Cyclic School Floor: Rooms Distribution

tolerance ($reward_{max} = 10$) leads to the best outcome compared to both overly aggressive as well as passive responses.

Figure 3.7 displays the average total casualties at the conclusion of the event (5 minute duration) for the different evacuee distributions in the Cyclic School Floor. Both distributions display a well-defined optimum around a $reward_{max}$ value of 10. The optimum balance between seeking safety and seeking escape, achieved with a $reward_{max}$ value of 10 is specific to the specific graphical environment, but independent of the distribution of evacuuees, the initial location of the shooter in the graph and the randomness of the shooter's movement.

Acycilic School Parametric Reward Shaping: Figure 3.8 displays the results of the same parameter combination and runs for the Acyclic School as those conducted in the Cyclic School Floor environment. These graphical environments are topologically distinct and the results are indicative of that fact. The Acyclic School contains more exits than the cyclic school floor. Moreover, the lack of possible cycles connecting the hallways significantly changes the topology of the graph. Figure 3.8 shows the results for the scenario when the evacuees are distributed in both the rooms and hallways. While the resulting performance of

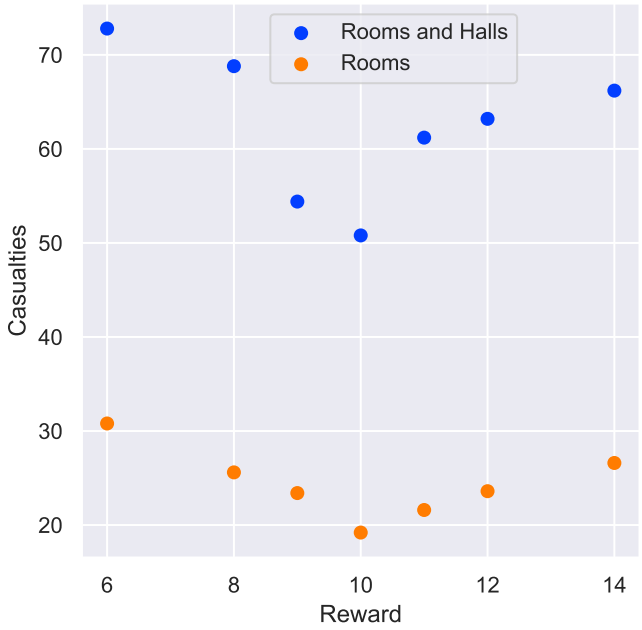


Figure 3.7: Escape Reward Value Affect at 300 Secs - Cyclic School Floor

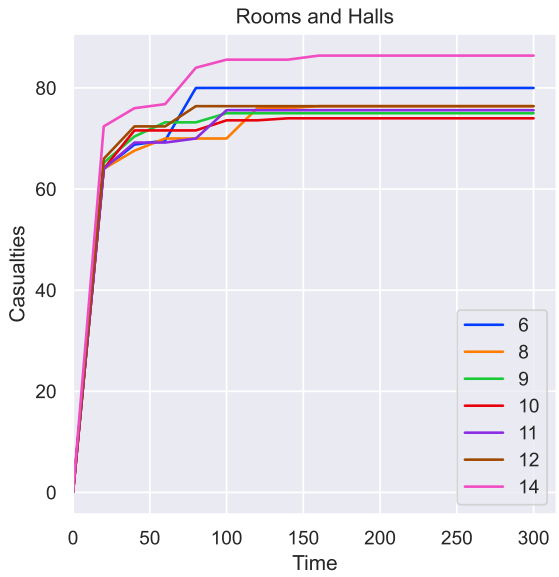


Figure 3.8: Escape Reward Value Affect Over Time - Acyclic School: Rooms and Halls Distribution

differing $reward_{max}$ values is very similar, it is important to note that the most risk taking behavior associated with a $reward_{max}$ of 14 performs the most poorly, especially early in the simulation demonstrating that breaking for an exit too soon is detrimental to survival. While other $reward_{max}$ values outperformed it within the first 100 seconds of simulation, a $reward_{max}$ value of 10 was again the best performing resulting in the least amount of casualties during the simulation.

The results of distributing the evacuees in rooms only at the start of the simulation is displayed in Figure 3.9. The real differences between the $reward_{max}$ values occurs in the first minute of the simulation with each of the values producing very different results. Interestingly, Figure 3.9 displays the only occurrence where 10 was not the best performing reward value among either graphical network or parameter combinations. The best performing values when the evacuees were distributed in rooms only were 8, 9, and 10 in order. While this result may cause one to question, in this case, whether smaller value of the reward outperformed the rest, you can see that a reward value of 6 was still poorer performing than all but 12 and 14. In a simpler environment such as this with fewer distinct pathing options, it is clear that being more risk averse is a better approach than being

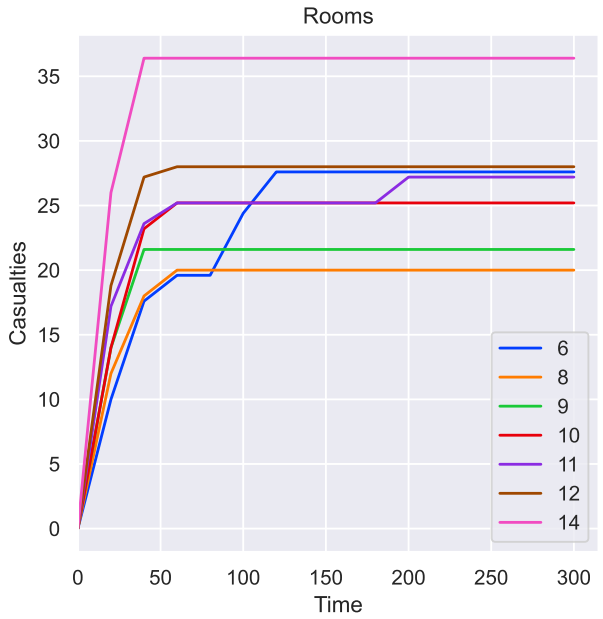


Figure 3.9: Escape Reward Value Affect Over Time - Acyclic School: Rooms Distribution

risk taking. In this environment, the balance between escaping and hiding is struck leaning towards hiding as it provides the best safety outcomes.

Just like Figure 3.7, Figure 3.10 shows the average casualties at 300 seconds for each of the reward values and each of the evacuee distributions. The rooms and halls distribution still has the concave spread centered on 10, but, for the first time, the rooms only distribution showed a concavity centered on 8. This demonstrates to us that, when starting in a safe position in the simpler layout of the Acyclic School, it is more effective to err on the side of safety and hiding than moving to escape through an exit. While these results differ slightly from those of the Cyclic SChool Floor, they do support the contention that striking the balance between hiding and escape provides the most effective strategy even if the layout requires a behavior leaning more towards safety than escape.

Based upon these results, we elected to use the $reward_{max}$ value of 10 for all of the remaining parameter exploration runs contained in this work. There may be benefit in future work to adjust the reward function based upon certain assumptions and circumstances. For example, if we assume that an event is only going to last for 60 seconds, the $reward_{max}$ may potentially need to be smaller as the smaller values in general did well early in the parameter exploration and then worse as time went on. Perhaps a more complicated building may

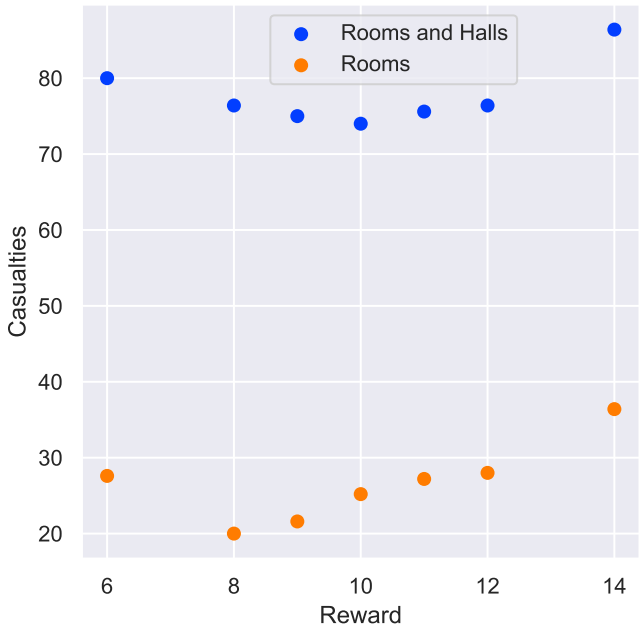


Figure 3.10: Escape Reward Value Affect at 300 Secs - Acyclic School

provide more hiding and movement opportunities than either of these graphical environment and would therefore encourage a larger $reward_{max}$ value. Tuning the $reward_{max}$ value to the building and the situation is the most likely way to improve the performance of the routing and the safety outcomes of the evacuees.

3.6.2 Aggregated Performance

The performance of the algorithms within the two graphical environments are affected by various factors which will be discussed in subsequent results sections. The analysis of the aggregated performance of the algorithms takes all of the various parameter combinations into account without isolating specific values or combinations. Table 3.4 displays the overall results for each of the algorithms and shows the parametric combinations of features with algorithmic specific performance based upon the average number of casualties and escapes in raw numbers and by percent of the total evacuees as well as the total number of seconds all of the evacuees spent in the shooter's line of sight. The casualties entries are the average. The three primary metrics that we captured during simulation were the number of casualties, the number of escapees, and the time spent in the line of sight of the shooter, whether the

Table 3.4: Aggregated Tabular Simulation Results Across Assets and Algorithms: This table shows the performance of each algorithm for each combination of parameters for both the Acyclic School and Cyclic School assets. The values in bold represent the best performance amongst all the algorithms for each possible parameter combination.

Shooter Spawn			Exit		Hall		Room	
Evacuee Distribution			Rooms	Rooms and Halls	Rooms	Rooms and Halls	Rooms	Rooms and Halls
Acyclic School	C-CASTERS	Casualties	13.4(4.06%)	**25.67(5.03%)**	**20.07(6.08%)**	**27.4(5.37%)**	23.6(7.15%)	**35.07(6.88%)**
		Escapes	316.6(95.94%)	**461(90.39%)**	**309.93(93.92%)**	**429.27(84.17%)**	**306.4(92.85%)**	**464.93(91.16%)**
		LOS	**44.63**	258.83	**75.63**	**254.8**	**45.67**	**212.1**
	Naive ASTERS	Casualties	**8.95(2.71%)**	51.5(10.1%)	23.91(7.24%)	64.71(12.69%)	**18.53(5.62%)**	93.33(18.3%)
		Escapes	**317.59(96.24%)**	447.67(87.78%)	306.09(92.76%)	404.45(79.3%)	301.33(91.31%)	418.91(82.14%)
		LOS	45.03	**217.72**	99.08	290.69	67.59	217.72
	Natural Response	Casualties	56.76(17.2%)	84.23(16.52%)	51.91(15.73%)	71.57(14.03%)	65.64(19.89%)	122.28(23.98%)
		Escapes	269.57(81.69%)	402.01(78.83%)	277.65(84.14%)	384.68(75.43%)	253.84(76.92%)	377.29(73.98%)
		LOS	405.97	789.83	225.97	452.69	307.28	542.01
Cyclic School	C-CASTERS	Casualties	**18.8(12.05%)**	**39.6(14.56%)**	**19.17(12.29%)**	**25.73(9.46%)**	**16.35(10.48%)**	**31.43(11.55%)**
		Escapes	79.8(51.15%)	140.2(51.54%)	83.97(53.82%)	142.2(52.28%)	83.3(53.4%)	151.08(55.54%)
		LOS	**23.4**	**100.8**	**33.77**	127.9	**27.5**	**55.5**
	Naive ASTERS	Casualties	27.36(17.54%)	55.56(20.43%)	27.16(17.41%)	37.15(13.66%)	22.84(14.64%)	50.28(18.49%)
		Escapes	106.52(68.28%)	182.04(66.93%)	110.39(70.76%)	187.61(68.98%)	118.58(76.01%)	198.01(72.8%)
		LOS	102.64	246.08	78.53	259.35	61.79	226.86
	Natural Response	Casualties	30.8(19.74%)	53.48(19.66%)	25.92(16.62%)	38.8(14.26%)	26.57(17.03%)	53.26(19.58%)
		Escapes	**121.2(77.69%)**	**218.52(80.34%)**	**130.08(83.38%)**	**221.87(81.57%)**	**125.43(80.4%)**	**218.74(80.42%)**
		LOS	429.76	726.6	605.35	**1002.2**	406.9	655.86

evacuee was close enough to be harmed or not. Our objective was to minimize the number of casualties and the time in line of sight as law enforcement experts agree that minimizing it provides the highest probability of survival. For all of the remaining discussions, an evacuee was deterministically considered to be a casualty if they, at any instant of time, were within three path-lengths of the shooter and were within the shooter's line of sight. If they were in the shooter's line of sight but were farther than three path-lengths away from the shooter then they were considered to be in his line of sight only, but were free to continue their movement. Evacuees who escape are those that reach an exit node before being harmed by the shooter. We did not place significant emphasis on the escape metric because it does not necessarily equate to the best outcome. For example, an aggressive routing plan may allow 75% of the evacuees to escape while leading the remaining 25% to become casualties, while a less aggressive routing plan may only result in 60% escapes but also reduce casualties to 10% with the remaining 30% remaining safely in the building and away from the shooter until the emergency has passed.

Cyclic School Algorithm Performance: Figure 3.11 shows the combination scatter plot and histograms for the interactions between the number of casualties and the time spent in line of sight of the shooter for the Cyclic School Floor graphical environment. The

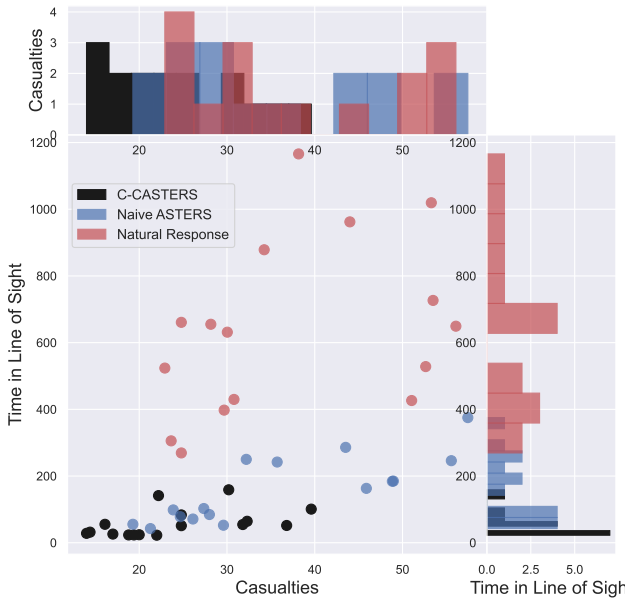


Figure 3.11: Aggregated Evacuation Effectiveness - Cyclic School Floor

casualty histogram displays how each of the three algorithms performed according to the casualties accumulated. The C-CASTERS algorithm outperforms the other two algorithms with an average of 24.01 casualties, while Naive ASTERS averages 35.52 casualties and Natural Response averages 37.36 casualties. The line of sight metric is the cumulative time spent in the shooter's line of sight for all of the evacuees throughout the simulation. The time in line-of-sight histogram shows the same result with C-CASTERS outperforming the others with a cumulative average of 58.83 seconds. Naive ASTERS comes in second with an average of 157.31 seconds, and the Natural Response algorithm, averaging 639.38 seconds is the worst performer. For the scatter plot, the best values are located in the lower-left and the worst are in the upper right quadrant. The plot demonstrates the way in which C-CASTERS outperforms the other two algorithms with consistently lower values for time in line of sight and fewer casualties and also lending support to the law enforcement point of view that decreased time in the shooter's line of sight results in fewer casualties.

Acyclic School Algorithm Performance: The Acyclic School graphical environment results are displayed in the same type of figure in Figure 3.12. Similar to the Cyclic School Floor environment, the C-CASTERS algorithm outperforms the other two with an average of 24.2 casualties with Naive ASTERS having an average of 43.49 casualties and the Natural

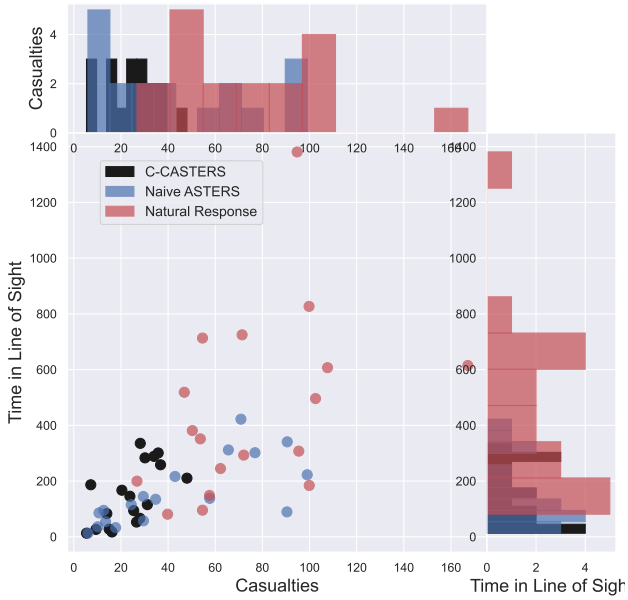


Figure 3.12: Aggregated Evacuation Effectiveness - Acyclic School

Response having 75.40 average casualties. The simpler layout of the Acyclic School broadens the differences between the algorithms and highlights the benefits of a capacity constrained approach. The C-CASTERS and Naive ASTERS were very close in terms the line-of-sight metric with C-CASTERS averaging 148.61 seconds and Naive ASTERS averaging 156.30 seconds. Natural Response was again severely outperformed averaging 453.96 seconds. Similar to the Cyclic School Floor, Figure 3.12's scatter plot shows the concentration of the C-CASTERS in the lower left quadrant with the Naive ASTERS stretched a bit more along the casualties axis and the Natural Response having an even greater spread. Accounting and planning for capacity constraints greatly improved the safety outcomes for the evacuees in both graphical environments and provided promising insights for future development.

3.6.3 Effect of Shooter's Start Location

The location where the shooter is first detected and/or reveals himself by using his firearm, can have a drastic effect on the performance of any routing algorithm. We chose several locations in different regions of the graphical environments to ensure that we captured all possible situations of shooter movement through the graphs. For each of the 20 runs for a given scenario, the shooter was spawned in the same location but was given a randomly chosen target node that he moved toward, thereby creating randomness and difference between each run. For the sake of this analysis, we categorized the shooter's starting locations into one of three possibilities: Exits, Halls, or Rooms.

Cyclic School Shooter's Start Location Effects: Figure 3.13 displays violin plots of the number of casualty for the different starting position categories of the shooter in the Cyclic School Floor graphical environment. Each of the element of the violin plots in essence is a vertical, smoothed histogram that identifies the mean and the quartiles. Longer, stretched "violins" are indicative of a large spread with greater variance of the given data while shorter, fatter "violins" are indicative of of densely distributed data that has smaller variation. The C-CASTERS algorithm provides the smallest mean and the shortest, thickest "violins" in all three shooter starting location categories. For all three categories, the C-CASTERS algorithm had more consistent results with smaller variance resulting in shorter "violins". Further, the quartiles were closer to the mean in the C-CASTERS performance

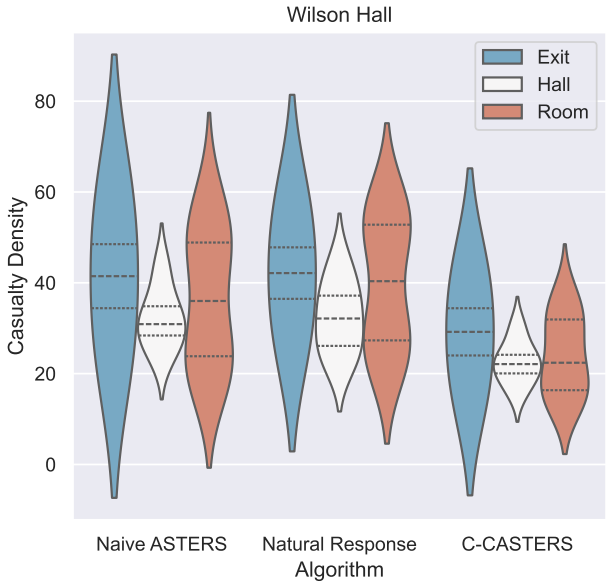


Figure 3.13: Shooter's Spawn Location Effect - Cyclic School Floor

than the other two algorithms indicating a more consistent performance over the 20 runs. Some themes common to all of the algorithms become apparent in Figure 3.13. The shooter starting in an exit is the worst case scenario for all of the algorithms and also has the greatest variance, longer "violins", meaning the number of casualties in each run can vary wildly. Further, a shooter starting in the hallway is the best case scenario for all of the algorithms and is also the most consistent with short, fat "violins". This is probably because in this case the algorithm benefits from the relatively higher confidence with which the shooter's movement can be predicted in the short term. The exits being centrally located makes the shooter's movement to either hallway equally likely and hence unpredictable.

Acyclic School Shooter's Start Location Effects: The Acyclic School graphical environment performance results for the different shooter starting locations are displayed in Figure 3.14. At first glance, it can be seen that the casualty distributions are less responsive to the shooter's start location in this graph compared to the cyclic graph. However, the C-CASTERS algorithm again performed best in each of the shooter start location categories, doing better than the other two algorithms in each situation. It may be worth remembering, that the Acyclic School environment contained almost twice as many evacuees as the Cyclic School Floor environment which explains the larger occurrences of casualties in the Acyclic

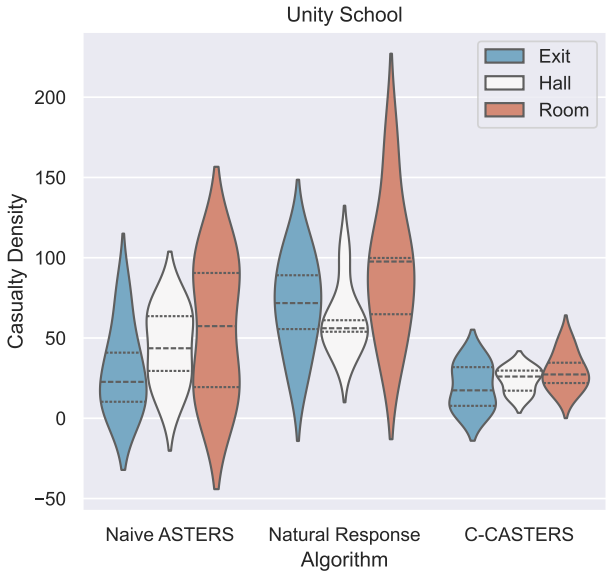


Figure 3.14: Shooter's Spawn Location Effect - Acyclic School

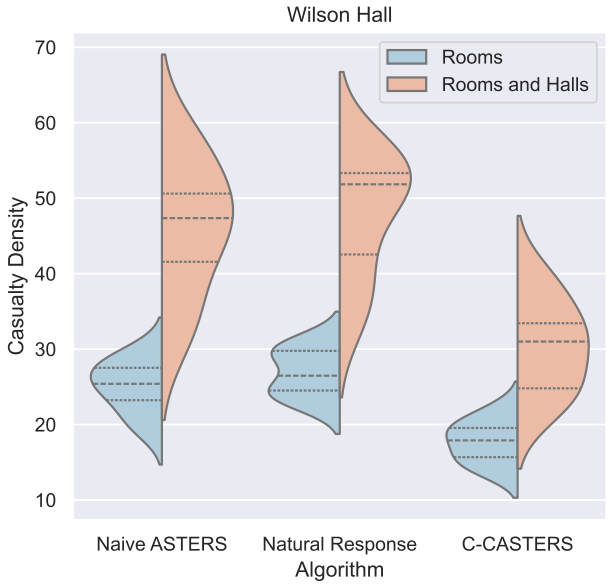
School results. The C-CASTERS algorithm was the best performing of the three routing algorithms across the two environments and the different shooter starting location categories.

It is very interesting to see that in the acyclic graph, no specific starting location could be identified that could qualify as "most" or "least" dangerous. The location of the three exits at the ends of the corridors assists in making predictable optimal choices for moving the evacuees. But at the same time, this illustrates the crucial role played by the capacity constrained optimization, since without that consideration, a large number of evacuees can be "caught" by the shooter near an exit.

3.6.4 Effect of Evacuee Distribution

As discussed in Section 3.5.5, to capture realistic scenarios in school evacuation planning, we modeled two distinct evacuee distribution scenarios for each simulated school environment. The two evacuee distribution scenarios are defined as follows:

Rooms Only Distribution: In this configuration, evacuees are spawned exclusively in the classroom nodes (30 to 68), with four evacuees assigned to each node. This setup reflects a scenario where students are in their respective classrooms during a class session. *Rooms and Halls* Distribution: In this scenario, evacuees are uniformly distributed across all nodes (1



4

Figure 3.15: Evacuee Distribution Effect - Cyclic School Floor

to 68), with four evacuees assigned to each node. This setup simulates a transition period when students are dispersed across both hallways and classrooms.

Figure 3.15 presents split violin plots representing the casualty distributions for all parameter combinations within the Cyclic School Floor graphical environment, stratified by evacuee distribution scenarios. The left (blue) side of each violin plot corresponds to the "Rooms Only" distribution, while the right (red) side represents the "Rooms and Halls" distribution. These plots provide a comparative visualization of the casualty distributions for each routing algorithm under different initial evacuee placements.

Performance Under the "Rooms Only" Distribution: In the "Rooms Only" distribution scenario, all algorithms demonstrated relatively consistent performance. The casualty distributions for each algorithm are narrow and dense, with interquartile ranges closely clustered around their respective means. Among the algorithms, C-CASTERS achieved the best and most consistent results, with an average of 18.15 casualties, Naive ASTERS followed with an average of 27.26 casualties. Natural Response performed the least effectively, with an average of 28.31 casualties. The comparatively small casualty range in the "Rooms Only" distribution underscores the advantage of evacuees remaining in classrooms, where they are initially positioned out of the shooter's line of sight. This

scenario highlights the importance of minimizing unnecessary movement when evacuees are already in relatively safe locations.

Performance Under the "Rooms and Halls" Distribution: The "Rooms and Halls" distribution presented a stark contrast, with significantly larger casualty ranges and less consistent algorithmic performance. In this scenario, evacuees were dispersed across both classrooms and hallways, exposing them to greater risk due to proximity to the shooter's potential path. The results for this distribution were as follows: C-CASTERS remained the best performer, with an average of 31.04 casualties. Naive ASTERS had a higher average of 47.62 casualties, while the Natural Response exhibited the poorest performance, with an average of 51.96 casualties. The increased spread in the casualty data for all algorithms in the "Rooms and Halls" distribution indicates the variability and heightened difficulty of routing evacuees when hallway movement is involved.

The results depicted in Figure 3.15 emphasize the critical role of initial evacuee distribution in casualty outcomes during active shooter events. The superior performance of C-CASTERS, particularly under the challenging "Rooms and Halls" distribution, highlights its efficacy in reducing casualties even in less favorable conditions. These findings demonstrate that reducing movement through hallways during active shooter events is a critical goal for effective evacuation planning. Strategies should prioritize keeping evacuees in safe locations, such as classrooms, and only initiate movement when it provides a demonstrable improvement in safety. This principle is particularly crucial in the design and implementation of routing algorithms aimed at mitigating casualties in high-risk scenarios.

Figure 3.16 shows the casualty results from runs in the Acyclic School graphical environment. Again, the "Rooms" distribution runs showed a significant shift in the order of algorithm performance compared to the "Rooms and Halls" with greatly reduced mean casualties and variability among all of the algorithms. Naive ASTERS was the best and most consistent performer with an average casualty value of 13.78, while the C-CASTERS followed closely achieving 15.53 average casualties and the Natural Response producing 53.16 average casualties in the "Rooms" distribution. The C-CASTERS algorithm significantly outperformed the other two algorithms in the "Rooms and Halls" case, with a very small and dense distribution of values. The C-CASTERS averaged 36.91 casualties while the

Naive ASTERS achieved 68.31 average casualties and the Natural Response turned in 101.78 average casualties. Again, across both graphical environments and evacuee distributions, the C-CASTERS algorithm was as good as or, more often, better than the other algorithms. The consistent theme across both graphical environments is the significantly better and more consistent outcomes produced when the evacuees start in the rooms regardless of algorithm.

3.6.5 Effect of Crowding

The effect of crowding and bottle-necking during an active shooter event is likely one of the most important elements related to the safety of the evacuees. The crowding of evacuees complicates the evacuation because it reduces the throughput of the affected nodes and slows down the movement of the evacuees. At the same time, it creates opportunities for potentially better routes to be used that avoid the larger crowds and would therefore be a faster means of exiting. In this paper we have explored a novel method for iteratively establishing routes for the evacuees and reserving their capacity along the route through time. Route reservation changes the possible best routes for other nodes in future iterations. The order in which routes are created and distributed to the nodes is a design choice. In this case, we have prioritized nodes based on their distance from the exit which leads to the

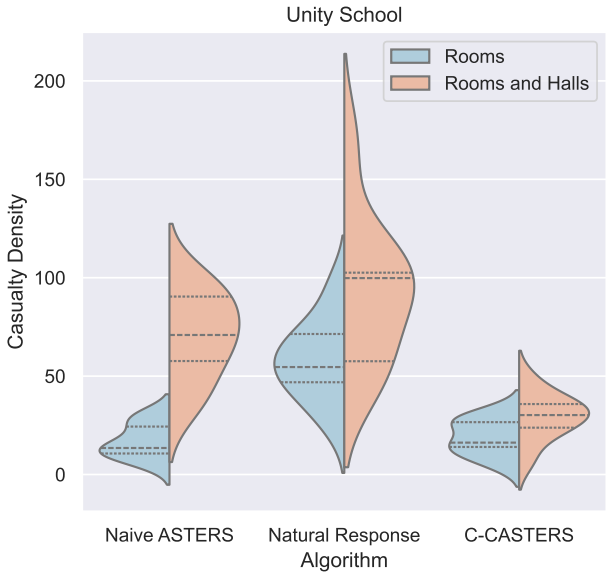


Figure 3.16: Evacuee Distribution Effect - Acyclic School

most efficient flow of evacuees. Below are the results of our node crowding analysis for each of the graphical environments.

Cyclic School Crowding Results: Figure 3.17 illustrates the impact of crowding in the Cyclic School Floor environment, with a focus on nodes adjacent to the exits: $node_2$, $node_{13}$, $node_{16}$, and $node_{27}$. The figure presents the average occupancy of each node, measured as the number of evacuees, over time from $t = 0$ seconds to $t = 75$ seconds. This time frame captures the critical period during which the majority of crowding occurs. Each graph contains three distinct curves representing different evacuation strategies: the blue line for C-CASTERS, the orange line for Naive ASTERS, and the green line for Natural Response.

To evaluate the effect of crowding on evacuee escape efficiency, we focus on spikes in node occupancy rather than steady-state values. Despite the relatively low number of evacuees in the Cyclic School Floor environment, crowding remains a significant factor. Even minor delays at exit-adjacent nodes can have life-threatening implications for evacuees.

The analysis reveals that, during each observed occupancy spike, C-CASTERS consistently achieves the lowest node occupancy values among the three routing algorithms. In many instances, its occupancy values are approximately half those of the Naive ASTERS and Natural Response algorithms. This superior performance is attributed to the iterative optimization strategy employed by C-CASTERS. By sequentially identifying optimal routes for nodes nearest the exits and farthest from the threat, and subsequently optimizing for nodes progressively farther from the exits and closer to the threat, the algorithm effectively reduces crowding at critical nodes. This reduction in crowding not only decreases casualties but also aligns with the improved performance metrics observed in preceding sections of the results.

The findings highlight the efficacy of C-CASTERS in mitigating the adverse effects of crowding during evacuation scenarios, thereby enhancing the safety and survival outcomes for evacuees.

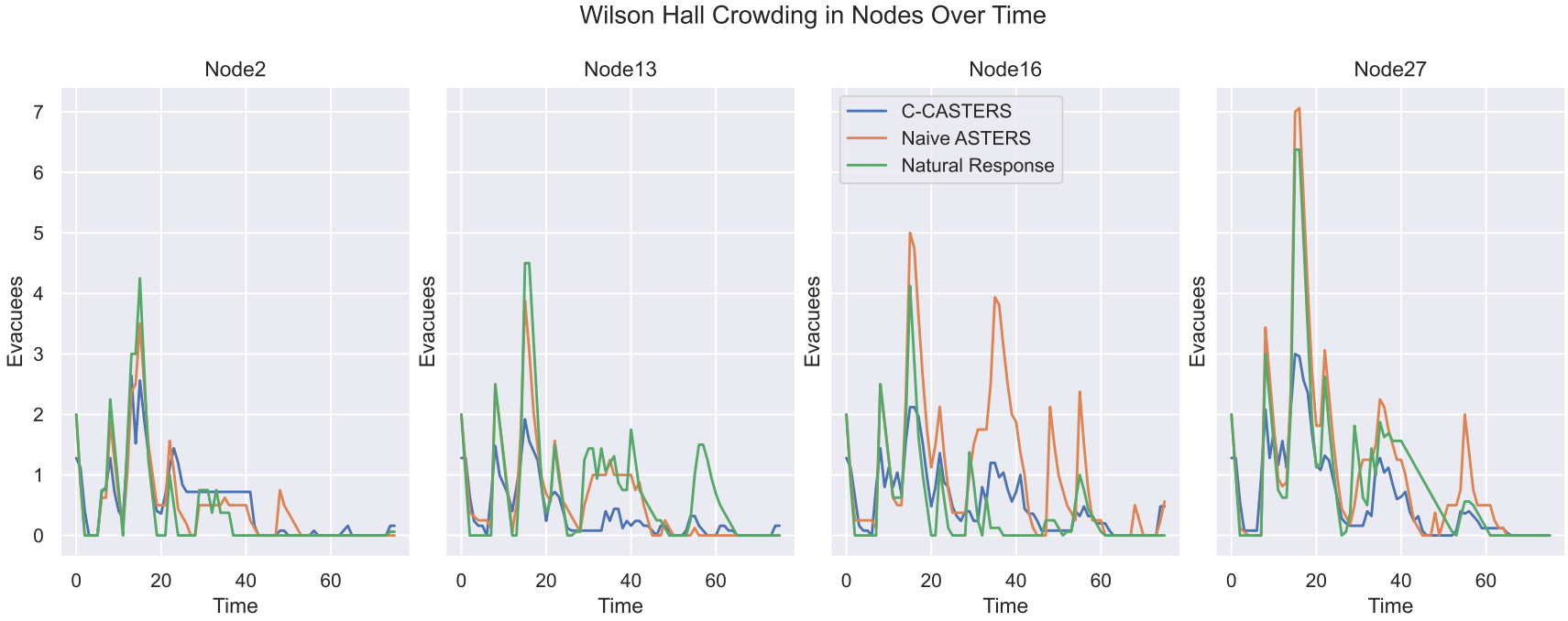


Figure 3.17: Node Crowding Effect - Cyclic School Floor

Acyclic School Crowding Results: The Acyclic School environment, characterized by significantly larger classroom sizes, hosts a greater number of evacuees compared to the Cyclic School Floor environment. The results of the simulations for this environment are presented in Figure 3.18. Similar to the previous analysis, the nodes of interest are those adjacent to the exits: $node_1$, $node_{12}$, and $node_{18}$. In this environment, the occupancy spikes observed in the node graphs are notably larger than those in the Cyclic School Floor environment. These spikes can result in delays of up to 15 seconds as evacuees wait to pass through the exits. Despite the larger crowd sizes, the C-CASTERS algorithm consistently achieves the lowest occupancy values during these spikes, often maintaining values less than half those observed with the Naive ASTERS and Natural Response algorithms. Notably, $node_1$ exhibits no significant crowding spikes under C-CASTERS, as the algorithm effectively regulates occupancy to prevent congestion and maintain steady movement.

These results underscore the effectiveness of C-CASTERS in addressing crowding challenges. By proactively managing evacuation routes and iteratively optimizing for nodes closest to exits, the algorithm minimizes congestion and reduces delays, even under high-occupancy conditions.

While the results highlight the promise of this approach, further refinements are necessary to enhance its performance. No evacuation algorithm can fully eliminate crowding in all scenarios, but the findings presented here demonstrate the potential of C-CASTERS to mitigate critical crowding issues and reduce the likelihood of tragic outcomes during emergency evacuations.

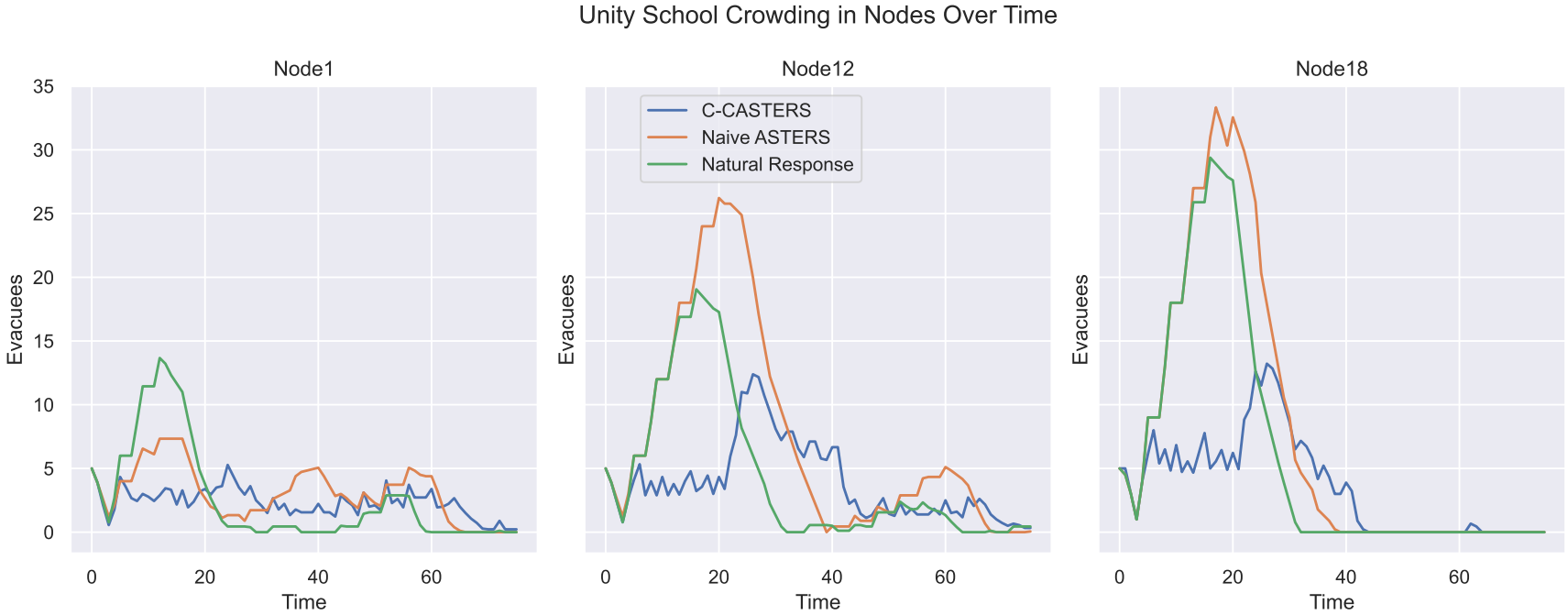


Figure 3.18: Node Crowding Effect - Acyclic School

3.7 Conclusion and Future Work

The study of capacitated transportation has far-reaching implications, and its relevance to emergency evacuations during active shooter scenarios cannot be overstated. Within this broader field, the focused investigation of routing individuals under such high-stakes circumstances demands sustained attention, innovation, and expansion. The importance of minimizing loss of life in these tragic events makes this an area of research worth pursuing. While we have addressed both non-capacitated and capacitated routing strategies, significant gaps remain in the literature, and meaningful solutions will require the collaborative efforts of a larger community of researchers.

Our research demonstrates the potential of reinforcement learning (RL) approaches to routing in active shooter situations. Specifically, the flexibility of C-CASTERS, which incorporates additional action options such as hiding in rooms or moving to safer nodes that are not necessarily exits, extends beyond the capabilities of traditional evacuation algorithms. Moreover, the algorithm's iterative identification of optimal routes, coupled with capacity reservation along these routes, substantially reduces crowding at key nodes, thereby preventing unnecessary fatalities.

3.7.1 Proposed Future Work

Behavioral Adjustments of Evacuees:
Current simulations assume evacuees follow routing instructions without hesitation. In real-world scenarios, trauma and panic may hinder evacuees' ability to act rationally or adhere to instructions. Modeling heterogeneous populations with varying probabilities of compliance, influenced by factors such as age, psychological state, and situational context, would add realism and complexity to evacuation models.

Integration of Deep Q-Learning (DQN) with Graph Neural Networks (GNNs):
Incorporating DQNs, which combine neural networks with Q-learning to handle high-dimensional state spaces, could facilitate real-time routing. Applying DQNs to GNNs, which are optimized for graph-based data, holds great potential for solving complex evacuation

problems. Training these systems in simulated environments and deploying them in real-time scenarios could lead to a practical solution for routing in active shooter situations.

Generalization of Learning Across Environments:
To enhance the applicability of the algorithm, it is essential to generalize the learning process beyond specific training environments. By conducting offline training on diverse graphical environments and transitioning to online learning in new scenarios, researchers can evaluate the transferability of learned strategies. Using graph characteristics to identify similarities between environments could further improve the algorithm's adaptability.

3.7.2 Conclusion

Our research highlights the potential of advanced routing algorithms like C-CASTERS in mitigating the devastating effects of active shooter incidents. While challenges remain, particularly in achieving real-time functionality, the proposed directions for future work aim to address these limitations and expand the scope of this critical research. By improving computational efficiency, incorporating behavioral realism, and exploring cutting-edge machine learning techniques, we can move closer to developing effective, scalable solutions for these emergencies.

The importance of this work lies in its ability to save lives, reduce casualties, and provide actionable solutions for individuals affected by active shooter events. We invite the broader research community to join this effort and contribute to the advancement of knowledge and technology in this essential domain.

Chapter 4

Model-Based and Model-Free Reinforcement Learning Optimized Evacuations in a Physics-based Simulation

4.1 Abstract

Active shooter incidents (ASIs) present a uniquely complex and time-critical challenge for emergency evacuation planning. This study evaluates the performance of three routing algorithms—Naive ASTERS, Q-Learning, and N-Step Temporal Difference Learning—within a high-fidelity, physics-based simulation environment built in Unreal Engine 5 (UE5). By transitioning from a discrete event simulation (DES) to UE5, we capture more realistic agent behaviors, continuous-time dynamics, and line-of-sight mechanics, enabling a more accurate assessment of algorithmic effectiveness. Across 90 experimental scenarios varying evacuee distributions, shooter spawn locations, and evacuee densities, we find that N-Step Temporal Difference Learning consistently outperforms the other methods in terms of survival rate, casualty reduction, and adaptability to dynamic threats. The results highlight the importance of incorporating temporal foresight and intermediate rewards into evacuation strategies. This work not only advances the state of the art in evacuation modeling but also lays the groundwork for future integration of real-time learning, virtual reality training, and multi-agent coordination in high-stakes emergency response systems.

4.2 Introduction

The previous thrusts of this work focused on the establishment of classical reinforcement learning methods as a viable method to guide evacuees in chapter 2 and introduce capacity constraints and novelly modify the previously established routing algorithm to incorporate multiple optimal routes in chapter 3. All of the previous work, training and evaluation, was completed within a discrete event simulation (DES) that I wrote in Python using a library called Simpy. Although the DES was an obvious starting point and proved to be a highly flexible means of conducting simulation experiments, it does not provide the realism required to arrive at strong conclusions based upon the experimentation results. A physics-based simulation solves this issue by injecting the realism lacking in DES into the simulation process through its ability to inherently handle crowding, collisions, and optimized movement

and way-finding. Transitioning to a physics-based simulation environment offers several advantages over the discrete event simulation (DES):

- Unlike DES, physics-based simulations operate in continuous time and space, governed by the laws of physics allowing for more accurate modeling of dynamic systems such as robotics, autonomous vehicles, or multi-agent interactions in complex environments
- Natural forces like gravity, friction, and collisions are naturally integrated, enabling agents to learn and adapt in environments that more closely resemble real world conditions
- Can lead to agents in Reinforcement Learning methodologies learning more nuanced and complex policies that are more transferable to other environments

Moving forward with a physics based simulation environment will provide the improvements identified above, but will also more experimentation on the ability to train offline within and a DES and then apply the learned policies to the physics based simulation. The focus of chapter 4 is this application as I have trained the model-based and model-free approaches on the DES I wrote and using the resulting policies in the Unreal Engine 5 (UE5) simulation for evaluation. The benefit of this approach is the cost of training offline in a DES is significantly "cheaper" in time and computation required than conducting the training in the UE5 simulation itself thus improving time to implement in new layouts and scenarios. Further, adoption of an evacuation and response plan validated on a physics based simulation is much more likely due to the increased and inherent realism over the DES. This chapter will layout the process by which I trained the evacuation

For the remainder of this paper, we summarize the current literature and other works in section III, describe the specifics of our methodology in section IV, discuss the experimental design, the simulation environment, and the different algorithms we evaluated in section V, discuss and compare the results from our simulations in section VI, and finally identify future plans to improve the algorithm in section VII.

4.3 Literature Survey

ASIs represent a serious threat to public safety that appears to only be increasing in frequency and severity, particularly in schools, universities, and other public spaces. These events are characterized by their unpredictability, rapid escalation, and devastating consequences, often leaving communities grappling with significant loss of life and psychological trauma. The increasing frequency of ASIs has underscored the urgent need for effective strategies to mitigate their impact, improve emergency preparedness, and enhance occupant safety.

Understanding human behavior during ASIs is critical for designing effective response strategies. Factors such as spatial awareness, decision-making under stress, and evacuation efficiency play pivotal roles in determining survival outcomes. However, studying these factors in real-world scenarios is inherently challenging due to ethical and logistical constraints. As a result, researchers are forced to turn to simulation-based approaches, leveraging advancements in technology such as reinforcement learning (RL), virtual reality (VR), and multi-agent systems to model ASIs and evaluate potential mitigation measures. Simulation-based modeling offers a unique opportunity to analyze the dynamic interactions between building design, occupant behavior, and threat progression during ASIs. By creating controlled environments that replicate real-world conditions, researchers can systematically test the effectiveness of architectural layouts, security features, and training programs. These models not only provide valuable insights into human behavior but also serve as decision-support tools for architects, emergency planners, and policymakers.

This literature review explores the current state of research on active shooter response and simulation modeling. The first section focuses on studies that examine human responses to ASIs and the role of training, spatial design, and personal factors in shaping evacuation strategies. The second section highlights advancements in simulation platforms, particularly those leveraging Unreal Engine, to create realistic and scalable environments for multi-agent systems and reinforcement learning. Together, these studies contribute to a growing body of knowledge aimed at enhancing resilience and preparedness in the face of active shooter threats.

School shootings represent a critical threat to public safety, necessitating innovative approaches to evaluate and improve building design and emergency preparedness. Recent work in ASIs includes Liu et al. [58] proposing a reinforcement learning (RL)-based simulation framework to assess the impact of school layouts on ASI outcomes. Their study uses four generalized school prototypes to simulate adaptive shooter behavior, capturing dynamic interactions between building design, occupant movement, and threat progression. The findings reveal that limited exit availability significantly increases harm, with classroom-adjacent exits playing a pivotal role in occupant survival. Additionally, architectural layouts such as bar and radial designs facilitate rapid shooter movement, leading to higher harm counts, whereas courtyard and cluster layouts limit shooter visibility and slow shooter progression, reducing harm rates. Topraklı and Satır [91] conduct a systematic review of building design strategies to mitigate ASI risks, focusing on simulation modeling. Using the PRISMA framework, they analyze 75 studies published since 2000, synthesizing findings across key themes such as exit design, spatial configuration, security features, and communication systems. Their review highlights the effectiveness of access control, surveillance systems, and bullet-resistant materials in enhancing occupant safety. Additionally, the authors critically evaluate simulation modeling approaches, identifying strengths, limitations, and knowledge gaps. The study provides valuable insights for architects, emergency planners, and policymakers aiming to optimize building design for resilience against ASIs. While these examinations of the building layouts is useful, their primary purpose is to influence future buildings architecture and not necessarily improve responses to ASIs. Sharma and Moses [82] present an immersive virtual reality (VR) training environment for active shooter response in university campus buildings. Developed in Unity 3D, the platform incorporates run, hide, and fight modes for emergency response. The authors introduce a multi-user VR platform where experiments for active shooter response can be conducted using computer-controlled (AI) agents and user-controlled agents. A user study involving 195 participants evaluates the effectiveness of the training environment, showing significant improvements in participants' knowledge, intrinsic motivation, and self-efficacy. The results highlight the potential of VR-based training to enhance spatial awareness and decision-making during high-stress situations, ultimately increasing survival

rates and evacuation success. Sharma and Moses [81] extend their work by developing a collaborative virtual reality environment (CVE) module for active shooter response training. The CVE module, implemented in Unity 3D, allows users to navigate immersive and non-immersive environments as autonomous agents. A user study evaluates the effectiveness of the module, showing increased sense of presence, intrinsic motivation, and self-efficacy among participants. The study demonstrates the potential of immersive VR environments to enhance emergency preparedness and decision-making during active shooter events. While useful in training, their research does not attempt to guide or direct the user-controlled agents to improve outcomes. Liu et al. [56] investigate the role of personal factors in shaping responses to ASIs using interpretable machine learning methods. Their study analyzes data from 107 participants, examining factors such as training methods, prior training experience, sense of direction, and gender. The results indicate that VR-based training leads to better responses compared to video-based training, with a better sense of direction and previous training experience correlating with a greater propensity to run and reduced vulnerability. Gender influences decisions and vulnerability slightly but significantly impacts pre-evacuation time, with females evacuating slower due to higher risk perception. Dai et al. [16] developed a multi-agent simulation model to assess mitigation measures for school shootings. Their study evaluates the impact of a gunshot detection system on evacuation rates and casualty reduction. Simulations reveal that the gunshot detection system significantly improves evacuation rates, increasing them from 16.6% to 66.6%, and reduces average casualty rates from 24.0% to 12.2% in six minutes. The research highlights the adaptability of the system in different floor layouts and provides a starting point to demonstrate the effectiveness of gunshot detection systems in improving school safety.

Beyond the ASI specific work, the use of video games engines as a model simulator increases the realism both visually and systematically. Wheeler [94] introduces a reinforcement learning framework for Unreal Engine, addressing the need for machine learning-based methods to control non-playable characters (NPCs). The framework integrates Unreal Engine levels with Python libraries for reinforcement learning, enabling game developers to create, train, and implement smarter AI characters. The study demonstrates the framework's capability by implementing, training, and evaluating on the AI-Gym

benchmarks, proving its scalability and viability compared to existing tools. Haley et al. [32] describe a synthetic simulation environment for defense applications, integrating thermal modeling capabilities into Unreal Engine. Their system generates realistic training data for object recognition and classification, supporting autonomous system development and validation. The simulation environment enables end-to-end autonomy training and development before hardware availability, addressing challenges such as high costs for training data generation and field autonomy integration. The development of synthetic images and videos to train. Johnson et al. [39] propose an architecture for transitioning from reduced reality training environments like a DES to realistic simulations using Unreal Engine 5. Their modular system supports multi-agent systems and incorporates Software-in-the-Loop (SITL) and Hardware-in-the-Loop (HITL) for detailed testing and validation. The study investigates case studies on single-agent autonomous tracking and auction-based multi-agent coordination, demonstrating the framework's scalability and effectiveness. This approach would be very helpful and could be applied more broadly when they support custom environments. Chen et al. [11] present UBER, an Unreal Engine-based simulation platform with extensibility and real-time capability. The platform leverages TCP communication to integrate Python scripts, enabling reinforcement learning and formation control algorithms in high-fidelity simulations. UBER's extensibility and real-time capability make it a valuable tool for simulating unmanned vehicle scenarios and testing autonomous systems. While we are not looking to do anything autonomously, the computer vision aspect of this approach is interesting from the perspective of crowd estimation and shooter identification. Hu et al. [36] introduce Unreal Multi-Agent Playground (Unreal-MAP), a general platform for multi-agent reinforcement learning (MARL) based on Unreal Engine. Unreal-MAP allows users to create customized multi-agent tasks and deploy state-of-the-art MARL algorithms. The platform is user-friendly, open-source, and compatible with a wide range of algorithms, advancing the field of MARL through its integration of existing algorithms with user-customized tasks.

4.4 Methodology

The focus of this chapter is to adapt and develop offline methods of evacuee route determination and use them in real-time environments within a physics-based simulation. This involves two approaches: modifying the previous algorithms I developed for the DES in order to make them compatible to run within UE5 and develop and train new, model-free approaches to incorporate into the library of potential solutions to evacuee routing for safety. The model-based approaches, Naive ASTERS from chapter 2 and C-CASTERS from chapter 3 made use of knowledge of the environment and perceived ideas of safety. The idea that a particular room may be safer than another, based upon materials used for the door for instance, could lead a model-based algorithm to "push" people toward the safer room. In theory, the room with better door materials may be safer, but it practice it may not matter and hence provide inaccurate representations of the environment. The problem with the model-based approach is that it requires known information about the environment in order to model it appropriately and if there is inaccuracy in any of the elements within the model, the results can be poor and incorrectly interpreted. The existence of this potential necessitated the addition of the model-free approaches to the library of methods for determining evacuee routing.

The model-free approaches, Q-Learning and N-Step Temporal Difference Learning, rely on the the agents learning from the actions and their results to determine optimal decisions. In the same scenario above with the door materials, the agent will learn to avoid that room, regardless of the perceived safety of the room's door materials, if the agent experiences poor outcomes when choosing to go to the room. Learning from the experience of their actions is a fantastic benefit for the agents of the model-free approaches. Making fewer assumptions about the environment and allowing the agents to learn the optimal actions in each situation addresses the problem of potentially incorrectly modeling the environment and thereby pulling incorrect interpretations of the results.

4.4.1 Model-Based Algorithms and Adjustments

In our work in chapter 3, we modified the Naive ASTERS value iteration approach to track capacities and iteratively identify optimal routes as path capacities were maxed out resulting in C-CASTERS. However, after discussing this aspect of the project with industry experts from the Department of Homeland Defense and other academic institutions interested in this topic, it is unrealistic to give people in the same node different routes as human nature would be to follow the crowds due to the belief that there is safety in numbers. For the experiments in this chapter, we will provide only a single route from each potential node within the layout. Incorporating this into all of the routing algorithms will do two things: 1) decrease the time to calculate optimal routes or train agents offline and 2) increase the realism of our results and brings us closer to an actual application that can eventually save lives. Our routing algorithms that we examined in this chapter have changed. We continued to examine the performance of Naive ASTERS in the new simulation environment and were to draw more definitive conclusions from its performance due to the more realistic dynamics of the UE5 environment. We will no longer examine the performance of the Natural Response as it has consistently performed worse than the other algorithms and was always meant as a benchmark to compare to in the absence of official direction beyond the blanket advice of "run, hide, fight".

Naive ASTERS Adjustment and Implementation

Naive ASTERS is a straightforward implementation of value iteration within the realm of reinforcement learning. The previous application of Naive ASTERS in chapter 2 used the proximity to shooter, safety factor of nearby nodes, and distance to the nearest exits as a means to determine what the optimal routing from each node was. The path that was provided was a list of consecutive nodes the evacuee was to travel and with a max cumulative travel time of 30 seconds by which point another path would have been calculated and provided. This version did not use the idea of capacity as the number of evacuees in experimentation did rise to the require it. In chapter 3, the number of evacuees was increased significantly where crowding and bottle-necking became a significant concern. For

this set of experiments, we modified Naive ASTERS to account for real time capacity and provide multiple routes from each node which became the implementation of C-CASTERS. We continued to analyze the performance of Naive ASTERS because it gave the benefit of complete offline route development. It still performed well, however, it was eclipsed in performance by C-CASTERS in most situations due to its lack of planning for any type of capacity constraints.

In order to implement the Naive ASTERS algorithm in the new physics-based environment, it required modifications to be able to pass the offline developed routes to the blueprint background of UE5. The route development was conducted in the same manner as in chapter 2 and chapter 3. A simplified layout of the UE5 environment was discretized into a series of nodes and edges that matched the eventual pathing layout within UE5. The Naive ASTERS algorithm was run to determine the optimal route from each node as far as they could get within the 30 seconds threshold. The path that resulted included the list of all intermediate nodes between their start point and their final destination which meant that there was usually a combination of room and hallway nodes in the list. Everything to this point is exactly the same process as in previous experiments. In order to give specific directions to the NPCs in UE5, we needed to create a look up table that passed the tuple *[evacuee node, shooter node, route node list]* to UE5. UE5 then interprets that tuple and provides the directions to the evacuees. However, UE5 will not pass a list of nodes to the NPCs to follow so I removed all of the nodes from the list except the next immediate node. During initial experimentation, this led to very jittery execution in the UE5 environment as a NPC would get to the next node and then wait to receive its next target node. This is where the performance difference between the DES and a real time simulation becomes evident. This waiting would not matter in the DES because every agent waits their turn at every discrete time interval, however, this causes significant hiccups in the UE5 simulation resulting in artificial crowding and bottle-necking. To address this, I reduced the initial list of nodes along the route down to the next room node and removed everything else. This allowed UE5 to handle the collisions and optimal movement of the NPCs between their current node and the next room node. Once they arrived at the next room node, the next tuple would be passed and the NPC would again move to the next room node.This modification allowed

the Naive ASTERS algorithm to determine the target node to travel to while also allowing UE5 the freedom to optimize the how to physically get the NPC there while dealing with the collisions and crowding. Again, it is important to note that the look up table is created in an offline process so UE5 is able to route the NPCs in real time.

C-CASTERS Approach

C-CASTERS was the best performing algorithm from chapter 3 experiments and it did so by creating multiple optimal routes and reserving capacity along the routes in an specified order that resulted in less crowding and better evacuation results. Theoretically, these were fantastic results and definitively proved the approach held merit with respect to determining the evacuation routing in an ASI. C-CASTERS takes the real time occupancy of every node and the shooter's position, calculates the optimal route for each node, reserves the evacuees occupancy in the future nodes in the path, and continues to iterate until every evacuee has an optimal route. While this approach did provide the best performance, it also took the longest time to compute and implement as the DES would pause every 20 seconds to allow it determine the routes and provide them to the evacuees before restarting and continuing with the experiment. Even restricting the algorithm to providing a single route for each node, which essentially makes it the same as the Naive ASTERS algorithm, still required from 8 to 20 seconds of run time. The goal of this research is to save as many lives as possible by providing real time directions to ensure the safety of evacuees which is impossible in the real world where an ASI has no pauses or breaks to run algorithms. There were no modifications that could be made to change or improve the implementation of C-CASTERs, and, because a single route version is essentially Naive ASTERS and it still would be capable of real time implementation, C-CASTERS will not be examined within the UE5 simulation environment. It provided a sound theoretical approach with great performance, however it was not realistic to be implementable in the real world.

4.4.2 Model-Free Algorithms

The model-free approaches that we incorporated in the DES training and UE5 simulation are Q-Learning and N-Step Temporal Difference Learning. We conducted the offline training for both approaches in the discrete event simulation environment and then used the results from that simulation to create the optimal policy for execution in the UE5 simulation. Each of the two approaches plays an important role in the experiment. The Q-Learning approach provides a simple action-result framework that is a tried and true method for training agents and determining optimal policies. The N-Step TD approach is very important because, unlike the Q-Learning, the value associated with its actions take into account the n previous actions so there is some decision history that plays a role in what actions are chosen. The resulting optimal policies from these two model-free algorithms present a means for determining routes that differ from the model-based approach and has the advantage of creating the very kind of lookup table that is required for routing in UE5 as the output to the algorithms. Further, we are continuing to pursue the traditional training methods using a tabular format of the Q table for reproducibility and explainability of the actions taken and why. We feel it is important to be able to explain the reasoning behind the decisions that are made and be able to explain why situations occur. This approach prevents the model-free algorithms from taking the occupancy of the nodes into account as part of the state because the state space would become too large and would require a Deep Q Network that uses a neural net to approximate the Q-values in the high dimensional space.

Model-Free Common Implementations

N-Step Temporal Difference Learning is a more advanced version of Q-Learning with a more complex means of keeping track of previous decisions and values and how they affect the current decision. That being said, Q-Learning and N-Step Temporal Difference Learning are extremely similar and as such have common aspects to their implementation in training and within UE5.

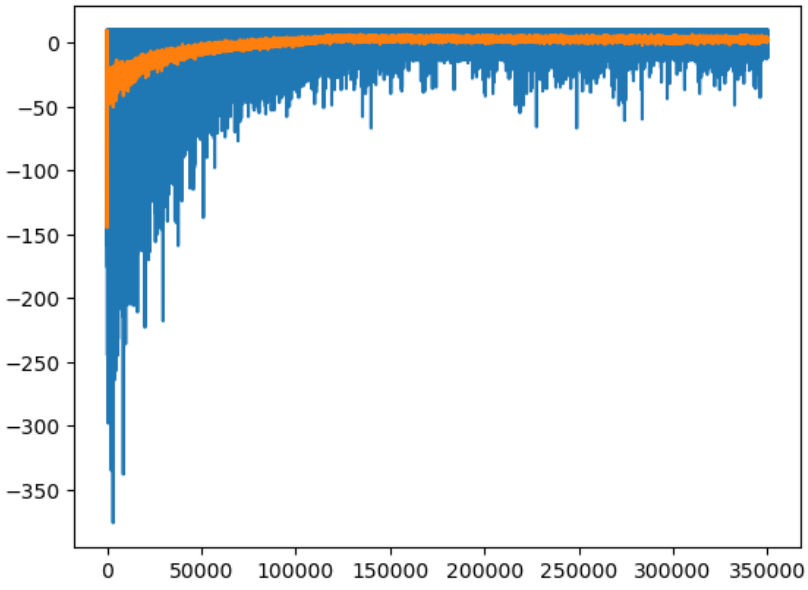
Reward Function The reward function is similar to the what was used in the C-CASTERS algorithm with some simplifications and is similar to many standard reward

functions. Being in close proximity to the shooter and being seen by him results or, reciprocally, arriving at an exit results in a large negative or positive reward respectively representing the worst and best case rewards. The remainder of the reward function is meant to reward behaviors we want (avoiding the shooter's line of sight or wasting time).

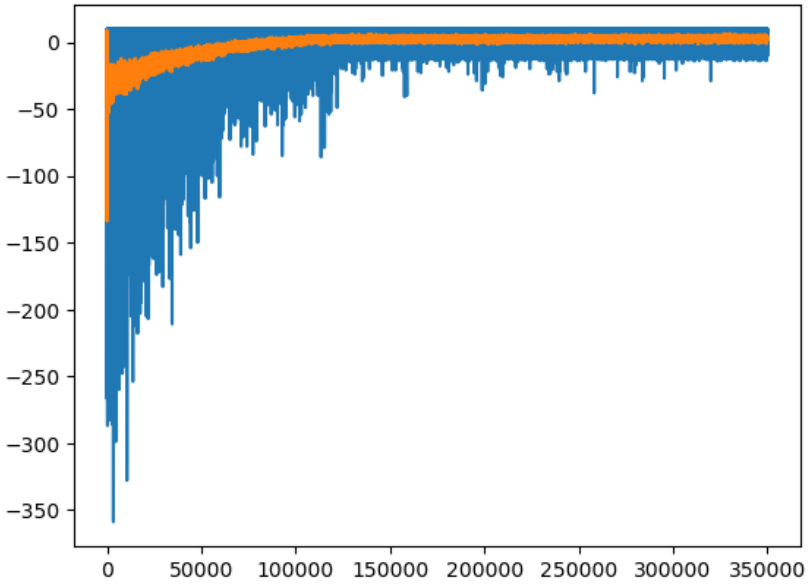
- Evacuee <= 3 nodes from shooter AND in shooter's line of sight: $reward = -10$
- Evacuee in shooter's line of sight: $reward = -3$
- Evacuee in room (not hallway): $reward = 0$
- Evacuee in exit: $reward = 10$
- Else: $reward = -1$

Training Environment and Parameters The process for training the agents for both algorithms was the same. Each of the agents trained through 350,000 episodes in the aforementioned DES. Each episode, the agent would spawn in one node and the shooter would spawn in another and the remaining nodes would be empty. The shooter would have a target node, would proceed on the shortest path to that target node, and, upon arrival, determine the next target node at random and continue this process until the episode was done (the agent escaped or was caught by the shooter). The agent would take actions according to an ϵ-greedy policy and observe the result and the reward achieved. Through the 350,000 episodes the agent would experience decisions in all of the possible combinations of the state space between the agent and the shooter. Below, Figures 4.1a and 4.1b depict the training progression of both model-free algorithms.

Both model-free algorithms make use of the ϵ-greedy choice for the agents actions. The ϵ-greedy policy means that the agent will choose an action at random when a random number generator creates a random number smaller than ϵ, otherwise the agent chooses the action most likely to result in the best reward. As each episode completes, ϵ deteriorates by a small amount making it more and more likely that the agent will choose the optimal action. After enough episodes have completed, ϵ becomes 0 and the only actions taken from that point



(a) Q-Learning Progression



(b) N-Step Temporal Difference Learning Progression

Figure 4.1: Learning Progression of the Model-Free Algorithms: These figures demonstrate the progression of learning for each of the model-free agents. The horizontal axis is the number of episodes and the vertical axis is the reward achieved for each episode. The blue line displays each of the individual 350,000 episodes and the resulting reward. The orange line is the running average reward of the last 50 episodes. For both algorithms, the results of the agents actions really solidify between the 100,000 and 150,000 episodes mark and, from that point forward, the agents perform extremely consistently.

on would be optimal. You can see this behavior in Figure 4.1a where the there is much less variation in the blue episode results and the orange running average.

The state spaces for both algorithms are also the same. The state space, s, is a tuple of the evacuees current node when the action is taken, n_e, and the shooter's current node when the action is taken, n_s, making $s = [n_e, n_s]$. More detail on the state space and it's implications and considerations will be addressed in the following experimental design section, 4.5.

Q-Learning Evacuation

Q-learning is a widely studied model-free reinforcement learning (RL) algorithm that enables an agent to learn an optimal policy for sequential decision-making tasks in a Markov Decision Process (MDP) framework meaning the previous decisions made do not affect the current decision. The algorithm for Q-Learning follows in Algorithm 3. The algorithm operates by iteratively estimating the action-value function through training the agent in the environment, commonly referred to as the Q-function, which quantifies the

---

Algorithm 3: Q-Learning Evacuation

Input: Learning rate α, discount factor γ, exploration rate ϵ
Initialize: Q-table $Q(s, a)$ arbitrarily for all $s \in \mathcal{S}, a \in \mathcal{A}$

1. **For** each episode **do**
   (a) Initialize state s
   (b) **Repeat**
      i. Choose action a from s using policy derived from Q (e.g., ϵ-greedy)
      ii. Take action a, observe reward r and next state s'
      iii. Update Q-value:

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right]$$

      iv. Set $s \leftarrow s'$
   (c) **Until** s is terminal
2. **End For**

---

expected cumulative discounted reward of executing a given action in a particular state and subsequently following the optimal policy. The strength of this method is its simplicity and its explainability; the changes to the Q-function can be easily followed as an agents experiences the environment and the optimal policy, once training is complete, provides a simple means of choosing the best action in any experienced situation. The need to experience a situation at least one time previously which is the primary reason for not including node occupancy in the state space s.

N-Step Temporal Difference Evacuation

N-Step Temporal Difference (TD) learning is a generalization of the standard one-step TD or TD(1) method that incorporates multiple future rewards into the value function update where Q-Learning only takes the single action and its reward into account. It provides a middle ground between the high bias of one-step TD and the high variance of Monte Carlo methods by considering a fixed number of future steps, denoted by n, when estimating returns.

Given a Markov Decision Process (MDP) with state space $\mathcal{S}$, action space $\mathcal{A}$, reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and discount factor $\gamma \in [0, 1)$, the n-step return from time step t is defined as:

$$G_t^{(n)} = \sum_{k=0}^{n-1} \gamma^k r_{t+k+1} + \gamma^n V(s_{t+n}),$$

where $V(s_{t+n})$ is the estimated value of the state reached after n steps. The value function is then updated using the following rule:

$$V(s_t) \leftarrow V(s_t) + \alpha \left[G_t^{(n)} - V(s_t) \right],$$

where $\alpha \in (0, 1]$ is the learning rate. This update incorporates both immediate and near-future rewards, allowing the agent to learn more efficiently in environments with delayed or sparse feedback.

N-Step TD methods are particularly useful in environments where intermediate rewards are informative but not sufficient for learning optimal behavior. This is a great example of

why it is successful in an ASI modeled environment. The intermediate rewards we discussed (avoiding line of sight and wasting time) help with survival, but don't necessarily result in great outcomes. Avoiding the shooter altogether and escaping are rewards that can take many steps to get to which lends itself to the use of this algorithm. The algorithm governing N-Step Temporal Difference Learning can be seen in Algorithm 4.

Algorithm 4: N-Step Temporal Difference Learning

Input: Learning rate α, discount factor γ, step size n
Initialize: Value function $V(s)$ arbitrarily for all $s \in \mathcal{S}$

1. **For** each episode **do**
   (a) Initialize state s_0
   (b) Initialize buffer to store (s_t, r_{t+1}) pairs
   (c) **For** $t = 0, 1, 2, \ldots$ until episode ends **do**
      i. Take action a_t according to policy π, observe r_{t+1} and s_{t+1}
      ii. Store (s_t, r_{t+1}) in buffer
      iii. **If** $t \geq n - 1$ **then**
         A. Compute n-step return:

$$G_{t-n+1}^{(n)} = \sum_{k=0}^{n-1} \gamma^k r_{t-n+1+k+1} + \gamma^n V(s_{t+1})$$

         B. Update value function:

$$V(s_{t-n+1}) \leftarrow V(s_{t-n+1}) + \alpha \left[G_{t-n+1}^{(n)} - V(s_{t-n+1}) \right]$$

   (d) **End For**
2. **End For**

4.5 Experimental Design

The eventual goal of this research is to see real world implementation that makes a difference and helps preserve the lives of those unfortunate enough to find themselves involved in an ASI. Pursuing this goal necessitated the switch from the DES environment to the physics-based UE5 simulation environment to increase the realism and the transferability of the

results for real world implications. The experimental design section follows where the simulation environment and the parameters and simulation inputs are described in greater details to include the justifications for the parameter choices.

4.5.1 UE5 Simulation Environment

The focus of this section is the translation from a DES that is underpinned by a graphical network, Figure 3.4b, to the UE5 environment in Figure 4.2. The transition to the UE5 environment provides a large number of improvements like the realistic movement of NPCs in crowds using inherent navigation and way-finding (Figure 4.3), NPC collisions (both environmental and other NPC), continuous time information and data, and visualization of the environment and the events that occur in it, however, it also requires a significant number of updates to previous parameter and characteristic choices that will affect the outcome and processing of the simulation. Each of the updates and changes follow with a description of what occurred in the DES and how it changed for UE5.

UE5 Update to Shooter Actions

The shooter's actions in the DES for both chapter 2 and chapter 3 followed a set pattern that repeated throughout the entirety of an episode. The shooter would spawn in a directed node and would have a directed target room node, both of which were controlled simulation

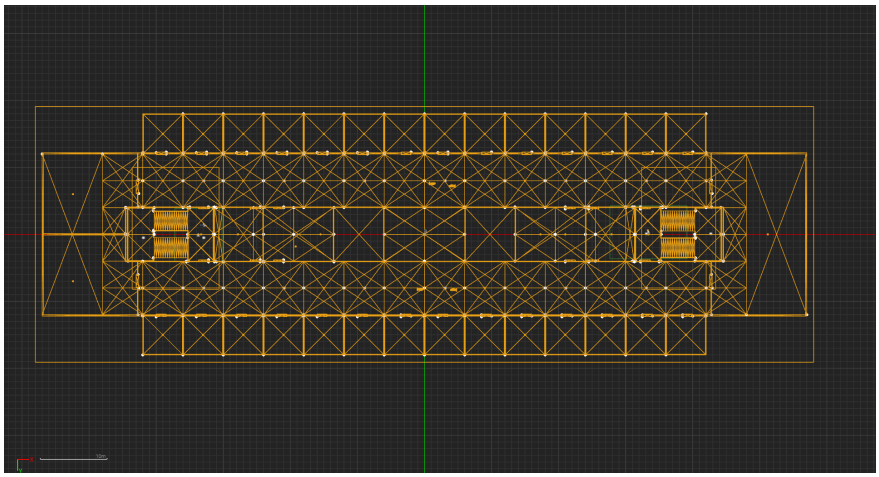


Figure 4.2: Wire frame Image of Cyclic Dorm Floor

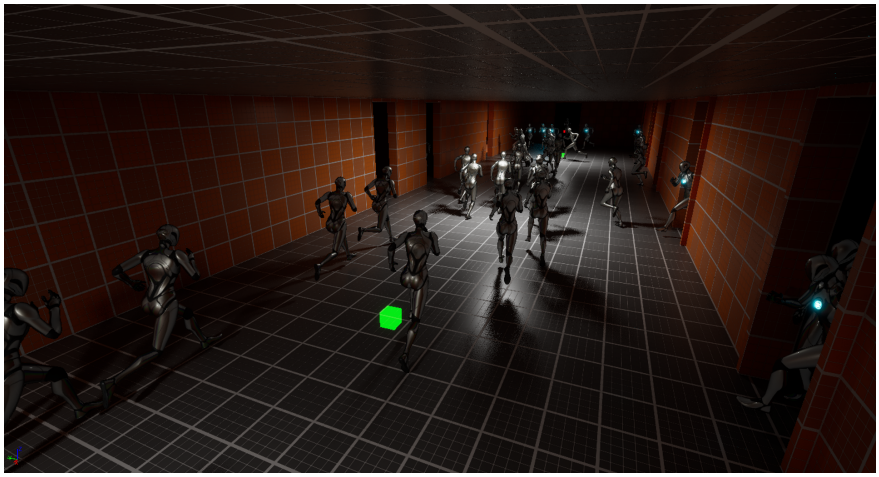


Figure 4.3: Cyclic Dorm Floor Visual with NPCs

parameters to ensure the entirety of the floor plan was explored. Assuming the shooter had knowledge of the floor plan, he would take the shortest path to the target room. Upon arrival at the target room, the shooter would loiter for a random number of seconds between 5 and 10. Upon completion of the loiter time, the shooter would randomly choose another target room from a list of unvisited rooms (closer rooms had a higher probability than farther rooms) and begin movement towards that room. This process continues until the end of the episode. The shooter would not alter his path or target regardless of the awareness of evacuees in other locations of the floor plan.

The changes that were required for the translation from the DES to UE5 were subtle, but they certainly increased the realism of the simulation. The shooter still spawned at a parameter controlled location, however, the initial target node was randomly chosen from the start instead of another controlled parameter. This slight change increased the randomness of the simulation and reduced the number of parameter combinations without sacrificing key data or results. We also make use of the awareness of the shooter once he has reached his target room. Upon arrival, the shooter will move towards the last place he witnessed any evacuees. While this is a more dangerous situation than what we executed in the DES, it is certainly more realistic and improves our ability to evaluate the performance of the algorithms.

UE5 Update to Evacuee Actions

The actions available to both algorithms are based upon the current node of the evacuee agent. The DES environment is underpinned by a graphical network, so the action space, a, at a given node are all of the adjacent nodes i.e. $a_5 = [4, 5, 6, 35, 64]$ from Figure 3.4b. Each node maintains a unique set of possible actions based upon the adjacent nodes to it. However, as discussed in the model-based modifications in section 4.4.1, providing intermediate nodes between key room nodes slows down the simulation and creates an awkward jitteriness to the NPCs movements. In order to simplify the Q-tables, I ran an algorithm that would follow the "crumbs" of the Q-table from the starting node until it reached another room and replace the recommended action with the next room node. For example, observe the scenario below for the series of actions that would likely follow:

Shooter at $node_{30}$, Evacuee at $node_6$, Chosen Action is a^*

- $a_6 = [5, 6, 7, 35]$ — $a^* = 7$
- $a_7 = [6, 7, 8, 22, 29, 36]$ — $a^* = 29$
- $a_{29} = [7, 8, 21, 22, 29]$ — $a^* = 21$
- $a_{21} = [8, 20, 21, 22, 29, 51]$ — $a^* = 51$
- $node_{51}$ is first room node, Q-table is updated: $Q[node_s = 30, node_e = 6, a^* = 51]$

The Q-table is updated to the very first room that is encountered by following the optimal policy until the first room is encountered. This allows the UE5 collisions and way finding to handle the minutia of movement in between the beginning and ending nodes but still maintains the logic behind the optimal policy.

UE5 Cyclic Dorm Environment

We replicated the Cyclic Dorm Floor from chapter 3 to scale in UE5. The underlying graphical network framework is exactly the same as in chapter 3 so the routing process will stay the same, however, as stated in the sections on adjustments to the Naive ASTERS,

Q-Learning, and N-Step Temporal Difference Learning, the algorithms will only pass the next room node to the UE5 processor. The floor plan is to scale so the doorways to the majority of rooms are single doors while the exit doors to the stairwells are double doors for increased throughput. Like the DES environment, the 29 hallway nodes range from 1 to 29, the 37 room nodes range from 30 to 67 minus 61, the two exit foyer nodes 61 and 68, and the two exit nodes are 69 and 70. Visually, the UE5 simulation is fantastically detailed and looks much like the most current generation video games with the amount of details and textures inherent in the gaming engine.

4.5.2 Parameters and Simulation Inputs

As expected, the simulations we ran in UE5 took longer than they did in the DES, so we reduced the number of parameters we examined through experimentation as well as the number of options per parameter type. We carefully chose the possible values for each parameter to ensure we efficiently explored the simulation space and were able to examine the plethora of possible scenarios. The discussion and explanation of the parameter values for the evacuee distribution, shooter spawn location, and number of evacuees follows.

Evacuee Distribution in UE5

At the beginning of each simulation, in both the DES and UE5 and across all of the different floor plans, the evacuees spawned into their starting locations. The locations that they spawned into fell into one of two categories: rooms or rooms and hallways. If the distribution of evacuees was identified as rooms, then the evacuees would only spawn in the rooms within the floor plan which in the case of the Cyclic Dorm Floor means they would spawn in the 37 room nodes from 30 to 67 minus 61. If the distribution was rooms and hallways, then the evacuees would spawn in the 66 nodes that do not include the exits or exit foyers which in this case means nodes 1 to 67 and excluding node 61. The thought process for these two different distributions is to examine the scenario while taking into account the purpose of the building being examined. In this case, we are examining a dorm floor so it is a series of bedrooms with some common areas including common rooms, bathrooms, kitchens, etc.

The rooms only distribution represents an evening when everyone is likely in their rooms while the rooms and hallways distribution is more indicative of some time in the middle of the day when there would be significantly more activity and movement within the rooms and hallways.

Shooter Spawn Location in UE5

The location where the shooter spawns plays an extremely important role in how an ASI plays in the real world and in simulation as well. We examined the floor plan for the Acyclic Dorm Floor to determine where the shooter should spawn in the UE5 simulation. Looking at Figure 3.4b, the building is conveniently symmetrical which means the number and types of nodes required to spawn the shooter in can be reduced. Figure 4.4 shows $node_2$ in the opposite hall as $node_{27}$, but also in the same position in the other end of the hall from $node_{13}$ which is opposite $node_{16}$. Because we distribute the evacuees evenly among the distribution nodes, there is no difference between any of those four nodes as the are just opposite or mirrors of the other nodes. This logic holds for basically all of the nodes, each one has one to three nodes that are in a symmetrical position to itself. With this in mind, the shooter spawns in $node_2$. $node_{29}$ (the only node that doesn't have another similar one), and $node_{34}$.

Number of Evacuees in UE5

The number of evacuees is new to our work in the UE5 simulation. In the previous experiments, we use a constant number per node so the only thing that changed the total number of evacuees was the distribution parameter. For this experiment, because we are

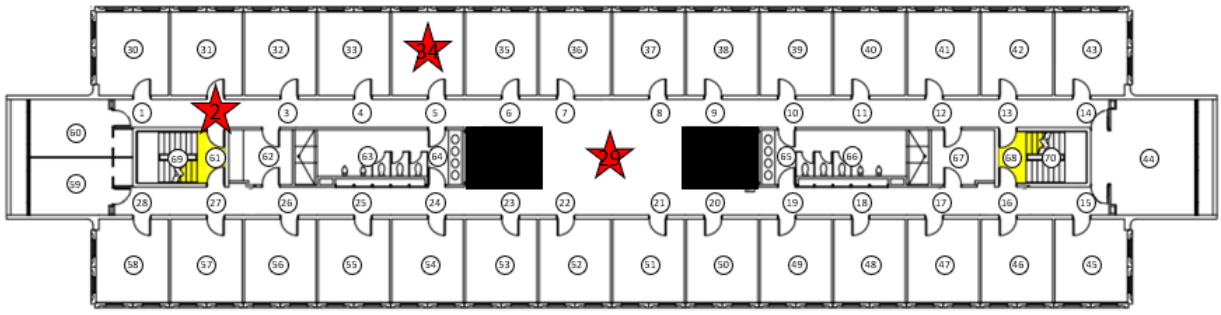


Figure 4.4: Shooter Spawn Locations

dealing with a dorm floor, we wanted to capture the potential varying occupancy of a dorm through its many phases throughout a day from being almost empty during a class day to be overly full on a rainy weekend. With this in mind, we varied the number of evacuees in each node from one to five.

When taking into account the three different algorithms, the two different distribution types, the three shooter spawn locations, and the 5 possible values for evacuees in nodes, we ran 90 different experiments to examine each of the parameter combinations in this simulation.

Shooter Identification and Tracking

The complementary program at Iowa State University has been developing and training a computer vision task to identify and track the shooter. A simplified version of that algorithm is in the UE5 simulation that identifies a green colored shooter like in Figure 4.5. The UE5 simulation reports the location of the shooter to the routing algorithms and they return the target node to the evacuees for them to begin moving. These updates to evacuees happen in milliseconds keeping the simulation as close to real time as it can be.

Figure 4.5: Green Shooter Example

4.6 Results

In this section, we comprehensively compare the overall performance of all three routing algorithms and then identify the key differences between their performance with respect to our parameters of interest. For the simulation results, we ran 10 randomized simulation episodes of 300 seconds for each possible combination of parameter values to capture enough data to provide significant insights into the performance of the algorithms and the effects of the parameters. Again, we simulated the Cyclic Dorm Floor in the UE5 environment and varied the starting location of the shooter, the starting distributions of the evacuees, and the number of evacuees in each node at the start of the simulation.

4.6.1 UE5 Aggregated Performance

The overall performance of all algorithms can be found in Table 4.1. Each entry in the table is the average of 50 runs due to the number of evacuees per node not being included in the table due to size restrictions. The Distribution and Shooter Spawn columns of table indicate the values for those parameters. The next three columns, Escapes, Safe, and Survived are all related: Escapes is the number of evacuees that were able to make it to an exit, Safe is the number of evacuees that were still safe at the end of the five minutes of simulation time, and Survived is the sum of the two previous metrics. Casualties is the number of evacuees who were caught within the threshold distance and were seen by the shooter. The line of sight in UE5 is a significant upgrade from the DES. The DES assumed an evacuee was within the shooter's line of sight if their current position **could** be seen by the shooter, not that he was necessarily looking at them, whereas the UE5 simulation actually uses the the orientation of the shooter's eyes to determine if the evacuee is seen. This improvement increases the realism of the simulation mechanics and provides more explainability to the potential casualties that occur. Finally, the Survival Rate is give by the equation $\frac{Survivors}{TotalEvacuees}$ with $TotalEvacuees = Survivors + Casualties$ being the total spawned evacuees for a single run. Each row of the table corresponds to a set of parameter values; the first row in each algorithm section corresponds to the same parameter combination so you could compare the algorithms performance against each other by observing the same row in each section. The

Table 4.1: Performance Comparison of Naive ASTERS, Q-Learning, and N-Step Temporal Difference: Table 4.1 displays the average performance of each algorithm over the course of the 50 runs per parameter combination. The bold values represent the best performance amongst the algorithms for each of the five metrics of interest with that parameter combination.

Distribution	Shooter Spawn	Escapes	Safe	Survived	Casualties	Survival Rate
Naive ASTERS						
Rooms	2	*58.92*	**19.08**	*78*	*33*	*0.67*
Rooms	29	*54.52*	20.68	*75.2*	*35.8*	*0.64*
Rooms	34	*58.24*	14.32	*72.56*	*38.44*	*0.63*
Rooms and Halls	2	*113.68*	**45.48**	*159.16*	*38.84*	*0.75*
Rooms and Halls	29	*108.72*	46.28	*155*	*43*	*0.74*
Rooms and Halls	34	*121.8*	39.48	*161.28*	*36.72*	*0.77*
Q-Learning						
Rooms	2	**79.6**	*4.96*	**84.56**	**26.44**	**0.76**
Rooms	29	**76.36**	*4.76*	81.12	29.88	0.73
Rooms	34	69.84	*11.52*	81.36	29.64	0.71
Rooms and Halls	2	**135.84**	*28.36*	164.2	33.8	0.81
Rooms and Halls	29	**131.72**	*33.76*	165.48	32.52	0.81
Rooms and Halls	34	127.68	**43.44**	171.12	26.88	0.83
N-Step Temporal Difference Learning						
Rooms	2	74.04	9.92	83.96	27.04	0.73
Rooms	29	68.2	**22.36**	**90.56**	**20.44**	**0.78**
Rooms	34	**73.36**	**22.6**	**95.96**	**15.04**	**0.83**
Rooms and Halls	2	134.96	40	**174.96**	**23.04**	**0.85**
Rooms and Halls	29	117.88	**56.64**	**174.52**	**23.48**	**0.83**
Rooms and Halls	34	**138.64**	*38.36*	**177**	**21**	**0.87**

bold values in the table represent the best performance for that metric for that combination of parameters and the italicized values represent the worst.

By examining Table 4.1, a few trends become immediately obvious when comparing the bold and italicized values. The Naive ASTERS algorithm performs poorly when compared to the other two algorithms with it being worse in all metric and combination of parameters except for the number of "safe" evacuees in both evacuee distributions and when the shooter spawned in $node_2$, otherwise it was the worst performing algorithm in Escapes, Survived, Casualties, and Survival Rate in every combination of parameters. Further, the N-Step Temporal Difference algorithm performed extremely well having the best performance in $\frac{2}{3}$ of the metric and parameter combination pairings with 20 bold values out of 30 possible,

having the worst performance in a single metric and parameter combination, and, most importantly, having the highest Survival Rate in all parameter combinations except for the Rooms evacuee distribution and when the shooter spawned in $node_2$. Interestingly, the Q-Learning algorithm performed well in most metrics except for the Safe metric where it was the worst performer. The primary difference between the Q-Learning and N-Step Temporal Difference algorithms, the temporal aspect of possible future rewards being incorporated into the value function, provides an improved safety aspect to the N-Step Temporal Difference as it not only out performs Q-Learning in the Safe metric, but also Survived, Casualties, and Survival Rate. Holistically, N-Step Temporal Difference learning was objectively the best performing algorithm. In the following sections, we examine the intricacies of how the parameters and their values affected performance.

4.6.2 Effect of Shooter's Spawn Location in UE5

As was discussed in section 4.5.2, the shooter spawns in three different nodes in the UE5 simulation: $node_2, node_{29}$, and $node_{34}$. After examining the interactions with the other parameters and the metrics of interest, the following figures are indicative of the algorithms performance with respect to changes to the shooter's spawn location. The following figures exploring the effects of shooter's spawn location are scatter plots indicating the number of casualties on the horizontal axis and the number of escapes on the vertical axis meaning that data elements in the upper left quadrant are ideal (high escapes and low casualties) and data elements in the lower right quadrant being unacceptable (low escapes and high casualties). Interestingly, examining the scatter plots doesn't provide any intuitions about the affects of the shooter's spawn locations. The scatter of the data elements for the different algorithms are all intermixed and its difficult to see any patterns. We overlaid three large X's on top of the scatter plot that represent the coordinates of the centroid of each algorithms Casualties by Survivors. Comparing the centroids of each algorithm's data points provides a much more clear picture.

Figure 4.6 displays the results when the shooter spawns in $node_2$ and the centroids show that, while there is little difference between the centroids, the N-Step Temporal Difference algorithm produced the centroid with the most survivors and fewest casualties followed by

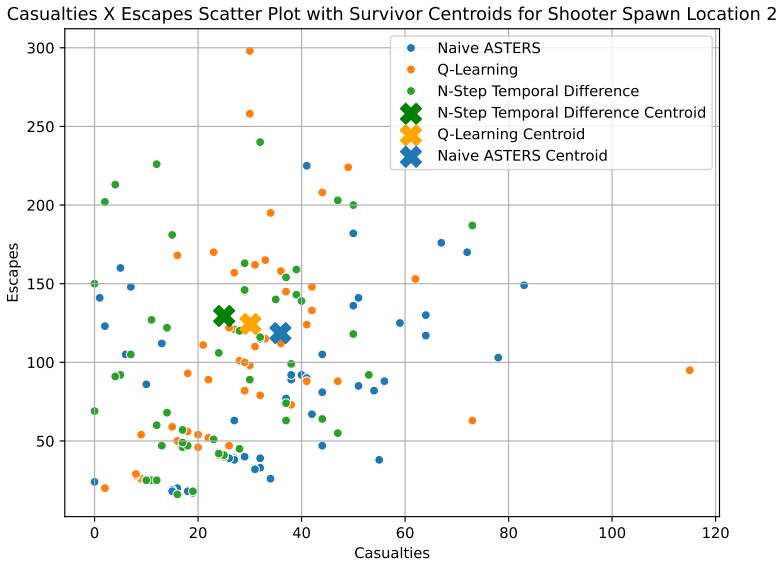


Figure 4.6: Performance Effects of Shooter Spawn-2

Q-Learning and Naive ASTERS. The tight spread and lack of any algorithm distancing itself from the others can be explained the close proximity to an exit when the shooter spawns making it more difficult to escape for half of the floor plan. Examining figure 4.7 yields the same result, the N-Step Temporal Difference algorithm performing the best followed by Q-Learning and Naive ASTERS respectively. The spread of the centroids is larger in this case with an approximate 13 casualty difference and 9 survivor difference between N-Step Temporal Difference and Q-Learning. The shooter spawn point being in the center of the floor plan explains the reason for the improved performance by all of the algorithms as it leaves a lot of options that the algorithms were able to exploit. The trend continues in figure 4.8 as the order of performance remains the same and the spread between them grows larger again. As has been seen in the previous experiments in chapter 2 and chapter 3, the shooter spawning in a room actually improves the algorithms ability to route evacuees to safety because the shooter is isolated in a room and has severely restricted line of sight. The consistently better performance by the N-Step Temporal Difference algorithm demonstrates a dominance and indicates that is the best routing algorithm we examined. The other fascinating aspect of these two graphs was that there are no significant differences among the algorithms i.e. in the following figures showing the densities for each of the algorithms is almost exactly the same.

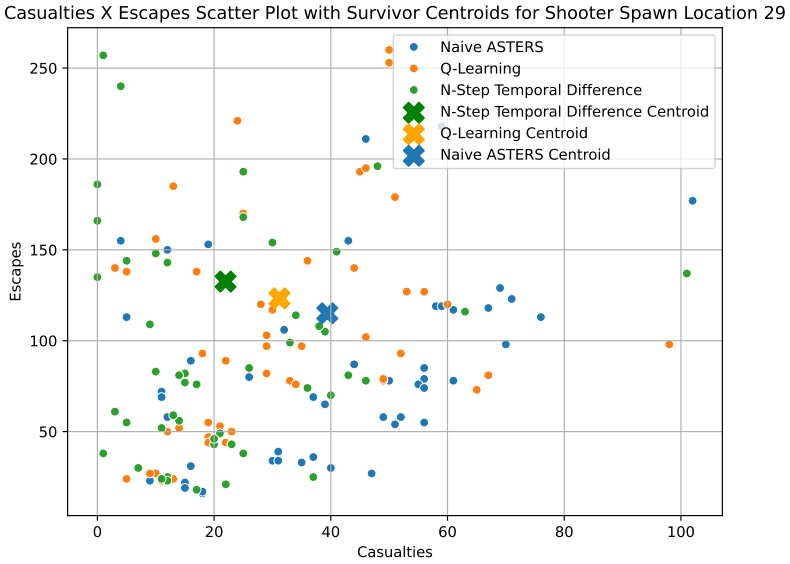


Figure 4.7: Performance Effects of Shooter Spawn-29

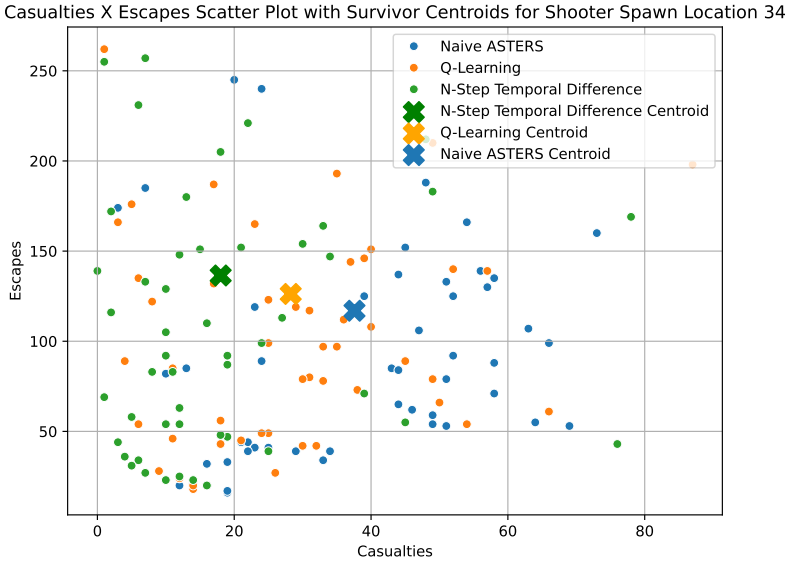


Figure 4.8: Performance Effects of Shooter Spawn-34

4.6.3 Effect of Evacuee Distribution in UE5

Varying the distribution of evacuees yields an interesting dichotomy where it seems to have little affect on the number of casualties, but significantly affects the number of escapes that the algorithms are able to execute. As indicated previously, we distributed the evacuees either in the 37 room nodes only or in the 66 room and hallway nodes that excludes the exits and the exit foyers. Figure 4.9 shows violin plots for the casualty densities for each of the algorithms with the left side in blue being the rooms only density and the right side in orange being the rooms and halls density. The rooms only density looks almost exactly the same as the rooms and halls density indicating that distributing the evacuees in rooms only does not seem to affect the quantity of casualties for any of the algorithms. However, Figure 4.10 displays the violin plots for the survivors metric for each of the algorithms. The difference in the densities is obvious and somewhat contrary to the previous DES experiments conducted in chapter 2 and chapter 3. In the previous experiments, the rooms only distribution consistently resulted in fewer casualties and more survivors. In the UE5 simulation for each of the algorithms, the rooms and halls distribution has a much wider spread, but it also has a significantly higher

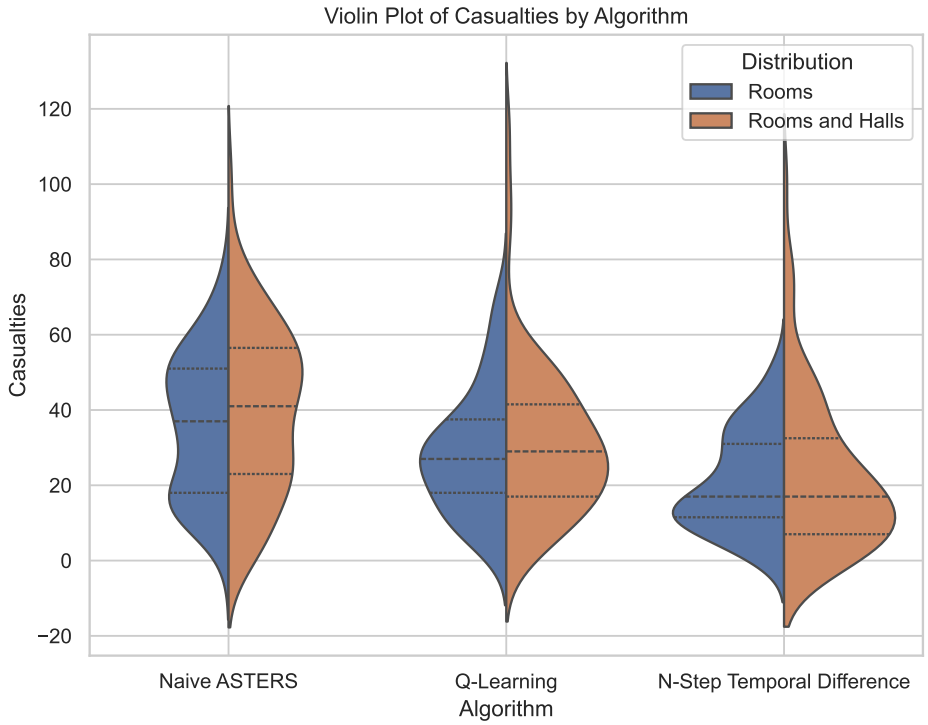


Figure 4.9: Casualty Effects of Evacuee Distribution

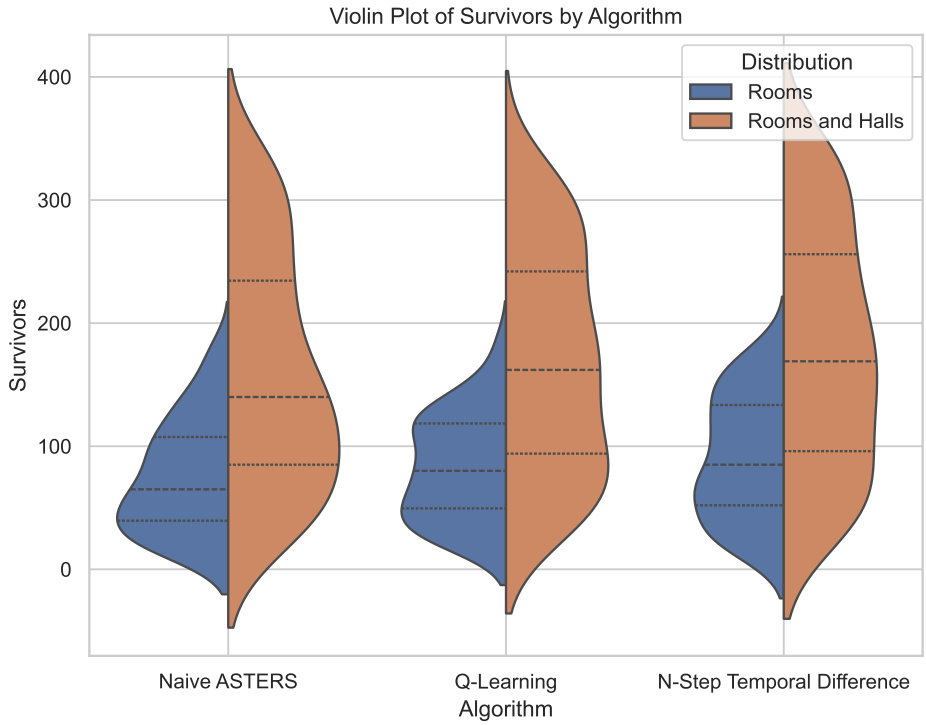


Figure 4.10: Survivor Effects of Evacuee Distribution

median and provides significantly more upside (higher probability of more survivors) with approximately the same downside.

4.6.4 Effect of Number of Evacuees Per Node in UE5

The final parameter we varied through the experiments was the number of evacuees that spawned in each node at the beginning of a simulation $num_{evacs} = [1, 2, 3, 4, 5]$. We varied this value to simulate the differing levels of occupation that a dorm may have i.e. almost empty on a Saturday football game or filled over normal capacity on the evenings before finals. Escapes and casualties would likely increase as we add more and more evacuees to the system, which they do, however, the relationship of increase for escapes remains consistent while the relationship with casualties slowly changes as the number of evacuees is increased. Figure 4.11 and Figure 4.12 are cumulative histograms that accumulate escapes and casualties respectively over time in the simulation across 90 runs for each value of evacuees per node. The shape of the cumulative curves are the same as the number

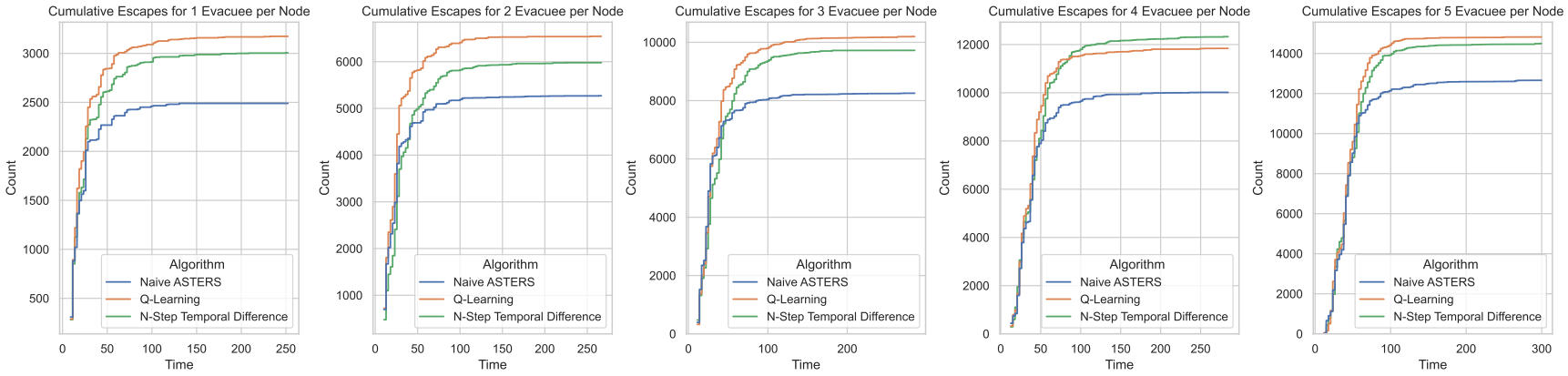


Figure 4.11: Evacuee Per Node Effects on Escapes

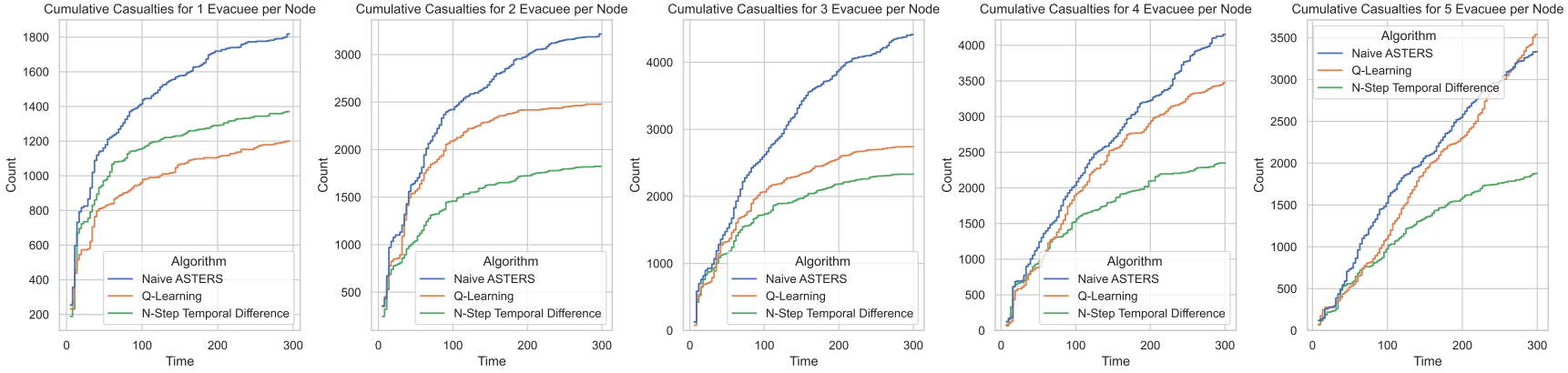


Figure 4.12: Evacuee Per Node Effects on Casualties

of evacuees per node goes up, however, the total accumulated escapes increases as expected. Comprehensively, the Naive ASTERS algorithm is the worst performer across all of potential values for evacuees per node. Conversely, the Q-Learning algorithm is the best performer for all values except for 4, but the N-Step Temporal Difference algorithm is close in performance to Q-Learning in the others. Its also interesting, regardless of the number of evacuees per node, that the vast majority of escapes occur within the first minute of the simulation.

The narrative around the effect evacuees per node had on casualties is significantly different and actually shows a difference as the number of evacuees increases. Figure 4.12 demonstrates the same information as Figure 4.11 utilizing the time that each evacuee becomes a casualty over 90 simulated runs. The shape of the casualty curves changes slightly as more evacuees are introduced to the simulation. Having only a single evacuee per node yields similar shapes to that of the escape curves with at least half of the casualties occurring within the first minute of the simulation. For that one parameter value, Q-Learning performs best with N-Step Temporal Difference being only slightly worse and Naive ASTERS being the worst which it does for all of the parameter values. As the number of evacuees increase from 1 to 5, the Q-Learning algorithm begins performing worse until eventually, at 5 evacuees per node, it and Naive ASTERS basically have a linear relationship between casualties and time meaning that the casualties continue to occur at a consistent rate throughout the five minutes. This shows how increasing the number of evacuees in the nodes seems to make it harder for Q-Learning and Naive ASTERS to keep evacuees safe. On the other hand, N-Step Temporal Difference Not only maintains its shape (at least half of the casualties occur in the first minute and then slower for the remainder of the time), but it also is able to maintain fewer casualties (within the neighborhood of 2000) through all of the possible parameter values.

4.7 Conclusion and Future Work

As we stated earlier, the goal of this research is to present practical applications that can be refined and implemented to improve the safety outcomes of individuals who become involved in an ASI. Transitioning the simulation environment to the visually more realistic

and physics-based game engine UE5 has absolutely moved us closer to that goal. The UE5 environment currently provides or will in the future:

- Visual presentation graphics on par with current generation video games (current)
- Physics-based handling of movement, crowding, and collisions (current)
- Realistic implementation of a shooter's line of sight and movements (current)
- Incorporation of full computer vision identification and tracking of shooter (future)
- High quality VR capability for human training and trials (future)

The UE5 simulation platform that we currently have is a huge improvement over the DES, but it can be further refined and improved to bring greater value and knowledge to this research area.

4.7.1 Proposed Future Work within UE5

VR Participants in Simulation The use of active participants in the simulation alongside NPCs for either data gathering or training will provide many research avenues to pursue. VR participants would allow us to begin examining behavioral aspects of ASIs and the individuals within them like the utility of a leader-follower dynamic, participants views of the trustworthiness of the algorithms, and does a herd mentality override the instructions given by a routing algorithm. Using the simulation as a training implement can improve the abilities of first responders to quickly assess situations, provide valuable experience to teachers who are in charge of small children (leader-follower), and develop and test building specific strategies for nullification of the shooter.

Agent Training in UE5 The experiments in chapter 4 were completed using offline training in the DES environment and still performed quite well in the UE5 simulation. Training the Naive ASTERS routing wouldn't change due to use of value iteration to calculate the route, however, the Q-Learning and N-Step Temporal Difference algorithms could significantly improve if they were trained directly within the UE5 environment. The

simple use of the UE5 shooter vision mechanic and training agents in the environment using that would reduce the number of casualties and very likely improve the number of escapes as the DES overestimates the shooter's ability to see and thus the agents are more risk averse than intended.

4.7.2 Conclusion

This research presents a comprehensive evaluation of evacuation routing algorithms in the context of ASIs, leveraging the enhanced realism and fidelity of a physics-based Unreal Engine 5 simulation environment. By comparing Naive ASTERS, Q-Learning, and N-Step Temporal Difference Learning across a range of scenarios—including varying evacuee distributions, shooter spawn locations, and evacuee densities, we demonstrate that N-Step Temporal Difference Learning consistently outperforms the other methods in terms of survival rate, casualty reduction, and overall adaptability. The transition from a discrete event simulation (DES) to UE5 not only improved the behavioral realism of agents and shooter dynamics but also enabled more nuanced assessments of algorithmic performance under complex, real-world conditions. These findings underscore the critical importance of incorporating temporal foresight and intermediate reward structures into evacuation strategies. Looking forward, integrating online learning within UE5, incorporating human participants through VR, and expanding the simulation to multi-agent reinforcement learning frameworks offer promising avenues for advancing both the theoretical and practical impact of this work. Ultimately, this research contributes to the growing body of knowledge aimed at enhancing public safety and preparedness in the face of emergent threats.

Chapter 5

Conclusion

5.1 Summary of Results

This dissertation presents a progressive exploration of optimized evacuation routing strategies during ASIs, beginning with model-based reinforcement learning in a DES, extending to capacity-constrained multi-route planning, and culminating in the evaluation of both model-based and model-free approaches in a high-fidelity, physics-based simulation environment. Chapters 2, 3, and 4 each build upon the previous, introducing increasing levels of realism, complexity, and computational rigor.

5.1.1 Chapter 2: Naive ASTERS and NHSMDP-Based Optimization

Chapter 2 introduced the Naive ASTERS algorithm, a model-based reinforcement learning approach grounded in a Non-Homogeneous Semi-Markov Decision Process (NHSMDP) framework. The algorithm incorporated dynamic shooter location, node safety (hardness), and exit proximity to generate optimal routing policies for evacuees. Simulations were conducted in a custom-built DES using two building layouts: a single-level acyclic school and a two-level cyclic hospital.

Across 12,484 parameterized scenarios, Naive ASTERS consistently outperformed a Naturalistic Response (NR) baseline derived from expert-informed "Run, Hide, Fight" logic. On average, Naive ASTERS reduced casualties by 56% and time spent in the shooter's line of sight by 52% compared to the best-performing NR variant. The algorithm's ability to recommend strategic waiting, enabled by the NHSMDP's temporal state expansion, proved critical in minimizing exposure to the shooter. These results validated the viability of model-based RL for ASI evacuation and highlighted the importance of incorporating shooter dynamics and node safety into routing logic.

5.1.2 Chapter 3: Capacity-Constrained Multi-Route Optimization (C-CASTERS)

Chapter 3 addressed a key limitation of Naive ASTERS: the assumption of unlimited capacity along evacuation routes. The C-CASTERS algorithm extended the NHSMDP framework to incorporate node and edge capacity constraints, enabling the generation of multiple optimal routes per node while reserving capacity through time. This iterative approach prioritized nodes based on proximity to exits and the shooter, reducing bottlenecks and improving flow efficiency.

Simulations were conducted in one of the previous environments and one new environment: the Acyclic School and the Cyclic Dorm Floor, with evacuee counts increased to reflect realistic crowding. C-CASTERS outperformed both Naive ASTERS and NR across all metrics. In the Cyclic Dorm Floor, C-CASTERS reduced casualties by 34.2% and 53.3% compared to Naive ASTERS and NR, respectively. It also achieved a 50% reduction in occupancy at key bottleneck nodes. These improvements were attributed to C-CASTERS' ability to distribute evacuees across multiple viable routes and delay movement when necessary to avoid congestion. The results demonstrated that capacity-aware planning is essential for realistic, scalable evacuation strategies.

5.1.3 Chapter 4: Model-Free Learning and Physics-Based Evaluation in UE5

Chapter 4 transitioned from DES to a physics-based simulation environment built in Unreal Engine 5 (UE5), enabling more realistic modeling of agent movement, crowding, and line-of-sight mechanics. This chapter also introduced model-free reinforcement learning approaches—Q-Learning and N-Step Temporal Difference (TD) Learning—trained offline in the DES and evaluated in UE5.

Across 90 experimental scenarios varying evacuee distributions, shooter spawn locations, and evacuee densities, N-Step Temporal Difference Learning consistently outperformed both Q-Learning and Naive ASTERS. It achieved the highest survival rate in all but one of the parameter combinations, with survival rates reaching up to 87%. Q-Learning performed

well in escape metrics but lagged in safety, while Naive ASTERS underperformed across all metrics due to its lack of real-time adaptability and inability to account for physical crowding.

The UE5 results validated the transferability of offline-trained policies in a DES to real-time, physics-based environments. N-Step Temporal Difference's temporal foresight allowed agents to make safer, more strategic decisions, particularly in high-density scenarios. These findings underscore the potential of model-free approaches for real-time deployment and highlight the importance of incorporating intermediate rewards and temporal context into evacuation strategies.

Together, the results from Chapters 2, 3, and 4 demonstrate a clear trajectory of improvement in evacuation routing performance—from static, single-route planning to dynamic, capacity-aware, and temporally informed decision-making. The progression from model-based to model-free approaches, and from abstract simulation to physics-based realism, provides a robust foundation for future work in real-time, deployable ASI evacuation systems. N-Step Temporal Difference Learning, in particular, emerges as a promising candidate for integration into intelligent guidance systems capable of saving lives in high-stakes emergencies.

5.2 Impact and Implications

This work represents a significant advancement in the field of emergency evacuation planning by introducing and evaluating reinforcement learning (RL)-based routing strategies tailored to the unique challenges of ASIs. While the primary focus is on ASIs, the methodologies and insights developed here have broader implications for emergency evacuations in general, including those triggered by fires, natural disasters, and other dynamic threats. The impacts of this work span theoretical modeling, algorithmic innovation, simulation fidelity, and practical applicability.

5.2.1 Implications for Emergency Evacuations in General

Traditional evacuation models often rely on static assumptions: fixed hazards, shortest-path routing, and homogeneous evacuee behavior. These simplifications, while computationally convenient, fail to capture the dynamic, uncertain, and often chaotic nature of real-world emergencies. This dissertation challenges those assumptions by introducing a framework that models evacuation as a sequential decision-making problem under uncertainty, where both the environment and the threat evolve over time.

The use of Non-Homogeneous Semi-Markov Decision Processes (NHSMDPs) allows for the incorporation of time-dependent transition probabilities and sojourn times, enabling the modeling of scenarios where waiting in place may be safer than moving. This is a critical departure from conventional shortest-path models, which typically assume that faster egress is always better. By quantifying the trade-offs between movement, exposure, and safety, this work provides a more nuanced understanding of optimal evacuation behavior.

Moreover, the introduction of capacity-constrained routing through the C-CASTERS algorithm addresses a long-standing limitation in evacuation planning: the assumption of infinite throughput. In reality, doorways, hallways, and stairwells impose physical limits on how many people can move through them at a given time. By iteratively generating multiple optimal routes and reserving capacity along them, C-CASTERS reduces bottlenecks and improves overall flow efficiency. This approach is broadly applicable to any evacuation scenario where congestion is a concern, including stadiums, airports, and urban evacuations during natural disasters.

The transition from discrete event simulation (DES) to a physics-based environment in Unreal Engine 5 (UE5) further enhances the realism and applicability of the findings. By incorporating continuous-time dynamics, collision handling, and line-of-sight mechanics, the simulation environment more accurately reflects the complexities of real-world evacuations. This fidelity is essential for validating the effectiveness of routing algorithms and for building trust among stakeholders who may implement these systems in practice.

5.2.2 Implications for Active Shooter Incidents

ASIs present a uniquely complex challenge for evacuation planning. Unlike fires or earthquakes, the threat in an ASI is intelligent, mobile, and often unpredictable. Traditional evacuation protocols such as "Run, Hide, Fight" offer general guidance but lack the specificity and adaptability needed to optimize survival outcomes in real time. This dissertation addresses that gap by developing algorithms that dynamically adapt to the shooter's location, movement, and line of sight.

The Naive ASTERS algorithm demonstrated that even a relatively simple model-based approach can significantly outperform expert-informed naturalistic responses. By incorporating shooter location and node safety into the reward structure, the algorithm was able to recommend strategic waiting and safer routes, reducing casualties and exposure time. This finding challenges the prevailing assumption that immediate evacuation is always the best course of action and underscores the importance of context-aware guidance.

The C-CASTERS algorithm further refines this approach by accounting for crowding and capacity constraints. In ASIs, bottlenecks near exits can become fatal choke points. By distributing evacuees across multiple routes and staggering their movement, C-CASTERS reduces the likelihood of mass casualties due to congestion. This is particularly important in environments like schools, where large numbers of individuals may attempt to evacuate simultaneously through limited egress points.

Perhaps the most impactful contribution comes from the evaluation of model-free approaches, Q-Learning and N-Step Temporal Difference Learning, in a high-fidelity simulation environment. These algorithms, trained offline in a DES and deployed in UE5, demonstrated the ability to make real-time decisions that adapt to dynamic threats. N-Step Temporal Difference Learning, in particular, consistently achieved the highest survival rates across a wide range of scenarios. Its ability to incorporate temporal foresight and intermediate rewards makes it especially well-suited for ASIs, where the optimal action may depend on anticipating future threat movements.

The success of these model-free approaches has profound implications for real-world deployment. Unlike model-based methods, which require detailed knowledge of the

environment and threat dynamics, model-free algorithms can learn optimal policies through experience. This makes them more robust to modeling errors and more adaptable to new environments. When paired with real-time shooter tracking systems, such as those using computer vision or gunshot detection, these algorithms could provide personalized, context-aware evacuation guidance to individuals via smartphones, digital signage, or public address systems.

5.2.3 Broader Societal and Policy Implications

Beyond the technical contributions, this dissertation has broader societal implications. The increasing frequency of ASIs in the United States and abroad has created an urgent need for evidence-based, technologically advanced solutions. The algorithms and simulation tools developed here offer a pathway toward more effective emergency preparedness and response strategies.

From a policy perspective, the findings support the integration of intelligent evacuation systems into building safety protocols. Schools, universities, hospitals, and other vulnerable institutions could benefit from deploying RL-based routing systems that adapt to real-time conditions. These systems could be integrated with existing infrastructure, such as surveillance cameras and public address systems, to provide dynamic guidance during emergencies.

The work also highlights the importance of simulation-based training for both evacuees and first responders. The UE5 environment, with its high visual fidelity and support for virtual reality, offers a powerful platform for immersive training. By simulating realistic ASI scenarios, stakeholders can test and refine evacuation plans, identify vulnerabilities, and build muscle memory for high-stress situations.

Finally, this research opens the door to important ethical discussions. As algorithms begin to influence life-and-death decisions, questions about fairness, transparency, and accountability become paramount. Future work must address these concerns by incorporating ethical frameworks into algorithm design and by engaging with diverse stakeholders to ensure that the systems developed are equitable, trustworthy, and aligned with community values.

In summary, this dissertation advances the science of emergency evacuation planning by introducing a suite of reinforcement learning algorithms that are context-aware, capacity-sensitive, and adaptable to dynamic threats. While the primary focus is on active shooter incidents, the methodologies and insights have broad applicability to a wide range of emergency scenarios. By bridging the gap between theoretical modeling and practical implementation, this work lays the foundation for intelligent evacuation systems that can save lives and enhance public safety in an increasingly uncertain world.

5.3 Future Work

5.3.1 Offline Learning to Physics-Based Evaluation

One of the most significant contributions of this dissertation lies in the successful demonstration of offline-trained reinforcement learning (RL) agents being deployed and evaluated in a high-fidelity, physics-based simulation environment. This advancement bridges a critical gap between theoretical algorithm development and practical, real-time application in complex, dynamic environments such as ASIs. The ability to train agents offline, where computational resources are abundant and safety is not a concern, and then apply those learned policies in a realistic, real-time simulation represents a major step forward in the development of deployable intelligent evacuation systems.

Offline Training: Efficiency and Scalability

Offline training offers several advantages over online or real-time learning, particularly in safety-critical domains like emergency evacuation. First, it allows for extensive exploration of the state-action space without the risk of real-world consequences. In this work, both Q-Learning and N-Step Temporal Difference Learning agents were trained over hundreds of thousands of episodes in a DES, enabling them to experience a wide variety of scenarios, including different shooter locations, evacuee distributions, and environmental layouts.

Second, offline training is computationally efficient. Training in a DES environment is orders of magnitude faster than training in a physics-based engine like UE5, where

each simulation step must account for continuous-time dynamics, collision detection, and rendering. By decoupling the training phase from the deployment environment, this approach enables rapid iteration, hyperparameter tuning, and policy refinement without incurring the computational overhead of high-fidelity simulation.

Third, offline training supports reproducibility and explainability. The tabular Q-values generated during training provide a transparent mapping from states to actions, allowing researchers and practitioners to audit the decision-making process. This is particularly important in high-stakes applications where trust and accountability are paramount.

Application to Physics-Based Simulation: Realism and Transferability

The successful deployment of offline-trained agents in UE5 demonstrates the feasibility of transferring learned policies from abstract simulation to realistic environments. This is a non-trivial achievement, as the transition from DES to UE5 introduces several complexities:

- Continuous time and space: Unlike DES, which operates in discrete time steps and grid-based movement, UE5 simulates continuous motion governed by physics.
- Agent embodiment: In UE5, agents are embodied entities with physical dimensions, subject to collisions, inertia, and crowding effects.
- Perception realism: Line-of-sight in UE5 is determined by the agent's orientation and field of view, rather than abstract visibility graphs.

Despite these challenges, the N-Step Temporal Difference Learning agent maintained strong performance in UE5, consistently achieving the highest survival rates across a range of scenarios. This suggests that policies learned in simplified environments can generalize to more complex settings, provided that the state and action representations are carefully designed. In this work, the state space was defined as a tuple of the evacuee's and shooter's locations, and the action space was constrained to room-to-room transitions, allowing UE5's native navigation system to handle low-level pathfinding and collision avoidance.

This modular architecture—where high-level decision-making is handled by the RL agent and low-level execution is delegated to the simulation engine—offers a scalable blueprint for future intelligent evacuation systems.

5.3.2 Toward Real-Time, Adaptive, and Human-Centered Systems

Building on the foundation established in this dissertation, several promising directions for future research emerge:

Online and Continual Learning in Physics-Based Environments

While offline training is efficient and safe, it lacks adaptability. In real-world scenarios, conditions may change in ways not anticipated during training—new building layouts, unexpected obstacles, or novel threat behaviors. Incorporating online or continual learning capabilities into RL agents would allow them to adapt to these changes in real time. This could be achieved through techniques such as experience replay, meta-learning, or hybrid architectures that combine model-free and model-based components.

Training directly within UE5, while computationally intensive, would allow agents to learn from the full richness of the environment, including nuanced crowd dynamics and emergent behaviors. Advances in parallel simulation and GPU acceleration could help mitigate the performance costs.

Multi-Agent Reinforcement Learning (MARL)

The current work focuses on single-agent decision-making, where each evacuee acts independently based on their own state. However, real evacuations involve complex interactions among multiple agents, including cooperation, competition, and communication. Multi-agent reinforcement learning (MARL) offers a framework for modeling these interactions, enabling the development of coordinated evacuation strategies.

Future work could explore centralized training with decentralized execution (CTDE), where agents are trained with access to global information but act independently at runtime. This approach could support the emergence of leader-follower dynamics, group cohesion, and adaptive role assignment.

Integration with Real-Time Sensing and Communication Systems

For RL-based evacuation guidance to be deployed in practice, it must be integrated with real-time sensing systems that provide accurate and timely information about the environment. This includes shooter detection (via computer vision or gunshot localization), evacuee tracking (via RFID or smartphone signals), and environmental monitoring (fire or smoke sensors).

Future systems could use this data to update the state of the environment in real time and re-plan routes accordingly. This would require robust data fusion, low-latency communication, and fault-tolerant decision-making algorithms.

Human-in-the-Loop and Ethical Considerations

As intelligent evacuation systems begin to influence human behavior in life-or-death situations, it is essential to consider the human factors and ethical implications. Future research should explore how evacuees perceive and respond to algorithmic guidance, how to design interfaces that foster trust and compliance, and how to balance individual autonomy with collective safety.

Ethical frameworks must also be developed to guide decision-making in scenarios where not all evacuees can be saved. Questions about fairness, prioritization, and transparency must be addressed through interdisciplinary collaboration with ethicists, psychologists, and community stakeholders.

Generalization Across Environments and Threat Types

Finally, future work should aim to generalize the learned policies across different building layouts, population densities, and threat types. This could involve training agents on a diverse set of simulated environments or using transfer learning techniques to adapt policies to new domains. The ultimate goal is to develop robust, general-purpose evacuation agents that can be deployed in a wide range of emergency scenarios, from ASIs to fires, earthquakes, and chemical spills.

This dissertation demonstrates that offline-trained reinforcement learning agents can be effectively deployed in high-fidelity, physics-based simulations to guide evacuees during active shooter incidents. By combining the efficiency of offline training with the realism of UE5, this work lays the groundwork for intelligent evacuation systems that are both practical and powerful. Future research will build on this foundation to develop adaptive, multi-agent, and human-centered systems capable of saving lives in the most challenging of circumstances.

5.3.3 Deep Learning Agents in the DES

While not part of the proposed work, we pushed forward into the field of deep learning agents in the DES after seeing the successful implementation of DES training with UE5 execution. So far, we have attempted two approaches with varying degrees of success and continue to modify and improve our approaches in this arena.

Multi-Task Deep Q-Learning

We have developed and began training a Multi-Output Deep Q-Learning agent in our custom DES. The architecture for the network is in Figure 5.1. The input layer has 71 inputs containing the occupancy of each of the 70 nodes in the Cyclic Dorm Floor and the final input is the shooter's location. The network passes the inputs to three fully-connected hidden layers utilizing a Leaky ReLU activation function. The final hidden layer passes the "results" to the output layer consisting of 70 nodes with each one representing evacuee decision at each of the 70 nodes. The benefit of this approach is that there is a single input layer and all of the output tasks use the same hidden layers i.e. a single network is learning all of the actions for the nodes. The agent is a Double Deep Q-Learning agent meaning it is making use of an additional target-network to stabilize the training process and remove error due to rapidly changing value functions. Further, the agent is making use of prioritized memory replay which prioritizes "valuable" memories when using the memory batches to learn.

Unfortunately, our initial results in Figure 5.2 leave a lot of room for improvement. Using the single network, while it seems like it should be more efficient, requires a much larger network resulting in much longer time to complete single episodes (upwards of 30

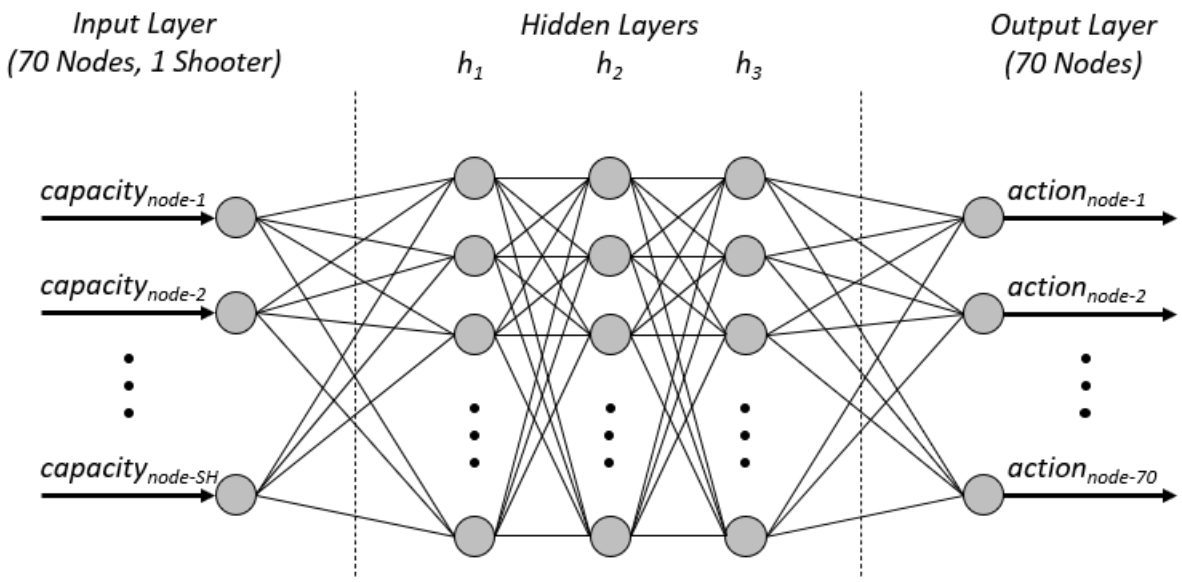


Figure 5.1: Multi-Output Deep Q-Learning Architecture

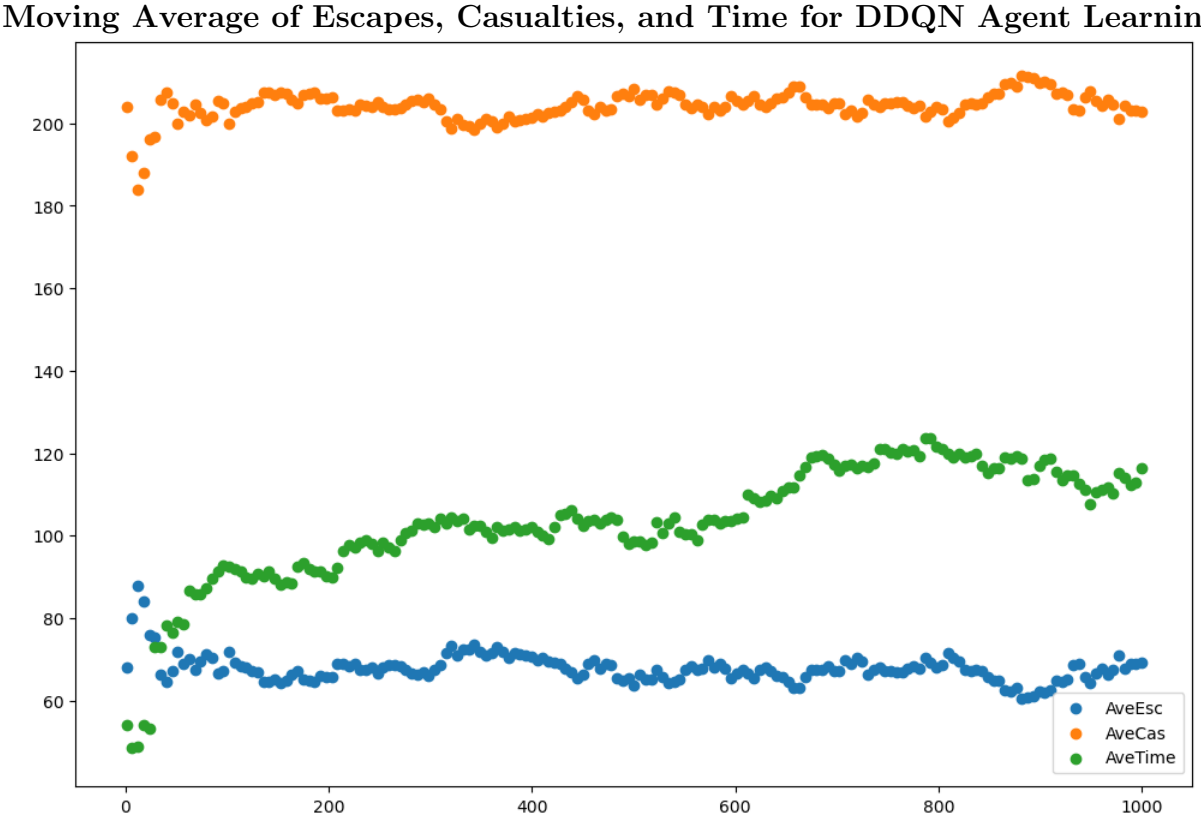


Figure 5.2: Multi-Task Double Deep Q-Learning Agent Learning

minutes). This makes training difficult, even in the DES, as it can take some time to see if the agent is learning or not. Figure 5.2 shows the learning of the agent over the first 1000 episodes. The graph is tracking the moving average of the last 50 episodes with regards to escapes, casualties, and the time to complete an episode. The casualties and escape seem to be in a steady state with very little changes, while the time to complete episodes is slowly increasing. This model approach may require significantly more training which we plan to continue to do on lab computers with better GPUs than what is currently being used.

Multi-Agent Deep Q-Learning

We are also training a Multi-Agent DQN with better success. Figure 5.3 shows a visualization of one of the architectures for the system. There are 70 individual agents in the system. Each agent's state space is made up of the occupancy of all of the adjacent nodes, including its own, and the shooter's current position so it is unique to each agent. We used this more restricted state space to try to simplify the inputs for the decision to encourage easier learning. The inputs are passed through three hidden layers and then to a single output layer that represents the decision for that particular node. Another potential benefit of this system is the individualized decision made by the agents would not require the computation of a larger system, further, computation requirements would be reduced if there are no evacuees in an agent's node because no action will be required.

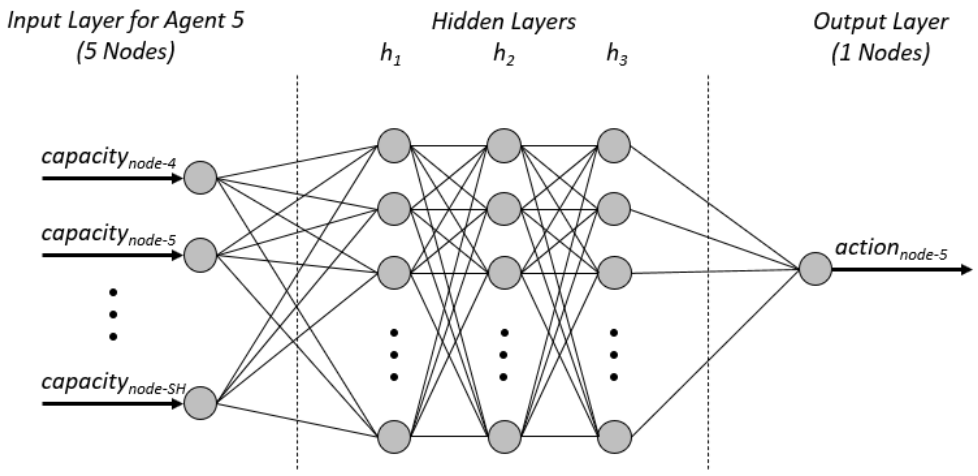


Figure 5.3: Multi-Agent Deep Q-Learning Architecture

The training for the multi-agent system started much better than the multi-output agent. Figure 5.4 shows the learning of the multi-agent system in the DES environment. The multi-agent system provided the benefit of much shorter episode times taking approximately 15 to 20 seconds for real-time to complete. Training was much faster and you are able to observe the improvements in the metrics of interest as escape slowly increases, casualties decreases, and the time per episode decreases. The ϵ-greedy policy that we used during training slowly decayed the epsilon value as actions were taken for each agent. In Figure 5.4, the ϵ decayed too quickly and around the 5000^{th} episode the agents regressed in their actions. It is likely the agents did not see enough training repetitions before ϵ decayed too far and the system became unstable. We plan to retrain the agents will a much slower ϵ decay in hopes that addresses the instability and continues to allow the agents to learn as they were prior to the 5000^{th} episode. This has also led us to push forward into other multi-agent systems.

Multi-Agent Actor Critic (MAAC) Approaches

Seeing the struggles the other two approaches had, we plan to continue to move towards more state of the art NN techniques for agent learning. The integration of DNNs and

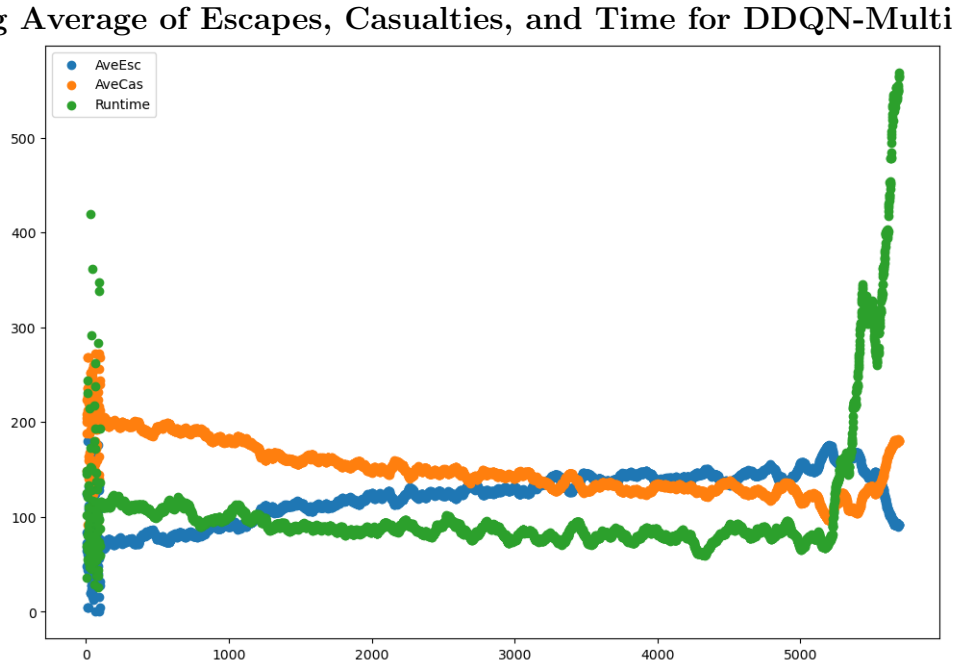


Figure 5.4: Multi-Agent Double Deep Q-Learning Network

MAAC frameworks could represent a further significant advancement in the development of intelligent evacuation systems, particularly for complex, high-stakes scenarios like ASIs. DNNs serve as powerful function approximators, enabling agents to generalize across large, continuous state spaces and learn directly from high-dimensional inputs such as spatial layouts which could include the occupancy and visualizations of crowds. This capability is especially valuable in dynamic environments where handcrafted state representations are insufficient. DQNs and recurrent architectures can be employed to process spatial and temporal information, allowing agents to make context-aware decisions that adapt to evolving threats and environmental conditions.

Multi-agent actor-critic approaches extend this capability by enabling decentralized coordination among evacuees. Using a centralized or shared critic with decentralized actor architectures, agents learn to cooperate, avoid congestion, and dynamically adjust their behavior based on shared goals and real-time updates. These methods are well-suited for modeling group dynamics, leader-follower behavior, and adaptive routing under uncertainty. When combined with real-time sensing systems and simulation platforms like UE5, DQNs and MAAC approaches offer a scalable, flexible foundation for next-generation evacuation planning—capable of learning from experience, adapting to new environments, and ultimately improving survival outcomes in ASIs and other emergency scenarios.

Bibliography

[1] Abdelgawad, H. and Abdulhai, B. (2009). Emergency evacuation planning as a network design problem: a critical review. *Transportation Letters*, 1:41–58. 26

[2] Abdulhalim, I., Mutch, C., González, V. A., and Amor, R. (2021). Improving post-earthquake evacuation preparedness for deaf and hard of hearing children: A conceptual framework. *International Journal of Disaster Risk Reduction*, 62:102360. 23, 60

[3] Abusalama, J., Razali, S., Choo, Y.-H., Momani, L., and Alkharabsheh, A. (2020). Dynamic real-time capacity constrained routing algorithm for evacuation planning problem. *Indonesian Journal of Electrical Engineering and Computer Science*, 20:1388. 28, 60

[4] Arteaga, C. and Park, J. (2020). Building design and its effect on evacuation efficiency and casualty levels during an indoor active shooter incident. *Safety science*, 127:104692. 5

[5] Arteaga, C., Park, J., Morris, B. T., and Sharma, S. (2023a). Effect of trained evacuation leaders on victims' safety during an active shooter incident. *Safety science*, 158:105967. 6

[6] Arteaga, C., Park, J., Morris, B. T., and Sharma, S. (2023b). Effect of trained evacuation leaders on victims' safety during an active shooter incident. *Safety Science*, 158:105967. 60

[7] Arteaga, C., Park, J., Morris, B. T., and Sharma, S. (2023c). Effect of trained evacuation leaders on victims' safety during an active shooter incident. *Safety Science*, 158:105967. 64

[8] Balboa, A., González-Villa, J., Cuesta, A., Abreu, O., and Alvear, D. (2020). Testing a real-time intelligent evacuation guiding system for complex buildings. *Safety Science*, 132:104970. 28, 60

[9] BELLMAN, R. (1957). A markovian decision process. *Journal of Mathematics and Mechanics*, 6(5):679–684. 15

[10] Casteigts, A., Himmel, A.-S., Molter, H., and Zschoche, P. (2021). Finding temporal paths under waiting time constraints. *Algorithmica*, 83(9):2754–2802. 13

[11] Chen, J., Shi, Z., Wang, X., and Wu, J. (2023). Uber: An unreal engine based simulation platform with extensibility and real-time capability. In *2023 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pages 322–327. IEEE. 108

[12] Chen, Z., Zhou, W., Wu, S., and Cheng, L. (2020). An adaptive on-demand multipath routing protocol with qos support for high-speed manet. *IEEE Access*, 8:44760–44773. 12

[13] Chou, J.-S., Cheng, M.-Y., Hsieh, Y.-M., Yang, I.-T., and Hsu, H.-T. (2019). Optimal path planning in real time for dynamic building fire rescue operations using wireless sensors and visual guidance. *Automation in Construction*, 99:1–17. 28, 60

[14] Cullen, D. (2020). *Parkland: Birth of a Movement*. Harper. 24

[15] Daamena, W. and Hoogendoorn, S. (2010). Capacity of doors during evacuation conditions. In *Procedia Engineering*, volume 3, pages 53–66. Elsevier Ltd. 66

[16] Dai, W., Singh, Y. P., and Zhang, R. (2024). Multi-agent simulation for mass school shootings. In *2024 IEEE International Conference on Big Data (BigData)*, pages 2714–2723. IEEE. 107

[17] Dalal, S., Seth, B., Jaglan, V., Malik, M., Surbhi, Dahiya, N., Rani, U., Le, D.-N., and Hu, Y.-C. (2022). An adaptive traffic routing approach toward load balancing and congestion control in cloud–manet ad hoc networks. *Soft Computing*, 26(11):5377–5388. 11

[18] Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271. 35

[19] Dong, K., Yang, D., Sheng, J., Zhang, W., and Jing, P. (2024). Dynamic planning method of evacuation route in dam-break flood scenario based on the aco-ga hybrid algorithm. *International Journal of Disaster Risk Reduction*, 100:104219. 10

[20] Dulebenets, M. A., Abioye, O. F., Ozguven, E. E., Moses, R., Boot, W. R., and Sando, T. (2019). Development of statistical models for improving efficiency of emergency evacuation in areas with vulnerable population. *Reliability Engineering & System Safety*, 182:233–249. 23, 60

[21] ElSherief, M., Saha, K., Gupta, P., Mishra, S., Seybolt, J., Xie, J., O'Toole, M., Burd-Sharps, S., and De Choudhury, M. (2021). Impacts of school shooter drills on the psychological well-being of american k-12 school communities: a social media study. *Humanities and Social Sciences Communications*, 8(1):1–14. 3

[22] Farmer, J. and Zhang, J. (2020). Run, hide, and fight to save your life. *Florida Journal of Educational Research*, 58(7):44–57. 3

[23] Follman, M., Aronsen, G., and Pan, D. (2022a). Can the next attack be prevented? a guide to mass shootings in america. 24

[24] Follman, M., Aronsen, G., and Pan, D. (2022b). Us mass shootings, 1982–2022: Data from mother jones' investigation. https://www.motherjones.com/politics/2012/12/mass-shootings-mother-jones-full-data/. 2, 24

[25] Follman, M., Aronsen, G., and Pan, D. (2022c). Us mass shootings, 1982–2022: Data from mother jones' investigation. 65

[26] Fox, J. A. and DeLateur, M. J. (2014). Mass shootings in america: moving beyond newtown. *Homicide studies*, 18(1):125–145. 2

[27] Froger, A., Jabali, O., Mendoza, J. E., and Laporte, G. (2022). The electric vehicle routing problem with capacitated charging stations. *Transportation Science*, 56(2):460–482. 13

[28] Georgiou, A. C., Papadopoulou, A., Kolias, P., Palikrousis, H., and Farmakioti, E. (2021a). On state occupancies, first passage times and duration in non-homogeneous semi-markov chains. *Mathematics*, 9. 25

[29] Georgiou, A. C., Papadopoulou, A., Kolias, P., Palikrousis, H., and Farmakioti, E. (2021b). On state occupancies, first passage times and duration in non-homogeneous semi-markov chains. *Mathematics*, 9. 64

[30] Gunn, S., Luh, P. B., Lu, X., and Hotaling, B. (2017). Optimizing guidance for an active shooter event. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4299–4304. IEEE. 6, 29, 60

[31] Haghpanah, F., Ghobadi, K., and Schafer, B. W. (2021). Multi-hazard hospital evacuation planning during disease outbreaks using agent-based modeling. *International Journal of Disaster Risk Reduction*, 66:102632. 28, 60

[32] Haley, J., Tucker, J., Nesper, J., Daniel, B., and Fish, T. (2023). Multi-agent collaboration environment simulation. In *Synthetic data for artificial intelligence and machine learning: tools, techniques, and applications*, volume 12529, pages 208–213. SPIE. 108

[33] Harris, L. M., Chakraborty, S., and Srinivasan, A. R. (2023). Use of immersive virtual reality-based experiments to study tactical decision-making during emergency evacuation. *arXiv preprint arXiv:2302.10339*. 60

[34] Hong, Y., Li, D., Wu, Q., and Xu, H. (2018). Dynamic route network planning problem for emergency evacuation in restricted-space scenarios. *Journal of Advanced Transportation*, 2018. 60, 62

[35] Hopcroft, J. E., Motwani, R., and Ullman, J. D. (2001). Introduction to automata theory, languages, and computation. *ACM Sigact News*, 32:60–65. 37

[36] Hu, T., Fu, Q., Pu, Z., Wang, Y., and Qiu, T. (2025). Unreal-map: Unreal-engine-based general platform for multi-agent reinforcement learning. *arXiv preprint arXiv:2503.15947*. 108

[37] Huang, P., Lin, X., Liu, C., Fu, L., and Yu, L. (2024). A real-time automatic fire emergency evacuation route selection model based on decision-making processes of pedestrians. *Safety science*, 169:106332. 7

[38] Islam, K. A., Chen, D. Q., Marathe, M., Mortveit, H., Swarup, S., and Vullikanti, A. (2024). Strategic routing and scheduling for evacuations. *arXiv preprint arXiv:2401.04371*. 11

[39] Johnson, S., Escamilla, L., Lehman, H., and Valasek, J. (2025). Reinforcement learning environment to realistic simulations for multi-agent system validation and deployment. In *AIAA SCITECH 2025 Forum*, page 1542. 108

[40] Karan, F. S. N. and Chakraborty, S. (2016). Dynamics of a repulsive voter model. *IEEE Transactions on Computational Social Systems*, 3(1):13–22. 60

[41] Karan, F. S. N., Srinivasan, A. R., and Chakraborty, S. (2017). Modeling and numerical simulations of the influenced sznajd model. *Physical Review E*, 96(2):022310. 56

[42] Khalili, S. M., Mojtahedi, M., Steinmetz-Weiss, C., and Sanderson, D. (2024). A systematic literature review on transit-based evacuation planning in emergency logistics management: Optimisation and modelling approaches. *Buildings*, 14(1):176. 9

[43] Kim, S., George, B., and Shekhar, S. (2007). Evacuation route planning. In *Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems*, pages 1–8. ACM. 63

[44] Kisko, T. M. and Francis, R. L. (1985). Evacnet+: A computer program to determine optimal building evacuation plans. *Fire Safety Journal*, 9. 27

[45] Kramer, E. N., Shipley, J., and Palm, C. (2023). Beyond "run, hide, fight". *Academic Psychiatry*, 47(5):490–491. 3

[46] Krolikowski, J., Martin, S., Medagliani, P., Leguay, J., Chen, S., Chang, X., and Geng, X. (2021). Joint routing and scheduling for large-scale deterministic ip networks. *Computer Communications*, 165:33–42. 12

[47] Lavalle-Rivera, J., Ramesh, A., Harris, L., and Chakraborty, S. (2022). The effectiveness of naive optimization of the egress path for an active-shooter scenario. *SSRN Electronic Journal*. 60, 64, 65, 79

[48] Lee, J. Y. (2019). Agent-based modeling to assess the effectiveness of run hide fight. Master's thesis, Purdue University. 3

[49] Lee, J. Y., Dietz, J. E., and Ostrowski, K. (2018). Agent-based modeling for casualty rate assessment of large event active shooter incidents. In *2018 Winter Simulation Conference (WSC)*, pages 2737–2746. IEEE. 60, 63

[50] Li, D., Zhang, Z., Alizadeh, B., Zhang, Z., Duffield, N., Meyer, M. A., Thompson, C. M., Gao, H., and Behzadan, A. H. (2024). A reinforcement learning-based routing algorithm for large street networks. *International Journal of Geographical Information Science*, 38(2):183–215. 10

[51] Li, G., Zhang, L., and Wang, Z. (2014). Optimization and planning of emergency evacuation routes considering traffic control. *Scientific World Journal*, 2014. 62

[52] Lin, J., Zhu, R., Li, N., and Becerik-Gerber, B. (2020). Do people follow the crowd in building emergency evacuation? a cross-cultural immersive virtual reality-based study. *Advanced Engineering Informatics*, 43:101040. 23, 60

[53] Lindekilde, L., Pearce, J., Parker, D., and Rogers, B. (2021). "run, hide, tell" or "run, hide, fight"? the impact of diverse public guidance about marauding terrorist firearms attacks on behavioral intentions during a scenario-based experiment in the united kingdom and denmark. *International Journal of Disaster Risk Reduction*, 60:102278. 3

[54] Liu, C., Mao, Z. L., and Fu, Z. M. (2016). Emergency evacuation model and algorithm in the building with several exits. In *Procedia Engineering*, volume 135, pages 12–18. Elsevier Ltd. 62

[55] Liu, H., Chen, H., Hong, R., Liu, H., and You, W. (2020). Mapping knowledge structure and research trends of emergency evacuation studies. *Safety Science*, 121:348–361. 23, 59

[56] Liu, R., Becerik-Gerber, B., and Lucas, G. M. (2025a). Investigating role of personal factors in shaping responses to active shooter incident using machine learning. *arXiv preprint arXiv:2503.05719*. 107

[57] Liu, R., Wu, W., Becerik-Gerber, B., and Lucas, G. M. (2024). Enhancing building safety design for active shooter incidents: Exploration of building exit parameters using reinforcement learning-based simulations. *arXiv preprint arXiv:2407.10441*. 5

[58] Liu, R., Wu, W., Becerik-Gerber, B., Lucas, G. M., Laboy, M., and Fannon, D. (2025b). Reinforcement learning for evaluating school safety designs in active shooter incidents. *Advanced Engineering Informatics*, 67:103575. 106

[59] Liu, Y., Zhang, H., Zhan, Y., Deng, K., and Dong, L. (2022). Evacuation strategy considering path capacity and risk level for cruise ship. *Journal of Marine Science and Engineering*, 10:398. 60, 63

[60] Lotero, S., Androulakis, V., Khaniani, H., Hassanalian, M., Shao, S., and Roghanchi, P. (2024). Optimizing fire emergency evacuation routes in underground coal mines: A lightweight network flow approach. *Tunnelling and Underground Space Technology*, 146:105637. 8

[61] Lowe, S. R. and Galea, S. (2017). The mental health consequences of mass shootings. *Trauma, Violence, & Abuse*, 18(1):62–82. 3

[62] Lu, Q., Huang, Y., and Shekhar, S. (2003). Evacuation planning: A capacity constrained routing approach. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2665. 27

[63] Lu, Q. and Shekhar, S. (2015). Capacity constrained routing for evacuation planning 1. 60, 63

[64] Lu, X., Astur, R., and Gifford, T. (2021). Effects of gunfire location information and guidance on improving survival in virtual mass shooting events. *International Journal of Disaster Risk Reduction*, 64:102505. 6

[65] Luca, M., Malhotra, D., and Poliquin, C. (2020). The impact of mass shootings on gun policy. *Journal of public economics*, 181:104083. 3

[66] Luh, P. B., Wilkie, C. T., Chang, S.-C., Marsh, K. L., and Olderman, N. (2012). Modeling and optimization of building emergency evacuation considering blocking effects on crowd movement. *IEEE Transactions on Automation Science and Engineering*, 9:687–700. 27

[67] Lujak, M. and Giordani, S. (2018). Centrality measures for evacuation: Finding agile evacuation routes. *Future Generation Computer Systems*, 83:401–412. 62

[68] Maulana, M. F. and Sari, D. N. (2024). Network analysis for determining the fastest evacuation routes in flood-prone areas of the tuntang watershed, indonesia. In *IOP Conference Series: Earth and Environmental Science*, page 012053. IOP Publishing. 9

[69] Metzl, J. M. and MacLeish, K. T. (2015). Mental illness, mass shootings, and the politics of american firearms. *American journal of public health*, 105(2):240–249. 2

[70] Michaels, S. (2015). The united states has had more mass shootings than any other country. 24

[71] Mirahadi, F. and McCabe, B. Y. (2021). Evacusafe: A real-time model for building evacuation based on dijkstra's algorithm. *Journal of Building Engineering*, 34:101687. 23, 27, 60

[72] Mizumura, T., Taguchi, H., and Nakamura, H. (2024). Confirming the safety improvement and evacuation time reduction effects by the method for evacuating inundated areas via the shortest possible route. *International Journal of Disaster Risk Reduction*, 101:104252. 9

[73] Müller, K., Vignaux, T., Scherfke, S., and Lünsdorf, O. (2023). Simpy. https://simpy.readthedocs.io/en/latest/. 38

[74] Naito, H., Takai, M., and Ishihara, S. (2024). Effect of aggressive evacuation behavior associated with information-sharing using a dtn on evacuation time. In *2024 International Conference on Information Networking (ICOIN)*, pages 434–439. IEEE. 10

[75] Naviagation360 and of Homeland Security, D. (2021). Active shooter definition. https://www.alicetraining.com/active-shooter/. 3, 30, 39

[76] Newman, B. J. and Hartman, T. K. (2019). Mass shootings and public support for gun control. *British Journal of Political Science*, 49(4):1527–1553. 3

[77] Nowakowski, R. and Winkler, P. (1983). Vertex-to-vertex pursuit in a graph. *Discrete Mathematics*, 43:235–239. 25, 60

[78] Pişirir, O. M., Bingöl, O., and Erkan, İ. (2024). Evacuation scenario optimization in buildings with human anthropometric characteristics. *Journal of Building Engineering*, 95:110033. 8

[79] Schildkraut, J., Elsass, H. J., and Meredith, K. (2018). Mass shootings and the media: Why all events are not created equal. *Journal of Crime and Justice*, 41(3):223–243. 2

[80] Shahabi, K. and Wilson, J. P. (2018). Scalable evacuation routing in a dynamic environment. *Computers, Environment and Urban Systems*, 67:29–40. 62

[81] Sharma, S. and Moses, P. (2025a). A collaborative virtual reality environment module for active shooter response training and decision making. *Electronic Imaging*, 37:1–7. 107

[82] Sharma, S. and Moses, P. A. (2025b). Immersive active shooter response training and decision-making environment for a university campus building. In *International Conference on Human-Computer Interaction*, pages 220–232. Springer. 106

[83] Sheu, J.-B. (2024). Mass evacuation planning for disasters management: A household evacuation route choice behavior analysis. *Transportation Research Part E: Logistics and Transportation Review*, 186:103544. 8

[84] Shultz, J. M., Thoresen, S., Flynn, B. W., Muschert, G. W., Shaw, J. A., Espinel, Z., Walter, F. G., Gaither, J. B., Garcia-Barcena, Y., O'Keefe, K., et al. (2014). Multiple vantage points on the mental health effects of mass shootings. *Current psychiatry reports*, 16:1–17. 3

[85] Silva, J. R. (2022). Global mass shootings: comparing the united states against developed and developing countries. *International Journal of Comparative and Applied Criminal Justice*, pages 1–24. 24

[86] Srinivasan, A. R. and Chakraborty, S. (2018). Study of hazard information propagation on evacuation efficiency with a hybrid simulation model. Technical report, University of Tennessee Knoxville. 60

[87] Srinivasan, A. R., Karan, F. S. N., and Chakraborty, S. (2017). Pedestrian dynamics with explicit sharing of exit choice during egress through a long corridor. *Physica A: Statistical Mechanics and its Applications*, 468:770–782. 56, 60

[88] Srinivasan, A. R., Karan, F. S. N., and Chakraborty, S. (2018). A study of how opinion sharing affects emergency evacuation. In *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*, pages 176–182. Springer, Cham. 27, 56, 60

[89] Sutton, R. and Barto, A. (1998). Reinforcement learning: An introduction. *IEEE Transactions on Neural Networks*, 9:1054–1054. 35

[90] Tamakloe, R., Hong, J., Tak, J., and Park, D. (2021). Finding evacuation routes using traffic and network structure information. *Transportation Research Part D: Transport and Environment*, 95. 62

[91] Topraklı, A. Y. and Satır, M. S. (2025). Architectural design for active shooter preparedness: A simulation-based systematic review. *Gazi University Journal of Science Part B: Art Humanities Design and Planning*, 13(1):97–122. 106

[92] Tsai, Y. L., Rastogi, C., Kitanidis, P. K., and Field, C. B. (2021). Routing algorithms as tools for integrating social distancing with emergency evacuation. *Scientific Reports*, 11. 60, 63

[93] Vidal, R., Shakernia, O., Kim, H., Shim, D., and Sastry, S. (2002). Probabilistic pursuit-evasion games: theory, implementation, and experimental evaluation. *IEEE Transactions on Robotics and Automation*, 18:662–669. 25, 60

[94] Wheeler, J. B. (2023). Reinforcement learning framework for the unreal engine. Master's thesis, California Polytechnic State University. 107

[95] Xiao, M., Shao, X., Gao, L., and Luo, Z. (2015). A new methodology for multi-objective multidisciplinary design optimization problems based on game theory. *Expert Systems with Applications*, 42:1602–1612. 25, 60

[96] Xiaobing, H., Liyan, Y., Hang, L., Yubo, Z., Yong, Z., and Qixuan, L. (2024). Optimization of emergency evacuation route based on ripple-spreading algorithm. *Journal of Transportation Systems Engineering and Information Technology*, 24(1):253. 8

[97] Yin, L., Chen, J., Zhang, H., Yang, Z., Wan, Q., Ning, L., Hu, J., and Yu, Q. (2020). Improving emergency evacuation planning with mobile phone location data. *Environment and Planning B: Urban Analytics and City Science*, 47:964–980. 27, 60

[98] Zacarese, L. M. (2015). Run, hide, fight... re-evaluate? *Campus Security Report*, 12(8):7–7. 3

[99] Zhang, Y.-Q., Wang, J.-H., Wang, Y., Jia, Z.-C., Sun, Q., Pei, Q.-Y., and Wu, D. (2024). Intelligent planning of fire evacuation routes in buildings based on improved adaptive ant colony algorithm. *Computers & Industrial Engineering*, 194:110335. 7

[100] Zhong, S., Cheng, R., Jiang, Y., Wang, Z., Larsen, A., and Nielsen, O. A. (2020). Risk-averse optimization of disaster relief facility location and vehicle routing under stochastic demand. *Transportation Research Part E: Logistics and Transportation Review*, 141:102015. 12

[101] Zhu, R., Lucas, G. M., Becerik-Gerber, B., Southers, E. G., and Landicho, E. (2022). The impact of security countermeasures on human behavior during active shooter incidents. *Scientific Reports*, 12(1):929. 6

Vita

Joseph Lavalle-Rivera grew up in Albion, ME. After attending high school in Fairfield, ME, he attended the United States Military Academy at West Point and received a Bachelor of Science degree in System Engineering. Upon graduation, he was commissioned as a 2nd Lieutenant in the US Army as an Air Defense Officer. He spent the first eight years in the US Army at Schofield Barracks, HI, FT Monroe, VA, and FT Sill, OK. He was selected to be an Operations Research and Systems Analyst and attended graduate school at the University of North Carolina at Chapel Hill receiving a Masters of Science degree in Interdisciplinary Statistics and Operations Research. Joseph then returned to the United States Military Academy to teach in the Department of Mathematical Sciences. After returning to FT Sill and leading a team of analysts, he was selected to return to graduate school to attain a Doctor of Philosophy degree. He chose to attend the University of Tennessee at Knoxville to pursue a Doctor of Philosophy degree in Data Science and Engineering. His research interests include Reinforcement Learning, Machine Learning, and Artificial Intelligence with a concentration on Agent and Multi-Agent modeling. After graduation, he will continue to serve in the US Army until he retires. He is incredibly grateful for all the support from his wife, children, and family that has carried him through this difficult process.