

A Deep Q-learning based Path Planning and Navigation System for Firefighting Environments

Manish Bhattarai^{1,2} and Manel Martínez-Ramón¹

¹University of New Mexico, Albuquerque, NM, 87106, New Mexico, U.S.A.

²Los Alamos National Laboratory, Los Alamos, NM, 87544, U.S.A.

Keywords: Path Planning, Navigation, Firefighting, Decision Making, Reinforcement Learning, Deep Q-learning, Situational Awareness.

Abstract: Live fire creates a dynamic, rapidly changing environment that presents a worthy challenge for deep learning and artificial intelligence methodologies to assist firefighters with scene comprehension in maintaining their situational awareness, tracking and relay of important features necessary for key decisions as they tackle these catastrophic events. We propose a deep Q-learning based agent who is immune to stress induced disorientation and anxiety and thus able to make clear decisions for firefighter navigation based on the observed and stored facts in live fire environments. As a proof of concept, we imitate structural fire in a gaming engine called Unreal Engine which enables the interaction of the agent with the environment. The agent is trained with a deep Q-learning algorithm based on a set of rewards and penalties as per its actions on the environment. We exploit experience replay to accelerate the learning process and augment the learning of the agent with human-derived experiences. The agent trained under this deep Q-learning approach outperforms agents trained through alternative path planning systems and demonstrates this methodology as a promising foundation on which to build a path planning navigation assistant. This assistant is capable of safely guiding firefighters through live-fire environments in fireground navigation activities that range from exploration to personnel rescue.

1 INTRODUCTION

Near-zero visibility, unknown hallways, deadly heat and flame, and people in dire need. These are the challenges firefighters face with every structure fire they respond to. Firefighters endure both extreme external conditions and the internal hazards of stress, panic, and disorientation as part of their daily job. Their central weapon against both internal and external hazards is their training on maintenance of situational awareness or understanding of the activities, and circumstances occurring in ones immediate vicinity. Maintaining situational awareness is key to a firefighters quick and apt response to an ever-changing environment and is critical to accurate decision-making. Situational awareness can be heavily impacted by both external hazards related to fire, and the corresponding internal stresses experienced by first responders. Loss of situational awareness is one of the main causes in the loss of life of firefighters on scene. Firefighters must make prompt decisions in high-stress environments, constantly assessing the situation, planning

their next set of actions, and coordinating with other colleagues, often with an incomplete picture of the situation. Situational awareness is the foundation of further decisions on how to coordinate both rescue operations and fire suppression. Firefighters on-scene pass their scene interpretations on via portable radio devices to field commanders for further assistance in decision making and the passing along of an inaccurate understanding of current conditions can prove disastrous. The limitation of this decision-making system is well reflected in the annual statistics by the US Fire Administration on the loss of human life¹. Existing fire fighting protocols present an excellent use case for institution of state-of-the-art communication and information technologies to improve search, rescue, and fire suppression activities through improved utilization of the data already being collected on-scene.

Firefighters often carry various sensors in their equipment, including a thermal camera, gas sensors, and a microphone to assist in maintaining their situational awareness but this data currently is used only

¹Firefighter Fatalities in the United States in 2017

in real-time by the firefighter holding the instruments. Such data holds great potential for improving the capability of the fire teams on the ground if the data produced by these devices could be processed with relevant information extracted and returned to all on scene first responders quickly, efficiently and in real-time in the form of an augmented situational awareness. The loss of situational awareness is at the core of disorientation and poor decision making. Advancement in computing technologies, small, cheap, wearable sensors paired with wireless networks combined with advanced computing methodologies such as machine learning (ML) algorithms that can perform all data processing and predicting utilizing mobile computing devices makes it not only possible, but quite feasible to create AI systems that can assist firefighters in understanding their surroundings to combat such disorientation and its consequences. This research presents a theoretical approach that can serve as the backbone upon which such an AI system can be built by demonstrating the power of deep Q-learning in building a path planning and navigation assistant capable of tracking scene changes and offering firefighters alternative routes in dynamically changing fire environments.

AI planning is a paradigm that specializes in design algorithms to solve planning problems. This is accomplished by finding a sequence of actions and addressing the needs and constraints to drive an agent from a specified initial state to a final state satisfying several specified goals. We utilize these paradigms to build a framework that teaches the agent about fire avoidance and deploys a decision process reactive enough to successfully guide the agent through simulated spaces that are as dynamic as those encountered in live fire events. Training in a simulated environment allows us the ability to test a multitude of situations and train the agent for exposure to a vast number of scenarios that would otherwise be impossible in real life. As a result, we get a vastly experienced pilot capable of presenting quick recommendations to a wide variety of situations. The presentation of this technology is meant to serve as the basis upon which to build a navigation assistant in future work.

2 PRELIMINARIES

The work in this paper is based on two distinct fields which are 1) path planning and navigation and 2) deep reinforcement learning.

2.1 Path Planning and Navigation

A large amount of work focused on path planning and navigation to aid firefighting has been done, but few works address dynamic, continuously changing environments. (Su and Su, 2012) proposes a mobile robot with various sensors to detect fire sources and use the so called A* search algorithm for rescue. An algorithm based on fire simulation to plan safe trajectories for an unmanned aerial system in a simulator environment is presented in (Beachly et al., 2018). (Jarvis and Marzouqi, 2005) shows the efficacy of the covert robotic algorithmic tool for robot navigation in high-risk fire environments. The usage of an ant colony optimization tool to automatically find the safest escape routes in an emergency situation in a simulator environment is shown in (Goodwin et al., 2015) whereas (Zhang, 2020) formulates the navigation problem as a "Traveling Salesman" problem and proposes a greedy-algorithm-based route planner to find the safest route to aid firefighters in navigation. (Ranaweera et al., 2018) proposes a particle swarm optimization for shortest path planning for firefighting robots, whereas (Zhang et al., 2018) proposes approximate dynamic programming to learn the terrain environment and generate the motion policy for optimal path planning for UAV in forest fire scenarios. A methodology for path reconstruction based on the analysis of thermal image sequences is demonstrated in (Vadlamani et al., 2020) which is based on the estimation of camera movement through estimation of the relative orientation with SIFT and Optical flow.

Despite the large quantity of work in the literature to aid the firefighters in path planning and navigation, most tend to solve the path planning considering a static environment where a one-time decision is made to guide the agent from source to destination. Such algorithms fail when the environment is dynamic. Furthermore, these algorithms do not allow for the agent to take immediate decisions when encountered with a sudden fire in the path of the chosen navigation path. We propose a deep reinforcement learning-based agent that is capable of taking an instantaneous decision based on learned experiences when subjected to sudden environment changes during navigation.

2.2 Deep Reinforcement Learning

Reinforcement learning (RL) is a technique that tends to learn an optimal policy by choosing actions based on maximizing the sum of expected rewards. Even though several works exist for path planning in fire environments, no RL based path planning implemen-

tations were found for fire scenarios. Outside of the fire scenario, several RL based path planning implementations do exist. (Romero-Martí et al., 2016) demonstrates an RL based navigation of a robot which is provided with a topological map. (Li et al., 2006) uses a Q-learning based path planning for an autonomous mobile robot for dynamic obstacle avoidance. An RL based complex motion planning for an industrial robot is presented in (Meyes et al., 2017). RL integrated with deep learning has demonstrated phenomenal breakthroughs that are able to surpass human-level intelligence for computer games such as Atari 2600 games (Mnih et al., 2013) (Mnih et al., 2015), AlphaGo zero (Silver et al., 2017) (Tang et al., 2017) along with various other games. In these frameworks, the AI agent was trained by receiving only the snapshots of the game and game score as inputs. Deep RL (DRL) has also been used for autonomous navigation based on inputs of visual information (Surrmann et al., 2020) (Kiran et al., 2020). (Bae et al., 2019) proposes a multi-robot path planning algorithm based on deep Q-learning whereas (Lei et al., 2018) demonstrates the autonomous navigation of a robot in a complex environment via path planning based on deep Q-learning (DQL) with SLAM. Most of these deep reinforcement learning-based path planning and navigation tasks are based on visual input i.e raw images/depth data which encodes the information about the environment. Based on this information, the navigation agents can establish the relationship between action and the environment. The agent in the DRL system embeds the action-policy map in the modal parameters of the neural nets.

Despite the efficacy of the DRL system in navigation, they are based on a learning experience of trial and error where the agent goes through numerous failures before actual success. Considering the hazardous behavior of the fire-environments, training, and evaluation of such DRL systems is very dangerous and practically infeasible. In addition to that, it is very expensive as well as time-consuming. To address these challenges, we developed the training environment for the RL agent in a virtual gaming environment Unreal Engine. The virtual environment depicts the actual firefighting scenario and enables the user to collect a large number of visual observations for action and reaction in various fire environments. The agent can interact with the environment through the actions and can also be trained with various user-defined rewards and goals. The framework also allows a plug and play option for the firefighting environment where one can depict a variety of fire scenarios from structural fire to wildfires for training the DRL agent.

In this paper, we introduce a DRL approach to train a virtual agent in a simulated fire environment. Taking advantage of simulation, we are able to expose our agent to a vast number of scenarios and dynamics that would be cost as well as safety prohibitive in real life but the results of the training can be applied to real life fire events. The resulting algorithm can be used in conjunction with other deep learning/machine learning approaches to produce a robust navigation assistant that can operate in real-time, effectively guiding fire fighters through a fire scene and aid their decision making by supplementing information gaps and situational awareness lags through the correct interpretation of the scenes they have passed and/or are currently in.

3 RL PROBLEM DESCRIPTION AND RL ARCHITECTURE

The virtual environment is achieved in a gaming platform Unreal Engine (Qiu and Yuille, 2016) depicting a fire scenario of burning objects and smoke. The gaming engine uses computational fluid dynamics (CFD) (Anderson and Wendt, 1995) based physics models to simulate a real-life dynamic situation where the parameters are a function of time. This gaming environment allows an external interaction where the agent can navigate in the scene via external controls such as a keyboard or head movement in a Virtual reality (VR) device. We take advantage of an interface software AirSim (Shah et al., 2018) that allows communication to and from the gaming environment to a deep learning framework (Tensorflow (Abadi et al., 2016)). AirSim can grab various parameters from the gaming environment such as RGB feed, Infrared feed, depth, and semantic map information corresponding to the scene and provides the feed to the python block. The python block then processes this information and dictates an agent's movement such as move forward, move backward, turn right, turn left, jump based on the deep reinforcement learning (RL) algorithm, and passes to the AirSim. AirSim further provides these control commands to the Unreal Engine environment which emulates these motions. We deploy a deep Q-learning agent that is trained on a policy-reward mechanism along with experience replay. For the experience replay, in addition to storing agent self play, we also recorded the user interactions with the environment where they were asked to safely navigate the environment, avoiding the fire and reaching the target in the given scene. With each new start, the user is asked to take different routes with the virtual agent to reach the destination while avoiding fire and

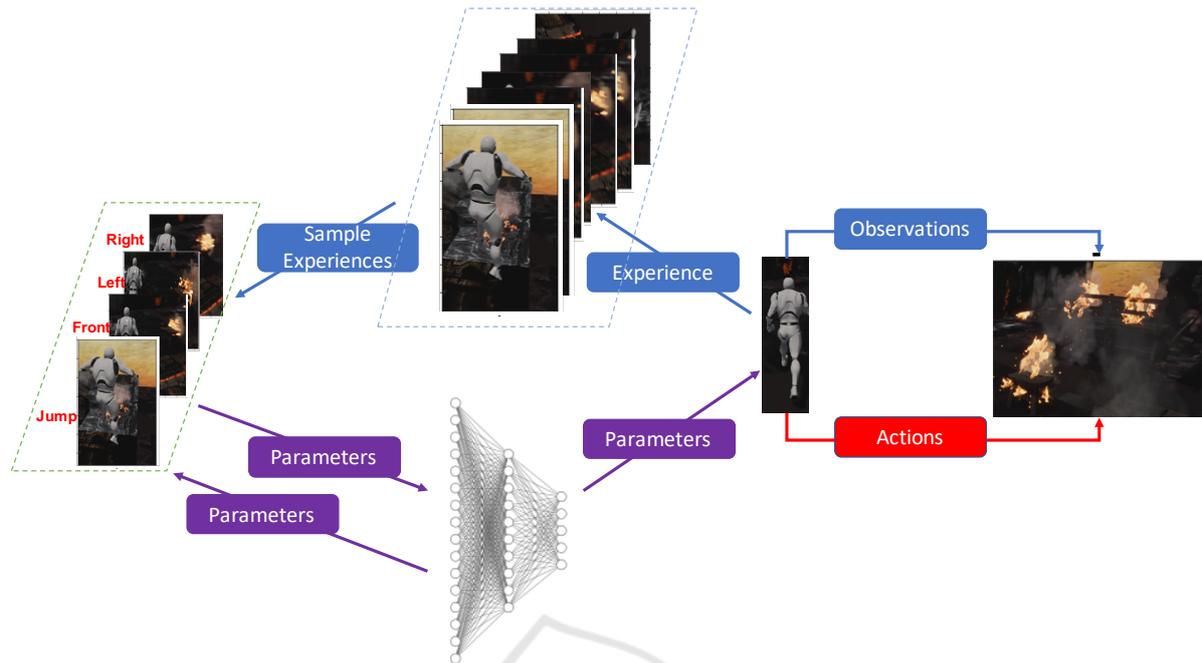


Figure 1: Deep Q-Network implementation.

the video frames and controls are recorded. During the training of RL agent, the sequence of frames and controls from the experience replay memory are provided to accelerate the training process and make the non-differentiable optimization problem converge in a reasonable time with better accuracy. The knowledge gained by the virtual agent on how to successfully navigate the virtual scene can then be transferred to a cyber-human system that can use this knowledge to interpret a real scene and provide step-by-step directions to firefighters to assist them in avoiding fire or other dangerous obstructions. The overview of the proposed DQN is shown in Figure 1.

Now, we define our objectives and various parameters associated with the proposed DRL framework.

3.1 Objective

The goal of the proposed deep Q-learning based agent is to reach the destination while safely navigating the fire in a dynamic environment. Safe navigation is defined as avoiding any contact with simulated fire. During the test, the agent needs to be prepared to make instantaneous decisions in instances where fire appears unexpectedly in the chosen navigation path. To achieve the best decisions under such situations, the agents can be subjected to many worst case situations during the training phase. The rewards need to be defined precisely to handle such task-driven learning.

3.2 Observations

The observations for the Q-learning framework are collected through the agent's field of view (FOV) from the virtual simulation environment (i.e Unreal Engine) using the AirSim app. The observations are received by the Python deep learning environment in the form of various feeds which include RGB, infrared, depth, and semantic map frames. Out of these, we are particularly interested in the infrared frames as the CNN framework is developed to perform recognition on thermal imagery. This CNN framework is based on (Bhattarai and MartiNez-Ramón, 2020). In real life scenarios, infrared cameras are the only feed type which can withstand extreme fire and smoke situations and improve visibility in heavy smoke. The virtual environment is also able to provide information about the camera position and the agent position, which will be helpful to locate the agent in the given 3D environment.

3.3 Actions

For ease of implementation and demonstration of proof of concept, we have transformed the action space from continuous to discrete space which comprises five primary agent motions. This discretization of the agent space also helps to reduce the model complexity. The five actions are move front, move back, turn left, turn right, and jump. With these motions, the agent can navigate in a structural building containing

obstacles like ladders and furniture. The same set of motions also enables the agent to navigate in wild-fire scenarios. The agent may take one or a combination of these actions to navigate along the fire scene to reach the destination.

3.4 Rewards

It is very important to define the direction of the goal while training an RL agent. To achieve a task-driven learning objective, it is vital to define the rewards to the navigating agent. The ultimate goal is to find a safe and minimal trajectory length to the navigation target. Unlike the trivial objective of finding the minimal trajectory length, the additional constraint of finding the safe path makes the optimization algorithm more complex. This results in a time-varying decision system whose instantaneous decisions are based on contemporary information of the environment. We introduce a reward and a penalty for familiarizing the agent to the fire environment. The fire has a penalty of -10 and a reward for reaching the goal is 10.

3.5 Problem Statement

The RL agent tends to pivot the actions in the direction of maximizing the rewards. The DRL system optimizes the hyperparameters of the Q-neural network to encode the experience of the agent for navigation. The backbone deep network aids the navigation by detection of the objects of interest for navigation. This information is then fed to the Q-network which then chooses the optimal actions to guide the agent. The idea of a DRL system is to provide an end to end learning framework for transforming the pixel information into actions (Mnih et al., 2015). Most of the DRL systems aim to learn the parameters for the neural network to find a transformation from state representations s to policy $\pi(s)$. Also, it is desirable to have an agent that can learn the navigation from a single environment and can generalize the experience to various environments. To achieve that, the aim is to learn a stochastic policy function π , which can process a representation of current state s_i and target state s_t to produce a probability distribution over action space $\pi(s_i, s_t)$. During the test, the agent samples an action from this distribution until it can achieve the destination. To summarize, the objective function that is used to assess the model performance is given in the form

$$z = g(x; \theta) = g(x; \beta(\theta); \theta) \quad (1)$$

where g is the navigation problem, which can be defined as finding the optimal actions with a DQN whose parameters θ and β are the parameters of the

navigation agent and x is the observation. z describes the navigation where a given fitness function $J(z)$ is applied to it in order to measure the optimality of the estimated decisions for navigation. For a given fixed set of neural net parameters β , the optimizer tends to seek optimal θ that determines the actions for the agent.

3.6 Model

The emphasis of this work is to find an optimal policy that can aid a firefighter to navigate in a fire setting via deep reinforcement learning. A deep neural network is trained for a non-linear approximation of the policy function π , where action a at time t is sampled as :

$$a \sim \pi(s_i, s_t | \beta) \quad (2)$$

where β corresponds to NN model parameters, s_i is the current observation frame, s_t is the target observation to which navigation is to be performed with action sequence a . Here, s_t belongs to a discrete set, π is a distribution function. The target scene can comprise fire victims which need help for rescue. So, once the deep learning (DL) model estimates the target to be rescued, the RL agent tends to propose navigation paths that successfully rescue the victim.

3.7 Q-learning and Deep Q-learning

We employ a variant of Q-learning called Deep Q-learning (DQL)(Mnih et al., 2013) to train an agent for navigating the fire to reach the destination safely. In this section, we briefly give an overview of the Q-learning and Deep Q-learning algorithms.

Q-learning learns the action-value function $Q(s, a)$ to quantify the effectiveness of taking an action at a particular state. Q is called the action-value function (or Q-value function). In Q-learning, a lookup table/memory table $Q[s, a]$ is constructed during training to store Q-values for all possible combinations of states s and actions a . An action is sampled from the current state, followed by computation of reward R (if any) and then the new state s . From the memory table, the next action a is determined based on the maximum of $Q(s, a)$. After this, an action a is performed to seek a reward of R . Based on this one-step look ahead, the target $Q(s, a)$ is set to

$$target = R(s, a, s') + \gamma \max_{a'} Q_k(s', a') \quad (3)$$

, where k corresponds to an episode. The update equations are called Bellman equations (Bellman, 1966) and are performed iteratively with dynamic programming. As this update is performed iteratively until convergence, a running average for Q is maintained.

Algorithm 1: Deep Q-learning algorithm for path planning agent.

```

1: Initialize replay memory  $\mathcal{R}$  to capacity N.
2: Initialize the Q-function  $Q(s, a)$  for all  $s, a$  with
   random weights.
3: for k in 1, 2, ... M do  $\triangleright$  each execution sequence,
   where k corresponds to an episode
4:   Initialize sequence  $s_1 = x_1$ 
5:   for t in 1, 2, ... T do  $\triangleright$  decision epoch
6:     With probability  $\epsilon$ , select a random action,
     otherwise select  $a_t = \max_a Q^*(s, a; \beta)$   $\triangleright$ 
     Exploration vs Exploitation Step
7:     Action  $a_t$  is performed by agent in the
     environment and corresponding rewards  $r_t$  and
     scene  $x_{t+1}$  is observed.
8:     Set  $s_{t+1} = s_t, a_t, x_{t+1}$ 
9:     Store  $s_{t+1}$  in  $\mathcal{R}$ .
10:    Sample a batch of transitions  $e_k =$ 
    ( $x_k, a_k, r_k, x_{k+1}$ ) from  $\mathcal{R}$ .
11:    if  $x_{t+1}$  is terminal then
12:       $y_k = r_k$ 
13:    else
14:       $y_k = r_k + \gamma \max_{a'} Q(s_{k+1}, a'; \beta)$ 
15:    end if
16:    Compute loss  $(y_k - Q(s_k, a_k; \beta_k))^2$  and
    then update neural net parameters  $\beta$  with gradient
    descent and back-propagation as per equations 4, 5 and 6.
17:  end for
18:
19: end for

```

However, for solving a real-world problem such as path planning and navigation, where the combinations of states and actions are too large, the memory and the computational requirements for Q is very expensive and intractable in some cases. To address that issue, a deep Q-network (DQN) framework was introduced to approximate Q(s, a) with the aid of neural network parameters. The associated learning algorithm is called Deep Q-learning. Based on this approach. we can approximate the Q-value function with the neural network rather than constructing a memory table for Q-function for state and actions.

An RL system needs to know the current state and actions to compute the Q-function. However, for our proposed simulation environment, the internal state information is not available. In one way, the state information can be constructed based on a recognition system that can identify the object of interest in the scene resulting in a discretization of the observation space by assigning pixels discrete values based on their identity. This objective is out of the scope of this paper and will be pursued in the future. For this

implementation, we only focus on observing a frame x_t from the emulator, which is a grayscale infrared image. Based on the action performed in the environment, the agent receives a reward r_t , along with a change in the internal state of the environment. Since we have defined a finite reward/penalty corresponding to specific states, the agent might need to go through a series of actions before observing any reward/penalty.

To estimate the Q-function, we consider the sequence of actions and observations for a game play episode. It is given as $s_t = x_1, a_1, x_2, a_2, \dots, a_{t-1}, x_t$. Considering t is a finite time where the game terminates either by reaching the target or getting burnt in a fire, this sequence can be formulated as a markov decision process (MDP). The goal of the agent is to choose the action that maximizes the sum of future rewards where the reward at time t is given as $R_t = \sum_{t_1=t}^T \gamma^{t-t_1} r_{t_1}$ for T being episode time. We then use a $Q^*(s, a)$ as optimal action-value function for a given sequence s and action a where $Q^*(s, a) = \max_{\pi} \mathbb{E}[R_t | s_t = s, a_t = a, \pi]$, π being the distribution over actions. To estimate this $Q^*(s, a)$, we use the deep neural network (Q-network) of parameters β as a non-linear function approximator in the form $Q(s, a; \beta)$ where it is expected that $Q(s, a; \beta) \approx Q^*(s, a)$. This network is trained with an objective of minimizing a sequence of loss functions $L_k(\beta_k)$ where,

$$L_k(\beta_k) = \mathbb{E}_{s, a \sim \psi(\cdot)} [(y_k - Q(s, a; \beta_k))^2], \quad (4)$$

Where $y_k = E_s[r + \gamma \max_a Q(s, a; \beta_{k-1}) | s, a]$ is the target for iteration k and $\psi(\cdot)$ is the probability distribution of sequences s and actions a.

The neural net then back-propagates the gradient given as

$$\nabla_{\beta_k} L_i(\beta_k) = \mathbb{E}_{s, a \sim \psi(\cdot)} \left[\left(r + \gamma \max_a Q(s, a; \beta_{k-1}) - Q(s, a; \beta_k) \right) \nabla_{\beta_k} Q(s, a; \beta_k) \right] \quad (5)$$

The parameters of the neural network are updated as

$$\beta_{k+1} = \beta_k - \alpha \nabla_{\beta_k} L_i(\beta_k) \quad (6)$$

where α is the learning rate of the neural network.

Furthermore, a technique called experience replay (Mnih et al., 2013) is used to improve convergence. This occurs through exposing the model to human-controlled navigation and decision making. To implement experience replay, the agent's experience $e_t = (s_t, a_t, r_t, s_{t+1})$ at each time step t is stored, where s_t is the current state, a_t is the action, r_t is the reward,

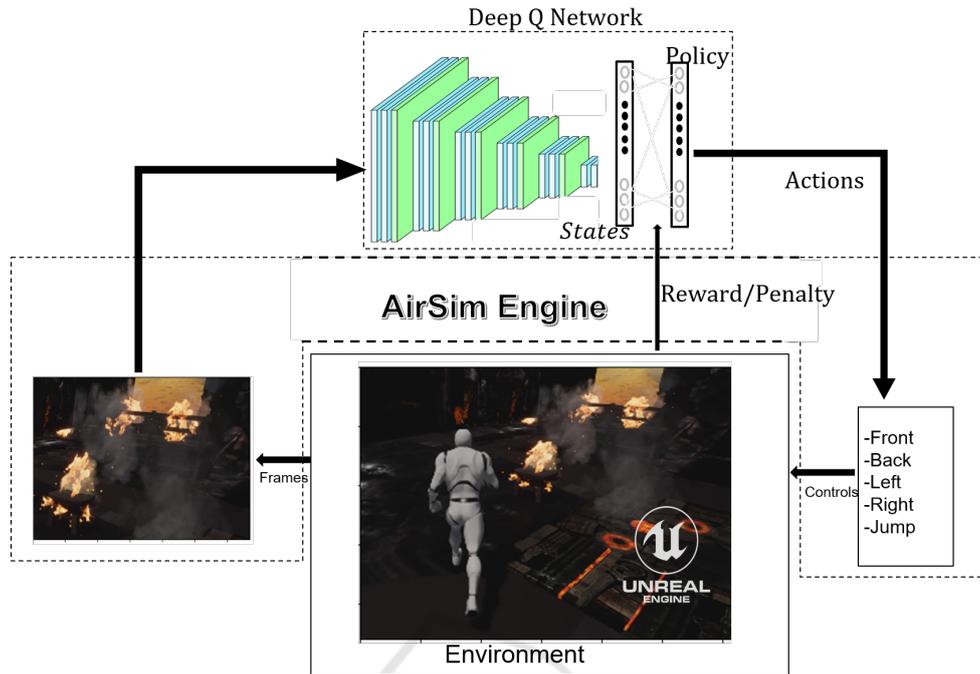


Figure 2: Architecture of Path Planning and Navigation system.

s_{t+1} is the next state on taking action a_t . The experience calculations presented result from the human-controlled navigation training and interaction with the environment. Based on the human interaction, various episodes $\{e_i\}_{i=1}^N$ are then stored in a memory buffer M . During the inner loop Q-learning updates, a sample of experiences are drawn randomly from the memory buffer M . The agent then selects and executes an action based on an ϵ -greedy policy. The approach of sampling randomly from experience replay enables the agent to learn more rapidly via improved exposure to reactions to different environmental conditions during an episode of training and allows the model parameters to be updated based on diverse and less correlated state-action data. The algorithm corresponding to the deep Q-learning is presented in Algorithm 1 and implementation methodology is presented in Figure 1.

3.8 Network Architecture

The DQN framework is built on top of a VGG-net-like framework (Bhattarai and MartiNez-Ramón, 2020) as a backbone and is shown in Figure 2. The backbone framework is used as a feature extractor that produces 4096-d features on a 224x224 infrared/thermal image. The VGG framework is frozen during the training. A stack of 4 history frames are used as state inputs to account for the past sequence of actions of the agent. Then the concatenated fea-

ture set comprising 4×4096 is projected into a 512-d embedding space. This vector is then passed through a fully-connected layer producing 5 policy outputs which give the probability over actions and value output.

4 EXPERIMENTAL RESULTS

The implementation of the DQN model was done in Tensorflow (Abadi et al., 2016) on a dual NVIDIA GeForce 1080Ti GPU. The DQN framework is trained with an RMSProp optimizer (Tieleman and Hinton, 2012) with a learning rate of 10^{-4} and batch size of 32. The training was performed with ϵ -greedy with ϵ started at 1 and decayed to 0.1 over 5,000 frames. The training afterwards was continued with ϵ of 0.1. The whole training was performed with as many as 100,000 frames with a replay memory of 20,000 frames.

While training, the gradients are back-propagated from the Q-layer outputs back to lower-level layers while the backbone model is frozen. The navigation performance was measured by the agent’s ability to reach 100 different targets set in a given fire environment. The target was placed at various locations in the virtual building on different floors where the agent needed to navigate using a combination of all actions. The actions of the agent in successful navigation are shown in Figure 3. The images correspond

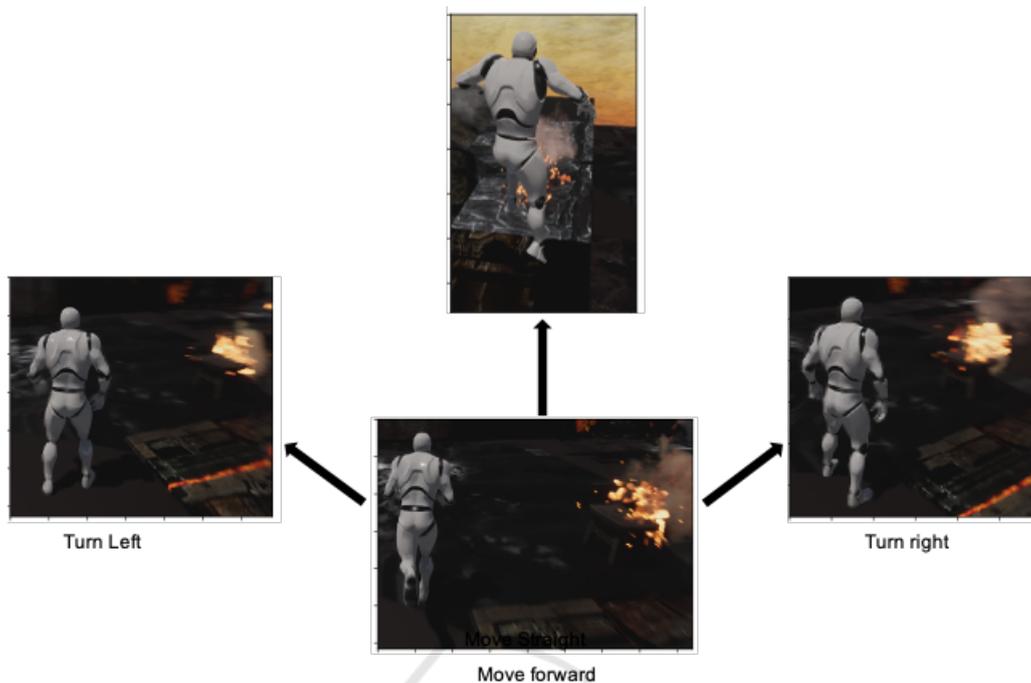


Figure 3: **Demonstration Of agent actions in fire Environment as dictated By reinforcement learning algorithm.** Four primary actions demonstrated respectively are turning left, right, jump, and moving forward. This enables the agent to avoid fire and obstacles and safely reach a given destination.

to the agent's action of move forward, turn left, turn right and jump. It is complicated to report the average trajectory length due to the constant changes occurring along the virtual path. In this simulation, the agent needs to avoid fire, and the trade-off for that is time. Imposing a time constraint and weighing the reinforcement model for rewarding or penalizing according to both strictures is a goal for future work. To prove the efficacy of the proposed method's learning, we worsen the situation by adding more fire occurrence. The fire volume per scene was increased from 10% to 80%, adding fire at random locations. The shortest path planning strategy failed to reach the destination for fire percentage 30%, and our proposed algorithm was able to navigate to the destination for the fire percentage as high as 76%. The agent was consistently able to navigate to the destination with a narrow escape. For extreme fire conditions, we carefully increased the rewards and defined additional penalties (distance to fire) to better the agent's learning condition. The deep Q-learning agent can only be trained under certain fire conditions, and it can exploit that knowledge to navigate under different fire conditions. For example, in this setup, the agent was trained for a fire percentage of 40%, and it was able to navigate for a fire percentage of as high as 76%.

The main goal of the proposed algorithm is to find the least number of combinations of actions that

helps the agent to navigate from the current position to the destination while avoiding fire. Due to the dynamic nature of the environment, when we attempted to solve this problem with other path planning techniques including shortest path technique, breadth-first search(BFS) (Beamer et al., 2012), depth-first search(DFS) (Tarjan, 1972), A*(LaValle, 2006) and random walk(Spitzer, 2013), the probability of the agent reaching the destination was very low (less than 5%) under the simulation environment. Since these methods use a single shot decision map to navigate the agent to the destination, the agent was unable to quickly adapt to the continuously varying surroundings. When the agent encountered fire which was not present before the decision, the agent failed to reach the target in most of the cases. In contrast, the agent governed by our proposed method was able to reach the destination with a probability greater than 80%.

During the evaluation of the agent's navigation performance with different path planning algorithms, including the proposed, we observed that a random initialization for the position of the agent and target point resulted in an increase of the distance between them with a larger probability. This leads to a higher chance of failing the agent to reach the destination. To address the agent's challenge, we selectively chose a distance parameter L between the agent and target, starting with a smaller value and increasing them over

the training allowing the agent to learn a better navigation experience for a continually changing environment leading to more extensive training samples distribution.

During the navigation, when the DQL trained agent is unable to find the path to proceed, we design penalties so the agent is constrained to either staying in the same position until the path is cleared or re-tracing its steps backwards to the previous possible path. We acknowledge that this part of the algorithm requires further attention, as remaining in one place in a real fire scenario is not realistic.

5 MOVEMENT PLANNING THROUGH DEEP Q LEARNING FOR FIREFIGHTING APPLICATION

We have demonstrated the potential deep Q-learning based algorithms hold as a base framework off of which a successful navigation assistant can be built. This methodology can provide an efficient decision-making system for aiding firefighters whose decision-making abilities may be impaired due to disorientation, anxiety, and heightened stress levels. This work presents a novel approach to eliminating faulty decisions made under duress through the application of AI planning paradigms. The paths followed by the firefighters are useful to determine their positions, which is particularly important in search and rescue.

Existing path planning algorithms can process the information of all paths followed by the firefighters but fail under the constantly changing nature of the fire ground which can make a previously defined rescue plan unavailable. Also, the presence of smoke and other visual impairments could make difficult the rapid identification of these incidents by a firefighter. Incidents in the fire ground are hardly predictable by a machine learning system. Machine learning does however, perform well in rapid assessment and production of a decision given the current set of circumstances. In other research outside of the scope of this paper, (Bhattarai and MartíNez-Ramón, 2020; Bhattarai et al., 2020) have developed a machine learning based methodology that detects and tracks objects of interest such as doors, ladders, people and fire in the thermal imagery generated by firefighter's thermal cameras. Such information may be valuable to further improve the reinforcement learning algorithm's ability to understand aspects of the environment that may be used in navigation or escape. Future work looks to incorporate a similar object detection work

with the path planning work described here to make a robust, navigation assistant that is capable of understanding the surrounding environment outside of fire presence and then recommend best paths to fire fighters. To deploy the agent in a real fire situation, we also aim to first construct the 3D map based on multimodal data (RGB, infrared and depth map) collected from various sensors attached to firefighter's body sensors. Such a map can be imported to the emulator to train the agent in more natural look-a-like environment.

Assuming that a path has been previously determined by the system by using the information coming from the camera of the rescuer, the rescuer has access to an initial rescue path. The system tells the rescuer to take a direction, which is the present action. The states will be represented by the objects present in the scene. The objects of interest can be represented in a matrix that contains the extracted feature's image. Each detected feature has a different reward. Fire and obstacles have associated penalties, while a clear path has a positive reward. If an obstacle is detected, then the firefighter is told to take a different direction. The new state will be computed for the action taken and a new path will be traced. The recursion repeats until the rescuer has reached the desired position. The plan for this part of the research will include an initial model constructed by simulation. This will be useful to determine the right design for the neural network in terms of stability and convergence speed in different simulated situations. When incorporating information from other paths followed by other fire fighters in a real scene, it is not evident that all obstacles can be determined by their past experiences due to the dynamic nature of a real fire scene. Nevertheless, the parallax estimation obtained from sequences of cameras in motion can be helpful. Parallax data can be used to determine the depth of a given path because it gives the distances between the camera and the key points detected by the SIFT algorithm. This information only needs to be stored and compared with future sequences of the same path. We can consider that an obstacle has been found in a previously clear path if the estimated depth has dramatically changed. In this case, the direction pointed by the camera will be given a low reward instead of a high one.

6 CONCLUSION AND FUTURE DIRECTIONS

We present a deep Q-learning based agent trained in a virtual environment that is able to make decisions for navigation in an adaptive way in a fire scene. The

Unreal engine was used to emulate the fire environment and AirSim was used to communicate data and controls between the virtual environment to the deep learning model. The agent was successfully able to navigate extreme fires based on its acquired knowledge and experience.

This work serves as the foundation on which to build a deep learning framework that is capable of identifying objects within the environment and incorporating those objects into its decision making process in order to successfully deliver safe, navigable routes to firefighters.

The learning process is currently slow and needs several hours of training. In the future, we aim to utilize A2C and A3C based reinforcement learning models to train a shared model utilized in parallel by multiple agents with multiple goals simultaneously. We also aim to use the deep learning-based results such as object detection, tracking, and segmentation to create a more informative situational awareness map of the reconstructed 3d scene.

The proposed system is intended to be integrated in a geographic and visual environment with data of the floor plan, which will also include scene information about the fire locations, doors, windows, detected firefighters, health condition of the firefighters and other features that are collected from the sensors carried in the fire fighter gear, which will be transmitted over a robust communication system to an incident commander to produce a fully flexed situational awareness system.

ACKNOWLEDGEMENTS

This work was supported by the National Science Foundation (NSF) Smart & Connected Communities (S&CC) Early-Concept Grants For Exploratory Research (EAGER) under Grant 1637092. We would like to thank the UNM Center for Advanced Research Computing, supported in part by the National Science Foundation, for providing the high-performance computing, large-scale storage, and visualization resources used in this work. We would also like to thank Sophia Thompson for her valuable suggestions and contributions to the edits of the final drafts.

REFERENCES

- Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283.
- Anderson, J. D. and Wendt, J. (1995). *Computational fluid dynamics*, volume 206. Springer.
- Bae, H., Kim, G., Kim, J., Qian, D., and Lee, S. (2019). Multi-robot path planning method using reinforcement learning. *Applied Sciences*, 9(15):3057.
- Beachly, E., Detweiler, C., Elbaum, S., Duncan, B., Hildebrandt, C., Twidwell, D., and Allen, C. (2018). Fire-aware planning of aerial trajectories and ignitions. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 685–692. IEEE.
- Beamer, S., Asanovic, K., and Patterson, D. (2012). Direction-optimizing breadth-first search. In *SC'12: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, pages 1–10. IEEE.
- Bellman, R. (1966). Dynamic programming. *Science*, 153(3731):34–37.
- Bhattacharai, M., Jensen-Curtis, A. R., and MartíNez-Ramón, M. (2020). An embedded deep learning system for augmented reality in firefighting applications. *arXiv preprint arXiv:2009.10679*.
- Bhattacharai, M. and MartíNez-Ramón, M. (2020). A deep learning framework for detection of targets in thermal images to improve firefighting. *IEEE Access*, 8:88308–88321.
- Goodwin, M., Granmo, O.-C., and Radianti, J. (2015). Escape planning in realistic fire scenarios with ant colony optimisation. *Applied Intelligence*, 42(1):24–35.
- Jarvis, R. A. and Marzouqi, M. S. (2005). Robot path planning in high risk fire front environments. In *TENCON 2005-2005 IEEE Region 10 Conference*, pages 1–6. IEEE.
- Kiran, B. R., Sobh, I., Talpaert, V., Mannion, P., Sallab, A. A., Yogamani, S., and Pérez, P. (2020). Deep reinforcement learning for autonomous driving: A survey. *arXiv preprint arXiv:2002.00444*.
- LaValle, S. M. (2006). *Planning algorithms*. Cambridge university press.
- Lei, X., Zhang, Z., and Dong, P. (2018). Dynamic path planning of unknown environment based on deep reinforcement learning. *Journal of Robotics*, 2018.
- Li, Y., Li, C., and Zhang, Z. (2006). Q-learning based method of adaptive path planning for mobile robot. In *2006 IEEE international conference on information acquisition*, pages 983–987. IEEE.
- Meyes, R., Tercan, H., Roggendorf, S., Thiele, T., Büscher, C., Obdenbusch, M., Brecher, C., Jeschke, S., and Meisen, T. (2017). Motion planning for industrial robots using reinforcement learning. *Procedia CIRP*, 63:107–112.
- Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.
- Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M.,

- Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533.
- Qiu, W. and Yuille, A. (2016). Unrealcv: Connecting computer vision to unreal engine. In *European Conference on Computer Vision*, pages 909–916. Springer.
- Ranaweera, D. M., Hemapala, K. U., Buddhika, A., and Jayasekara, P. (2018). A shortest path planning algorithm for pso base firefighting robots. In *2018 Fourth International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB)*, pages 1–5. IEEE.
- Romero-Martí, D. P., Núñez-Varela, J. I., Soubervielle-Montalvo, C., and Orozco-de-la Paz, A. (2016). Navigation and path planning using reinforcement learning for a roomba robot. In *2016 XVIII Congreso Mexicano de Robotica*, pages 1–5. IEEE.
- Shah, S., Dey, D., Lovett, C., and Kapoor, A. (2018). Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and service robotics*, pages 621–635. Springer.
- Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017). Mastering the game of go without human knowledge. *nature*, 550(7676):354–359.
- Spitzer, F. (2013). *Principles of random walk*, volume 34. Springer Science & Business Media.
- Su, H.-S. and Su, K.-L. (2012). Path planning of fire-escaping system for intelligent building. *Artificial Life and Robotics*, 17(2):216–220.
- Surmann, H., Jestel, C., Marchel, R., Musberg, F., Elhadj, H., and Ardani, M. (2020). Deep reinforcement learning for real autonomous mobile robot navigation in indoor environments. *arXiv preprint arXiv:2005.13857*.
- Tang, Z., Shao, K., Zhao, D., and Zhu, Y. (2017). Recent progress of deep reinforcement learning: from alphago to alphago zero. *Control Theory and Applications*, 34(12):1529–1546.
- Tarjan, R. (1972). Depth-first search and linear graph algorithms. *SIAM journal on computing*, 1(2):146–160.
- Tieleman, T. and Hinton, G. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31.
- Vadlamani, V. K., Bhattarai, M., Ajith, M., and Martinez-Ramon, M. (2020). A novel indoor positioning system for unprepared firefighting scenarios. *arXiv preprint arXiv:2008.01344*.
- Zhang, L., Liu, Z., Zhang, Y., and Ai, J. (2018). Intelligent path planning and following for uavs in forest surveillance and fire fighting missions. In *2018 IEEE CSAA Guidance, Navigation and Control Conference (CGNCC)*, pages 1–6. IEEE.
- Zhang, Z. (2020). Path planning of a firefighting robot prototype using gps navigation. In *Proceedings of the 2020 3rd International Conference on Robot Systems and Applications*, pages 16–20.