ÇANKAYA UNIVERSITY
COMPUTER ENGINEERING DEPARTMENT

CENG 407

LITERATURE REVIEW

A Review of the Computer Science Literature Relating to AI-Based Firefighting Vehicle

Efekan Karakoç

Ceren Erdoğan

Şeyma Sert

Esra Yalçın

Ömer Faruk Yıldırım

Abstract

The effectiveness of emergency responses has a critical role in providing fast and reliable fire services. Especially in heavy urban traffic, fire trucks must reach fire scenes quickly and safely for life-saving interventions. This study presents a literature review of a simulation project that uses the Unity game engine and applies reinforcement learning (RL) techniques. The simulation focuses on the adaptive and rational response of fire trucks to various dynamics in traffic. It focuses on how these vehicles can effectively navigate in traffic.

1. Introduction

The literature review examines the fundamentals of RL algorithms and their application to the Unity simulation environment, traffic simulations and navigation strategies, used RL algorithms and models, performance evaluation metrics, current challenges, future directions, and potential real-world applications.

The main reason why Unity was chosen is that it has the flexibility and power to simulate real-world conditions, thanks to its advanced graphics capacity, comprehensive physics engine, and AI integration tools such as the ML-Agents Toolkit. This simulation environment in Unity can model real-time traffic flow, various road and weather conditions, and emergency scenarios in detail. These features provide an ideal platform to analyze the adaptation and performance of RL algorithms to complex problems.

This literature review reveals the evolution of simulation-based fire truck navigation studies from the past to the present. It provides a comprehensive look at how RL algorithms can be used to provide safe and effective navigation in interaction with traffic. The practical challenges of implementing RL algorithms in the Unity environment and research to overcome these challenges are also discussed. The simulation is designed to improve behavior in accordance with traffic rules and the ability to make quick decisions in emergency scenarios, and the methods and algorithms used to achieve this goal are examined in detail.

2. Unity and Simulation Environments:

Thanks to its extensibility and versatility, Unity can also be used as a testing laboratory. The design of test environments is critical for developing and validating RL algorithms. In particular, when creating a fire truck simulation, it offers a flexible platform that can emulate various traffic and environmental conditions.

Testing Environment:

In the project, firstly, the test environment will be used for the initial training and validation phases of RL algorithms was developed. This testbed will conduct controllable experiments on traffic scenarios, road conditions, and vehicle behavior models. By creating complex cityscapes and traffic flow models, Unity provides an ideal framework for evaluating the agent's adaptation to different scenarios.

The fire truck's agent will encounter increasingly complex tasks in this test environment, starting with simple scenarios. Simple scenarios may involve moving directly on a single road or while remaining in a specific lane. Complex scenarios include making the right turns at intersections, prioritizing other vehicles in traffic, and reacting to sudden obstacles.

City Simulation:

After successfully completing the test environment, the second phase of the project, a realistic city simulation, is commissioned. This phase aims to demonstrate potential real-world applications of the project using the advanced features Unity offers. City simulation includes realistic city plans, various vehicle and pedestrian traffic densities, and various emergency scenarios.

This simulation environment will test whether the fire truck's agent can internalize traffic rules and apply the navigation strategies it has learned in real-world conditions. For example, it includes scenarios where the agent is randomly launched at different points in the city and is expected to find the fastest and safest path to the fire.

Unity's graphics and physics engines provide rich visual feedback and detailed analysis to evaluate the performance of the RL agent. In addition, Unity allows real-time data collection and monitoring features to examine the agent's decision-making processes and the results of these processes in detail.

In conclusion, a Unity-based testbed and city simulation provide the opportunity to evaluate whether RL-based navigation strategies are effective in theory and practical and realistic scenarios. The literature review should include research on how such simulations can be used for agent training and evaluation and guidance for the transition of simulations to real-world applications.

3. Fundamentals of Reinforcement Learning:

Reinforcement Learning (RL) is a paradigm in the fields of artificial intelligence and machine learning that relies on an agent learning optimal action strategies through trial and error. In this process, the agent acts according to the feedback from its environment, and this feedback is usually in the form of reward or punishment. The main purpose of RL is to develop behavior policies that will maximize the agent's total reward.

Basic concepts:

Agent: An entity that learns and makes decisions.

Environment: The physical or virtual space in which the agent interacts and receives state information.

State: All observable and/or unobservable information the environment has at any moment.

Action: The intervention that the agent can apply to the environment in any situation.

Reward: The feedback the agent receives as a result of an action. Positive rewards indicate correct behavior, and negative rewards indicate incorrect behavior.

Policy: A set of rules that determine what action the agent will choose in a situation.

Value Function: It refers to the expected total reward to be obtained under a policy, starting from a certain situation or situation-action pair.

Q-Value: It represents the value of a specific situation-action pair and is used to find the optimal policy.

Learning process:

In the RL process, the agent interacts with the environment and receives a reward after each interaction. Based on these rewards, the agent tries to measure the long-term consequences of its actions and learns the strategy that will achieve the maximum total reward.

Some of the basic RL algorithms used in this process are:

Monte Carlo Methods: Methods that learn at the end of each episode and try to maximize the average reward.

Temporal Difference (TD) Learning: Methods that are a combination of Monte Carlo and dynamic programming methods and use partial results to update predictions.

Q-Learning: It is a model-free algorithm in which the agent learns the best Q-value regardless of the policy.

Deep Q-Network (DQN): By combining Q-Learning and deep learning methods, it makes it possible to learn from high-dimensional data in complex environments.

Policy Gradients: Methods that directly optimize policy parameters and are generally preferred for continuous action areas.

Actor-Critic Methods: It is a two-component method that improves both value function and policy.

Application and Fire Truck Scenario:

Reinforcement Learning provides an effective solution to the problem of finding the optimal path for the vehicle to reach the fire in traffic in the fire truck scenario. The agent learns how to reach the fire scene as quickly as possible, obeying city traffic rules and without damaging other vehicles. During the training process, the agent's interactions with other vehicles in traffic in every situation it encounters are shaped by the rewards and punishments it receives from the environment. This learning takes place in simulation environments designed taking into account complex traffic dynamics and emergency scenarios.

4. Traffic Simulations and RL:

Traffic simulations are computer-based systems for analyzing the movement of vehicles and traffic control mechanisms by modeling real-world traffic flow. Reinforcement Learning (RL) is used in traffic simulations to develop strategies that optimize traffic flow, reduce traffic congestion, and clear paths for emergency vehicles.

Role of RL in Traffic Simulations:

Traffic simulations are ideal environments for developing and testing RL algorithms because they offer the opportunity to model complex traffic scenarios in real time safely. In these simulations, RL provides a flexible learning mechanism for the agent (fire truck) to learn the dynamics within the traffic and determine the optimal path.

In particular, in the traffic wayfinding problem for emergency vehicles, RL is investigated under the following headings.

Control at Traffic Lights: Algorithms that speed up the passage of emergency vehicles by optimizing traffic lights.

Road and Lane Selection: Policies that determine the fastest route, considering factors such as traffic density and road condition.

Agile Maneuvering: Algorithms that learn to maneuver without hitting other vehicles or violating traffic rules.

Simulation Tools and RL Algorithms:

Traffic simulation tools, software such as SUMO (Simulation of Urban Mobility) and VISSIM, are capable of performing traffic simulations on a large scale and offer rich visual experiences by integrating with Unity. RL algorithms are incorporated into these simulation platforms and used for agents to learn complex scenarios in traffic.

Using RL for Fire Truck Scenario:

In a fire truck scenario, the RL algorithm might follow these steps:

Observation: The agent observes information such as the traffic situation in its environment, nearby vehicles, the status of traffic lights, and road conditions.

Action: The agent selects a movement from a predetermined set of activities (e.g., speed up, slow down, change lanes).

Reward: The agent receives a positive or negative reward due to the action he chooses. Getting to the fire faster receives a positive reward, while breaking traffic rules or causing accidents gets a negative reward.

Learning: The agent updates its action strategies using the reward information it has collected.

This process allows the fire truck to navigate effectively in the complex traffic and emergency scenarios of the simulation. In particular, it aims to proceed quickly and safely in traffic, not to endanger other vehicles and to disrupt the traffic flow as little as possible.

Reinforcement Learning has the potential to produce innovative solutions to road-clearing and way-finding problems in traffic simulations, especially for emergency vehicles such as fire trucks. The development of RL algorithms and their integration with traffic simulations can provide valuable insights and strategies for real-world traffic management applications.

5. Fire Truck Navigation Strategies:

Navigation strategies of fire trucks focus on determining the shortest and safest route to optimize response time to emergencies. Shortest path algorithms play a fundamental role in solving this type of navigation problem.

Shortest Path Algorithms:

Dijkstra's Algorithm: Finds the shortest distance from the starting point to the destination, considering the distance from each node to neighboring nodes. Dynamically changing the weights according to the traffic situation allows this algorithm to be used effectively in traffic simulations.

Algorithm A: Predicts the least-cost path from the origin to the destination using a heuristic function. Fire trucks often prefer to minimize time and find an effective route without hitting obstacles.

Bellman-Ford Algorithm: It allows finding paths with negative edge weights in weighted graphs but works slower. It can be used when unexpected situations in traffic (for example, the sudden closure of a road) need to be taken into account.

Floyd-Warshall Algorithm: Finds the shortest paths between all pairs of nodes and provides a general solution for complex city maps, but may not be practical in large graphs due to its computational intensity.

Dynamic Routing Algorithms: They constantly update the best route using real-time traffic information. These algorithms can instantly react to dynamic events such as traffic congestion and road works.

Simulation and Real-Time Application:

In the Unity simulation environment, the application of the shortest path algorithm for the fire truck takes place on the simulated city map. These algorithms are integrated with Unity's physics engine and map data structures, serving as real-time navigation and decision-making mechanisms. Particularly in traffic simulations, the vehicle's environmental sensing capabilities (e.g., a virtual fire truck equipped with cameras and sensors) and artificial intelligence control points that manage traffic flow are critical to improving the algorithm's accuracy.

Shortest route finding for fire trucks is a critical factor in reducing emergency response times. Applying these algorithms in the Unity simulation environment can mimic real-world conditions and effectively model the challenges and response strategies that fire department responders may face. Thus, it increases the operational efficiency and effectiveness of the fire truck both in simulation and in real-world applications.

6. RL Algorithms and Models Used

In this project, we will try various Reinforcement Learning algorithms to enable the fire truck to navigate traffic effectively. These algorithms learn optimal decisions in complex traffic scenarios and adapt over time.

Basic RL Algorithms:

Q-Learning: It is one of the oldest and most common RL algorithms. It aims to maximize the reward obtained from actions taken from a certain situation. For the fire truck, the table of Q-values is updated for each condition (vehicle position, traffic situation, etc.) and action (change of direction, speed adjustment, etc.).

Deep Q-Networks (DQN): Adding deep learning techniques to the classical Q-Learning algorithm increases its applicability for large state spaces and complex tasks. It plays an essential role in helping a fire truck learn the complex traffic dynamics.

Policy Gradients: Optimizes a strategy that determines what action to take in each situation by making direct improvements over learned policies. It is helpful for fire trucks to learn traffic rules and yield scenarios.

Actor-Critic: It consists of two separate components: Actor and Critic. While the Actor determines the actions to be taken, the Critic estimates the value of these actions. This approach helps the fire truck not ignore its long-term goals while making instant decisions in traffic.

Proximal Policy Optimization (PPO): It is an advanced algorithm from the Policy Gradients family that increases the stability and reliability of the algorithm. It makes it easier for the fire truck to adapt to sudden changes in traffic quickly.

Twin Delayed Deep Deterministic Policy Gradients (TD3): It is an algorithm that combines the advantages of DQN and Actor-Critic methods and provides more stable learning by reducing noise in action selection. It is especially useful in situations that require sudden maneuvers in traffic.

Model-Based RL Approaches:

Model-based RL algorithms predict the consequences of actions using a model of the environment. They can thus provide learning with fewer trials. This is important for practical applications in real traffic because every wrong action can have serious consequences.

Simulation Integration:

Implementing these algorithms in the Unity environment allows the fire truck to experience realistic traffic conditions in a virtual environment and develop optimal decision-making mechanisms. Tools such as Unity's ML-Agents Toolkit provide the necessary interface and infrastructure for the integration and optimization of RL algorithms.

RL algorithms have the potential to improve the performance of fire trucks in traffic. Choosing the right algorithm and training it effectively is the key to successfully applying the strategies learned in the simulation environment in the real world. These algorithms are critical for developing robust and flexible decision-making capabilities for various situations to be encountered in traffic scenarios.

7. Performance Evaluation and Metrics:

A. Performance Metrics:

On-Time Arrival Rate (TVO): The percentage of the fire truck arriving at the fire scene on time is the most direct indicator of success.

Average Travel Time (AVT): The average time it takes to reach a fire is a measure of the efficiency of traffic flow.

Number of Collisions: Collisions between vehicles and the environment are an important metric in terms of safety.

Compliance Rate with Rules: The rate of compliance with traffic rules shows how much the vehicle complies with traffic laws.

Ride Smoothness (JS): Sudden accelerations, braking, and lane changes reflect the ride quality and impact the vehicle has on other drivers around it.

Response Time: The time it takes for the fire truck to reach the fire scene is one of the most critical performance metrics. A short response time ensures quick and effective fire response.

Priority Situations: The fire truck's ability to prioritize emergency vehicles can increase the speed of reaching the fire scene. This metric is used to evaluate the vehicle's interaction with other emergency vehicles.

Optimal Route Use: The fire truck's ability to choose the shortest and fastest route in traffic increases its effectiveness. This metric evaluates the distance of the vehicle's chosen route to the fire scene.

Disrupting Traffic Flow: How much the fire truck disrupts traffic flow is an important factor for other drivers. The ability to not disrupt traffic flow can increase the efficiency of urban traffic.

Total Rewards: The amount of rewards collected by RL algorithms during the learning process is a metric that reflects the performance of the agent. A higher reward total indicates better performance.

B. Simulation Based Evaluation:

Scenario Replay: Specific traffic and fire scenarios can be played multiple times in the simulation to evaluate the model's decision-making ability.

Multi-Environment Tests: The model should be tested in various conditions, such as different traffic density and different city maps.

Realism: How well the simulation reflects real-world conditions in terms of traffic flow, driver behavior and emergency procedures should be evaluated.

C. Statistical Analysis:

Decision Distribution Analysis: The statistical distribution of which decisions the fire truck prefers in which situations is valuable for understanding policy. It is especially important to examine under what conditions decisions are made, such as complying with traffic rules or reaching the fire zone quickly, to understand the priority order of decisions. This analysis will help us better understand the fire truck's behavior and make improvements where necessary.

Confidence Intervals and Variance: Statistical confidence intervals and variance of performance metrics indicate the consistency and reliability of the model. This statistical analysis helps us determine how much the simulation results are affected by random variations. Narrower confidence intervals and lower variance indicate that the model is more stable and reliable. This helps better predict the real-world performance of the fire truck.

D. Comparative Analysis:

Benchmarking: The model's performance should be compared with industry standards or results from similar studies. These comparisons help us identify areas where the developed model can be improved. It is also used to measure the ability to perform above or below industry standards. Benchmark results help us better understand the competitiveness and feasibility of the model.

A/B Tests: Comparative analysis of results obtained using different RL algorithms or hyperparameter settings. These tests help us determine where different model configurations perform better. For example, one algorithm may have reached the fire zone more quickly than others, while another algorithm may have followed traffic rules more strictly. This type of analysis helps us understand how the model should be tuned to achieve the best performance.

E. Feedback from Human Experts

Expert Evaluation: Fire experts and traffic managers can review the simulation results and evaluate them for realism and feasibility. Additionally, these experts can provide important insight into aligning fire truck behavior with real-world fire crews. These experts can offer important perspectives on how the fire truck can be better integrated into firefighting operations.

Surveys and Snapshots: Surveys and snapshots from drivers in real traffic situations can be used to understand the impact of the model. Surveys and interviews can be conducted to help drivers evaluate how they perceive the fire truck's behavior in traffic and the impact of these behaviors on their safety. This can be an important source of feedback on how the model can better serve the needs of society.

A comprehensive evaluation of the fire truck's performance reveals the strengths and weaknesses of the developed model and enables necessary improvements to be made before implementation in the real world. The metrics measured must accurately reflect the vehicle's effectiveness in traffic and its ability to react in emergency situations.

8. Challenges, Future Directions and Applications:

A. Current Challenges:

Simulation Realism: Virtual environments created in Unity do not fully reflect real-world complexity and unpredictability, which can make the transfer learning process difficult.

Computational Resources: High-resolution simulations and complex RL algorithms require significant processing power and can occasionally lead to resource constraints.

Sensing and Sensor Integration: Integrating precise sensor data that mimics real-world conditions into simulation remains a significant engineering challenge.

Algorithm Stability: The stability and reliability of RL algorithms are critical, especially in dynamic and constantly changing traffic environments.

B. Potential Directions for Future Research:

Transfer Learning: Developing methods that will enable successful transfer of the model from the simulation to the real world.

Multi-Agent Systems: Researching strategies that will enable multiple fire trucks to act in a coordinated manner.

Deep Learning Integration: Combining RL algorithms with deep learning techniques can improve perception and decision-making mechanisms.

C. Transfer Potential to Real World Applications:

Emergency Management: Integration of RL-based solutions in real-time traffic management and emergency response strategies.

Intelligent Transportation Systems: Using RL algorithms to optimize traffic flow and improve safety.

D. Technological Advances:

Artificial Intelligence and Machine Learning: Advances in AI models will increase the applicability of RL for traffic navigation.

Computer Hardware: Advanced hardware such as GPUs and dedicated processors will speed up simulation and model training processes.

Shortest Path Algorithms:

Dijkstra's Algorithm: Finds the shortest distance from the starting point to the destination, taking into account the distance from each node to neighboring nodes. Dynamically changing the weights according to the traffic situation allows this algorithm to be used effectively in traffic simulations.

Algorithm A: Predicts the least-cost path from the origin to the destination using a heuristic function. It is often preferred for fire trucks to minimize time and find an effective route without hitting obstacles.

Bellman-Ford Algorithm: It allows finding paths with negative edge weights in weighted graphs, but it works slower. It can be used when unexpected situations in traffic (for example, the sudden closure of a road) need to be taken into account.

Floyd-Warshall Algorithm: Finds the shortest paths between all pairs of nodes and provides a general solution for complex city maps, but may not be practical in large graphs due to its computational intensity.

Dynamic Routing Algorithms: They constantly update the best route using real-time traffic information. These algorithms can instantly react to dynamic events such as traffic congestion and road works.

9. Conclusion

This review guides future research on how RL algorithms can be applied in simulation and real-world scenarios in the Unity environment and provides a bibliography of essential studies in the field.

References

[1] Sutton, R.S., and Barto, A.G., "Reinforcement Learning: An Introduction", MIT Press, latest edition.

[2]Mnih, V. et al., "Playing Atari with Deep Reinforcement Learning", DeepMind Technologies, 2013.

[3]Silver, D. et al., "Mastering the game of Go with deep neural networks and tree search", Nature, 2016.

[4]Lillicrap, T.P. et al., "Continuous control with deep reinforcement learning", arXiv, 2015.

[5]Pan, Y., et al., "Virtual to Real Reinforcement Learning for Autonomous Driving", arXiv, 2017.

[6]Dosovitskiy, A., et al., "CARLA: An Open Urban Driving Simulator", arXiv, 2017.

Books:

"Deep Learning" by Ian Goodfellow, Yoshua Bengio, and Aaron Courville

"Artificial Intelligence: A Modern Approach" by Stuart Russell and Peter Norvig

"Python for Data Analysis" by Wes McKinney

"Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow" by Aurélien Géron

Online Resources and Documentation:

Unity Official Documentation and User Guides

Google AI Blog and DeepMind Publications

OpenAI Research Blog and Publications

IEEE Xplore Digital Library

arXiv e-Print archive