Comps Proposal

James Derrod
jderrod@oxy.edu
Occidental College

1 Problem Context

The prospect of integrating artificial intelligence (AI) into autonomous vehicles (AVs) heralds a potential transformation across various industries, notably in transportation. Autonomous vehicles promise to enhance road safety and efficiency by significantly reducing human error, the primary cause of traffic incidents. Yet, the transition from human-driven to fully autonomous vehicles involves navigating a complex landscape filled with technical, ethical, and logistical challenges.

My proposed project centers on deploying Deep Reinforcement Learning (DRL) to develop an AI agent capable of autonomously navigating a vehicle within a Unity-based simulated environment. Unity, a platform extensively utilized in game development, supports this endeavor through its MLAgents toolkit, which is adept at integrating machine learning agents for training, evaluation, and visualization purposes. The simulation setting provided by Unity offers a controlled environment to meticulously manage and observe variables impacting driving behaviors, essential for crafting robust AI systems.

Driving, inherently complex, involves dynamic and unpredictable elements such as changing road conditions, erratic human behaviors, and quick decision-making processes. These aspects elevate autonomous driving to a high-stakes, high-impact task that demands an AI system capable of precise and dependable decision-making. In this project, the driving challenge is conceptualized as a Markov Decision Process (MDP), which models decision-making in scenarios where outcomes are influenced by both randomness and strategic choices. This structured framework aids in training the AI to handle safe and efficient driving maneuvers as well as adapt to complex, unpredictable real-world scenarios.

While autonomous vehicles hold tremendous potential, they also introduce significant ethical and practical concerns, including data bias, security vulnerabilities, and the socio-economic implications of widespread automation. This project, however, operates at a much smaller scale, utilizing reinforcement learning to explore these broader issues within a manageable and contained framework. The goal is to refine and validate AI behaviors in simulated environments before any real-world application, thus mitigating risks associated with full-scale implementation.

Ultimately, this project aims not only to advance the capabilities of AI within controlled settings but also to contribute thoughtfully to the ongoing discussions about the ethical and sustainable integration of AI technologies into societal frameworks. This approach ensures that while striving for technological advancements, the broader implications and responsibilities are not overshadowed.

2 Technical Background

The development of an autonomous vehicle (AV) system via Deep Reinforcement Learning (DRL) incorporates a blend of computer science, mathematics, and engineering principles, especially focusing on areas like artificial intelligence and machine learning. This section outlines the foundational concepts and computational structures underlying the project, designed to be accessible to a computer science student familiar with data structures and computer organization, but not necessarily specialized in machine learning or AI.

2.1 Markov Decision Process (MDP)

Central to this project is the Markov Decision Process (MDP), a mathematical framework used for modeling decision-making in environments where outcomes are partly random and partly under the control of a decision-maker. An MDP is defined by its state space, action space, transition function, and reward function, each of which plays a crucial role in the learning process:

- State Space (**S**): In the context of autonomous driving, the state space represents all possible scenarios the AV can encounter. This includes variables such as the vehicle's position, velocity, the geometry of the road, traffic conditions, and the presence of obstacles. Each state encapsulates a snapshot of the environment from the perspective of the vehicle at any given time.
- Action Space (**A**): This is the set of all actions the AV can take in response to its current state. For driving, actions might include accelerating, braking, steering left, steering right, or maintaining current speed and direction. Actions are typically quantified, allowing

the AI to execute them to varying degrees (turning the steering wheel slightly vs. fully).

- Transition Function (**P**): The transition function models the probability of moving from one state to another given a particular action. This function is crucial for understanding the dynamics of the environment, as it encapsulates how actions affect the state of the vehicle and its surroundings. For example, if the AV decides to accelerate, the transition function would provide probabilities of resultant new states, which might include changes in speed or encountering new obstacles.
- Reward Function (**R**): This function assigns a numerical value to each action taken in a particular state, guiding the AI's learning process. Positive rewards are given for desirable outcomes (maintaining safe distances, adhering to traffic laws), and penalties are assigned for undesirable actions (collisions, speeding). The design of the reward function significantly impacts the learning and behavior of the AI, as it seeks to maximize cumulative rewards over time.

2.2 Deep Reinforcement Learning (DRL)

Reinforcement learning (RL) involves learning what to do, how to map situations to actions, so as to maximize a numerical reward. Deep Reinforcement Learning integrates deep learning with RL, utilizing neural networks to approximate the functions involved in the MDP, primarily the policy function (action selection given a state) and the value function (expected reward given a state and action).

The typical DRL cycle involves:

1. Observation: The agent observes the current state of the environment.
2. Decision: Based on a policy (initially random), the agent selects an action to perform.
3. Action: The agent performs the action, and the environment transitions to a new state.
4. Reward: The agent receives a reward based on the action's effectiveness.
5. Learning: The agent updates its policy based on the reward received and the new state of the environment.

2.3 Computational Aspects

The computational backbone of this project leverages the Unity platform's MLAgents toolkit, which provides the necessary infrastructure for implementing and testing DRL algorithms. The algorithms are typically implemented using Python, taking advantage of libraries such as TensorFlow or PyTorch for creating and training neural networks. These networks process large volumes of data generated during simulated driving scenarios, learning to predict the best actions from complex sensory inputs.

By situating this project within a controlled simulation environment like Unity, it allows for detailed manipulation of variables and systematic testing under varied conditions, essential for the iterative process of model refinement and validation in AI development.

This technical foundation supports the ambitious goal of training an AI to navigate complex driving scenarios autonomously, addressing both the immediate challenges of AI control and broader concerns regarding safety and ethics in AI applications.

3 Prior Work

Tesla's development of Autopilot and Full Self-Driving (FSD) systems represents a substantial real-world application of autonomous driving technologies. Although not based strictly on Deep Reinforcement Learning (DRL), Tesla's methodologies utilize advanced machine learning strategies that parallel DRL principles, especially in perception and decision-making. For the current project, Tesla's implementations provide a practical framework from which to evaluate the efficacy of DRL models in real-time, complex traffic environments, offering a benchmark for performance and safety metrics.

Other research into DRL technology within cars includes:

- An exploration of DRL-based control systems, particularly those employing Deep Q-Networks (DQN), offers insights into managing vehicle control and decision-making processes under dynamic conditions. These systems are crucial for understanding how autonomous vehicles can adapt to real-world driving scenarios with high reliability. For this project, the use of DQN can guide the development of algorithms capable of making split-second decisions in simulated environments that mimic real-world unpredictability. [1]
- A survey summarizing various DRL algorithms provides a structured approach to understanding their application in autonomous driving tasks. This includes a taxonomy of automated driving tasks, such as obstacle avoidance, path planning, and traffic navigation, addressing challenges like algorithmic delay and computational overhead. The insights from this survey are directly applicable to this project, aiding in the selection and optimization of DRL algorithms tailored to specific driving tasks in the simulated environment. [2]
- A review focusing on DRL applications to highway driving challenges, such as lane changing and ramp merging, highlights the necessity for precise and safe maneuvering capabilities in high-speed conditions. This prior work provides a foundation for this project's aim to implement and refine DRL strategies that effectively manage lane transitions and merging behaviors

in high-speed simulations, ensuring both safety and efficiency. [3]

4 Methods

This project utilizes Unity and the ML-Agents toolkit to simulate an environment for training an autonomous vehicle using Deep Reinforcement Learning (DRL). The following steps outline the methodology:

- **Environment Setup:** Using Unity, a virtual driving environment is created, complete with varied traffic scenarios, road conditions, and obstacles. This setup is designed to mimic real-world driving complexities, providing the AI with a comprehensive range of experiences.
- **Agent Design and Sensors:** The autonomous vehicle, or agent, is equipped with sensors (cameras, LIDAR) to perceive the environment. These sensors feed data into the neural network, enabling the vehicle to make informed decisions based on its surroundings.
- **DRL Implementation:** The core of the training involves a DRL algorithm, likely a variant of DQN or an actor-critic method, which will be used to train the agent in the simulated environment. The agent's objective is to maximize its cumulative reward by learning the optimal actions in various driving scenarios.
- **Data Handling and Neural Network Training:** Input data from the agent's sensors is processed and fed into a deep neural network, which learns to predict the best actions from different states. Training involves adjusting weights within the network based on the rewards received for actions taken in the environment.
- **Simulation and Testing:** After initial training phases, the agent is tested within the simulation to assess performance metrics such as safety, compliance with traffic laws, and efficiency. The simulation allows for controlled manipulation of variables to test the agent's response to different driving conditions.
- **Iterative Refinement:** Based on testing outcomes, the model undergoes further refinements to enhance decision-making accuracy and adaptability. This iterative process continues until the agent reliably meets the desired performance standards.

Through this methodology, I aim to develop a robust autonomous driving system capable of learning and adapting to complex driving environments, leveraging Unity's simulation capabilities to safely and effectively train and test AI models.

5 Evaluation Metrics

To effectively gauge the success of the autonomous vehicle system developed in Unity, this section outlines a comprehensive set of evaluation metrics designed to measure safety, efficiency, and compliance with real-world driving standards. These metrics not only benchmark the AI's performance against existing systems such as Tesla's Autopilot but also provide insights into areas requiring further improvement. Through both quantitative assessments and statistical analysis, the evaluation process aims to validate the AI's capabilities across diverse driving scenarios, ensuring its readiness for potential real-world application.

The evaluation of the autonomous vehicle project in Unity will focus on various performance metrics to ensure that the AI system meets safety and efficiency standards comparable to real-world systems like Tesla's Autopilot.

1. Safety Compliance: Metrics such as collision rate, adherence to traffic signals, and proper lane keeping will be quantified. A collision-free rate will be calculated as the percentage of test runs without incidents.
2. Efficiency Measures: Time to destination and fuel (or energy) efficiency will be tracked. These will assess how effectively the AI navigates routes in terms of both speed and resource utilization.
3. Rule Adherence: Compliance with traffic laws will be evaluated by checking the AI's adherence to speed limits, stop signs, and other traffic controls.
4. Behavioral Metrics: To evaluate the AI's ability to handle complex driving situations, scenarios such as emergency braking, obstacle avoidance, and response to sudden environmental changes will be measured.
5. Statistical Validation: Using statistical methods such as t-tests, the AI's performance will be compared against baseline models and real-world benchmarks like Tesla to determine significant improvements or the need for adjustments.
6. Generalization Capability: The model's ability to perform well across various unseen environments will be tested to ensure robustness. This involves evaluating the model on scenarios not included in the training set.

By assessing these metrics, my project aims to refine the AI's driving algorithms continually, ensuring they meet high safety and efficiency standards before any real-world application.

6 Timeline

- **(May 16 - May 31)** Initial Setup: Configure Unity environment and set up initial scenarios. Begin integration of basic ML-Agents toolkit.

- **(June 1 - June 15)** Agent Design: Develop and implement initial designs for the autonomous vehicle's sensor and control systems.
- **(June 16 - June 30)** Basic DRL Implementation: Implement initial Deep Reinforcement Learning algorithms and conduct preliminary tests.
- **(July 1 - July 15)** Extended Scenario Implementation: Introduce more complex driving scenarios into the simulation for broader training.
- **(July 16 - July 31)** Data Collection Phase: Begin intensive data collection from simulation runs for training the DRL models.
- **(August 1 - August 15)** Model Refinement: Refine neural network architectures and reinforcement learning algorithms based on early test results.
- **(August 16 - August 31)** In-depth Testing: Conduct in-depth testing and begin tuning parameters for optimization.
- **(September 1 - September 15)** Advanced Scenario Training: Introduce advanced driving scenarios and edge cases to train the AI.
- **(September 16 - September 30)** Performance Evaluation: Start comprehensive evaluation of AI performance against established metrics.
- **(October 1 - October 15)** Iterative Refinements: Implement feedback loops for iterative improvements based on evaluation results.
- **(October 16 - October 31)** Final Adjustments: Make final adjustments to algorithms and simulation environment based on beta testing feedback.
- **(November 1 - November 15)** Preparation for Final Presentation: Prepare for final presentation, focusing on documenting results and lessons learned.
- **(November 16 - November 30)** Final Presentation Rehearsals: Conduct rehearsals and final tweaks to the presentation based on feedback.
- **December 1 - December 14** Final Presentation and Documentation: Deliver final presentation and submit comprehensive project documentation.
- **(December 15)** Project Submission: Submit final version of the project along with all related documentation and code.

References

[1] Jin, Wei et al. "A Comparative Study on DRL-Based Autonomous Driving Under Single-Vehicle and Human-Vehicleroad Coordination". In: *2023 2nd International Conference on Automation, Robotics and Computer Engineering (ICARCE)*. 2023, pp. 1–5. DOI: 10.1109/ICARCE59252.2024.10492530.

[2] Kiran, B Ravi et al. "Deep Reinforcement Learning for Autonomous Driving: A Survey". In: *IEEE Transactions on Intelligent Transportation Systems* 23.6 (2022), pp. 4909–4926. DOI: 10.1109/TITS.2021.3054625.

[3] Liao, Jiangdong et al. "Decision-Making Strategy on Highway for Autonomous Vehicles Using Deep Reinforcement Learning". In: *IEEE Access* 8 (2020), pp. 177804–177814. DOI: 10.1109/ACCESS.2020.3022755.