Reinforcement Learning for Switching of Semiautomated Vehicles Under Uncomfortable Driving Situations

Ceren Ugurlu
Delft University of Technology

Abstract

Over the last two decades, autonomous driving has progressed from science fiction to a real possibility and rapidly developing. However, autonomous driving technology has significant weaknesses and is not safe in unexpected conditions. As a result, automobile manufacturers insist that the driver remains in the driver's seat even while the vehicle is in autonomous mode. Semi-autonomous driving helps in this situation. Semi-autonomous vehicles require minimum human intervention and cooperate with human drivers. It provides multiple levels of automation to the driver to give the optimal decision about who should be in charge in a particular scenario. However, it has limitations and does not work perfectly in all complex scenarios. This paper focuses on this limitation and provides a reinforcement learning strategy to solve an existing complex scenario. In this specific scenario, Mediator initiates a shift of control to a different automation level when uncomfortable driving situations are detected. Markov Decision Process strategy used to formulate the decision problem, and the reinforcement learning strategy compared with a decision-tree based baseline strategy for the evaluation. The outcome was collected using driver safety and comfort metrics. The outcome supports the hypothesis, demonstrating that a learned reinforcement learning strategy can be used to solve complex decision-making scenarios in semi-autonomous driving.

1 Introduction

Automated vehicle technology is rapidly evolving for all modes of transportation and has huge potential for safety. However, the automated system may not (yet) work properly in all cases, and if necessary, the human driver will be expected to take over. Mediator is a project which aims to develop a mediating system for drivers in semi-automated vehicles [1]. The project's goal is safely switching between the human driver and automated system in real-time based on who is fittest to drive. For this, the system continuously assesses the driving fitness of both in a variety of driving scenarios. This paper focuses on the decision logic module of the Mediator, which determines the actions concerning the transfer of control to ensure both driver safety and comfort.

The focus of this paper is on reinforcement learning techniques for dealing with decision-making problems that arise in the Mediator system. The current state of knowledge about the Mediator project's decision logic problem is limited. The most noteworthy project is work of Vermunt [2] which focuses on creating a proof of concept of a Markov Decision Process (MDP) that implements

the decision logic. The work presents a detailed background of the Mediator project and existing research challenges. It shows that MDP is a feasible model in this area but it does not solve the decision-making problem. The existing research about switching between driving entities in semi-autonomous vehicles has shown that switching between driving entities has benefits regardless of trip length [3]. They also create an MDP model to reduce the expected safety cost of a trip while taking into account the changes in the road/environment during the trip. However, it does not investigate specific complex scenarios. The previous researches [4, 5] indicates that reinforcement learning is appropriate for solving decision problems which occurs in semi-autonomous vehicles, and MDP can be used to model the decision problem.

The decision problem focused on this research is "What action should the Mediator system take when an uncomfortable situation is detected?". Specifically, Mediator initiates a shift of control based on drivers' personal preferences when uncomfortable driving situations are detected. This study focuses on uncomfortable events and the uncomfortable events are likely to make the driver feel uncomfortable. A previous study on comfort in autonomous driving identifies driving styles that are perceived as comfortable for automated driving [6]. It also suggests that a recommendation can be made for an autonomous driving style to feel driver comfortable. Since it is becoming more important to figure out how semi-automated vehicles should drive to keep the drivers comfortable, the action suggestions that the Mediator system will make to feel the driver comfortable is also important.

This paper aims to model the decision problem and to evalute the performance of RL algorithms compared to decision-tree based baseline policy. RL learns from its actions' results and selects its actions based on prior experiences (exploitation) and new options (exploration) [7]. There are numerous existing RL algorithms that can be applied to a wide range of problems. Deep Q-Learning, Advantage Actor Critic and Trust Region Policy Optimization algorithms are chosen from among those algorithms for this case. For that purpose, this paper constructs an MDP model to provide a mathematical formulation of the decision problem, then implements and evaluates the computational efficiency of the designed RL algorithm.

The main research question that will be investigated in this paper is "What is the performance of RL when solving the MDP problem in the specific use case of Mediator?". The use case is "Uncomfortable driving situations are detected and Mediator concludes that driver comfort might be impaired and reacts by suggesting an automation takeover (e.g., traffic jam) or handover to the driver (e.g.,time pressure). The following sub-questions accompany the main research question:

- How can the decision problem be formulated as an MDP?
- Based on the formulated MDP model, how to make the reinforcement learning algorithm efficiently address the decision problem?
- What is the performance of the learned policy regarding safety, comfort and efficiency?

The paper is structured as follows. First, a formal description of the problem is described, as well as theoretical background for MDP. This is followed by the explanation of the built MDP model. After that, the baseline and RL policies are explained. Then, the experimental setup and the results of the algorithms where performances were also evaluated are presented. The next section addresses the ethical aspects of the research and discusses the methods' reproducibility.

Finally, the research question is summarized and answered, along with some additional questions and improvement suggestions.

2 Methodology

In this section, uncomfortable driving situations for the semi-automated vehicle are defined and the assumptions are mentioned. The decision problem that involves the switching between the driver and automated vehicles is modeled as Markov Decision Process.

2.1 Uncomfortable Driving Situations

This research focuses on uncomfortable situations while semi-automated driving. Some uncomfortable situations have been picked to use during the research. The simplification was made since all uncomfortable situations cannot be examined. According to a previous study [8], the uncomfortable situations can be classified into two groups: potentially uncomfortable situations in manual driving and potentially uncomfortable situations in automated driving. Using the data from previous research, three events that occur in manual mode and one event that occur in automation mode are picked to use in this research. These events are as follows:

- Uncomfortable situations in manual driving:
  - **Traffic jam:** when the traffic is heavy, it might sense uncomfortable for the driver to be in control at all times.
  - **Poor visibility:** when the driver cannot see clearly, it might sense uncomfortable and unsafe for the driver to be in control at all times.
  - **Long trips:** when the trip is too long and monotonous, it might sense uncomfortable for the driver to be in control at all times.
- Uncomfortable situation in automated driving:
  - **Time pressure:** when there is time pressure, the driver might want to go faster or to use other roads. Therefore, the driver might sense uncomfortable in high automation.

Assumptions

Throughout the research, the following assumptions regarding uncomfortable events and driver comfort are made:

- When there is no uncomfortable event, assumed that the driver is comfortable and nothing is done.
- When the uncomfortable events are "Traffic jam", "Poor visibility" or "Long trips", assumed that the driver is in manual mode since these events potentially occur in manual mode. It is also assumed that the driver will not feel comfortable in manual mode under these situations, thus shift to the optimal level is suggested or shifted to the optimal level.
- When the uncomfortable event is "Time pressure", assumed that the driver is in automation mode since this event potentially occurs in automation mode. It is also assumed that the driver cannot feel comfortable in automation mode under this situation, thus shift to manual mode is suggested or shifted to manual mode.

2.2 MDP Formulation

A Markov decision process (MDP) is a discrete-time stochastic control process which provides a mathematical framework to model decision making. MDP models contain finite number of states and actions. At each point time, the MDP can be utilized by taking actions and observing the outcome states and rewards. The goal of solving the MDP is to minimize costs or maximize rewards. In order to formulate a problem as an MDP, 4 different elements which are S, A, T, R need to be defined. Their definitions are as follows:

- S - State Space
- A - Action Space
- $T(s'|s,a)$ - Transition Function
- $R(s,a)$ - Reward Function

State Space

The state space is the customized version of the Mediator system's components. It contains sensor observations from the human driver, the vehicle, and the environment. The observations and action types are used for communication between the decision logic agent and the Mediator simulator. Four main components of the state space are as follows:

- **Automation State:** vehicle's current driving capabilities. Includes various indicators such as current and maximum automation level.
  - Auto Level: indicates current automation level. There are 4 possible level values between 0 and 3. Values 0, 1, 2, 3 represent L0, L2, L3, L4 respectively. L0 is manual driving, L2 is partial automation, L3 is conditional automation and L4 is high automation.
  - Level Max: indicates current maximum available automation level.
- **Context State:** gives context information about road/environment. Includes various indicators such as uncomfortable events and leaving operational design domain (ODD).
  - Uncomfort Event: the uncomfortable events are likely to make driver feel uncomfortable. There are 5 possible event values between 0 and 4. 0 represents no event when there is no uncomfortable event. Values 1, 2, 3, 4 represent traffic jam, poor visibility, long trips and time pressure respectively.
- **Time Metrics State:** predicts the time for state changes. The values are maximum of 9999 seconds for simplicity to represent arbitrarily large values. Includes various indicators such as TTDD, TTDF, TTAFs, and TTAUs.
  - TTDD: indicates time to driver discomfort. Uncomfortable events are likely to increase driver discomfort.
  - TTDF: indicates time to driver fitness, which describes the estimated time before a driver is able to safely perform the manual driving task.

  - TTA2F, TTA3F, TTA4F: indicates time to automation fitness, which describes how much time is left before an automation level will become available. Automation fitness is mainly impacted by ODD.
  - TTA2U, TTA3U, TTA4U: indicates time to automation unfitness, which describes how much time is left before an automation level will no longer be available.

- **Feedback State:** driver's interactions with the system. Includes various indicators such as driver response and last action.
  - Driver Response: decision logic (DL) generates a shift action with preference to the human-machine interface (HMI), and HMI negotiates with the driver about the action. The feedbacks that HMI can provide to decision logic is the driver's choice. The default value is -1, after DL action values between 0 and 2. Values 0, 1, 2 represent no response, accept, and reject respectively.
  - Last Action: indicates the type of decision logic action at the previous time. The default value is doing nothing, determined by DL action.

Action Space

The action space refers to the set of actions the system can perform. Instead of providing different actions per automation level, the optimal automation level (L_{opt}) term is used in actions. L_{opt} states the optimal level to shift to in the current state. L_{opt} is defined as follows:

- If the uncomfortable event occurs in manual driving, L_{opt} corresponds to L_{max}.
- If the uncomfortable event occurs in high automation, L_{opt} corresponds to L0.

The action space consists of the following actions:

- **Do Nothing (DN):** No change takes place withinside the state of the vehicle
- **Suggest Shift L_{opt} (SSL):** The mediator suggests the driver shift to the optimal automation level.
- **Shift L_{opt} (SL):** The mediator performs a shift from the current level to the optimal automation level.

Transition Function

In this section, the transitions will be represented separately for each action. A transition consist of three parts: initial state, action and resulting state. In the following figures, the circles represent states, the squares represent actions and lastly the arrows represent transition probabilities.

The decision logic only shifts to automation levels that are considered safe and comfortable. The assumptions about which levels are considered comfortable are mentioned in section 2.1. The transitions designed per action and their probabilities are as follows:

Do Nothing: no change in the resulting state. The initial state and the resulting state are the same. Regardless of the other variables, the transition probability is always 1 in the 'Do Nothing' action. The transition with DN action is shown in Figure 1.

$$P(S'|S, DN) = 1$$

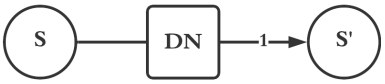


Figure 1: Representation of transition with DN action

Suggest Shift L_{opt}: the transition can end up in three different resulting states depending on the driver response. There are three possibilities: the driver may not respond, accept or reject. The probabilities of 'no response', 'accept' and 'reject' responses in the environment are considered to be 0.1, 0.8, and 0.1, respectively. Therefore, the transition probabilities of the S_N and S_R resulting states obtained in the 'Shift L_{opt}' action are 0.1. Finally, for the resulting state S_A, the transition probability in the 'Suggest L_{opt}' action is 0.8. The transition with SSL action is shown in Figure 2.

$$P(S_N|S, SSL) = P(S_R|S, SSL) = 0.1, P(S_A|S, SSL) = 0.8$$

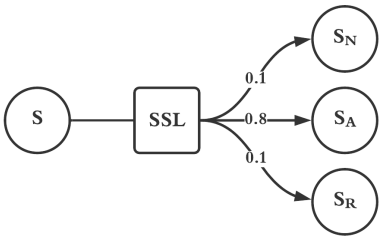


Figure 2: Representation of transition with SSL action

Shift L_{opt}: the automation level changes to L_{opt} in the resulting state. Since 'Shift L_{opt}' is an enforced action, the transition probability is always 1. The transition with SL action is shown in Figure 3.

$$P(S'|S, SL) = 1$$

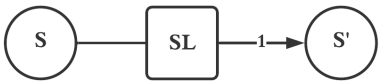


Figure 3: Representation of transition with SL action

Reward Function

In this section, the reward function will be represented using decision tree. The tree's leaf values represent reward values, while the other nodes represent conditions. The colors of the leaves reflect whether or not the action taken is desired. As a result, green leaves state rewards while red leaves state costs. The designed reward tree is shown in Figure 4.

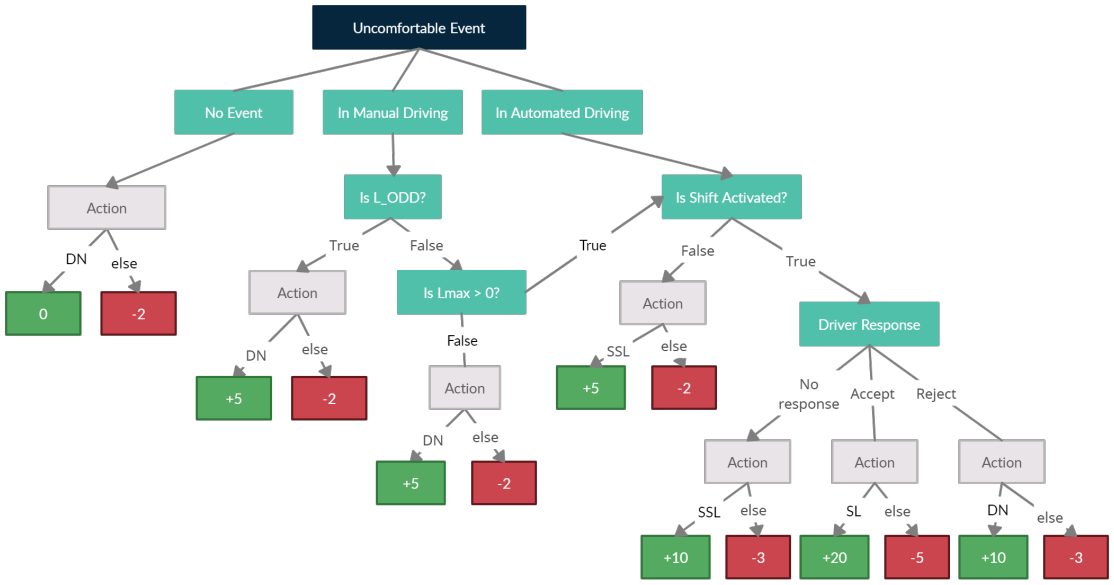


Figure 4: Reward decision tree

3 Experimental Setup

This section shows the steps taken to set up and perform the experiment. Section 3.1 describes baseline policy. After that Section 3.2 explains reinforcement learning policies that will be used on the MDP. Finally, the evaluation metrics to evaluate the policies are defined and the steps taken to conduct the experiment are described.

3.1 Baseline Policy

In this section, after the decision-making problem is formulated as an MDP, the baseline policy is developed to deal with the problem. A policy defines the learning way at a given time. It maps the perceived states of the environment to the actions taken on those states. In a policy-based approach, the goal is to find the optimal policy for the maximum future rewards. A policy should be applied where the action carried out at each step helps maximize future rewards.

The baseline policy uses a decision tree to go from observations represented in the branches to actions represented in the leaves. In this way, the action to be taken in the next step is decided using fixed conditions. The designed decision-tree based baseline policy for the uncomfortable events and related actions is shown in Figure 5.

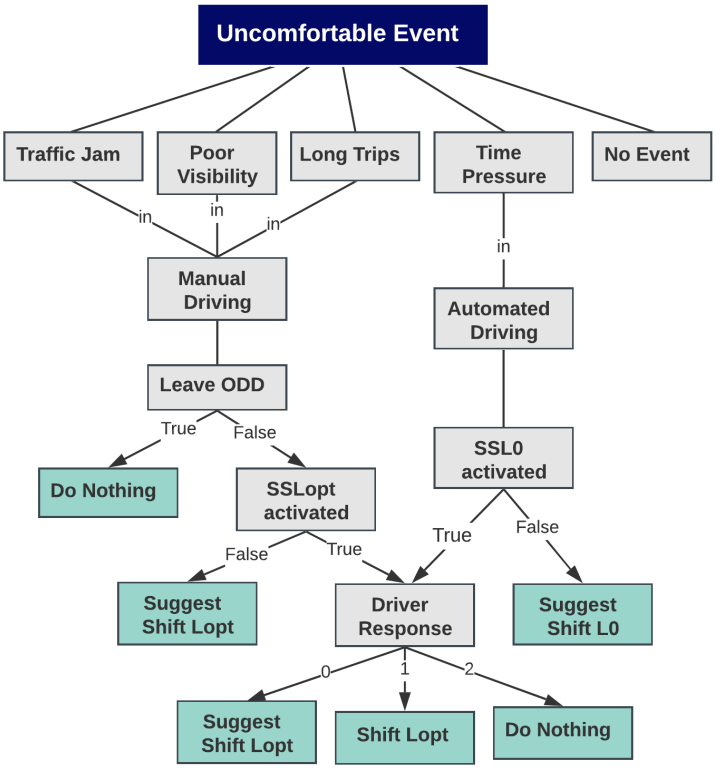


Figure 5: Decision-tree based baseline policy

The decision tree starts with the type of uncomfortable event. If there is no event the result action is 'Do Nothing'. Otherwise, there are two options: uncomfortable events in manual driving or automated driving. If the event occurs in manual driving and the vehicle will leave ODD, then the result action is 'Do Nothing'. If the vehicle will not leave ODD, the action is decided according to whether the shift is activated or not. This part of the decision tree is the part where manual driving and automatic driving are merged. In both cases, if the shift is not activated the result action is 'Suggest Shift'. If the shift is activated, the action will be decided upon drivers' response to this suggestion. There are three different possibilities: if the driver does not respond, the result action is 'Suggest Shift', if the driver accepts the suggestion, the result action is 'Shift' and lastly, if the driver rejects the suggestion, the result action is 'Do Nothing'. These are all possible paths in the designed decision tree.

3.2 Reinforcement Learning Policy

Several reinforcement learning algorithms were taken into account to tackle the decision-making problem utilizing a reinforcement learning-based policy. For this purpose, Baselines which is a set of high-quality implementations of reinforcement learning algorithms are used in this research [9]. All of these algorithms employ function approximation to estimate how good an action is depending

on the existing state, allowing the problem to be solved in a large state space. The chosen algorithms to use on MDP and their short descriptions are as follows:

- Deep Q-Learning (DQN): an RL algorithm that combines Q-Learning with deep neural networks to let RL work for complex, high-dimensional environments [10].
- Advantage Actor Critic (A2C): synchronous, deterministic RL implementation that waits for each actor to finish its segment of experience before performing an update, averaging over all of the actors. It is shown that this algorithm performs better than the other asynchronous baselines implementations [11].
- Trust Region Policy Optimization (TRPO): an effective RL algorithm for optimizing large nonlinear policies as neural networks. It tends to give monotonic improvement, with little tuning of hyperparameters [12].

Hyperparameters

The method of reinforcement learning requires several pre-defined parameters. In this way, the RL algorithm can learn on its own as it interacts with the environment during the learning phase. These parameters are known as hyperparameters [13]. According to the findings of the experiments carried out throughout the research, chosen parameters are:

- **Total timesteps:** It refers to total number of samples to train on. Since higher values provide better performances, this value progressively increased. As a result, the total timesteps value is set to 1000000.
- **Gamma:** It varies between 0 and 1. When gamma is close to zero, the agent is more likely to focus on immediate rewards. If gamma approaches one, the agent will give future rewards more weight and be ready to postpone the reward. High gamma values in this model resulted in decreased rewards. As a result, the gamma value is set to 0.1.
- **Batch size:** It refers to how many training examples are used in single iteration. Batch size can set to one or the total number of examples in the training dataset or a value in between. A value in between resulted best in this model. As a result, the batch size value is set to 32.

3.3 Evaluation Metrics

In this section, evaluation metrics are designed to evaluate the efficiency of reinforcement learning algorithms. Using these metrics, the performance of the learned policies regarding safety and comfort will be evaluated and compared against each other. The evaluation metrics grouped into two categories: reward and domain-specific metrics.

Reward

- Training performance: the mean reward versus timesteps. The plot is used to represent the evaluation metric.

Domain-specific Metrics

Domain-specific metrics are also divided into 3 parts: driving safety, driver comfort, and decision efficiency. The definitions of metrics are as follows:

Driving Safety

- The number of unsafe actions: the total number of shifts to automation when TTAF != 0 or manual driving when TTDF != 0 over the total number of actions. The unit of the metric is the percentage (%).

Driver Comfort

- Time to fix discomfort scenario: the mean duration of being in uncomfortable situations. The total time till the uncomfortable situation is resolved. The unit of the metric is timestep (ts).

Decision Efficiency

- The number of fixed scenarios: the number of scenarios fixed and no longer in uncomfortable situations over the total number of scenarios. The unit of the metric is the percentage (%).
- Time to correct action: the average time it takes to find the correct action. The unit of the metric is timestep (ts).
- Action distribution: the frequencies of actions per uncomfortable driving situation over the total number of actions. The unit of the metric is the percentage (%) and the histogram is used to represent the evaluation metric.

The experiments' goal is to compare and assess the performance of the learned policies specified in section 3.1. For this purpose, experiments carried out using these algorithms. Finally, the experimental results were obtained by using the evaluation metrics in section 3.2. The results will be explained and discussed in the next section.

4 Experimental Results

This section discusses the results of the experiments. The experiments trained one million timesteps to obtain these results. After that, for unsafe actions, the total number of unsafe shifts divided by the total number of actions. For time to fix, the average time till the fixed uncomfortable event is calculated. For fixed scenarios, the number of fixed scenarios divided by the total number of scenarios. Lastly, for time to correct action, the average time till the correct action is calculated.

As presented in section 3.2, evaluation metrics grouped into three categories: driving safety, driver comfort, and decision efficiency. When there are many metrics in a category, the data are combined and the average is calculated. The evaluation results are shown in Table 1.

Table 1: Evaluation results of the baseline and RL algorithms

Algorithms	Driver Safety (%)	Driver Comfort (ts)	Decision Efficiency (%)
Baseline	32.63	2.1117	67.99
DQN	32.66	**1.1112**	**91.65**
A2C	**21.81**	2.0070	81.65
TRPO	32.66	2.1096	69.32

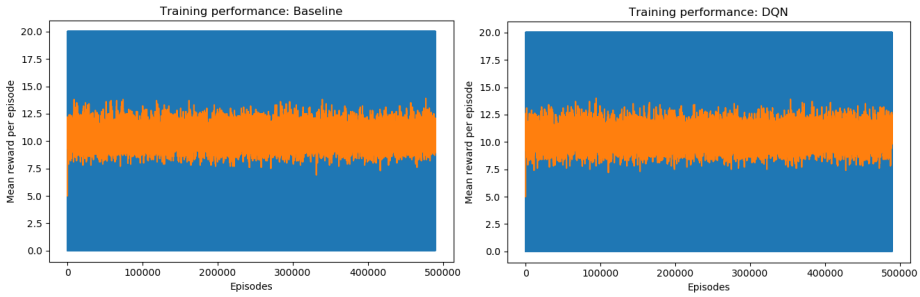
The performances of DQN and TRPO in terms of driving safety are identical. A2C, on the other hand, outperforms DQN and TRPO. The performance of the Baseline algorithm is comparable to DQN and TRPO. So there is a more feasible RL algorithm, A2C, than Baseline in terms of driver safety. The performance of TRPO, A2C, and the baseline algorithms in terms of driver comfort are extremely similar. DQN, on the other hand, has the best performance. So, in terms of driver comfort, DQN is a more feasible RL algorithm than Baseline. The baseline and TRPO algorithms perform the worst in terms of decision efficiency. While DQN has the best performance, followed by A2C. As a result, RL algorithms, A2C and DQN outperformed the baseline algorithm in all three metrics.

After the evaluation of results in terms of driving safety, driver comfort and decision efficiency; the algorithms are compared in terms of training performance and action distribution. The rewards gained throughout episodes are taken into account to compare the training performances of algorithms. The results are shown in Figure 6 and Figure 7. The blue lines in plots indicate the reward for each step, while the orange lines indicate the moving average, which is the average of the last 100 episodes. The reward range, the mean reward, and the median reward for each policy are as follows:

- **Baseline** rewards in the range of [0, 20], the mean reward 5.108, and the median 5
- **DQN** rewards in the range of [0, 20], the mean reward 5.098, and the median 5
- **A2C** rewards in the range of [-2, 20], the mean reward 2.872, and the median 0
- **TRPO** rewards in the range of [0, 20], the mean reward 5.099, and the median 5

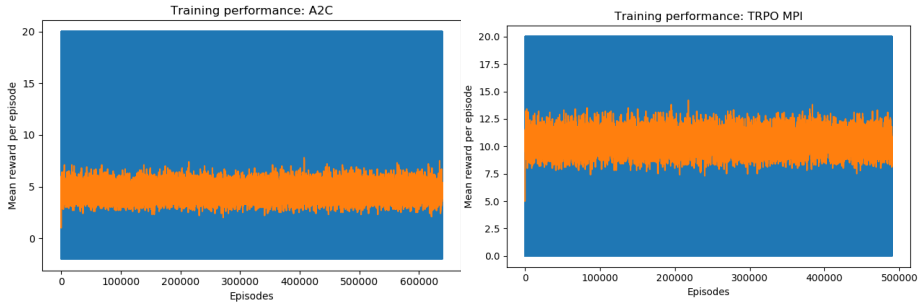
The baseline, DQN, and TRPO algorithms all have similar results for training performance. However, when compared to the others, the A2C algorithm underperforms.

The percentages of performed actions per uncomfortable situation are taken into account to compare the action distribution of algorithms. The results are shown in Figure 8 and Figure 9. The baseline, DQN, and TRPO algorithms all produce similar findings when it comes to action distribution per uncomfortable driving situation. The algorithms are supposed to perform the SL (shift) action because the goal is to fix the uncomfortable driving situations. However, when the A2C algorithm's action distribution is considered, the shift action is rarely performed. The algorithm performs DN action more than SL action. Therefore, when compared to the others, A2C performs poorly.



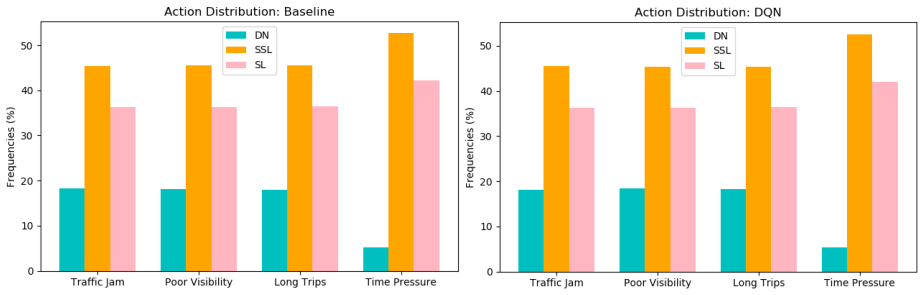
(a) (b)

Figure 6: Left: training performance and mean reward of baseline algorithm. Right: training performance and mean reward of DQN algorithm.



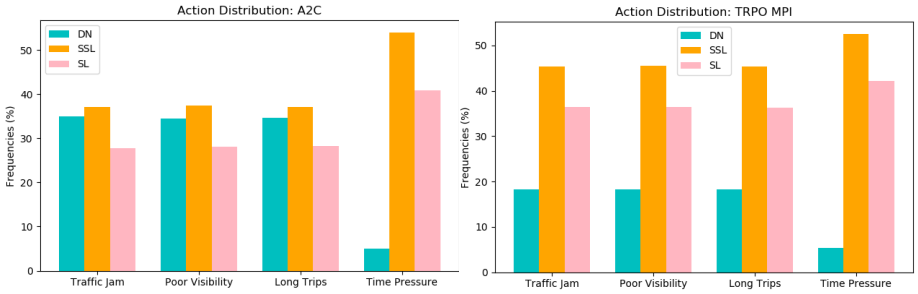
(a) (b)

Figure 7: Left: training performance and mean reward of A2C algorithm. Right: training performance and mean reward of TRPO MPI algorithm.



(a) (b)

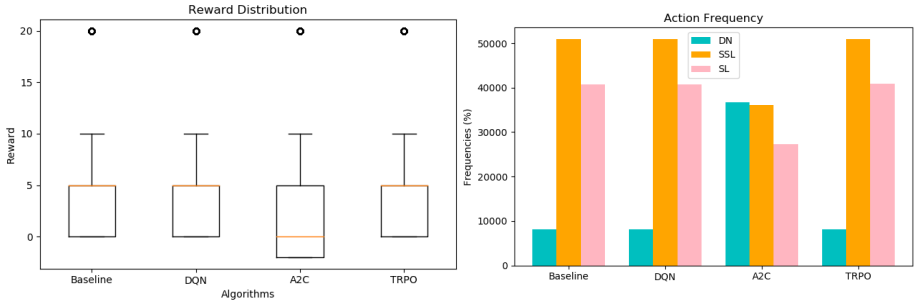
Figure 8: Left: action distribution of A2C algorithm. Right: action distribution of TRPO MPI algorithm.



(a) (b)

Figure 9: Left: action distribution of A2C algorithm. Right: action distribution of TRPO MPI algorithm.

The reward distribution and action frequency of the chosen algorithms are visualized in Figure 10. The reward distribution is shown using a box plot. It visually shows the distribution of numerical data and skewness by displaying the data quartiles and averages [14]. The whisker is shorter on the lower end of the box in baseline, DQN, and TRPO, indicating that the distribution is positively skewed. The whisker is shorter on the upper end of the box in A2C, indicating that the distribution is negatively skewed. Positively skewed data will have a mean that is higher than the median. The mean of negatively skewed data will be less than the median. When looking at the action frequency histogram, it is obvious to see how A2C behaves differently. Instead of 'Suggest' or 'Shift' actions, the algorithm performs 'Do Nothing' action.



(a) (b)

Figure 10: Left: reward distribution of all algorithms. Right: action frequency of all algorithms.

When all of these assessment criteria are considered, the DQN algorithm always outperforms the Baseline method. The TRPO algorithm always performs similarly to the Baseline algorithm. Except for having the highest performance in terms of driver safety, A2C performs worst in other metrics. In conclusion, as compared to the baseline policy, learned reinforcement learning policies are feasible and efficient in handling complicated decision-making problems.

5 Responsible Research

This section consists of two parts. The first part reflects the ethical aspects of the research. Then, the second and last part discusses the reproducibility of the methods used in this research.

5.1 Ethics

In this study, a Mediator simulator is used for the experiments and evaluation, and no real-world data collected. Therefore, the experiments carried out in this study do not appear to raise any ethical or privacy concerns. If the decisions obtained using Reinforcement Learning algorithms are used in the real world, ethical considerations must be addressed. Due to the traffic and other vehicles, making poor decisions while driving might be risky for the driver. At the moment, most of the reinforcement learning algorithms cannot give any guarantees about how they will behave since they can be unpredictable. As a result, there could be an error in the sensors or signals, which could lead to a wrong decision. Before using an RL algorithm, the required safety procedures should be done in this context.

5.2 Reproducibility

This research is a small piece of a larger project and still in a test phase. It is open source as much as possible to make it reproducible. Since the project is still in the development stage, the code used in this research is not publicly available right now. In the future, it will be available upon request. Currently, the OpenAI Baselines algorithms used for RL algorithms are freely available on [9].

6 Discussion

There were several limitations in this study that may have influenced the results. Because of the time constraints, only a few reinforcement algorithms were chosen and used in the experiments. If there was more time, more algorithms could be tested , and some of them might be more feasible to handle the complex scenarios in semi-automated driving. Another limitation is that this study is a small part of the larger project and still in a test phase. As a result, there are a lot of risks, and the system is not yet safe. Even for testing, it requires full awareness of drivers. For this reason, a simulator is used in this research, and no data obtained from real people. This situation may lead to misleading results in some cases. Finally, the decisions chosen in this study are based on assumptions because the question of when the car should take over is still an open ethical discussion. In summary, autonomous driving is a way ahead of the future, and a lot of features are needed to align to make it properly work.

7 Conclusion

This paper attempts to show that learned reinforcement learning policies can be used to solve complex decision-making scenarios in semi-autonomous driving. In this research, the "What action should the Mediator system take when an uncomfortable situation is detected?" question was investigated as a decision-making problem. For this purpose, the decision problem is formulated as a

Markov Decision model. Then three different reinforcement algorithms, DQN, A2C, TRPO, were tested on the designed MDP model and compared to a decision tree-based baseline policy. Experimental evaluation shows that DQN algorithm performs best and can solve the decision problem faster than the baseline policy. It also performs well in terms of driver safety, comfort, and decision efficiency. However, the TPRO algorithm performs worse than the baseline policy in terms of driver safety. Lastly, A2C performs worst in terms of other evaluation metrics except for driver safety. In conclusion, results showed that reinforcement learning algorithms might be unpredictable. The algorithm can perform well in one aspect but not in the other. But this can be seen as an exception because the other RL algorithm performed better than the baselines policy. In particular, the DQN algorithm outperformed. This outcome supports the hypothesis and demonstrates that a learned reinforcement learning policy can be used to solve complex decision-making scenarios in semi-autonomous driving.

References

[1] "Mediating between driver and intelligent automated transport systems on our roads." `https://mediatorproject.eu/`. visited on 2021-05-15.

[2] D. Vermunt, "A markov decision process approach to human-autonomous driving control logic," 2020.

[3] F. van Wyk, A. Khojandi, and N. Masoud, "Optimal switching policy between driving entities in semi-autonomous vehicles," *Transportation Research Part C: Emerging Technologies*, vol. 114, pp. 517–531, 2020.

[4] C. You, J. Lu, D. Filev, and P. Tsiotras, "Advanced planning for autonomous vehicles using reinforcement learning and deep inverse reinforcement learning," *Robotics and Autonomous Systems*, vol. 114, pp. 1–18, 2019.

[5] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. A. Sallab, S. Yogamani, and P. Perez, "Deep reinforcement learning for autonomous driving: A survey," 2021.

[6] H. Bellem, B. Seitz, M. Schrauf, and J. Krems, "Comfort in automated driving: An analysis of preferences for different automated driving styles and their dependence on personality traits," *Transportation Research Part F Traffic Psychology and Behaviour*, vol. 55, 2018.

[7] N. Bagdure and B. Ambudkar, "Reducing delay during vertical handover," pp. 200–204, 2015.

[8] A. Borowsky, T. Oron-Gilad, H. Chassidim, C. Ahlstrom, J. Karlsson, B. Bakker, M. Beggiato, N. Rauh, M. Christoph, D. Cleij, and A. Tinga, "Behavioural markers for degraded human performance," 2020.

[9] P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, Y. Wu, and P. Zhokhov, "Openai baselines." `https://github.com/openai/baselines`, 2017.

[10] J. Schulman and S. Sidor, "Openai baselines: Dqn." `https://openai.com/blog/openai-baselines-dqn/`, 2017.

[11] A. Radford, J. Schulman, Y. Wu, P. Zhokhov, E. Mansimov, and S. Liao, "Openai baselines: Acktr & a2c." `https://openai.com/blog/baselines-acktr-a2c/`, 2017.

[12] J. Schulman, S. Levine, P. Moritz, M. Jordan, and P. Abbeel, "Trust region policy optimization," 2017.

[13] R. Liessner, J. Schmitt, A. Dietermann, and B. Baker, "Hyperparameter optimization for deep reinforcement learning in vehicle energy management," 2019.

[14] S. McLeod, "What does a box plot tell you?." `https://www.simplypsychology.org/boxplots.html`, 2019.

A Appendices