Modelling Autonomous Vehicle Interactions With Reinforcement Learning and Game-Theoretic Decision Making

William Chan
University of Western Ontario, Richmond Hill, Ontario, Canada
Email: williamchan@rogers.com

Mingfeng Yuan and Jinjun Shan
York University, Department of Earth and Space Science and Engineering, Toronto, Ontario, Canada
Email: mfyuan, jjshan@yorku.ca

Abstract—In the near future, autonomous vehicles will operate in traffic together with human-driven vehicles. However, modern autonomous vehicles lack the framework to address automation-automation and human-automation interactions. The goal of this research is to achieve an adaptive control strategy that enables interactions between a variety of drivers, using game theory and reinforcement learning. The environment used is an online simulator that hosts a four way unsignalized intersection. Using the level-k theory, multi-agent interactions were enabled and the Deep Q Network was used to train the policies needed for the framework. To complete the framework, two neural networks were trained to predict 1) the significance of the observation space in predicting the opponent vehicles' levels and 2) the vehicles' levels. The results from this process show us that this framework allows the ego vehicle to predict the levels of opponent vehicles and deploy the correct policy in order to pass through the intersection. Ultimately, this research paves a path towards investigating more complex scenarios using this framework.

Index Terms— Autonomous vehicles, reinforcement learning, game theory, action space, observation space, Deep Q Network, and level-k theory

I Introduction

With the rise of autonomous vehicles on public roads, the importance creating a framework that enhances road safety needs to be addressed. Over the past years, society has experienced 5 autonomous vehicle collisions, all of which were level 2 autonomous vehicles [1]. Level 2 autonomous have semi-autonomous functions that control steering and acceleration, falling short of being fully autonomous which is essential to optimizing road safety. To realize full autonomy, this research looks into reinforcement learning and game theory.

The implementation of reinforcement learning was inspired by the success of DeepMind's research in implementing the Deep-Q-Network (DQN) algorithm to create an agent that can successfully interact in seven Atari gaming environments [2]. The key takeaway of the DQN is its ability to allow an agent to learn from its experiences with a non-biased approach. The neural network does this by learning from random experiences rather than sequential experiences. In this research, the DQN enabled the ego vehicle to learn an optimal policy to traverse through the intersection. However, the limitations of single-agent reinforcement learning became apparent when the complexity of the scenario was increased to include opponent decision-making vehicles.

To counteract the issue of interacting with multiple decision-making vehicles, the concept the level-k theory was implemented. The level-k theory models the decision-making of individuals through a hierarchy of levels. Given a situation, a level k individual will always assume surrounding individuals are level k-1 and take action based on what a level k-1 individual would do. Using this concept, the decision-making process of the ego vehicle was optimized to interact with opponent vehicles based on the belief that they were level k-1.

In this paper, the proposed framework aims to contribute to current research being done in the field by introducing a framework that 1) optimizes the learning and prediction process and 2) can generalize results in a variety of traffic scenarios. The research focuses on this aspect since current research has completed extensive work on implementing deci-

sion trees with game theory. This implementation becomes computationally heavy as the scenario becomes more complex, making it an infeasible approach to use for autonomous vehicles.

This paper is organized as follows. In Section II, the problem is defined and the vehicle model used is discussed. In Section III, the framework used to achieve the adaptive control strategy is introduced. In Section IV, the results from the simulation and future work are discussed.

II Problem Definition

The problem being resolved is the modelling of multi-agent interactions in a four way unsignalized traffic scenario to achieve an adaptive control strategy. The environment used is an open-source repository on GitHub [3]. In this simulation, there are only three vehicles: one ego vehicle on the west side and two opponent vehicles on the north and south sides. To simplify the scenario, the vehicle have been predefined to travel in a straight direction. In this section, the action space, observation space, and reward function will be discussed as they were modified to model this scenario.

A Action Space

Drivers were modeled to have the following actions:

1) "Wait" (maintain a velocity of 0 [m/s])

2) "Move Forward Slowly" (proceed with a velocity of 1 [m/s]

3) "Go" (proceed with the default velocity of 5 [m/s]

These actions were inspired from this research paper that evaluated three different actions spaces in the intersection scenario [4]. The purpose of implementing these specific actions is because they provide a simple approximation of the actions taken by human drivers in an intersection scenario.

B Observation Space

In traffic scenarios, there are three key elements that drivers pay attention to before taking an action: the distance, direction, and motion of surrounding vehicles. The goal of the observation space is to model what a human driver would observe in the given scenario. Therefore, metrics such as the distance aren't precise, but rather they're approximations.

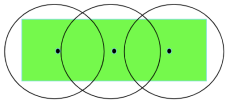


Fig. 1: Contour approximation

In order to create the observation space, three circles were created along the longitudinal axis of each present vehicle in the simulation (See Figure 1). Specifically, a circle with a predefined radius was created at the centre point of the vehicle. From there, a circle was created at the midpoint between the center and the front of the vehicle. Next, a circle was created at the midpoint between the center and the rear of the vehicle. Using these circles, the contours of each vehicle were approximated, allowing for the creation of the desired observation space.

Below is the breakdown of the observation space that was created for this simulation:

Distance: an algorithm was created to calculate all the permutations of distances between the circles on each vehicle using the euclidean distance formula for one-dimension:

$$distance = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{1}$$

Given the calculated set of distances, the shortest distance is chosen to reduce the margin of error. Once chosen, the distance is discretized into the following values (See Figure 2):

1) "Close" if the distance falls within the threshold of 3 [m]
2) "Nominal" if the distance falls between the threshold of 15 [m] and 3 [m]
3) "Far" if the distance falls beyond the threshold of 15 [m]

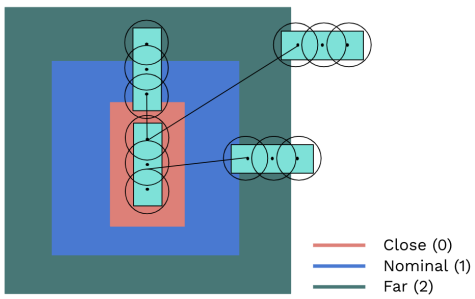


Fig. 2: Distances

These values are approximations of the distance which allows for efficient processing by the neural network and is representative of human observations. Moreover, they were inspired from a paper on implementing game theory in autonomous vehicles [5].

Direction: the surrounding area of each vehicle was broken down into 8 sectors that each have an interior angle of 22.5° (See Figure 3).

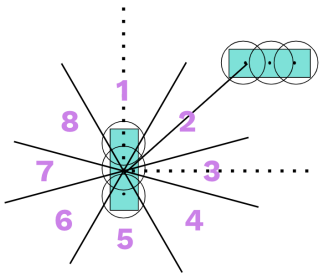


Fig. 3: Direction

Using the distance calculated beforehand, the angle between two vehicles on a predefined axis can be calculated through using this formula:

$$\theta = \pm cos^{-1}(x_{opp} - x_{ego})/distance \tag{2}$$

Given the angle, it can be discretized into one of the 8 categories:

1) "Front" *if* $\theta > -22.5°$
2) "Front Right" *if* $\theta > 22.5°$
3) "Right" *if* $\theta > 67.5°$
4) "Rear Right" *if* $\theta > 112.5°$
5) "Rear" *if* $\theta > 157.5°$ *or if* $\theta < -157.5°$
6) "Rear Left" *if* $\theta > -157.5°$
7) "Left" *if* $\theta > -112.5°$
8) "Front Left" *if* $\theta > -67.5°$

These discrete values allow for vehicles to acquire an approximation of where surrounding vehicles are located.

Motion: the motion of surrounding vehicles can be calculated by examining the historical data of the distances between vehicles. Based on the findings, the motion can be discretized into the following categories:

1) "Stable" $if distance_i == distance_{i-1}$
2) Approaching $if distance_i < distance_{i-1}$
3) Moving Away $if distance_i > distance_{i-1}$

These discrete values allow for the vehicles to approximate surrounding vehicles' motion.

C Reward Function

A reward function represents the goals of a driver. In this research, it was concluded that the main goals of a driver are to 1) Have zero collisions and 2) Complete the intersection in minimal time. Given this, the reward function is defined as:

$$R = \omega_1 c + \omega_2 t + \omega_2 a \tag{3}$$

The terms *c, t, and a* represent, collision, time, and arrival respectively. Collision indicates when the ego vehicle has collided with another vehicle. Time represents the number of steps taken until the end of the simulation. Arrival indicates when the ego vehicle has arrived at its desired destination.

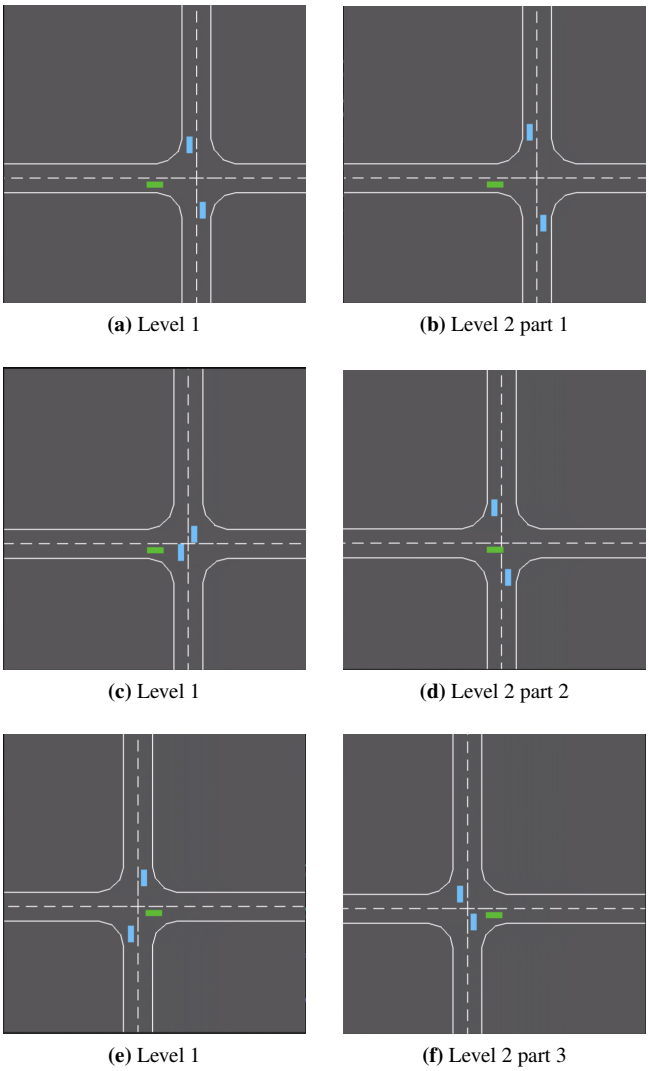
The weighting terms, ω_i remain constant. ω_1 has a weight of -1000 to emphasize that collisions are undesirable. ω_2 has a weight of -0.01 which provides a slight punishment for prolonged simulations. ω_3 has a weight of 1 which is the only positive reinforcement that rewards the vehicle when it arrives at its destination.

III Adaptive Control Strategy Framework

The adaptive control strategy framework allows the ego vehicle to deploy strategies accordingly to the given scenarios. In the following subsections, the development of this framework is explained.

A Deep Reinforcement Learning *[insert figure here]*

The DQN algorithm was implemented in order to train a level 1 and level 2 policy for the ego vehicle. Level 1 indicates a passive driver that executes the action "Move Forward Slowly" before the intersection and then proceeds through the intersection (See Figure 4). To train a level 1 policy, opponent vehicles were set to their default settings which was a level 0 policy. Hence, this is representative of the level-k theory where the ego vehicle is level k and the opponent vehicles are level k-1. Similarly, level 2 indicates an aggressive driver that executes the action "Go" before the intersection. To train a level 2 policy, opponent vehicles were to set the level 1 policy. After this process was completed, the ego vehicle gained two policies that it could deploy in any given scenario.



(a) Level 1 **(b)** Level 2 part 1

(c) Level 1 **(d)** Level 2 part 2

(e) Level 1 **(f)** Level 2 part 3

Fig. 4: Time steps of level 1 and level 2 vehicles proceeding through the four way unsignalized intersection

B Adaptive Control Strategy

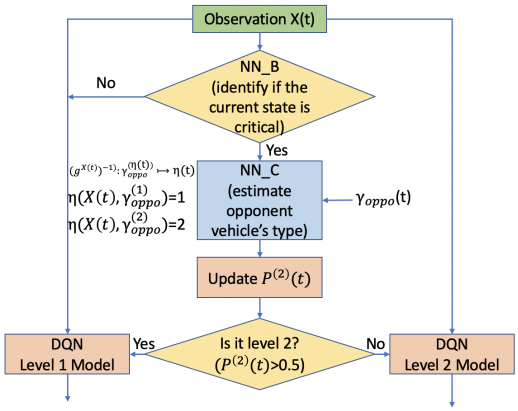


Fig. 5: Structure for Adaptive Control Strategy

To create an adaptive control strategy, a structure consisting of two neural networks was constructed (See Figure 5). This structure was also inspired by a paper on adaptive control strategies [6].

Neural network B is used to identify whether or not the given state, which is the input, is critical to identifying the opponent vehicles' levels. The reason for creating this neural network is because vehicles that have a level 1 or level 2 policy may share the same actions in a given time step. In this scenario, it is not possible to predict the levels. If the state is not critical, the structure will assume that the opponent vehicles are level 2 which causes the ego vehicle to deploy a level 1 strategy. The rationale for this action is because the ego vehicle should be cautious when approaching a vehicle whose actions it cannot predict. This is also reminiscent of human driver decisions. To train this neural network, the states in a simulation for all possible scenarios (level 1 vs level 1, level 1 vs level 2, level 2 vs level 2, and level 2 vs level 1) were collected. With each observation collected, if a level 2 and level 1 vehicle took the same actions in a given time step, the observation was given a label of 0 indicating that the state is not critical.

Neural network C is used to predict the level of the opponent vehicles only if the given state for neural network A was critical. To train this neural network, a similar process was used as neural network A. The states of the vehicles in all possible scenarios were collected and the states were given a label based on the level of the vehicle the state was associated with.

The probability function indicates the probability of the opponent vehicles being level 2 and is defined as follows:

$$P^{(2)}(t+1) = (1-\beta)P^{(2)}(t) + \beta\,\mathbb{1}\{\eta(t)=2\} \qquad (4)$$

t represents the time step, β is a constant set to 0.6, and $\eta(t)$ represents the output of neural network B. $P^{(2)}$(t) is set to 0.5 and is the probability of the opponent vehicles being level 2 at the beginning. This is a predefined value. From this function, it can be seen that as neural network B predicts the opponent vehicle is level 2 consistently, the probability will increase. If the probability is greater than or equal to 0.5, then the ego vehicle will deploy a level 1 strategy. Otherwise, it will deploy a level 2 strategy.

IV Results and Discussion

Currently, results have only been achieved using the environment's default observation space which consisted of the absolute x and y position and the x and y velocities of the vehicles. The ego vehicle was able to interact in an environment given that the opponent vehicles were level 2 (See Figure 4). However, in this scenario, each vehicle didn't have their own state which is the cause of not being able to generalize a successful result across all scenarios. Thus, the next steps for this research include:

1) Giving each vehicle their own states
2) Use the new observation space that was created (distance, direction, and motion)
3) Increase the complexity of the scenario (adding more vehicles and allow turning)
4) Replace DQN with Actor-Critic

Overall, this paper provides an approach to achieving an adaptive control strategy using reinforcement learning and game theory which is a step towards achieving full autonomy.

V APPENDIX A GitHub Code

https://github.com/willie3838/self-driving-car

References

[1] List of self-driving car fatalities. 2020.

[2] David Silver Alex Graves Ioannis Antonoglou Daan Wierstra Martin Riedmiller Volodymyr Mnih, Koray Kavukcuoglu. Playing atari with deep reinforcement learning. 2013.

[3] Edouard Leurent. An environment for autonomous driving decision-making. `https://github.com/eleurent/highway-env`, 2018.

[4] Akansel Cosgun Kaushik Subramanian David Isele, Reza Rahimi and Kikuo Fujimura. Navigating occluded intersections with autonomous vehicles using deep reinforcement learning. 2018.

[5] Mengxuan Zhang Yildiray Yildiz Ilya Kolmanovsky Nan Li, Dave W. Oyler and Anouck R. Girard. Game theoretic modeling of driver and vehicle interactions for verification and validation of autonomous vehicle control systems. *IEEE Transactions on control systems technology*, 26(5), 2018.

[6] Nan Li Ilya Kolmanovsky Anouck Girard Ran Tian, Sisi Li and Yildiray Yildiz. Adaptive game-theoretic decision making for autonomous vehicle control at roundabouts. 2018.