RL for autonomous driving simulator

1. Research Question or Problem

Over the past few years, we have seen an unprecedented growth of unmanned vehicles applications. In this project we seek to compare different Reinforcement Learning (RL) methods and Observation types to find the most suitable settings to allow vehicles to autonomously navigate successfully.

2. Research Goals and Objectives

Our goal is to find out the reinforcement learning algorithm and observation space types that are better for autonomous driving in different scenarios through horizontal comparison.

3. Research Methods and Design

A. Background

We mainly use four algorithms, A2C, DQN, PPO and QR-DQN. Here is a brief description of these algorithms.

a) A2C

Advanced Actor-Critic (A2C), is a simple variant of AC, train two models to do decide and evaluate part respectively. Only changing the critic approach to advanced function, trying o gain a better efficiency of update.

b) DQN

Deep Q-Network (DQN) [1] is an approach that has several advantages over standard online Q-learning. First, with neural network (NN) to compute the Q-table defined in Q-learning, simplify the calculation, which allows for greater efficiency. Second, NN allows the approach implement experience replay, the state in the past can also be the input of NN to gain a new Q-table, improve the utilization rate of data.

c) PPO

Proximal Policy Optimization (PPO) [2] is a optimization of Trust Region Policy

Optimization (TRPO)[3], which was proposed to solve the instability of the update process of some RL algorithms like DQN. Using the gradient function bellow, we can get a reasonable local optimal with stable optimization in each step of update. But it requires the new action policy is not too far from the old one, so that it will not exceed the range of the data. So, we should add a constrain to make sure it will converge correctly. But the constrain is complex in calculation, so PPO propose a simpler form of the constrain which as effective as TRPO, that allows PPO has performance comparable to or better than the state of art algorithms, but the calculations are very simple and very efficient.

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t\left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\right]$$

d) QR-DQN

The application of QR algorithm in distributed reinforcement learning eliminates the gap between theory and practice of distributed reinforcement learning. Through the optimization of quantile distribution, the optimization of Wasserstein measurement is realized in a real sense. QR-DQN [4] combines QR algorithm with DQN algorithm to apply the idea of distributed reinforcement learning in practice.

e) Partial Observable Markov Decision Process

Partially observable Markov decision processes (POMDP) is an ideal model for sequential decision making in dynamic uncertain environments in which state is partially knowable. Its core point is that the agent cannot know the state of the environment in which it is located. It is necessary to obtain its own state by means of additional sensors or interaction with other agents, so as to objectively and accurately describe the real world, which is an important branch of stochastic decision process research.

POMDP is a mathematical model. It is an abstract representation of a problem in reality. According to this model, you can study all kinds of specific problems, such as planning techniques, such as reinforcement learning.

B. POMDP modeling

We use POMDP model to complete our simulation of automatic driving, and the main method is as follows:

a) Action Space Specification.

Typically, the control profile of UAVs has two degrees of freedom, including accelerator and break, and steering, which refer to speed change and rotations around transverse axes, respectively. Correspondingly, there are two variables to control the motion of the car, denoted as $\varphi = [v, \theta]$, $[v]$ denotes the speed change and $[\theta]$ denotes the steering signal. However, this control is complex, and usually not needed, since the car most only do five types of actions, namely lane left, lane right, slower, faster, and idle. Therefore, in this project we simplify the continuous actions to five discrete meta-actions mentioned above.

b) State Space Representation.

There are a lot of method to observe the state space, we choose three types of observations to represent the state. Kinematics observation, is a sorted $V \times F$ array that describes a list of V nearby vehicles by a set of features of size F, with features of $[presence, x, y, v_x, v_y, cos\theta, sin\theta]$, $[x, y]$ are the relative x, y coordinates with the agent, $[v_x, v_y]$ are the relative speed with the agent. Occupancy grid observation, is a $W \times H \times F$ array, that represents a grid of shape $W \times H$ discretizing the space (X, Y) around the ego-vehicle in uniform rectangle cells. Each cell is described by F features, same as the features listed in Kinematics observation. Time to collision observation, is a $V \times L \times H$ array, that represents the predicted time-to-collision of observed vehicles on the same road as the ego-vehicle. These predictions are performed for V different values of the ego-vehicle speed, L lanes on the road around the current lane, and represented as one-hot encodings over H discretized time values (bins), with 1s steps.

c) Reward Design: Incorporating Domain Knowledge.

Reward acts as a signal evaluating how good it is when taking an action at a state. In this project, we design a non-sparse reward that incorporates our domain knowledge about the navigation problem and meanwhile preserving a relatively satisfactory policy. We generally focus on two features: a vehicle should progress quickly on the road and avoid collisions. Thus, the reward function is often composed of a velocity term and a collision term:

$$R(s,a) = a\frac{v - v_{min}}{v_{max} - v_{min}} - b\, r_{collision}$$

where v, v_{min}, v_{max} are the current, minimum and maximum speed of the ego-vehicle respectively, $r_{collision}$ is the collision penalty, and a, b are two coefficients.

4. Simulation Results

In this section we present the experiment settings and simulation results, along with some discussions.

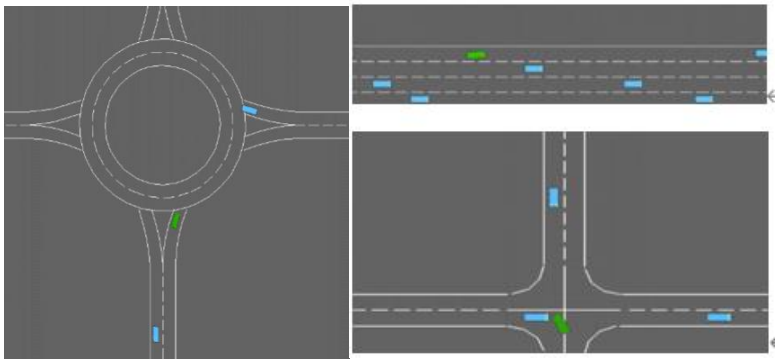


Fig. 1 environment:the left one is the roundabout, the right top is the highway, the right bottom is the intersection

A. Experiment Settings

We construct three types of experiment environments, depicted in Fig. 1, namely highway, intersection and roundabout. And we apply four different models and three different types of observations on each of the environment with a total timestep of 20000.

C. Navigation Behavior of Different Settings on Highway

Environment.

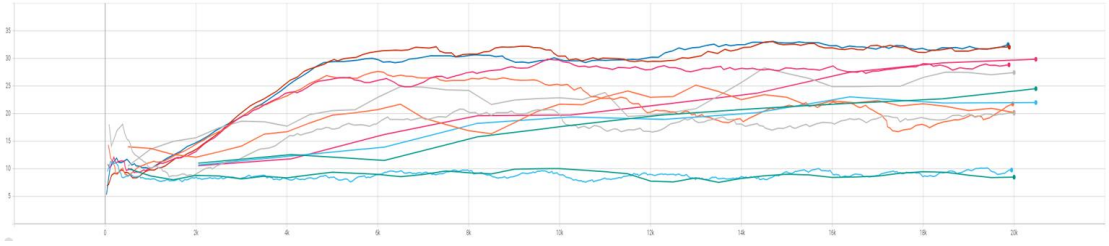
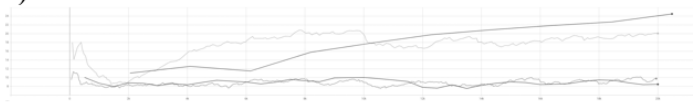


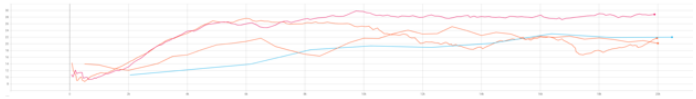
Fig. 2 the normalized return of the highway environment

As shown in Fig. 2, we can see a diversity of returns is generated by different settings. We can group them by three observation types, as dissipated in Fig. 3. The time to collision observation gains the largest reward returns, and the kinematics gains the lowest reward. And there are two models, A2C and DQN with kinematics observation were not able to converge. We believe that the reason that time to collision gains the largest reward is because in highway environment, the other cars are usually drives with a stable speed, and the time to collision observation is stable for this environment. And the reason that two models won't converge is because the kinematics observation is too complicate with too many cars that by simply flatten it to learn by algorithms like A2C and DQN is not able to converge.

a) kinematics



b) occupancy grid



c) time to collision

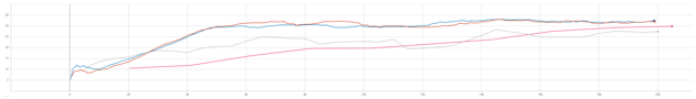


Fig. 3

D. Navigation Behavior of Different Settings on Roundabout Environment.

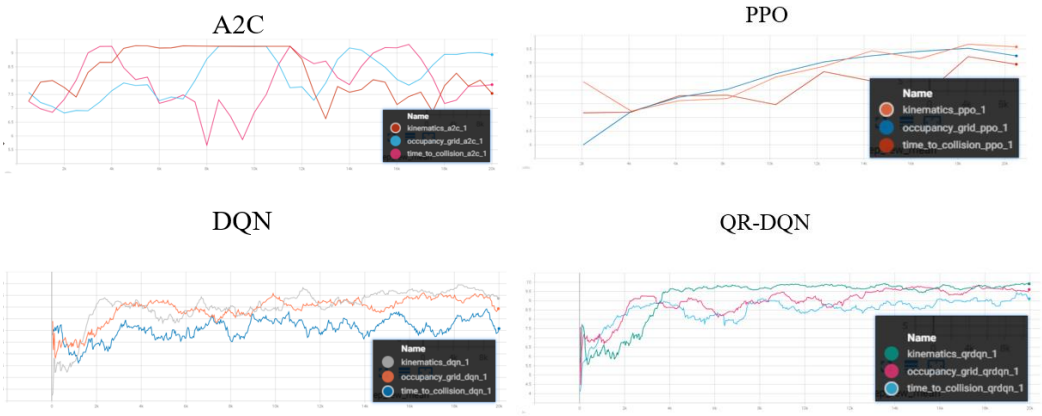


Fig. 4

We also apply four main algorithms to the roundabout environment, all based on three different observation spaces. As can be seen from Fig. 4, the navigation performance of A2C and DQN algorithms are unsatisfactory. After the car enters the roundabout, even if there is no car around to cause interference to it, it will also appear unexpected deceleration, and even stop. We hold the opinion that because their model is relatively simple, which cause a situation that it is difficult to learn good strategies in limited timesteps. On the contrary, the PPO and QR-DQN algorithms met our expectations. Without unnecessary action, they are able to avoid the potential collision correctly, and drove out of the roundabout as fast as possible. In addition, according to the training results, PPO can reach the same level as QR-DQN in terms of navigation performance, but PPO's training speed is 20% faster than QR-DQN. And as mentioned above, because the random action strategy in DQN algorithm is eliminated, its stability is better.

E. Navigation Behavior of Different Settings on Intersection Environment.

In the intersection environment, the performance difference of different observation spaces becomes more pronounced under the same algorithm. Fig. 5

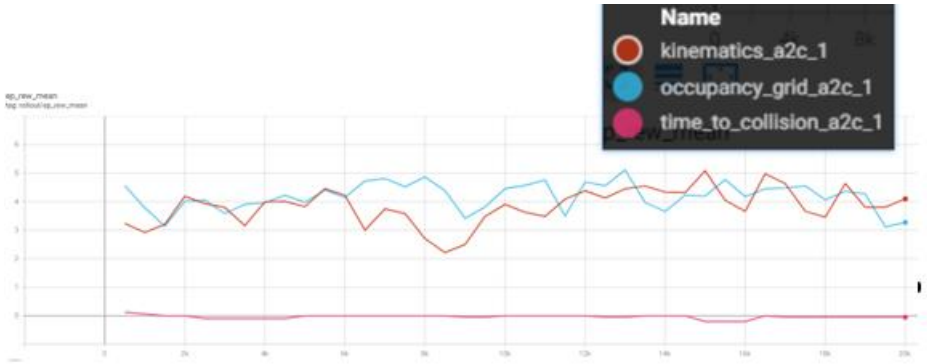


Fig. 5

shows that under the A2C algorithm, using a "collision time", and the car's reward is almost zero. We observe the simulation effect, the car virtually stagnant. After analyzing principle, we consider that because of the characteristics of intersection environment, every car will appear to slow down as it passed the intersection, change lanes, or even stop. It results the state give a very different time to collision table in each frame, which lead to this situation that the A2C algorithm doing nothing in this observation space.

And we found from Fig. 5 the rewards are nearly equal by using these two observation spaces, the occupancy grid and kinematics, but when we see the actual simulation situation, found that there are two completely different strategies. In kinematics observation space, the car usually tends to pass through the intersection at the fastest speed. And the number of other vehicles is not much, which make it goes through the intersections successfully at most of the time. Then its reward is very high, but it is not what we expected, because the traffic safety is very important, we need to find a conservative strategy to avoid collisions every time completely when a vehicle passes through an intersection. Therefore, in the occupancy grid observation, the vehicle's strategy nearly reach what we would expect, the car will normally slow down to avoid the collision, wait, and pass the intersection at a proper speed.

E. A little attempt based on continuous action

As the above tests are based on discrete element actions, and it is known that

continuous actions are more in line with real scenes, we also try to use NAF algorithm to realize the navigation under continuous actions. NAF algorithm is based on the principle of DQN algorithm to realize continuous action strategy. We tested this algorithm in different environments, including highway, parking lot, racetrack and so on. After adjusting the super parameters and reconstructing the reward function for many times, we still could not reach a satisfactory result. The speed of the car could not be improved all the time, and when it was very close to the goal, it usually tended to deduct a few points to reach the destination faster, which was a pity. In the future, we will also make more attempts on continuous action, hoping to achieve a better performance.

5. Staffing

An Guangyan is responsible for integrating, writing code and analyzing the experimental results.

Liu Tong is in charge of searching for available algorithms, understanding their principles and make PPT.

Yu Kaiwei is responsible for adjust parameters, analyzing experimental results and make PPT.

6. Timeline

11.27 Take a further understanding of the algorithms and the code to be used

12.11 Adjust parameters and train them

12.18 Some details should be improved

12.25 Complete the preparation of the project report

7. Conclusion

In general, PPO and QR-DQN algorithm have better results, and PPO has higher speed and stability. In different environments, appropriate observation space can achieve better results. For example, in the environment of highway, time to collision observation is able to reach a better performance, and occupancy grid observation is better to use at intersections. In addition, the structure of reward function is very important in reinforcement learning. In future work, we will

continue to improve the model and try to get better performance.

8. Reference

[1] Adamski, I., Adamski, R., Grel, T., Jędrych, A., Kaczmarek, K., & Michalewski, H. (2018). Distributed Deep Reinforcement Learning: Learn How to Play Atari Games in 21 minutes. High Performance Computing, 10876, 370-388.

[2] Schulman, J. , Wolski, F. , Dhariwal, P. , Radford, A. , & Klimov, O. . (2017). Proximal policy optimization algorithms.

[3] Schulman, J. , Levine, S. , Moritz, P. , Jordan, M. I. , & Abbeel, P. . (2015). Trust Region Policy Optimization. ICML.

[4] Dabney, W. , Rowland, M. , Bellemare, M. G. , & Munos, R. . (2017). Distributional reinforcement learning with quantile regression.