AI3603 Project1: Autonomous Driving Midterm Report

Dexuan He Wenxuan Miao Zhen Li

SJTU SEIEE

Date: November 28, 2023

Abstract

Autonomous driving is one of the most promising techs in the 21st century. This project aims to explore the application of reinforcement learning in enabling autonomous driving in 4 common driving environment: intersection, highway, racetrack ,and parking. The project's main focus is on enhancing safety, efficiency, and adaptability to changing traffic conditions. The intermediate results of our project show significant progress in developing an autonomous driving system that can effectively navigate a simulated highway environment. We also discuss the challenges encountered during the project and the future direction of our research.

Keywords: autonomous driving, reinforcement learning, highway

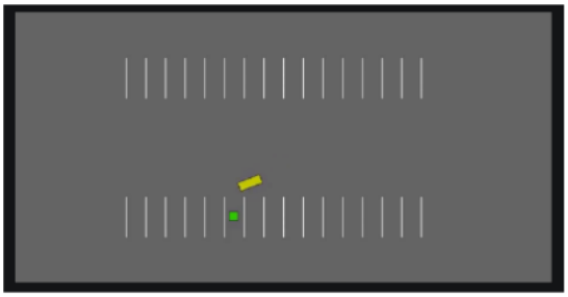
1 Introduction

The rapid advancement of autonomous driving technology presents an opportunity to create safer, more efficient, and intelligent transportation systems. Autonomous driving on highways involves navigating dynamic traffic scenarios, making rapid decisions, and real-time planning, posing challenges for traditional rule-based and fixed control strategies. In this context, reinforcement learning algorithms such as Deep Q-Network (DQN), Proximal Policy Optimization (PPO), Soft Actor-Critic (SAC) and Hindsight Experience Replay (HER) offer a promising approach for learning adaptive driving strategies. We have already read some papers [1] [2] [3] [4].

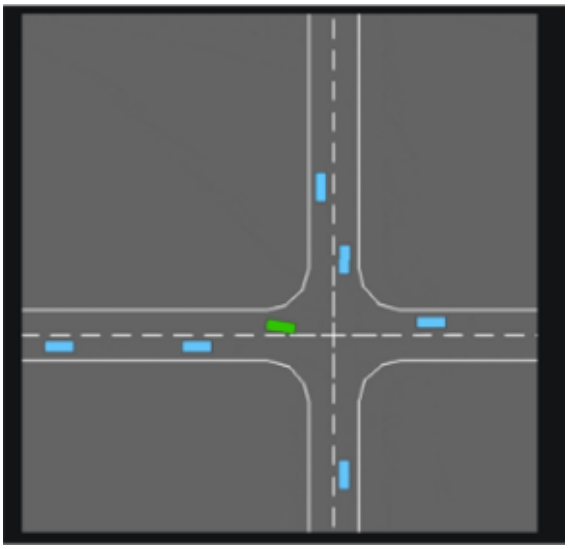
1.1 Environment

In this project, we will deal with four different driving scenarios, including highway, parking, intersection and racetracking as shown below.

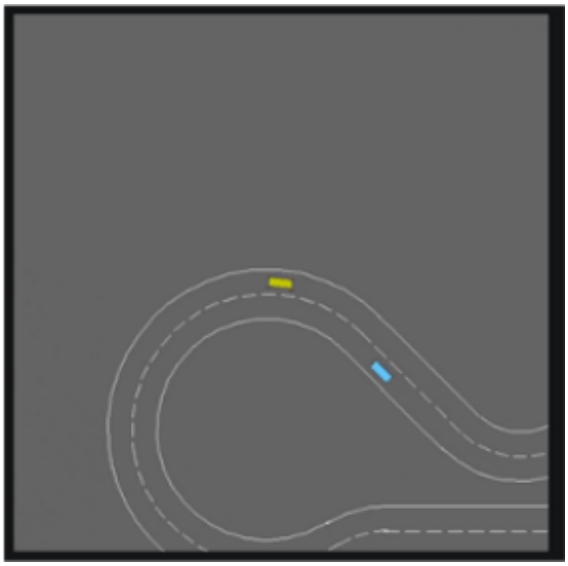
(a) highway



(b) parking



(c) intersection



(d) racetracking

Figure 1: Four driving scenarios

1.2 Requirments and scoring

We aim to implement this as fast as possible while keeping safe. In hightway, intersection and racetracking, we want to move as fast as possible. In parking, we need to park the car in a specific position. All tasks require us to avoid collisions.

Overall Scoring Criteria

A total of 100 points are allocated among four environments, with each environment contributing a specific score. The scores are distributed as follows: ”Highway,” ”Intersection,” and ”Racetrack” each contribute 30 points, while ”Parking” contributes 10 points.

Evaluation Criteria 1: Single-Step Rewards in "Highway," "Intersection," and "Racetrack"

The single-step reward is calculated as follows:

$$R(s,a) = \omega_1 \cdot \frac{v - v_{min}}{v_{max} - v_{min}} + \omega_2 \cdot \text{collision}$$

where v, v_{min}, and v_{max} represent the current speed, minimum speed, and maximum speed of the ego vehicle, respectively. ω_1 and ω_2 are coefficients set to 0.5 each. The state s and action a are used to calculate the reward at each step.

Evaluation Criteria 2: Single-Step Rewards in "Parking"

The single-step reward in the "Parking" environment is computed as follows:

$$R(s,a) = -||s - s_g||^p_{W,p} - \omega_1 \cdot \text{collision}$$

where

$$s = [x, y, v_x, v_y, cos\psi, sin\psi], s_g = [x_g, y_g, 0, 0, cos\psi_g, sin\psi_g]$$

. These parameters represent position, velocity, and orientation respectively. We use weighted P-norm represents the target state with components for position, velocity, and orientation with $p = 0.5$, $W = [100, 100, 5, 5, 1, 1]$, W is the weight of each dimension. $\omega_1 = 0.5$.

2 Implement

There are several algos to make a auto-drive agent with many ways to implement them. Now we only used basic RL models like DQN, and later we gonna try more advanced models. For the lack of time, here we use stable-baselines3 APIs to save time for more trials of models, and we're going to implement our own model in the latter half of this term.

2.1 Models

2.1.1 DQN

Both Q Learning and Sarsa use a list to store and learn $Q(s,a)$, which takes large time and storage cost, especially in continue time and multiple action problems. Deep Q-Network (DQN) use NNs to learn $Q(s,a)$

The unique point of DQN is the two networks: the evaluate network for $Q(s,a)$, and the target network for $r + gamma * max(Q(s',a'))$. In the learning process, parameters in the evaluate network are synchronously update, while those in the target network are not. This feature maintains the stability of the gradient when the network parameters are updated. Besides, DQN uses a replay buffer to randomly learn from the experience in the past, in order to eliminate the timing dependency of samples.

2.1.2 PPO

The stability of strategy is a problem for common policy gradient algo. Natural Policy Gradient Algo introduce the KL distance to limit the difference between strategy distributions. After that, Trust Region Policy Optimization (TRPO) import a check of improvement, a linear search to ensure the KL-restriction, and the conjugate gradient method to accelerate learning. Building on these works, Proximal Policy Optimization (PPO) use a CLIP function to directly restrict the changing range of strategy, i.e.

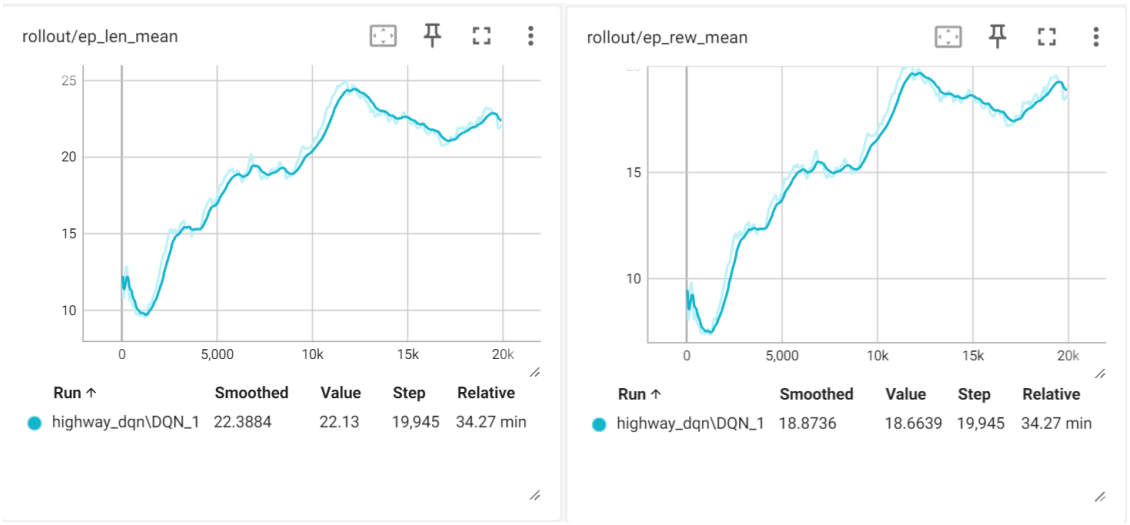
$$\mathcal{L}_{\pi_\theta}^{CLIP}\left(\pi_{\theta_k}\right) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T}\left[\min\left(\rho_t\left(\pi_\theta, \pi_{\theta_k}\right) A_t^{\pi_{\theta_k}}, \operatorname{clip}\left(\rho_t\left(\pi_\theta, \pi_{\theta_k}\right), 1-\epsilon, 1+\epsilon\right) A_t^{\pi_{\theta_k}}\right)\right]\right] \quad (1)$$

where $\rho_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)}$ is the importance sampling. This CLIP function returns the limit when the sampling exceeds this limit, and it maintains the balance between strategy stability and exploration.

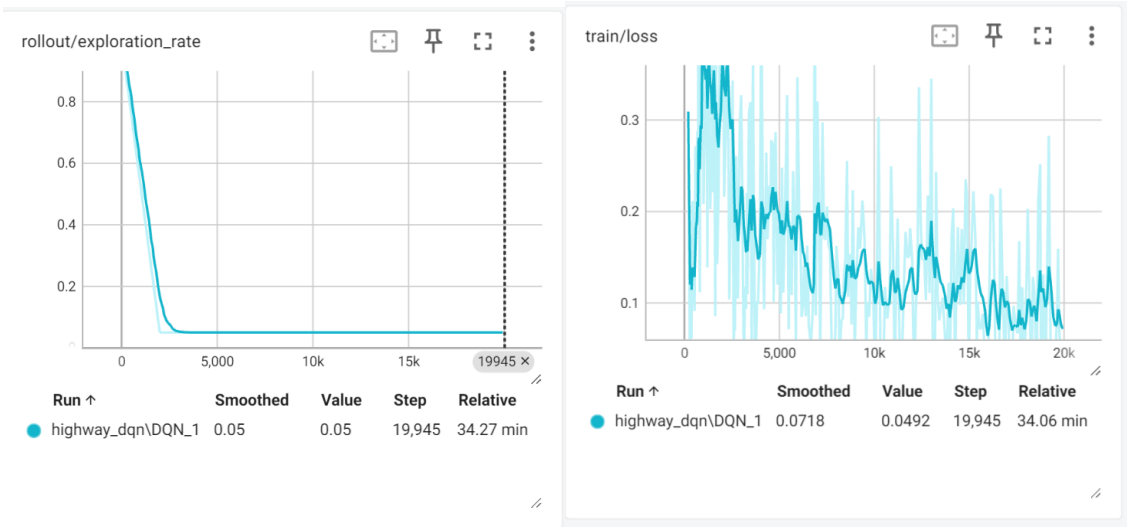
2.2 Current progress

Until now we have trained two effective model using two algos for two environment, one DQN for highway, and the other one PPO for racetrack. Both of them get remarkable result in their task. There are the training results.

2.2.1 highway



(a) episode length **(b)** episode reward

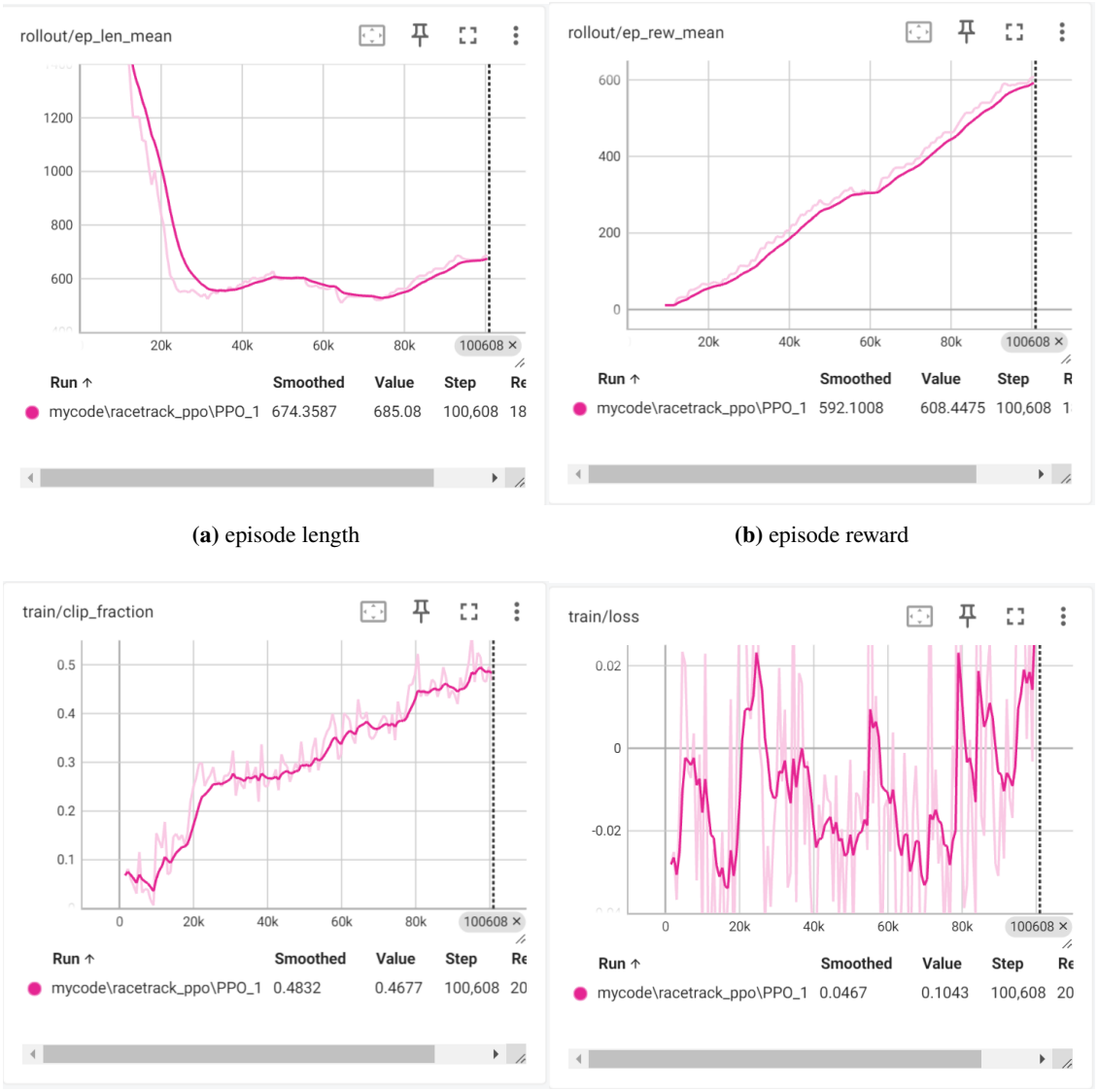


(c) exploration rate **(d)** loss

Figure 2: highway DQN

The highway environment is the most common RL auto-driving task so here we use traditional DQN to leave a chance for model improvement. We see that the simple model works well on this task, ensuring safety and having ability to adapt to changing road condition, while being too cautious to change lanes or accelerate.

2.2.2 racetrack



(a) episode length

(b) episode reward

(c) clip fraction

(d) loss

Figure 3: racetrack PPO

The racetrack environment requires safe driving with a speed as fast as it can. We apply PPO to this task but the result is not that great. The agent seems to be afraid of changing lanes or accelerating, and always drive in the other lane of the opponent. We try to upward adjust the exploration rate but get little effect.

3 Plan

3.1 Model expected to use

For the lack of time, we don't try many models, which is a tactical failure. In the near future, we're going to apply models like C51, DDPG, Social DQN, SAC and HER. We have been looking on the HER model.

3.1.1 HER

As the reference code of task "parking" use a brand new algorithm that hasn't been mentioned in class named HER algorithm , we decide to learning how it works and what improvement it can provide for the task , thus we may use it in our own code in the future.

The core concept of HER algorithm is using experience replay to accelerate the training process.

In details , HER firstly use a buffer to store the episodes the strategy generated , which is similar to what DQN does.However , HER uses a extra step to enlarge the buffer: it will go through the buffer . For each episode , if episode didn't went to the goal successfully , it will assume that one or some of states in this episode is a new goal , and add this new episode into the buffer . Thus , in most sparse reward tasks , HER can transfer a large amount of useless data that doesn't go to the goal into useful date that go to a virtual goal successfully , which can enrich the data set, thus accelerate the training process of deep learning and enhance the credibility and stability of the network.

3.2 Cross environment algo

We expect to find a cross environment model that can drive on highway, intersection, racetrack. Parking is significantly different from the others so not be in consideration. The difficulty seems to be the adaptation within environments and the underfitting of the model when training with experience from different tasks. A possible way of solution is to use the method of transfer learning, and we still have a long way to go.

Bibliography

[1] Marcin Andrychowicz et al. *Hindsight Experience Replay*. 2018. arXiv: 1707.01495 [cs.LG].

[2] Marc G. Bellemare, Will Dabney, and Rémi Munos. *A Distributional Perspective on Reinforcement Learning*. 2017. arXiv: 1707.06887 [cs.LG].

[3] Edouard Leurent. *An Environment for Autonomous Driving Decision-Making*. https://github.com/eleurent/highway-env. 2018.

[4] John Schulman et al. *Proximal Policy Optimization Algorithms*. 2017. arXiv: 1707.06347 [cs.LG].