Driving Decision and Control for Autonomous Lane Change based on Deep Reinforcement Learning

Tianyu Shi1, Pin Wang2, Xuxin Cheng1, Ching-Yao Chan2

Abstract—To fulfill high-level automation, an autonomous vehicle needs to learn to make decision and control its movement under complex scenarios. Due to the uncertainty and complexity of the state space, most classical rule-based methods cannot solve the problem of complicated decision tasks. Deep reinforcement learning has demonstrated great achievements in many fields such as playing games and robotics. However, a direct application of reinforcement learning algorithm still face challenges in handling complex autonomous driving task. In this paper, we proposed a deep hierarchical reinforcement learning based architecture for decision making and control of lane changing situations. We divided the decision and control process into two correlated processes: 1) when to conduct lane change maneuver 2) how to conduct the maneuver. To be specific, we apply Deep Q-network (DQN) with the consideration of safety during the task for deciding whether to conduct the maneuver. Furthermore, we design two similar Deep Q learning frameworks with quadratic approximator for deciding how to select a comfortable gap and just follow the preceding vehicle. Finally, a polynomial lane change trajectory is generated and Pure Pursuit Control is implemented for path tracking. We demonstrate the effectiveness of this framework in simulation, from both the decision-making and control layers. The proposed architecture also has the potential to be extended to other autonomous driving scenarios.

I. INTRODUCTION

As automated driving systems become closer to deployment, the issues of safety and robustness of these systems operating in the real world are drawing greater attention. To fulfill the level 4 or 5 automation, the vehicle needs to learn when to make the right decision and how to execute the action safely. Especially when the vehicle is in the interactive environment, such as a lane changing scenario, the action of the surrounding vehicles may be highly unpredictable. A study shows that nearly 10 percent of all highway crashes are caused by lane change maneuvers [1]. Therefore, a safe, smooth and efficient lane change maneuver is an essential task for autonomous vehicles. To realize this function, the vehicle architecture should allow efficient and robust execution to deal with the uncertainties in the operating environment, making proper decision and performing reasonable action in response to the potentially adversarial or cooperative actions exhibited by the surrounding vehicles.

A considerable body of literature apply pre-defined rule-based model to address the path-planning and trajectory tracking problems under interactive situations. For example, the approaches of a potential field model and model predictive control are suggested in work by Rasekhipour, Y. et al [2], Kim, B. et al [3], and Ji, J., Khajepour. et al [4]. However, as in a real-world scenario, some irrational and unforeseen behaviors (e.g. suddenly overtaking) may render the aforementioned methods inefficient.

Machine learning methods have demonstrated their capacity for solving complexed problems without rigid programming rules [e.g.5, 12, 13]. Zhu, Y. et al [5] demonstrated the good performance of a model-free deep reinforcement learning method that leverage a small amount of demonstration to assist a reinforcement learning agent. However, without a well-trained model and proper policies design, the behavior of the agent may still be unsatisfactory.

In this paper, we utilize the methods of classical control and machine learning in a hierarchical way by leveraging their advantages and disadvantages. In our framework, we first use a DQN to decide whether to perform the lane change immediately, and then use a unique Q-learning approach with a quadratic Q-function to handle the challenge of the continuous control action. The ego vehicle will receive inputs from a higher level, such as a route planning module, and issue a command for the corresponding lane change. Then, the ego vehicle will still need to make judgement whether it is safe to execute the lane change maneuver. If so, it will adjust its position and speed to prepare for the lane change. Otherwise, it will choose to follow the preceding vehicle. Finally, if the ego vehicle is proceeding to make the lane change, a reference lane change trajectory will be generated. Then, the geometry based path tracking method, Pure Pursuit Control will be implemented for path tracking. The integrated system will be optimized at each stage but follow a decision-making policy based on a relatively long-time horizon. In such a way, it can maintain the quick response under unforeseen and dangerous situations while perceiving the task goal further in the future. We made mainly two contributions in our work. First, we decompose the problem into two hierarchical layers, one is to decide on the action and the other is to execute the action. The decision layer can generate a reasonable decision and the execution layer can provide reliable guarantee. Secondly, unlike previous works [e.g. 9, 10, 11, 12], our agent has more autonomy that can learn to take actions, while following the preceding vehicle or adjusting its movement to move into the target gap. Our results show this design provide a reasonable and effective approach for the task of lane change.

II. RELATED WORK

Many studies have been carried out in the domain of autonomous vehicle decision making and control problems under interactive environment. Some researchers consider it as an optimal control problem, applying some predefined

*Research supported by Berkeley Deep Drive, PATH. 1.Tianyu Shi and Xuxin Cheng are with Beijing Institute of Technology, Zhongguancun Street, Haidian District, Beijing, 100081, {tianyu.s,chengxuindavid}@outlook.com
2.Pin Wang (corresponding author) and Ching-Yao Chan are with California PATH, University of California, Berkeley, 1357 South 46th Street, Richmond, CA. 94804, US. {pin_wang, cychan} @berkeley.edu
functions for simplification and optimization. To be specific, Nilsson, J., et al [6], presented a lane change maneuver algorithm which could determine whether a lane change was desirable and if an inter-vehicle gap and time instance to perform was appropriate based on the predefined utility function. Luo, Y., et al [7], proposed a dynamic automated lane change maneuver to accomplish an automated lane change and eliminate potential collisions during a lane change process. Their algorithm could convert the planning problem into a constrained optimization problem using the lane change time and distance. These aforementioned algorithms algorithm can define a smooth path for lane changing theoretically. However, these methods are not robust enough in some dynamic and undefined situations. Also, they cannot incorporate a dynamic prediction of the traffic environment which include uncertainty and sensor noise.

Machine learning approach have been applied to various problems for dealing with unforeseen situations, on the condition that the algorithms are properly trained on a large set of sample data without explicit boundaries or control rules. Vallon, C., et al [8] used Support Vector Machine (SVM), a supervised learning method, to predict the lane-change-initiation behavior for autonomous driving based on personalized human driver data. However, if the labeled data is noisy or the human driver in the experiment shows irregular behaviors, the learned results may be unreliable. The training behavior will also be more personalized rather than common to all drivers. Some researchers explore imitation learning for driver behavior cloning to generate driver-like trajectories. Bhattacharyya, et al [9], extended Generative Adversarial Imitation Learning (GAIL) to address these shortcomings and used a Parameter Sharing approach to guarantee imitation learning in a multi-agent context. As a result, a new algorithm called PS-GAIL was generated. Although they address challenges of instability to some extent in the learning environment through a training curriculum, the emergent values (e.g. collision rate, hard brake rate) climb up with the increase of number of agents.

Some researchers proposed the deep reinforcement learning framework to achieve a robust and reliable autonomous driving policy. Wang, P., et al [10] proposed to apply Deep Reinforcement Learning (DRL) techniques for finding an optimal driving policy by maximizing the long-term reward in an interactive environment. Wu, C et al [11] integrated a traffic simulator with DRL library to develop reliable policy for complex multi-agent problems, such as mixed autonomy traffic (involving both autonomous and human driving vehicles). However, both of the above works uses a rule-based car following model to simplify the longitudinal control. In fact, this is a drawback which cannot help the vehicle to learn a more naturalistic behavior. In one previous work [12], we presented a reinforcement learning based approach for automated lane change maneuver, which let the agent explore the unforeseen environment and make the correct decision based on the Q-values. However, the ego vehicle’s lane change process is without in-depth exploration of decision making. Meanwhile, we didn’t pay much attention to the motion control layer, therefore the control performance is not well tuned to the dynamic environment.

III. PROBLEM DESCRIPTION

In a typical lane change scenario as shown in Fig 1, the ego vehicle must be able to adjust its actions to fit into the dynamic traffic environment. Therefore, the feasible lane change policy π to be learned should be able to:

- Avoid collision with surrounding vehicle;
- Conduct smooth movement;
- Achieve high travel efficiency;

For the decision-making problem, it includes strategic and tactical initiation. The strategic decision means that the lane change will be initiated by the longer-term goals such as travel efficiency (e.g. front vehicle is too slow), the remaining time in the current lane (e.g. a need to exit from a nearby ramp), and other relevant factors on route planning. The tactical decision making means that the ego vehicle already have the strategic intention for lane changing but it needs to follow through. In this paper, we consider the tactical problem.

Firstly, we define the state space, which includes the relative distance between the surrounding vehicle $\Delta x_{\text{leader}}$, $\Delta x_{\text{target}}$, $\Delta x_{\text{follow}}$, the relative velocity in the ego lane $\Delta v_{\text{ego}}$ and in the target lane $\Delta v_{\text{target}}$, the speed $v_r$ and acceleration $a$ of the ego vehicle.

$$s = (\Delta x_{\text{leader}}, \Delta x_{\text{target}}, \Delta x_{\text{follow}}, \Delta v_{\text{ego}}, \Delta v_{\text{target}}, v_r, a) \in S$$

(1)

The autonomous vehicle should consider two layers of decision tasks based on the input features.

To begin with, we should consider whether the lane change movement is beneficial, which we refer to as the ‘decision making layer’. Therefore, for the decision module, the action space is defined as:

$$a_j = \{0, 1\} \in A_j$$

(2)

Where ‘0’ represents to ‘not now’, meaning remain in the ego lane and ‘1’ represents to ‘now’, meaning to change to the target lane now.

Secondly, the next task is to consider how to conduct an appropriate lane change action. In this task, we divide it into
two layers, ‘preparation layer’ and ‘execution layer’. The ‘preparation layer’ function is to find suitable opportunity to conduct the lane change. Meanwhile, the ‘execution layer’ is to provide safe and smooth lane change movement. To be specific, if the lane change decision is 0, i.e. not to change lane immediately, a ‘car following module’ will function and the ego vehicle will remain in the lane and wait for another gap. If the lane change decision is 1, i.e., change lane at this moment, the ‘gap adjustment module’ will function and the ego vehicle should select a gap to make the move. In this process, the ego vehicle will adjust its longitudinal acceleration so that its position and speed will be appropriate to conduct the lane change movement. Finally, in the ‘execution layer’, a planned trajectory will be generated and a controller based on Pure Pursuit Control will track the reference trajectory.

We defined the continuous action space of the ‘preparation layer’ with longitudinal acceleration $a_E$ as:
$$a_E = a_{ego} \in A_E$$

We separate the training into three phases. First we train the lower modules of car following module and gap adjustment module. After they all achieved good performance, we train the decision module.

IV. METHODOLOGY

A. Decision making layer

Deep Q-learning is a model-free reinforcement learning technique by applying deep convolutional neural network to represent Q-value.

In the decision module, the discrete decision is made based on a Deep Q-learning by adopting two fully connected layer neural networks. In the reward function of the decision module, we mainly consider three factors, i.e., relative distance $\Delta x_{ego}$ between ego vehicle and leading vehicle, the relative speed $\Delta v_{ego}$ between ego vehicle and leading vehicle, and the target gap $d_{gap}$ on the target lane. For example, if the relative distance $\Delta x_{ego}$ is big enough, the relative speed $\Delta v_{ego}$ is small, and the gap $d_{gap}$ is narrow, the ego vehicle will tend to remain in the ego lane and keep following the front vehicle. On the other hand, if the relative distance $\Delta x_{ego}$ is small and the gap $d_{gap}$ is big enough, the ego vehicle will tend to change to the target lane.

Therefore, we can construct the reward function for the decision making problem in the following equation:
$$r_{safe} = \begin{cases} w_1 |d_{ego} - x_{leader} - x_{ego}| + w_2 |v_{ego} - v_{leader}|, & a_t = 0 \\ w_3 (d_{target} - d_{gap}) + w_4 |v_{ego} - v_{target}|, & a_t = 1 \end{cases}$$

where $d_{target}$ is the desired distance in the target lane, which can be calculated based on a safety threshold:
$$d_{target} = v_{ego} t + (x_{target} - x_{ego}) + \tau (v_{target} - v_{ego}) + d_0$$

in which $d_{ego}$ is the desired distance in the target lane that in turn can be calculated based on:
$$d_{ego} = v_{ego} \tau + \frac{v_{ego}^2}{2a} + d_0$$

Where $a$ is maximum acceleration; $\tau$ is the reaction time of the human action; $d_0$ is the minimum distance and $t$ is the total time of the lane change.

B. Preparation layer

In the preparation layer, we need to output the continuous action for lane change opportunity preparation. However, the traditional Q-learning method cannot handle problems with continuous action space.

There are policy gradient based algorithms, e.g. actor-critic, to directly learn the policy without resorting to a value function, but it still needs much effort on designing the policy network. In the preparation layer, we use a modified Q-function network structure, for Q-learning to handle the continuous action space, which is similar to our previous work [12]. To be specific, we design a Q-function in a quadratic form so that the greedy action has an optimal solution. Our method can simplify the network design effort in the learning algorithm.

The Q-function approximator is expressed as follows:
$$Q(s, a) = A(s) \cdot (B(s) - a)^2 + C(s)$$

where $A$, $B$, and $C$ are coefficients and designed with neural networks with the state information as input, as illustrated in Fig 3. Because any smooth Q-function can be Taylor expanded in this form near the greedy action, there is not much loss in generality if we stay close to the greedy action in the Q-learning exploration.

As shown in Fig 3, in our model, $A$ is designed with a single-hidden-layer neural network with neurons in the input layer (i.e. the dimension of the state space $S$) and 150 neurons in the hidden layer. Particularly, $A$ is bounded to be negative with the use of a soft-plus activation function, multiplied by a negative sign, on the output layer. $C$ is also a single-hidden-layer neural network with the same number of neurons and
layers as $A$, but its output layer is a fully connected layer so that $C$ network can output any scalar number. $B$ is a neural network with two hidden layers with 200 neurons in each layer and ReLU activation function.

$$r_{\text{dis}} = -w_{\text{dis}} \cdot |x_{\text{leader}} - x_{\text{ego}} - d_{\text{ego}}|$$  \hspace{1cm} (8)

where $R_{\text{dis}}$ is the reward related to relative distance, $w_{\text{dis}}$ is the weight of the reward, $x_{\text{leader}}$ is the longitudinal position of the leader vehicle; $x_{\text{ego}}$ is the longitudinal position of the ego vehicle; $d_{\text{ego}}$ is the desired distance to the leader vehicle.

$$R_{\Delta v} = -w_{\Delta v} \cdot |v_{\text{ego}} - v_{\text{leader}}|$$  \hspace{1cm} (9)

$R_{\Delta v}$ is the reward related to relative speed, $-w_{\Delta v}$ is the weight of the reward, $v_{\text{ego}}$ and $v_{\text{leader}}$ are the speed of ego vehicle and leader vehicle.

$$R_C = R_{\text{dis}} + R_{\Delta v}$$  \hspace{1cm} (10)

$R_C$ is the total reward for the car following module. It is the summation of $R_{\text{dis}}$ and $R_{\Delta v}$.

2) Gap adjustment module

While in a gap adjustment situation, we want to keep the same speed but to adjust to another position unlike that in the previous car following module. The ego vehicle should adjust its longitudinal acceleration or deceleration $a_g$ in order to fit into the target gap while remain safe distance to the leader vehicle. The reward function is defined as:

$$r_{\text{dis}} = -w_{\text{dis}} \cdot \min(\Delta x_{\text{leader}}, \Delta x_{\text{target}}) - \Delta x_{\text{follow}}$$  \hspace{1cm} (11)

which means the ego’s desired position is the middle between the follow vehicle and the closer between the leader vehicle and the target vehicle, as described in Figure. 1.

$$R_{\Delta v} = -w_{\Delta v} \cdot \min(v_{\text{ego}}, v_{\text{leader}})$$  \hspace{1cm} (12)

$R_{\Delta v}$ has the same meaning as in car following module.

$$R_A = R_{\text{dis}} + R_{\Delta v}$$  \hspace{1cm} (13)

The total reward $R_A$ is the sum of $R_{\text{dis}}$ and $R_{\Delta v}$.

C. Execution layer

As soon as acquiring the lane change signal from the decision layer, a reference trajectory will be generated. The initial state is $(x_i, \dot{x}_i, \ddot{x}_i, y_i, \dot{y}_i, \ddot{y}_i)$ while the terminal state is $(x_f, \dot{x}_f, \ddot{x}_f, y_f, \dot{y}_f, \ddot{y}_f)$ The format of the reference trajectory is as follows:

$$x(t) = a_2 t^5 + a_4 t^3 + a_5 t^2 + a_7 t + a_0$$  \hspace{1cm} (14)

$$y(t) = b_2 t^5 + b_4 t^3 + b_5 t^2 + b_7 t + b_0$$  \hspace{1cm} (15)

Then we can define the time parameter matrix:

$$T_{6 \times 6} = \begin{bmatrix}
t_0^5 & t_0^4 & t_0^3 & t_0^2 & t_0 & 1 \\
5 t_0^4 & 4 t_0^3 & 3 t_0^2 & 2 t_0 & 1 & 0 \\
20 t_0^3 & 12 t_0^2 & 6 t_0 & 2 t_0 & 1 & 0 \\
5 t_0^4 & 4 t_0^3 & 3 t_0^2 & 2 t_0 & 1 & 0 \\
20 t_0^3 & 12 t_0^2 & 6 t_0 & 2 t_0 & 1 & 0
\end{bmatrix}$$  \hspace{1cm} (16)

For the lane change motion control, we adopted the Pure Pursuit control method to follow the reference trajectory. By
generating the target waypoints with velocity and position. We can control the ego vehicle to keep track of the target trajectory (as shown in the following Fig. 4).

![Image](image.png)

Figure 4. Path tracking based on Pure Pursuit Model

By calculating the angle $\alpha$ between the vehicle body and the target waypoint, we can control steering to track the target point.

According to the geometry relationship, we can get:

$$\frac{l_d}{\sin \alpha} = 2R$$

$$\delta(t) = \tan^{-1}\left(\frac{2L \sin(\alpha(t))}{l_d}\right)$$

Where, $l_d$ is the look ahead distance; larger $l_d$ will make the tracking smoother and smaller $l_d$ will make the tracking more accurate; $\delta(t)$ is the controlled front steering wheel angle, which keeps track of the orientation; $L$ is the wheel base of the vehicle.

IV. IMPLEMENTATION ON LANE CHANGE SCENARIO

A. Simulation Scenario Building

We build a simulation environment with the lane change vehicles and surrounding vehicles. The surrounding vehicles are controlled based on the Intelligent Driver Model (IDM) [12]. When a vehicle finishes the lane change movement, it will use IDM for car following.

B. Implementation on Lane change scenario

First we build our simulation environment as depicted in Figure 5. The red cars are ego vehicles, the green cars are vehicles associated with the selected gap, and the blue cars are normal vehicles controlled by IDM.

![Image](image.png)

Figure 5. Simulation environment

In this simulation platform, the segment length is 1000m and each lane width is 3.75m. The initial speed of vehicles varies from 75km/h to 136km/h. The initial distance between leader vehicle and ego vehicle is set to $\int_0^t v_{leader} \, dt$, $t \sim U(0,1)$, $7$.

These settings allow the environment to generate distinctive situations so that the behavior learned by agent is robust. In addition, as shown in the Fig 3, A, B is initialized as 0, and C is initialized as -60.

During the training process, we choose the learning rate as 0.0005, replay pool size 5000, batch size 64, and a decayed $\epsilon$ from 1 to 0.1 in 300,000 steps. At first the agent tends to take random actions to explore the environment and acquire information of the environment. As the training process moves on, the agent is more likely to exploit learned information to make the best decision.

The loss curve and accumulated reward per episode are depicted as Fig 6, from which we see that the loss converges and the agent successfully learns to take actions to maximize the reward.

![Image](image.png)

Figure 6. Evaluation of training loss and reward function

To better evaluate the performance of the model, we chose a typical initial condition where the initial speed of leader vehicle is 35.5 m/s while the initial speed of the ego vehicle is 17 m/s and the relative distance between them is 68 m.

Fig 7 demonstrates that the ego vehicle successfully learns to adjust its speed and relative distance to a proper value after about 100 time steps (10s). Even though there is a bid discrepancy in speeds at the start, we can see that the ego vehicle achieves good performance.

We train the adjustment module in a similar manner and then the decision module with a traditional discreet action space DQN approach. From the animation results, we can conclude that the proposed structure is able to solve the lane change problem with respect to the reward function designed. In the future the effect of the structure on the agent performance should be evaluated.

From the results of the experiments, the decision layer and the preparation layer both achieve good performance. The ego vehicle is able to recognize the safety of a gap when lane change command is committed.

V. CONCLUSION AND DISCUSSION

For the problem of lane change, a single deep reinforcement learning method is not able to complete the whole task due to the task’s complexity. Our proposed structure of DQN based hierarchical reinforcement learning decomposes the entire procedure into the decision layer and the execution layer. Each
module in the execution layer can be trained to gain a relative good performance. Then the upper layer, or the decision layer can be trained subsequently and finally learn to make the right decision.

To implement the proposed methods, we developed a simulation environment, and allow it to adapt to the training of execution layer and decision layer. The loss curve and accumulated reward imply that the algorithm has come to a promising convergence.

The next step of our research is to focus more on other types of decisions (e.g. overtaking) rather than reactive decision to accommodate the adjacent traffic. Different reinforcement learning methods can also be applied to continuous action space (e.g. Deep Deterministic Policy Gradient) to improve performance. To better exploit the capability of reinforcement learning in discrete space, we can also use a more complicated hierarchical framework (e.g. Option Graph) to handle only decision making, and give control and execution to traditional control methods (e.g. MPC). The suggested approach potentially can be adapted to various situations other than lane change.

REFERENCES

[1] S. Hetrick. Examination of driver lane change behavior and the potential effectiveness of warning onset rules for lane change or “side” crash avoidance systems. Dissertation, Virginia Polytechnic Institute & State University, 1997.
[2] Rasekhpour, Y., Khajepour, A., Chen, S. K., & Litkouhi, B. (2016). A potential field-based model predictive path-planning controller for autonomous road vehicles. IEEE Transactions on Intelligent Transportation Systems, PP(99), 1-13.
[3] Kim, B., Necsulescu, D., & Sasiadek, J. (2002). Model predictive control of an autonomous vehicle. 2001 IEEE/ASME International Conference on Advanced Intelligent Mechatronics. Proceedings (Cat. No.01TH8556). IEEE.
[4] Ji, J., Khajepour, A., Melek, W. W., & Huang, Y. (2017). Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints. IEEE Transactions on Vehicular Technology, 66(2), 952-964.
[5] Zhu, Y., Wang, Z., Merel, J., Rusu, A., Erez, T., Cabi, S., ... & Heess, N. (2018). Reinforcement and imitation learning for diverse visuomotor skills. arXiv preprint arXiv:1802.09564.
[6] Nilsson, J., Silvin, J., Brannstrom, M., Coelingh, E., & Fredriksson, J. (2016). If, when, and how to perform lane change maneuvers on highways. IEEE Intelligent Transportation Systems Magazine, 8(4), 68-78.
[7] Luo, Y., Xiang, Y., Cao, K., & Li, K. (2016). A dynamic automated lane change maneuver based on vehicle-to-vehicle communication. Transportation Research Part C, 62, 87-102.
[8] Vallon, C., Ercan, Z., Carvalho, A., & Borrelli, F. (2017). A machine learning approach for personalized autonomous lane change initiation and control. Intelligent Vehicles Symposium (pp.1590-1595). IEEE.
[9] Bhattacharyya, R. P., Phillips, D. J., Wulfe, B., Morton, J., Kuefler, A., & Kochenderfer, M. J. (2018). Multi-Agent Imitation Learning for Driving Simulation. arXiv preprint arXiv:1803.0104.
[10] Wang, P., & Chan, C. Y. (2017, October). Formulation of deep reinforcement learning architecture toward autonomous driving for on-ramp merge. In Intelligent Transportation Systems (ITSC), 2017 IEEE 20th International Conference on (pp. 1-6). IEEE.
[11] Wu, C., Keddieh, A., Parvate, K., Vinitsky, E., & Bayen, A. M. (2017). Flow: Architecture and benchmarking for reinforcement learning in traffic control. arXiv preprint arXiv:1710.05465.
[12] Wang, P., Chan, C. Y., & Fortelle, A. D. L. (2018). A reinforcement learning based approach for automated lane change maneuvers.
[13] Xi, C., Shi, T., & Chen, J. (2019). A Data Driven Approach for Motion Planning of Autonomous Driving Under Complex Scenario. arXiv preprint arXiv:1904.08784.