The following publication Y. Zhang, Y. Hou and I. W. -H. Ho, "Federated Learning Based Decision Making for Autonomous Driving in Extreme Scenarios," 2025 IEEE 102nd Vehicular Technology Conference (VTC2025-Fall), Chengdu, China, 2025, pp. 1-7 is available at https://doi.org/10.1109/VTC2025-Fall65116.2025.11310761.

# Federated Learning Based Decision Making for Autonomous Driving in Extreme Scenarios

Yuting Zhang[1,2], Yun Hou[1], Ivan Wang-Hei Ho[2]
[1]Department of Computer Science, The Hang Seng University of Hong Kong, Hong Kong
[2]Department of Electrical and Electronic Engineering, The Hong Kong Polytechnic University, Hong Kong
yu-ting.zhang@connect.polyu.hk, aileenhou@hsu.edu.hk, ivanwh.ho@polyu.edu.hk

***Abstract*—Autonomous driving systems must make precise and reliable decisions to ensure safety and prevent collisions. In this paper, we address the challenge of effectively training autonomous vehicles to handle rarely encountered extreme scenarios in a virtual environment. To achieve this, we propose a hybrid approach that integrates reinforcement learning (RL) for autonomous driving within a federated learning (FL) framework. This approach enables individual vehicles to collaboratively develop a global model capable of handling diverse extreme tasks at intersections. Additionally, it allows local vehicles to train models in conjunction with roadside units (RSUs) without compromising sensitive data, such as perception information. Simulation results demonstrate that the proposed FL framework not only boosts the convergence speed during the training phase by up to $114.26\%$, but also improves autonomous driving performance, as evidenced by higher reward values, lower collision rates, and reduced travel time, compared to benchmark RL schemes.**

***Index Terms*—Autonomous Driving, Reinforcement Learning, Federated Learning.**

## I. Introduction

Safety has always been the primary concern for autonomous driving technology, especially regarding higher levels of autonomy, where driving decisions are handled entirely by the system rather than by a human driver. To meet safety requirements, autonomous driving systems must make highly accurate decisions to avoid collisions. This is typically achieved through reinforcement learning (RL). However, RL generally requires many interactions between the agent and the training environment to learn the optimal policies. Training such models in real-world conditions can be excessively costly and complicated. Therefore, we aim to generate extreme cases in a virtual environment to facilitate the training of autonomous vehicle agents. To our knowledge, most issues tackled by RL in autonomous driving focus on highway scenarios, dealing with tasks such as lane changing, ramp merging, and platooning. Intersection scenarios, particularly those involving broken traffic lights, have received relatively little attention. In addition, most research concentrated on managing traffic flow rather than decision-making for individual vehicles. This paper focuses on intersection scenarios with broken traffic lights and explores the extreme cases that RL models may encounter.

This work was supported in part by the Faculty Development Scheme (FDS) (Project No. UGC/FDS14/E02/23) established under the University Grant Committee (UGC) of the Hong Kong Special Administrative Region (HKSAR), China. The work of I. W.-H. Ho was supported in part by the Research Institute for Artificial Internet of Things (RIAIoT) (Project P0050293), The Hong Kong Polytechnic University.

Since vehicles are highly mobile, quick decision-making is needed in autonomous driving scenarios. If a vehicle relies solely on local computation capacity to train its model, it may miss the optimal instant for decision-making, leading to accidents. Therefore, we require a novel model training architecture to enhance the efficiency of local model training and improve decision-making accuracy. Vehicle edge computing offers a promising solution by enabling vehicles to quickly perform local computing tasks and collaborate with roadside units (RSUs). Meanwhile, we aim to protect the privacy of local vehicles, allowing them to train models with RSUs without uploading their sensitive data. Federated learning (FL) addresses these needs effectively. This paper proposes combining the FL framework with RL tasks for autonomous vehicles. The approach involves vehicles pre-training their models using local data and then uploading the trained model for aggregation at RSUs. This cycle continues until the model converges. This method not only safeguards the privacy of local vehicles but also uses a collaborative model to speed up the convergence and improve the overall performance.

The main contributions of this paper are threefold.

- We utilize the SUMO simulator to design an intersection scenario featuring various extreme cases for the agent. This approach tackles the long-tail problem of scarce extreme case data and high data collection costs.
- We integrate the RL process for autonomous driving into a FL framework, allowing individual vehicles to work together to develop a generic model for various extreme tasks in intersection scenarios. Our framework outperforms two benchmark RL schemes, demonstrating faster model training convergence and improved overall effectiveness in terms of average reward, collision rate and average travel time.
- We investigate the impact of various aggregation intervals between local vehicles and the RSU on the average reward achieved by the FL model for various tasks. Our results reveal that the relationship between the aggregation interval and the reward obtained exhibits a biphasic response, where the aggregation interval should be neither too short nor too long to achieve the best reward.

The remainder of the paper is structured as follows. Section II reviews the literature on simulators and FL for autonomous driving. Section III presents the methods for extreme case

generation and the federated reinforcement learning (FRL) framework for autonomous driving intersection navigation tasks. Section IV presents the experimental results. Finally, section V provides a summary of the paper and an overview of future work.

## II. Related Works

### A. Simulators for Autonomous Driving

Simulators are often used for research in autonomous driving due to the advantages of saving time, cost, and labor. Compared to real-world scenarios, simulators can rapidly synthesize large-scale sensor data and replicate extreme weather conditions, unpredictable behavior of aggressive drivers, and intricate traffic scenarios. In addition, simulators have the unique advantage of replicating traffic accidents without posing safety risks [1].

Much research has explored the use of high-fidelity driving simulators to provide cost-effective and manageable alternatives to real-world data. These simulators utilize the power of game engines to generate realistic images with built-in physics. Examples include CARLA [2], DeepDrive [3], and NVIDIA Drive Sim [4]. However, the above simulators often need to spend much time on rendering 3D models into real-world background images, and the RL agents need to interact with the environment a lot to learn the optimal policy. This will inevitably prolong the model training time and degrade the overall efficiency.

In addition, some simulators focus on processing sensor data and support multimodal perception tasks, such as Sim4CV [5] and AirSim [6]. A clear advantage of this class of simulators is that it can speed up data collection under challenging conditions, especially in bad weather conditions, thus saving valuable time. For efficient generation of extreme cases, we focus on modifying scenarios during the decision stage, once perception is complete and annotations of surrounding vehicles are available. By altering the parameters or behavioral models of detected vehicles within the experimental environment, we can systematically create extreme cases. Moreover, we can directly utilize the simulator's data capture function to input the acquired data as state vectors for RL.

Therefore, considering that our simulator needs to implement traffic control and vehicle-to-everything (V2X) communication, as well as be compatible with Gym [7], a RL platform created by OpenAI, we adopt SUMO [8] as the traffic simulator in this work. By setting specific parameters in SUMO's road network and vehicle route files as variables, we can flexibly modify the traffic scenarios to create different extreme cases.

### B. Federated Learning

An important aspect of the learning phase for autonomous driving is its significant dependence on location. Therefore, aggregating training results from vehicles operating in different scenarios is highly beneficial. The cooperative vehicle infrastructure system (CVIS) is increasingly recognized as a valuable tool for achieving advanced autonomous driving with improved accuracy. Wireless edge computing systems collect locally trained model parameters from surrounding vehicles through edge nodes and then aggregate them using FL.

FL is still in the early research phase concerning autonomous driving. Zhang et al. [9] proposed a real-time, end-to-end FL approach with a unique asynchronous model aggregation mechanism. Another study [10] applied FL to predict turn signals. Recent research [11] [12] has explored the use of FL in 6G-enabled self-driving cars. In another study [13], researchers utilized FL to address the problem of distributed dynamic map fusion for intelligent connected vehicles. Work [14] proposed a distributed machine learning model-sharing architecture for self-driving cars that corrects localization errors by sharing a vehicle localization error evolution model. Liu et al. [15] proposed a lifelong federated reinforcement learning (LFRL) architecture to enable robots to integrate and migrate experiences, effectively utilizing prior knowledge and quickly adapting to new environmental challenges. Furthermore, reference [16] employed a joint RL approach to expedite cross-vehicle training for more efficient cooperative vehicle perception.

Although applying FL to autonomous driving can speed up the training process and improve the overall performance, problems such as communication burden and local model quality exist. Therefore, in this paper, in addition to applying the FL framework to the RL process for autonomous driving, we examine the aggregation interval between local vehicles and edge nodes within a specific timeframe to assess the convergence speed of the model training process and the model's test performance in various extreme cases. This will help us determine the optimal aggregation interval and strive for a balance between communication cost and the quality of the global model.

## III. Methodology

In this section, we divide the entire FRL framework into three parts and describe the methods and specific configurations in each part.

### A. Extreme Case Generation

Drawing upon the pre-crash scenario descriptions defined by the National Highway Traffic Safety Administration (NHTSA) [17], we develop a four-way intersection scenario in SUMO to investigate the behavior of autonomous vehicles in extreme cases related to malfunctioning traffic signals. Unlike highway scenarios, which have been extensively studied, existing intersection scenarios typically concentrate on traffic flow management with intelligent traffic signals. As a result, there is a lack of understanding regarding how autonomous vehicles can navigate signal failures through local or collaborative strategies.

We design three extreme cases to address this gap by parameterizing variables within the SUMO simulation. The scenario was implemented using SUMO's network files (*.net.xml*) and vehicle route files (*.rou.xml*). The details of the configuration

files are described below, and the corresponding parameter settings are shown in Table I.

- For the network file, we establish a bidirectional single-lane four-way intersection without traffic lights, allowing vehicles to turn left, right, or go straight at the intersection. In addition, we set the parameter *collision.check-junctions* to *True* to ensure that the system can detect collisions at intersections.
- For the vehicle route file, we define two vehicle types, i.e., RL agent and human-driven vehicles, which differ in their *speedmode* and their routes. For *speedmode*, it is set to 0 for RL agents, which means that while RL agent vehicles observe right-of-way rules when approaching intersections and adhere to the speed limit, they ignore specific regulations such as safe speed, maximum acceleration, deceleration, and hard braking to prevent running a red light. At the same time, to mimic the effect of signal failure, human-driven vehicles are assigned a *speedmode* of 39, promoting aggressive acceleration through occupied intersections. For their routes, RL agent vehicles have three different routes, left turn, go straight and right turn at the intersection, corresponding to the three different tasks of the agent. On the other hand, human-driven vehicles emerge from both sides of the intersection and go straight, at an arrival rate of one vehicle per 10 time steps.

TABLE I: Parameter settings for SUMO.

| Vehicle Type | Max Speed | Color | Depart Speed | Speed Mode |
|---|---|---|---|---|
| Human | 20.1168 | Yellow | Random | 39 |
| Agent | 20.1168 | Red | 0 | 0 |

### B. Reinforcement Learning

After creating a training environment for RL agents, we present the chosen RL algorithm for the driving tasks and the definition of its state space, action space, and reward function.

*1) Reinforcement Learning Algorithm:* This paper employs the Deep Deterministic Policy Gradient (DDPG) algorithm [18] for autonomous driving tasks due to its proficiency in managing continuous control, avoiding local optima, processing high-dimensional sensor inputs, and optimizing deterministic policies. Furthermore, incorporating experience replay and target networks [19] enhances training stability, while the off-policy design allows efficient data reuse. We use the Stable-Baselines3 [20] RL library to implement this framework as the local vehicle training module. Key hyperparameters, such as learning rates, batch sizes, and coefficients for exploration noise, are optimized for performance and convergence, as outlined in Table II.

*2) State Space:* Given the limited sensing range of the onboard sensors in autonomous vehicles operating in the real world, we define a grid coordinate system centered on the RL agent vehicle extending within a specified range, where the influence of roadside buildings on the sensing field is not considered. The center of the front bumper of each vehicle

TABLE II: Hyperparameter settings for DDPG.

| Hyperparameter | Value |
|---|---|
| Action noise | Ornstein-Uhlenbeck action noise |
| Policy | MLP policy |
| Network architecture | [512, 512, 256, 64] |
| Buffer size | 100,000 |
| Learning starts | 1,000 |
| Batch size | 256 |
| Tau | 0.001 |
| Gamma | 0.99 |
| Learning rate | 0.0001 |
| Training time steps | 10,000 |
| Testing episodes | 100 |

serves as its designated location. A schematic representation of this coordinate system is illustrated in Fig. 1. The figure shows that the observation range for the agent vehicle extends 40 m in front, 5 m behind, and 80 m to the left and right. Considering that the length and width of vehicles defined in SUMO are 5 m and 1.8 m, respectively, we set the dimension of one grid as 6.15 m in length and 2.5 m in width to ensure a safe distance between vehicles. As a result, at the vertical and horizontal span, there are in total 18 and 26 grids. We use 0 or 1 to represent if there is a vehicle observed in a grid, and at each grid, if there is an observed vehicle, we also record the vehicle's states, i.e., its velocity $v$ and steering angle $\theta$. Finally, the state space is a 3-D $26 \times 18 \times 3$ matrix. The state space $S_t$ and the state vector for each observed vehicle $s_t^{(i,j)}$ is represented as follows.

$$S_t = \begin{bmatrix} s_t^{(-80,40)} & s_t^{(-79,40)} & \dots & s_t^{(80,40)} \\ s_t^{(-80,39)} & s_t^{(-79,39)} & \dots & s_t^{(80,39)} \\ \vdots & \vdots & s_t^{(i,j)} & \vdots \\ s_t^{(-80,-2)} & s_t^{(-79,-2)} & \dots & s_t^{(80,-2)} \end{bmatrix} \tag{1}$$

$$s_t^{(i,j)} = \left[ v_t^{(i,j)},\ \theta_t^{(i,j)}, 0/1 \right]$$

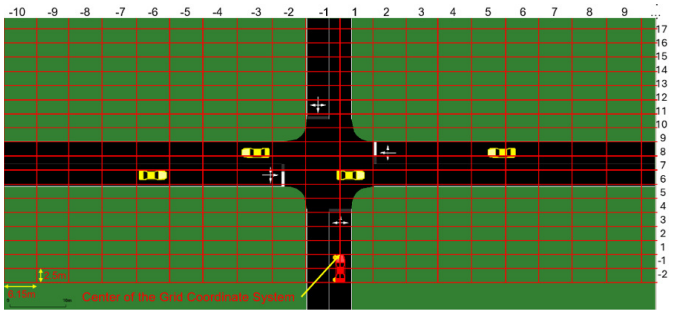



Fig. 1: Schematic diagram of the grid coordinate system.

*3) Action Space:* Due to the constraints of SUMO in vehicle kinematic control, this paper focuses exclusively on managing vehicle acceleration. Given that the DDPG algorithm is suitable for continuous action spaces, we define the agent's action space $A_t$ as vehicle acceleration, which ranges continuously from -1 to 1.

*4) Reward Function:* In tasks involving intersection navigation, the primary objective for the agent vehicle is to reach its destination as quickly as possible while avoiding collisions. To reflect this goal, the reward function incorporates several components: the time cost $t$, the requirement for maintaining an appropriate speed $v$, the penalty for collision $c$, and the reward for successfully reaching the destination $d$. Additionally, recognizing that real-world drivers strive to minimize disruptions to other vehicles, we include a penalty if the RL agent vehicle causes other vehicles to brake and wait due to improper driving, which are represented as $b$ and $w$, respectively. The specific formulation of the reward function is detailed below, where $R(S, A)$ and $r(S_t, A_t)$ represent the accumulated reward gained through one episode and the reward earned per time step, respectively. The coefficients in front of each item in the formula represent the corresponding weight values.

$$\begin{aligned} R(S,A) &= \sum_{t=1}^{T} r(S_t, A_t) \\ r(S_t, A_t) &= \alpha \cdot t + \beta \cdot v + \gamma \cdot (b+w) + \lambda \cdot c + \mu \cdot d \end{aligned} \tag{2}$$

### C. Federated Learning

The workflow of the FL framework is outlined in detail below. As illustrated in Fig. 2, the process can be divided into two primary components: local vehicle model training and edge node model aggregation.

*1) Local Vehicles Model Training:* The local vehicle training process follows the standard RL approach described in Section III.B, where the agent vehicle learns the optimal policy through numerous interactions within the environment, ultimately achieving the highest reward value.

*2) Edge Node Model Aggregation:* In the edge node model aggregation process, the Federated Averaging (FedAvg) algorithm [21] demonstrates advantages in improving the robustness and generalization of the global model, which reduce the risk of overfitting to a single data source by aggregating the models trained with data from different scenarios. Strategic adjustments such as optimizing local training duration (epoch) can be implemented to address convergence challenges arising from data disparities. Furthermore, FedAvg accommodates device heterogeneity including variations in computational capabilities and network conditions, while maintaining a generic model architecture across clients. The algorithm's flexibility and simplicity make it a versatile choice for distributed learning systems. Specifically, we employ FedAvg to aggregate models at the edge node through parameter averaging, which can be formulated as:

$$\omega_{\text{agg}} = \frac{1}{N} \sum_{i=1}^{N} \omega_i \tag{3}$$

where $\omega_i$ denotes the parameters of the $i$-th locally trained model and $\omega_{\text{agg}}$ represents the aggregated global model. To investigate the interplay between communication efficiency and model performance, we systematically vary the aggregation interval between local vehicles and the edge node while maintaining a fixed total training budget of 10,000 time steps. This experimental design allows us to analyze how aggregation interval influences reward convergence characteristics under heterogeneous conditions.

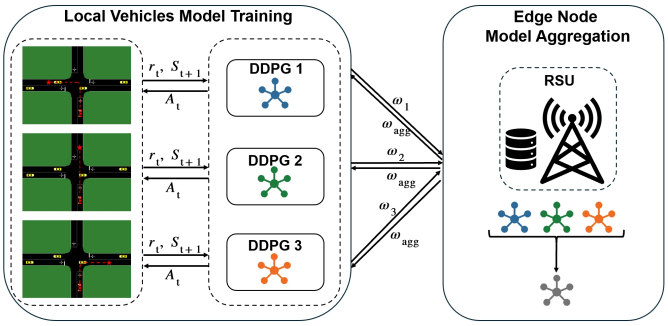



Fig. 2: The proposed federated reinforcement learning framework via model aggregation over vehicles with specific tasks.

## IV. Numerical Results

In this section, we conduct simulations to evaluate the effectiveness of the proposed RL reward function as well as the FL framework in the context of road intersection navigation under extreme settings in SUMO, as described in Section III.A.

### A. Reward Function Analysis

First, we run experiments to investigate the impact of various reward function settings. Applying various weights on travel time and collisions in the reward function can lead to different driving styles, e.g., cautious driving resulting in low collision rate and prolonged travel time, or traversing the intersection quickly with a significantly elevated collision rate. To analyze these trade-offs effectively, we fine-tune the coefficients $\alpha$ and $\lambda$ in (2), where $\alpha$ addresses the time costs, while $\lambda$ penalizes collisions. The remaining coefficients in the reward function take the values shown in Table III. Additionally, we refer to another reward function from literature [22] designed for self-driving navigation at unsignalized intersections. This function also enables the vehicle to reach its destination as soon as possible without collisions. Each reward function setting is trained with 10,000 time steps and tested with 100 episodes.

TABLE III: Values of the coefficients used in the experiments.

| Coefficient | Value |
|---|---|
| $\beta$ | 0.04 |
| $\gamma$ | -0.025 |
| $\mu$ | 5 |

Fig. 3 shows that a light weight on time cost leads to a prolonged average travel time, i.e., 74.83 time steps, when $\alpha$ being set to -0.005 and $\lambda$ being set to -10. Although this can ensure safety during operation as no collisions were found, driving at a significantly low speed could adversely affect

the driving of other vehicles on the road. Another important case is observed when $\alpha$ is -0.05 and $\lambda$ is -1, such that the collision rate for the agent vehicle rises to 9% due to the lightly set penalty coefficient for collisions. The average travel time is 44.87 time steps in this case, due to the agent vehicles not reaching their destinations, thus recording a shorter time. Modest weights, i.e., $\alpha$ = -0.05 and $\lambda$ = -10, give a balanced performance over travel time and collision rates. Compared to the reward function detailed in [22], the travel time is reduced by 6.49%, despite a 1% increase in collision rate.

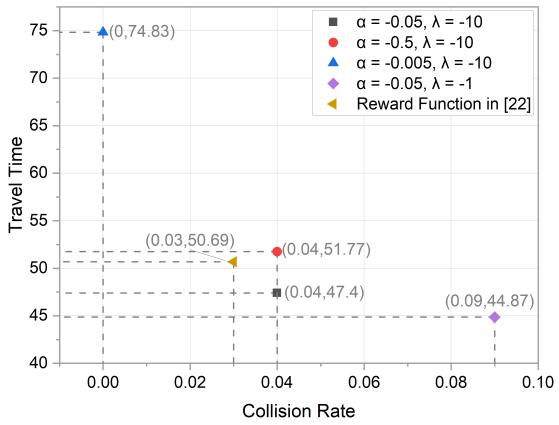



Fig. 3: Coefficient tuning effects and reward comparison.

### *B. Federated Learning Evaluation*

We conduct simulations to evaluate the effectiveness of the proposed FL framework by comparing it to two benchmark training frameworks using the reward function in (2) with $\alpha$ = -0.05 and $\lambda$ = -10. All three schemes aim to address the challenge of passing through intersections efficiently, minimizing collisions and disruptions to other vehicles, using the RL models outlined in Section III.B.

1) **Centralized RL Model (`Central-DDPG`)**: This scheme utilizes one centralized server to train the RL agent for random driving tasks, including going straight, turning left, and turning right across multiple episodes. This approach requires only one model, which simplifies training and deployment. However, a potential downside is that the agent may struggle to specialize in specific tasks, possibly resulting in slower convergence and suboptimal performance in extreme scenarios.
2) **Separated Task-Specific RL Models (`Sep-DDPG`)**: The framework utilizes three task-specific agents trained independently for distinct driving tasks: `DDPG1` for turning left, `DDPG2` for going straight, and `DDPG3` for turning right. The advantage of this approach is that each model can achieve high performance in its designated task. However, a notable disadvantage is that maintaining three separate models during both training and deployment adds complexity and increases resource requirements. For a comprehensive evaluation, performance metrics are averaged across all three models.
3) **Proposed FRL Framework (`FedAvg_DDPG`)**: The federated scheme trains RL models collaboratively, where three task-specific agents periodically upload their locally trained models to the edge server for aggregation into a single global model. This approach combines the benefits of task-specific specialization with the efficiency of a unified model, allowing for simplified implementation and high performance during testing or deployment. However, it requires maintaining three models during the training phase, and the collaborative training by local vehicles and RSUs incurs inevitable communication overhead during model transmission.

We begin by comparing the training efficiency of the three schemes. Fig. 4 illustrates the training process at each time step. For the Sep-DDPG scheme, we average the instantaneous rewards obtained by the three task-specific agents and plot the resulting averaged reward curve. As shown in the figure, all three schemes exhibit a converging and upward trend over time, but they converge at different speeds. For instance, considering an average reward of 6.0 as a benchmark, the FedAvg-DDPG scheme reaches this value at the 4593-th time step, while Sep-DDPG and Central-DDPG achieve the same reward at the 6068-th and 9841-th time steps, respectively. This demonstrates that FedAvg-DDPG converges significantly faster, with a speed improvement of up to 32.11% and 114.26%, respectively. In addition, the proposed FedAvg-DDPG scheme provides a gain of 21.00% in terms of the finally achieved reward value, compared to the other two benchmark schemes.

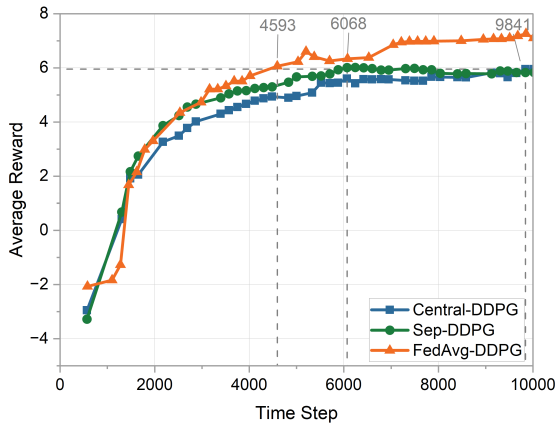
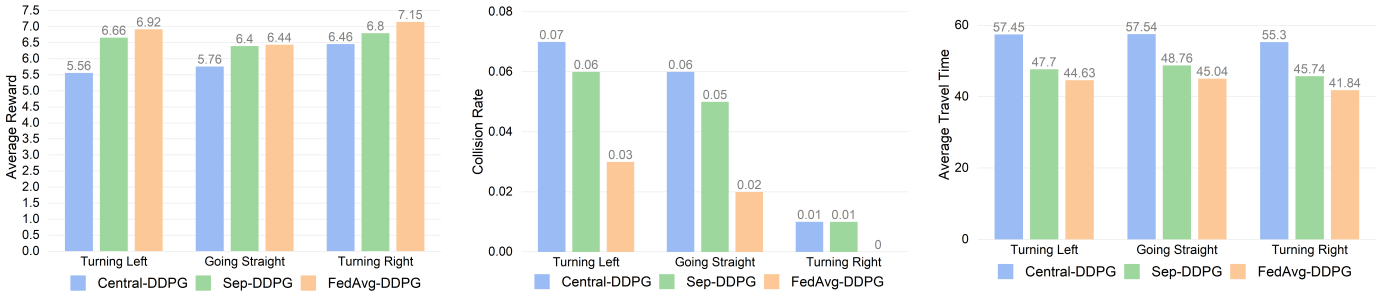



Fig. 4: The average reward obtained throughout training.

Next, we evaluate the test performance of the three schemes to assess their effectiveness in achieving rewards, avoiding collisions, and reducing travel time. For each task—going straight, turning left, or turning right—the trained models are tested in 100 episodes using random traffic flow passing through the intersection with an arrival rate of one every 10 time steps. The average reward values, collision rates, and travel times are presented in Fig. 5(a), (b), and (c), respectively. The results demonstrate that the centrally trained model (Central-DDPG) performs poorly across all three metrics com-




(a) Average Reward (b) Collision Rate (c) Average Travel Time

Fig. 5: Comparison of average reward, collision rate, and average travel time across different extreme scenarios.

pared to the task-specific models, whether trained separately (Sep-DDPG) or collaboratively using FL (FedAvg-DDPG). This is attributed to the centralized approach's inability to specialize in specific tasks, leading to not only slower convergence but also suboptimal performance in extreme scenarios. Between the two task-specific approaches, the proposed FedAvg-DDPG model outperforms Sep-DDPG in all metrics. Notably, FedAvg-DDPG significantly reduces collision rates across all three tasks, highlighting its ability to enhance road safety. Furthermore, the FL framework achieves this improvement while maintaining only one global model, thereby alleviating the system burden associated with managing multiple task-specific models. These results underscore the effectiveness of the proposed FL framework in improving both performance and efficiency in autonomous intersection navigation tasks.

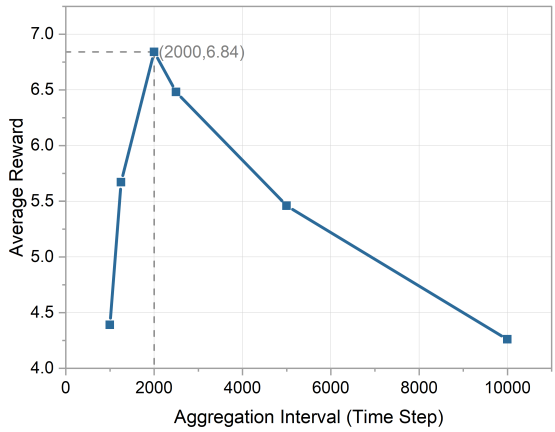



Fig. 6: The average reward obtained by FedAvg-DDPG vs aggregation intervals.

In FL, determining the optimal frequency for aggregating client models is a critical consideration. To address this, we investigate the impact of varying aggregation intervals—defined as the number of local training epochs performed on each client between two consecutive model uploads for aggregation—on the overall learning performance. As illustrated in Fig. 6, the average reward curve exhibits a biphasic trend, with the highest average reward of 6.84 achieved at a aggregation interval of 2,000 time steps. Both excessively low and excessively high aggregation intervals are found to negatively impact the model's performance, leading to a reduction in the average reward. This suggests that an intermediate aggregation interval strikes an optimal balance, enabling sufficient local training while ensuring timely model aggregation to maintain global performance.

## V. Conclusion and Future Work

This paper introduces a FRL framework to address the timeliness requirements for autonomous driving at intersections. We tackle the long-tail problem of data scarcity using SUMO simulation. The integration of FL enables collaborative training among distributed vehicles, resulting in a global model that outperforms individually trained agents in extreme scenarios, achieving higher rewards, lower collision rates and faster convergence. Our analysis of the aggregation interval between vehicles and RSUs provides insights for system optimization, identifying an optimal aggregation interval for enhanced performance. Currently, experiments are limited to simplified scenarios and have not yet considered real-world environmental factors, such as roadside buildings, which can affect communication quality. These complexities introduce additional constraints when optimizing the global model. Our future research will focus on addressing these challenges and allowing autonomous systems to navigate more effectively in varied environments.

## References

[1] Y. Li, W. Yuan, S. Zhang, W. Yan, Q. Shen, C. Wang, and M. Yang, "Choose your simulator wisely: A review on open-source simulators for autonomous driving," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 5, pp. 4861–4876, 2024.

[2] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.

[3] D. Team, "Deepdrive: A simulator that allows anyone with a pc to push the state-of-the-art in self-driving," 2020.

[4] NVIDIA, "Nvidia drive end-to-end platform for software-defined vehicles," 2024.

[5] M. Müller, V. Casser, J. Lahoud, N. Smith, and B. Ghanem, "Sim4cv: A photo-realistic simulator for computer vision applications," *International Journal of Computer Vision*, vol. 126, pp. 902–919, 2018.

[6] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics: Results of the 11th International Conference*. Springer, 2018, pp. 621–635.

[7] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *ArXiv Preprint arXiv:1606.01540*, 2016.

[8] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 2575–2582.

[9] H. Zhang, J. Bosch, and H. H. Olsson, "Real-time end-to-end federated learning: An automotive case study," in *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE, 2021, pp. 459–468.

[10] S. Doomra, N. Kohli, and S. Athavale, "Turn signal prediction: A federated learning case study," *ArXiv Preprint arXiv:2012.12401*, 2020.

[11] L. Barbieri, S. Savazzi, M. Brambilla, and M. Nicoli, "Decentralized federated learning for extended sensing in 6g connected vehicles," *Vehicular Communications*, vol. 33, p. 100396, 2022.

[12] L. U. Khan, Y. K. Tun, M. Alsenwi, M. Imran, Z. Han, and C. S. Hong, "A dispersed federated learning framework for 6g-enabled autonomous driving cars," *IEEE Transactions on Network Science and Engineering*, vol. 11, no. 6, pp. 5656–5667, 2024.

[13] Z. Zhang, S. Wang, Y. Hong, L. Zhou, and Q. Hao, "Distributed dynamic map fusion via federated learning for intelligent networked vehicles," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 953–959.

[14] Y. Fu, F. R. Yu, C. Li, T. H. Luan, and Y. Zhang, "Vehicular blockchain-based collective learning for connected and autonomous vehicles," *IEEE Wireless Communications*, vol. 27, no. 2, pp. 197–203, 2020.

[15] B. Liu, L. Wang, and M. Liu, "Lifelong federated reinforcement learning: A learning architecture for navigation in cloud robotic systems," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4555–4562, 2019.

[16] M. K. Abdel-Aziz, C. Perfecto, S. Samarakoon, M. Bennis, and W. Saad, "Vehicular cooperative perception through action branching and federated reinforcement learning," *IEEE Transactions on Communications*, vol. 70, no. 2, pp. 891–903, 2021.

[17] W. G. Najm, R. Ranganathan, G. Srinivasan, J. D. Smith, S. Toma, E. D. Swanson, A. Burgett *et al.*, "Description of light-vehicle pre-crash scenarios for safety applications based on vehicle-to-vehicle communications," United States. Department of Transportation. National Highway Traffic Safety, Tech. Rep., 2013.

[18] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *ArXiv Preprint arXiv:1509.02971*, 2015.

[19] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *ArXiv Preprint arXiv:1312.5602*, 2013.

[20] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021. [Online]. Available: http://jmlr.org/papers/v22/20-1364.html

[21] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[22] C. Spatharis and K. Blekas, "Multiagent reinforcement learning for autonomous driving in traffic zones with unsignalized intersections," *Journal of Intelligent Transportation Systems*, vol. 28, no. 1, pp. 103–119, 2024.