Mingfeng Yuan

✉ mfyuan@yorku.ca
☎ +1(437)9865257
🔗 Google Scholar
🔗 Personal Website
▶ YouTube Channel
📍 4700,KeeleStreet,Toronto,ON,Canada,M3J1P3

Education

2019 – present

- **PhD Earth and Space Science, York University, Toronto, Canada**
  Intelligent decision-making, planning and control
  Thesis title: *Adaptive Decision-Making for Autonomous Driving Considering Interaction and Uncertainty of Surrounding Vehicles.* **Resulted in 5 publications**

2016 – 2019

- **MSc Control Engineering, Tianjin University of Technology, Tianjin, China**
  Nonlinear Dynamics and Chaos-Based Cryptography
  Thesis title: *Modeling and Analysis of Complex Chaotic System and Design of Pseudo-random Number Generator using FPGA.* **Resulted in 4 published papers.** Nominated by the Degree Committee of the People's Government of Tianjin for the **Outstanding Master's Thesis Award in Engineering; National Scholarship**.

2013 – 2014

- **International Exchange Student Program, Electrical Engineering, Lovely Professional University, Phagwara, India**
  Student exchange program funded by **Beijing Municipal Government Scholarship: 20,000 RMB**.

2011 – 2015

- **BSc Electrical Engineering, Beijing Union University, Beijing, China**
  Graduation project title: *Electrical System Design for A Campus Living Area.*
  Full Mark in graduation project.

Employment History

Academia:

2019 – Present

- **Research and Teaching** York University, Canada
  1) Researcher on intelligent decision-making, planning and control for multi-robot.
  2) Serving as Lab instructors for many courses including Introduction to Control Systems (undergraduate & graduate levels), Feedback Control Systems (undergraduate & graduate levels), Engineering Mechanics (undergraduate), and Physics (undergraduate).

Industry:

2021 – Present

- **R & D Cooperation.** Quanser Company, Canada
  1) This is a Mitacs project with joint collaboration between **Quanser company, R&D department, and York University**.
  2) Self-driving car plateform development (including sensors, actuators, controllers, and traffic scenarios) (ROS1 & ROS2)
  3) Virtual labs: Developing virtual mechatronics systems (rotary servo, self-driving cars) that behave in the same way as the physical hardware and can be measured and controlled using MATLAB/Simulink and Python.

Ongoing Projects

2020 – Present

- **Effective Human-Machine Cooperation with Intelligent Adaptive Autonomous Systems (IDEaS project).**
  Department of National Defence (DND) and York University, Canada
  The main objectives are:
  1) Trust Model Development & Experimentation (Demo)
  2) Intelligent Adaptive Automation Aid (Search and Rescue)

Ongoing Projects (continued)

2022 – Present

Miniature Imaging Fabry-Perot Spectrometer - Canadian Payloads
Canadian Space Agency (CSA), York University, and MPB Communications.
The main objective is to obtain high-resolution measurements of molecular oxygen using a miniature imaging Fabry-Perot spectrometer (Link).

2019 – Present

Decision Making for Autonomous Vehicles in Dynamic and Interactive Environments Using Learning-Based Method
NSERC Alliance Program, Mitacs Accelerate Program, and York University.
The main objectives are:
1) Making deep reinforcement learning more efficient, both computationally and statistically, in a principled manner to enable its applications in critical domains;
2) Scaling deep reinforcement learning to design and optimize societal-scale multi-agent systems, especially those involving cooperation and/or competition among humans and/or robots (Demo1, Demo2).

Competitions

Unmanned Aerial Vehicle (UAV) Competition at ICUAS 2023, Poland, Warsaw. (out of 39 teams worldwide, 3rd place, the only finalist Canadian team)
Objectives: The ICUAS 2023 competition is centered around an UAV challenge to detect cracks in an unknown environment (both ROS simulation and hardware implementation).
Benchmark 1: unknown 3D environment exploration; (Demo1, Demo2)
Benchmark 2: crack detection;
Benchmark 3: UAV pose estimation;

"Freescale" Cup National Smart Car 2013, 2nd place, Beijing Union University
The main purpose of this competition is to build a self-balancing car using a microcontroller and essential sensors to keep two wheels upright and achieve vision-based tracking. The car is primarily divided into six main modules: control module, sensor module, power supply module, motor driver module, and debugging module. The core control unit of the car is Kinetis K60. The main sensors used include gyroscopes, accelerometers, rotary encoders, linear CCD, etc;

Supervision and Mentoring

2023 – Present

MSc Student
Transferring Multi-Agent Reinforcement Learning Policies for Autonomous Driving using Sim-to-Real
1) Guide the MSc student throughout his research.
2) Increase his self-confidence and allow him to follow his passion.
3) Provide support and advice to validate his work experimentally.

2020 – 2023

USRA Program Students
S1: Game-Theoretic Decision-Making for Autonomous Vehicles, UofT
S2: Deep Reinforcement Learning Based Decision-Making: Sim-to-Real Study, Western University
1) Explain and provide proper resources to the undergraduate students, so they can understand the underlying principles of their projects.
2) Help them implement their algorithms experimentally (hardware validation).
3) Provide advice on how to present their works at the Lassonde Undergraduate Research Award (LURA/USRA) Conference at the end of the internship.

Honours and Awards

2019 – Present

York University Graduate Fellowship
York University - $25,000

Honours and Awards (continued)

2023
- **Best Demo Award**
  IEEE International Symposium on Personal, Indoor & Mobile Radio Communications
- **Academic Excellence Fund**
  York University - $1701
- **Research Evaluation Conference, PhD category, 1^{st} Prize**
  York University - $750

2022
- **The 9th China International College Students' "Internet+" Innovation and Entrepreneurship Competition**
  China - $1, 500

2020 – 2021
- **Mitacs Award (work with Quanser company), Canada**
  Mitacs - $10, 000

2019
- **Carswell Scholarship, York University, Canada**
  York University - $10, 000

Professional Service and Memberships

- **IEEE Transactions on Industrial Electronics**, Reviewer
- **IEEE Transactions on Intelligent Transportation Systems**, Reviewer
- **Applied Mathematical Modelling**, Reviewer
- **Nonlinear Dynamics**, Reviewer
- **IEEE Robotics and Automation Letters**, Reviewer
- **IEEE International Conference on Robotics and Automation (ICRA)**, Reviewer
- **IEEE Robotics and Automation Society (IEEE RAS)**, Student member.
- **IEEE Industrial Electronics Society (IEEE IES)**, Student member

Research Publications

Journal Articles:

1. **Yuan, M.**, & Shan, J. (2023a). Scalable Game-Theoretic Decision-Making for Self-Driving Cars at Unsignalized Intersections. *IEEE Transactions on Industrial Electronics*.
   🔗 doi:10.1109/TIE.2023.3290255

2. **Yuan, M.**, & Shan, J. (2023b). From Naturalistic Traffic Data to Learning-based Driving Policy: A Sim-to-Real Study. *IEEE Transactions on Vehicular Technology*. 🔗 doi:10.1109/TVT.2023.3307409

3. **Yuan, M.**, & Shan, J. (2021). Deep Reinforcement Learning Based Game-Theoretic Decision-Making for Autonomous Vehicles. *IEEE Robotics and Automation Letters*, *7*(2), 818–825.
   🔗 doi:10.1109/LRA.2021.3134249

4. Jiao, X., Zhao, Y., Wang, X., **Yuan, M.**, Tao, J., Sun, H., ... Chen, Z. (2024). Learning-Based Acoustic Displacement Field Modeling and Micro-Particle Control. *Expert Systems with Applications*, *237*, 121503.

5. Li, Y., **Yuan, M.**, & Chen, Z. (2023). Constructing 3D Conservative Chaotic System with Dissipative Term Based on Shilnikov Theorem. *Chaos, Solitons & Fractals*, *171*, 113463.

6. Li, Y., **Yuan, M.**, Chen, Z., & Chen, Z. (2023). Coexistence and Ergodicity in A Variant Nosé-Hoover Oscillator and Its FPGA Implementation. *Nonlinear Dynamics*, *111*(11), 10583–10599.

7. Jiao, X., **Yuan, M.**, Tao, J., Sun, H., Sun, Q., & Chen, Z. (2023). Memristor Hyperchaos in A Generalized Kolmogorov-Type System with Extreme Multistability. *Chinese Physics B*, *32*(1), 010507.

8. Li, Y., **Yuan, M.**, & Chen, Z. (2022). Multi-Parameter Analysis of Transition from Conservative to Dissipative Behaviors for A Reversible Dynamic System. *Chaos, Solitons & Fractals*, *159*, 112114.

9. Li, Y., Chen, Z., & **Yuan, M.** (2022). The Transition from Conservative to Dissipative Flows in Class-B Laser Model with Fold-Hopf Bifurcation and Coexisting Attractors. *Chinese Physics B*, *31*(6), 060503.

10 Dong, E., & **Yuan, M.** (2019a). Topological Horseshoe Analysis and FPGA Implementation of A classical Fractional Order Chaotic System. *IEEE Access*, *7*, 129095–129103. **(Corresponding author)**.

11 Dong, E., Zhang, Z., & **Yuan, M.** (2019). Ultimate Boundary Estimation and Topological Horseshoe Analysis on A Parallel 4D Hyperchaotic System with Any Number of Attractors and Its Multi-scroll. *Nonlinear Dynamics*, *95*, 3219–3236.

12 Dong, E., & **Yuan, M.** (2018). Topological horseshoe analysis, Ultimate Boundary Estimations of A New 4D Hyperchaotic System and Its FPGA Implementation. *International Journal of Bifurcation and Chaos*, *28*(07), 1850081. **(Corresponding author)**.

13 Dong, E., & **Yuan, M.** (2019b). A New Class of Hamiltonian Conservative Chaotic Systems with Multi-Stability and Design of Pseudo-Random Number Generator. *Applied Mathematical Modelling*, *73*, 40–71. **(Corresponding author)**.

Conferences:

1 **Yuan, M.**, & Shan, J. (2023d). Learning Adaptive Cruise Control for Autonomous Vehicles Using End-to-End Deep Reinforcement Learning, Singapore: The 49th Annual Conference of the IEEE Industrial Electronics Society.

Book (chapter):

1 **Yuan, M.**, & Shan, J. (2023c). Game-theoretic Decision-making for Autonomous Driving Vehicles. In *Autonomous vehicles and systems-a technological and societal perspective* (pp. 269–301). River Publishers.

Under review:

1 **Yuan, M.**, & Shan, J. (2023e). *Enhancing Deep Reinforcement Learning via MPC Guidance for Autonomous Driving*. (Journal).

2 Kio, O. G., **Yuan, M.**, Shan, J., & Allison, R. S. (2023). *Performance-Based Data-Driven Assessment of Trust*.

Selected Teaching

2019-2023 **Instructor, LE/ENG4550 and LE/ENG5550 - Introduction to Control Systems (Fall, undergraduate & graduate levels)**
Department of Earth & Space Science, York University (Link)

2020-2023 **Instructor, LE/ENG4650 and GS/ESS5650 - Feedback Control Systems (Winter, undergraduate & graduate levels)**
Department of Earth & Space Science, York University

2020-2021 **Instructor, SC/PHYS1800 B - Engineering Mechanics (undergraduate level)**
Department of Physics and Astronomy, York University

2021-2022 **Instructor, SC/PHYS 1421 - Physics with Life Science Applications (undergraduate)**
Department of Physics and Astronomy, York University

Selected Press

2023 **Miniature Imaging Fabry-Perot Spectrometer, Canadian Space Agency, Canada**
Scientific Instruments Developed at Lassonde Fly High above the Clouds During Strato-Science 2023 Campaign [link1][link2]

2023 International Conference of Unmanned Aircraft Systems (ICUAS) Unmanned Aerial Vehicle (UAV) Competition, Warsaw, Poland
Lassonde Students Achieve High-Flying Success at International UAV Competition [link]

2019 **Carswell Scholars, York University, Canada**
Five Lassonde School of Engineering students Named Carswell Scholars [link]

I am passionate about working on different robots including aerial robots (e.g., drones), ground robots (e.g., self-driving cars), and industrial robots (e.g., robotic arms). Each type of robots has its own challenges; however, there are common challenges that are blocking the way of having highly intelligent robots in our daily life. Safety, security, privacy, and public trust are among the challenges that require more work in terms of regulations and public awareness. On the other hand, robustness against disturbances and uncertainties, cooperative tasks, and collisions avoidance between objects, different robots, and/or humans in the environment are technical challenges.

My research vision is to conduct **excellent and beneficial research** in the field of **"Intelligent Decision-Making, Planning and Control"**. This vision includes dealing with different types of robots such as self-driving cars, unmanned aerial vehicles (UAVs), unmanned ground vehicles (UGVs), and industrial robotic arms. To this end, and building on my research expertise, following research directions represent the core of my research plan:

1. **Human behaviour modelling, estimation, and prediction using AI methods:**
   (a) Human-in-the-loop learning and learning from human feedback;
   (b) Behavior/intention prediction for heterogeneous traffic participants;
   (c) Reinforcement learning, imitation learning, and inverse reinforcement learning.
2. **Learning for Safe and Robust Control:**
   (a) Decision-making under uncertainty;
   (b) Model predictive control combined with learning techniques
   (c) Safe exploration for model learning and control
3. **Digital Twin in Verification and Validation of Autonomous Technologies:**
   (a) Simulation verification and validation of autonomous technologies;
   (b) Standardization of data and interfaces for validation;
   (c) Sim-to-Real Transfer
4. **Autonomous Industrial Inspection Robot:**
   (a) Autonomous navigation in unknown environments
   (b) Multi-sensor-based fault detection
   (c) Obstacle avoidance considering multiple fast-moving obstacles

Although above research directions are based on my expertise, they are not just extrapolation of my previous work. They are meant to be broader extensions of significant contribution to the robotics field.

1 Human behaviour modelling, estimation, and prediction using AI

Taking autonomous driving vehicles (ADVs) as an example, modeling human-like driving behaviors is of great significant to improve driving safety since ADVs can exhibite human-like behaviors in order to be predictable for other human road users. A significant challenge in autonomous driving is ensuring safe and cooperative interaction with human traffic participants. A crucial aspect of this challenge is accurately predicting the intentions and trajectories of other road users and using these predictions to make informed decisions. This prediction task is exceptionally demanding due to factors like dynamics, road conditions, and the behavior of surrounding agents, often displaying various possible outcomes. Additionally, leveraging these prediction models for real-time and interaction-aware decision-making is equally challenging, **ensuring safety, minimizing energy consumption, compliance with traffic rules, cooperation with diverse road users, and passenger satisfaction**. AI methods offer promising solutions to address these challenges, enhancing system performance, scalability, and intelligence, enabling autonomous vehicles to navigate complex driving environments.

1.1 Related Work

A considerable part of my PhD was dedicated to investigate learning-based human-like driving policy, especially deep reinforcement learning (RL), imitation learning (IL), and game theory. I have proposed an efficient training scheme called Deep Recurrent Q-learning from Demonstration algorithm (DRQfD) [1] for lane-changing decision-making to address the low sample efficiency in RL and the poor generalization capability in IL. LSTM is used to predict future states of surrounding vehicles, helping to address the Partially Observable Markov Decision Process problem in autonomous driving. The experimental results show that our proposed method outperforms IL in terms of safety, travel efficiency, and human likeness. In addition, I also proposed a novel approach for implementing game-theoretic decision-making in combination with deep reinforcement learning to allow vehicles to make decisions at an unsignalized intersection, achieving an End-to-end self-play training [2]. The game-theoretic model allows anticipating reactions of additional vehicles to the movements of the ego-vehicle without using any specific coordination or vehicle-to-vehicle communication. The overall decision-making framework proposed in this work exhibits great potential to enhance the practical application of RL-driven human-like autonomous driving (AD).

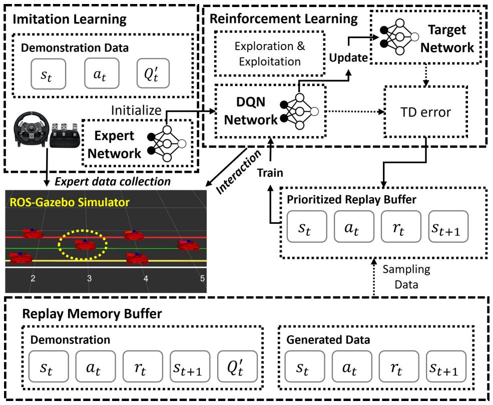


Fig. 1: RL Learning from Experts

1.2 Future Directions

- **Extend** my work and utilize human driving data to learn expert policies and guide the training of RL agents;
- **Address** the partially observable markov decision process problem in autonomous driving;
- **Improve** expression ability of reward function in RL for training complex human driving behaviors;

2 Learning for safe and robust control

Safety-critical tasks are prevalent in practical robotic applications, especially when robots operate near humans with limited environment knowledge. This includes scenarios like robots working alongside humans in **manufacturing**, **autonomous cars** in urban settings, or **quadrupeds in worksites**. Consequently, it's crucial to develop controllers that can adapt to dynamic environments while ensuring theoretical safety guarantees for real-world robotic applications.

2.1 Related Work

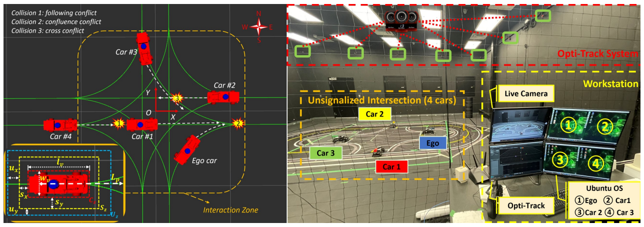


Fig. 2: Sim-to-Real Transfer (ROS-Gazebo)

Regarding the safe and robust control, I proposed a novel scalable robust adaptive decision-making framework based on game-theory, model predictive control, and interaction graph for resolving driving conflicts at unsignalized intersections [3]. The work considered **robustness against uncertainties such as simplified kinematic models and unknown driving preferences of surroundings**. In the payoff function design of decision-making, multiple driving features are considered including driving safety, fuel consumption, travel efficiency, and driving aggressiveness. To reduce the computational complexity of game theory, the concept of switching directed graphs is incorporated into the decision-making framework. Finally, **the algorithm is verified on both hardware and a high-fidelity simulator with multiple vehicles**. According to the testing results, it can be conducted that **the proposed algorithm makes robust adaptive decisions for ADVs, meanwhile, the performance of the algorithm in terms of interpretability, computational efficiency, and scalability can be guaranteed.**

2.2 Future Directions

- **Design and implement controllers** that combine RL and MPC in order to exploit the advantages of both, and therefore, obtain a controller that is optimal and safe;

- **Extend** my work and design robust controllers for UAVs (and different robots such as UGVs and industrial robots) considering uncertainties;

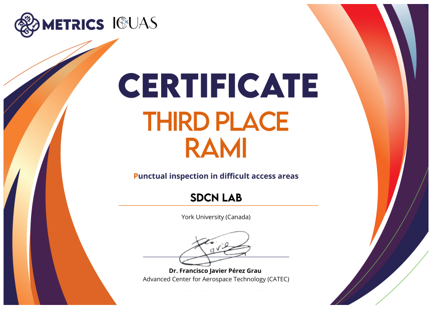
- **Propose** algorithms to achieve safe exploration for model learning and control;

3 Autonomous Industrial Inspection Robot:

Autonomous industrial inspection robot is a cutting-edge technological solution designed for the automation and optimization of inspection tasks within industrial settings. These robots are equipped with advanced sensors, computer vision, and artificial intelligence capabilities, allowing them to navigate autonomously, detect defects, gather data, and perform a wide range of inspection tasks with a high degree of accuracy and efficiency. They are a transformative addition to various industries, such as **manufacturing**, **quality control**, and **infrastructure maintenance**, as they not only **improve productivity** but also **enhance safety by minimizing human involvement in hazardous or repetitive inspection processes**.

3.1 Related Work

In 2023, our team, SDCNLAB, achieved the 3^{rd} place out of 39 teams worldwide in ICUAS 2023 Unmanned Aerial Vehicle (UAV) Competition, which is based on challenges faced by UAVs performing infrastructure inspection in an 3D unknown environment [4]. From this scenario three benchmarks arise that will test the perception capabilities, speed and intelligence of UAVs. To perform a successful inspection, an UAV needs to navigate through a dense 3D environment, estimating its pose and avoiding obstacles by using onboard sensors, including RGB-D camera and IMUs. Upon reaching points of interest, the UAV needs to scan the area and detect any defects in the infrastructure. However, the planner implemented for this UAV competition is designed for static environments and can tackle slowly moving obstacles (below 0.5 m/s) without any modification. A efficient planner for dynamic environment navigation is also needed.



METRICS ICUAS

CERTIFICATE

THIRD PLACE

RAMI

Punctual inspection in difficult access areas

SDCN LAB

York University (Canada)

Dr. Francisco Javier Pérez Grau

Advanced Center for Aerospace Technology (CATEC)

Fig. 3: UAV Competation

3.2 Future Directions

- **Building on** the obtained results of depth sensors and overcome associated practical problems such as making the machine learning model light enough to be deployed using CPUs instead of GPUs.

- **Extending** the results of avoidance maneuvers to consider cases with multiple fast-moving obstacles (UAVs).

4 Digital twin in verification and validation of autonomous technologies

Autonomous technologies are rapidly evolving in the last years, fueled by progress in key enabling technologies, such as accurate positioning, advanced environment perception, vehicular communications, and cybersecurity. However, the automotive industry faces a major challenge in ensuring the safety and reliability of the developed functions. The automotive industry is governed by strict test and validation rules, which require a thorough evaluation of all possible situations that an automated function will face in the real world. Testing all possible scenarios is unfeasible and unaffordable. Consequently, verification and validation (V&V) procedures and methodologies remain a key unresolved challenge for the validation of highly automated driving functions. The impact of rapid advances in Artificial Intelligence (AI) in the last years also raises a question about how to include them in a V&V procedure. New methodologies are required to improve their predictability and transparency to ensure their trustiness in a safety-critical field such as driving and search & rescure. Appropriate $V\&V$ procedures are required to put the latest autonomous driving functions into practice. Virtual or hybrid simulation environments, testing, data production and management, adoption of standards, and the use of Machine Learning and Al have some of the key roles in the current paradigm shift of Autonomous technologies validation.

4.1 Related Work

The road-testing of autonomous driving vehicles is costly and energy-consuming, not to mention posing a threat to pedestrians. In light of this, I build a high-fidelity simulation platform based on ROS-Gazebo that simulates dynamic environments with human-like road participants, different kinds of sensors, and vehicle dynamics for training and evaluating of different driving technologies [3, 5, 6]. The aforementioned works about modeling various human-like driving behaviors could be used to test autonomous driving technologies in high-fidelity simulation environments, which is of great significance to improve the safety of autonomous driving systems, as well as reduce their dependence on road tests. Additionally, this digital twin could reduce the gap in Sim2real transfer, especially for training deep reinforcement learning-powered prediction, decision-making, and motion planning of autonomous vehicles.

4.2 Future Directions

- **Propose** a standardised evaluation approach, including standard evaluation metrics and test beds for autonomous driving, which exposes candidate models to a range of novel and dangerous scenarios to fully evaluate their driving performance and safety levels.
- **Achieve** data and metadata generation for validation;
- **Transfer** multi-agent reinforcement learning policies for autonomous driving using sim-to-real;

References

[1] **Mingfeng Yuan** and Jinjun Shan. From naturalistic traffic data to learning-based driving policy: A sim-to-real study. *IEEE Transactions on Vehicular Technology*, 2023.

[2] **Mingfeng Yuan** and Jinjun Shan. Deep reinforcement learning based game-theoretic decision-making for autonomous vehicles. *IEEE Robotics and Automation Letters*, 7(2):818–825, 2021.

[3] **Mingfeng Yuan** and Jinjun Shan. Scalable game-theoretic decision-making for self-driving cars at unsignalized intersections. *IEEE Transactions on Industrial Electronics*, 2023.

[4] THE 2023 INT´L CONFERENCE ON UNMANNED AIRCRAFT SYSTEMS. https://uasconferences.com/2023_icuas/, 2023.

[5] **Mingfeng Yuan** and Jinjun Shan. Learning adaptive cruise control for autonomous vehicles using end-to-end deep reinforcement learning. Singapore, 2023. The 49th Annual Conference of the IEEE Industrial Electronics Society.

[6] **Mingfeng Yuan** and Jinjun Shan. Game-theoretic decision-making for autonomous driving vehicles. In *Autonomous Vehicles and Systems-A Technological and Societal Perspective*, pages 269–301. River Publishers, 2023.

Dear Professor,

The **decision-making module is crucial for safe and efficient driving in autonomous vehicles** (AVs). However, AVs face significant challenges in **coexisting with human driven vehicles and making fast and optimal driving decisions in complex and unknown traffic environments with only partial observations** (unknown driving behaviours of surrounding vehicles).

Predictions of the future motion of the other road users are not possible with absolute certainty. Uncertainties due to simple vehicle dynamic model used within the framework, and/or an improper estimation of the driver level of the other agents can lead to unsafe control actions taken by the AV. A high fidelity dynamic model can be used to eliminate certain degree of uncertainty but is accompanied by increased computational burden. Also, most likely, the interactions between vehicles in a given traffic scenario might be short for the AVs to estimate an accurate driver model.

Modeling human-like driving behaviors is of great significant to improve driving safety since AVs can exhibit human-like behaviors in order to be predictable for other human road users. Thus, designing decision-making algorithms for autonomous vehicles in complex traffic scenarios are vital and will be my research focus. Here are the three recent most significant contributions:

- **Contribution #1 (IEEE RA-L):**

  I proposed a novel approach for implementing game-theoretic decision-making in combination with deep reinforcement learning to allow vehicles to make decisions at an unsignalized intersection with partial observability, achieving an end-to-end self-play training. Sim2real transfer is successfully achieved by building a high-fidelity simulator and domain randomization.

  video link: `https://www.youtube.com/watch?v=mPtoojXh2-s&t=6s`

- **Contribution #2 (IEEE TVT):**

  For the first time, to the best of our knowledge, I proposed an efficient training scheme called Deep Recurrent Q-learning from Demonstration algorithm (DRQfD) for lane-changing decision-making to address the low sample efficiency in reinforcement learning and the poor generalization capability in imitation learning.

  video link: `https://www.youtube.com/watch?v=Svp2S1OaSB8&t=22s`

- **Contribution #3 (IEEE TIE):**

  An adaptive decision-making algorithm is designed using receding horizon optimization, level-k game theory, and directed switching graph to address interactions between autonomous vehicle and vehicles with varying driving preferences in complex unsignalized intersections, addressing the challenges of computational complexity and scalability faced by game-theoretic algorithms.

  video link: `https://www.youtube.com/watch?v=q6vKrjqHD54&t=6s`

In the next pages, I will share three recent journal publications.

Sincerely,
MingfengYuan
Department of Earth and Space Science and Engineering
Lassonde School of Engineering, York University, Toronto, Canada

Deep Reinforcement Learning Based Game-Theoretic Decision-Making for Autonomous Vehicles

Mingfeng Yuan, Jinjun Shan, *Senior Member, IEEE*, and Kevin Mi

***Abstract*—This letter presents an approach for implementing game-theoretic decision-making in combination with deep reinforcement learning to allow vehicles to make decisions at an unsignalized intersection by use of 2D Lidar to obtain their observations of the environment. The main novelty of this work is modeling multiple vehicles in a complex interaction scenario simultaneously as decision-makers with conservative, aggressive, and adaptive driving behaviors. The game model allows anticipating reactions of additional vehicles to the movements of the ego-vehicle without using any specific coordination or vehicle-to-vehicle communication. The solution of the game is based on cognitive hierarchy reasoning and it uses a deep reinforcement learning algorithm to obtain a near-optimal policy towards a specific goal in a realistic simulator (ROS-Gazebo). The trained models have been successfully tested on the simulator after training. Experiments show that the performance of the lab cars in the real-world is consistent with it in the simulation environment, which may have great significance to improve the safety of self-driving cars, as well as may reduce their dependence on road tests.**

***Index Terms*—Deep reinforcement learning, cognitive hierarchy theory, LSTM network, self-driving car, decision making.**

I. Introduction

WITHIN half a century, autonomous driving vehicles (ADVs), together with human-driven vehicles (HDVs), will be employed in traffic scenarios, where the interactions of ADVs and HDVs will constantly occur. Current mainstream level-4 autonomous driving solutions limits these interactions rather than accelerate it. For example, in complicated interactive cases, the ADV inclines to slow down and pause rather than spontaneously find another way through [1]. Solving decision-making problems for ADV in dynamic and interactive environments is challenging since it is almost impossible by predefining codes or rules. In addition, it has been estimated that ADVs need to be running for 275 million miles without fatality to assure the same rate of reliability as existing HDVs [2]. There is no doubt that only the road-testing phase is already costly and energy-consuming, not to mention posing a threat to pedestrians. Therefore, to model various driving behaviors and to test decision-making algorithms in high-fidelity simulating environments are of great significance to improve the safety of ADVs, as well as reduce their dependence on road tests. Motivated by the necessity of developing simulating tools for verifying and validating the autonomous driving systems running in traffic with both ADVs and HDVs, we mainly focus on modeling vehicle interactions based on deep reinforcement learning (DRL) and game theory (GT), since learning agents can potentially discover such complex interactions automatically through exploration, as behaviors and actions evolved, leading to more successful driving experiences with data collected through interactions in multiple agents environments (over time and/or in simulation).

The interaction process of ADVs in real traffic scenes has the following characteristics. First, the actions to be taken by ADV will be affected by the actions of surrounding vehicles, and vice versa. Secondly, vehicles not only cooperate to avoid the collision but also compete due to their different driving strategies, thus producing rich dynamic interaction behaviors. The driving policies of HDVs are unknown to ADVs, which can only be estimated by observing actions taken by opponents.

Several approaches are reported in the open literature to model multi-vehicle interactions, including decision trees [3], [4], dynamic Bayesian networks [5], partially observable Markov decision processes [6], model predicted control [7], and data-driven method [8], which serve primarily as high-level controllers. Game theory, mathematical models of strategic interaction among rational decision-makers, can be used to study the strategic reasoning of multiple vehicles and to model the interaction behavior between vehicles. In traditional approaches, the interaction behavior between agents is modeled by a one-shot normal-form game, in which each vehicle will choose driving actions (“move” and “wait”) through the payoff matrix without considering the dynamic characteristics of the vehicle. In addition, with the increase of the number of vehicles, the computational complexity will increase exponentially, making it challenging to realize real-time decision-making. In the current study, [9], [10] exploited a game-theoretic approach to modeling

Manuscript received September 8, 2021; accepted November 29, 2021. Date of publication December 13, 2021; date of current version December 20, 2021. This letter was recommended for publication by Associate Editor B. K. H. Low and Editor J. Kober upon evaluation of the reviewers’ comments. This work was supported in part by NSERC Alliance under Grant ALLRP 555847-20, and in part by Mitacs Accelerate under Grant IT26108. *(Corresponding author: Jinjun Shan.)*

Mingfeng Yuan and Jinjun Shan are with the Department of Earth and Space Science, Lassonde school of Engineering, York University, ON M3J 1P3, Canada (e-mail: mfyuan@yorku.ca; jjshan@yorku.ca).

Kevin Mi is with the Division of Engineering Science, University of Toronto, ON M5S 1A1, Canada (e-mail: kevin.mi@mail.utoronto.ca).

Digital Object Identifier 10.1109/LRA.2021.3134249

vehicle interactions in highway or intersection cases. Li *et al.* [11] modeled the interactions among vehicles at unsignalized intersections using the leader-follower game. Tian *et al.* [12] integrates a game-theoretic formalism, receding-horizon optimization, and an imitation learning algorithm to obtain control policies. Sankar and Han [13] proposed an adaptive control strategy that accounts for the uncertainties in the vehicle dynamic model and the driver model estimation. Our method largely differs from previous works that usually rely on relatively simple simulations without considering the noise existing in observation or dynamics in physical ADVs, thus imposing a large gap on their application to real vehicles. The decision-making scheme proposed in this letter relies solely on the onboard sensors, without assuming any coordination, communication, or shared control with the surrounding cars. Also, the actual actions and movements of the vehicle are performed on a realistic simulator.

Another line of research learns driving behavior in simulation, making it suitable for reinforcement learning (RL) because it is possible to learn from failure cases during learning in a safe environment. In addition, RL has proven effective at designing control policies for an increasing number of tasks in both single-agent systems and multi-agent systems, including navigation [14] and wireless communication [15]. Leveraging such methods for learning autonomous driving policies is emerging as a particularly promising approach [16]. The unapologetic nature of the trial-and-error process in RL compounds the difficulty of ensuring functional safety. These adversities call for learning that first takes place in simulation before transferring to the real world [17]. This transfer, often referred to as sim2real, is challenging due to discrepancies between conditions in simulation and the real world (such as vehicle dynamics and sensor data) [18]. Simultaneously, the act of colliding or nearly colliding is essential to the learning process, enabling future policy rollout to incorporate these critical experiences. How are we to provide safe multi-vehicle learning experiences without forgoing the realism of high-fidelity training data? There is a shortage of work that addresses this challenge.

In this letter, we apply DRL along with GT to modeling decision makers with different reasoning levels in an unsignalized intersection case. What distinguishes our method from existing studies is that all the drivers in a multi-move scenario make strategic decisions simultaneously, instead of modeling the ADV as a decision-maker and assuming predetermined actions for the rest of the drivers. This is achieved by combining a cognitive hierarchy theory also called level-k reasoning with a RL called Dueling Double Deep Q-Network with Prioritized Experience Replay (D3QN PER). Earlier studies [6], [9], [19]–[21] were trying to combine RL with level-k theory so that agents can autonomously learn policies with rich interaction behaviors. They pointed out that it's worth exploring how to extend to a more general setting where a level-k agent selects its best response to the action of the other agent who reasons according to a distribution over lower levels instead of only at level-(k-1) [19]. Based on this, this letter introduces the LSTM network into a RL to solve the hidden state problem, so as to effectively train a policy that can adapt to the autonomous interaction in the multi-strategy mixed environment.

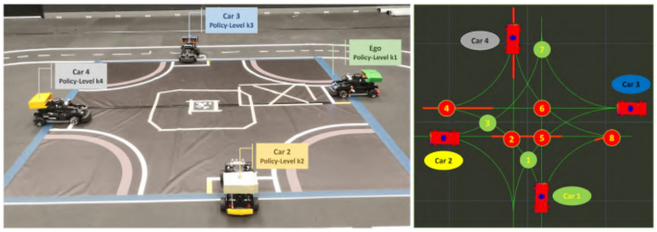


Fig. 1. Four way intersection: Car 1 is ego car; Car 2, 3 and 4 are opponents with different reasoning levels. Number 1 to 8 are the collision areas from the perspective of ego.

This letter is organized as follows. In Section II, the problem being treated in this letter is defined. In Section III, we apply level-k game theory, LSTM network along with RL algorithm to modeling decision-making for four cars in an intersection scenario. The algorithm is tested in Section IV by both simulator and hardware. The letter is summarized and concluded in Section V.

II. Problem Definition

A. Modeling Scenarios

The unsignalized intersection was chosen as the scenario in our work, since it's much more complex than other traffic scenarios, where each car chooses to enter the intersection area, and drivers constantly interact with the surrounding road users to safely and efficiently cross the intersection.

In this subsection, we will introduce a scenario consisting of 12 paths and four cars, among which car 1 is ego vehicle and the rest of the vehicles are opponents as shown in Fig. 1. To better show the results, the problem can be simplified as follows. The number of opponents ego encounters at the intersection is generated randomly, which can be 0, 1, 2 or 3. Ego has three tasks: turning left, going straight, and turning right. All opponents go straight but have two optional trained policies, namely conservative and aggressive driving behavior which is unknown to the Ego vehicle. The details about training of various driving policies are presented in Section III. It's worth noting that the method proposed in this letter can naturally be applied to more complicated scenarios, where all cars have no limitation on the path selection. In this research, there are 81 scenarios in total as mentioned above since there are 24 combinations in the case of ego versus three opponents, 36 combinations in the case of ego versus any two opponents, and 18 combinations in the case of ego versus one opponent. There are also 3 combinations when the ego passes through an empty intersection. To focus more on complex scenarios, the probability of having four cars, three cars, two cars, and one car interaction scenario will occur with probability 40%, 30%, 20%, and 10% respectively during the training.

B. Observation Space

Lidar is one of the most important sensors in the development of self-driving car because of its ability to adapt to different lighting conditions and its robustness to the environment. The point clouds generated by Lidar belong to long sequences information. To process the point cloud data, the LSTM network with

512 cell units is used in this letter to deal with partially observed environments. Due to various strategies of the opponent cars, the ego car may be confused if it is heavily rewarded for selecting an action in one state and then penalised for choosing the same action in the same situation next time, making the training process unstable. Therefore, the action chosen by ego depends not only on the current observation but also on a fixed number of the most recent observations.

The point cloud information is controlled to be 360 dimensions; the frequency is 50 Hz; the detection distance is [0.1, 3] m. In the intersection scenario, there are 8 collision areas where ego vehicle and opponent vehicle's path overlap, as shown in Fig. 1.

To combat limitations of using only 2D Lidar, each vehicles' distance to the collision points were added. Also, vehicle's own state information, and velocities of the opponent vehicles, represented as an array [v_1, v_2, v_3, v_4, p_1, p_2, p_3, p_4], where v_1 to v_4 are the speeds of car 1 to car 4 at each time step, and p_1 to p_4 correspond to the path each car selected (obtaining from turn signal) to further supplement the observation space. It can be extended to more complex cases without loss of generality.

C. *Action Space*

The action space includes 5 possible actions that each vehicle can undertake:

1 Maintain: Maintain current speed.

2 Accelerate: Increase speed of vehicle at 1.5 m/s^2, ignored if maximum velocity is already reached.

3 Fast Accelerate: Increase speed of vehicle at 3 m/s^2, ignored if maximum velocity is already reached.

4 Brake: Reduce speed of vehicle at 1.5 m/s^2, ignored if vehicle is stationary.

5 Hard Brake: Reduce speed of vehicle at 3 m/s^2, ignored if vehicle is stationary.

D. *End-to-End Control Scheme*

The end-to-end scheme in this letter is proposed to model different driving policies of self-driving cars at uncontrolled intersection. Firstly, the features of 2D Lidar point cloud data and car states are extracted through LSTM to generate observation information. Secondly, Q value is output through the full connection layer with dueling structure. An improved algorithm is proposed by combining the advantages of the traditional deep Q network (DQN), double DQN and Prioritized Experience Replay (PER) algorithm, which is called D3QN PER. Finally, the best action in each state can be generated by choosing the maximum Q value. Since the intersection can be represented by 12 paths, we used the pure pursuit controller to generate steering commands. The research framework is shown in Fig. 2.

III. Driver Interaction Model

The driver interaction model developed in this letter enables the modeling of driver-to-driver and driver-to-autonomous vehicle interactions through the use of level-k reasoning and D3QN PER algorithm. The model is a "policy" which is a stochastic map from the observation space of the driver to their action space (see Section II). In other words, this map assigns a probability distribution over possible actions for every observation. In the following sections, we explain how this model is generated.

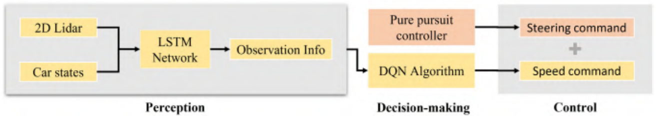


Fig. 2. End-to-end control scheme.

A. *Level-k Reasoning*

In order to model the strategic decision-making process of human drivers, a game-theoretical concept named level-k reasoning is used. The level-k approach is a hierarchical decision-making concept and presumes that different levels of reasoning exist for different humans. The lowest level of reasoning in this concept is called level-0 reasoning. A level-0 agent is a non-strategic/naive agent since their decisions are not based on other agents' possible actions but consist of predetermined moves. In one level higher, a strategic level-1 agent exists, who determines their actions by assuming that the other agents' reasoning levels are level-0. Hence, the actions of a level-1 agent are the best responses to level-0 actions. Similarly, a level-2 agent considers other agents as level-1 and makes their decisions according to this prediction. The Process continues following the same logic for higher levels. In some experiments, humans are observed to have at most level-3 reasoning, which may, of course, depend on the type of game being played. To generalize, all level-k agents, except level-0, presume that the rest of the agents are level-(k-1) and make their decisions based on this belief. Since this belief may not always hold true, the agents have bounded rationality [22], [23].

B. *Dueling Double Deep Q Network With Prioritized Experience Replay Algorithm (D3QN PER)*

In this letter, we use a more efficient algorithm by combining all advantages of Dueling DQN, Double DQN, and PER called D3QN PER [24] to train our ego vehicle. The comparison of learning efficiency of different algorithms is presented in Section IV.

The DQN algorithm [25] uses Q-learning to provide labeled samples for the deep Q-network: $Q(s, a; \theta)$ with parameters θ, which can be estimated by optimizing the following sequence of loss functions at iteration i:

$$L_i\left(\theta_i\right) = \mathbb{E}\left[\left(y_i^{DQN} - Q\left(s, a; \theta_i\right)\right)^2\right], \tag{1}$$

with

$$y_i^{DQN} = r + \gamma \max_{a'} Q\left(s', a'; \theta^-\right), \tag{2}$$

where r is the reward for taking action a in given state s; a' is next action taking in next state s'; γ is discount factor; θ^- represents the parameters of target network $Q(s', a'; \theta^-)$ which are frozen for a fixed number of iterations while updating the online network $Q(s,a;\theta_i)$ by gradient descent. The specific gradient update is

$$\nabla L_i\left(\theta_i\right)=\mathbb{E}\left[\left(y_i^{DQN}-Q\left(s,a;\theta_i\right)\right)\nabla_{\theta_i}Q\left(s,a;\theta_i\right)\right]. \quad (3)$$

However, both Q-learning and DQN have overestimation issue, since the max operator uses the same values to both select and evaluate an action. Double DQN solves this problem by using the following target instead of (2):

$$y_i^{Double}=r+\gamma Q\left(s',\underset{a'}{\arg\max}Q\left(s',a';\theta_i\right);\theta^-\right). \quad (4)$$

To accelerate the training, the key idea behind Dueling structure is that for many states, it is unnecessary to estimate the value of each action choice. Because, at some states, all actions of agent lead to the task to fail. Dueling structure can help to decompose estimator as the sum of:

$$Q(s,a;\theta,\alpha,\beta)=V(s;\theta,\beta)+A(s,a;\theta,\alpha), \quad (5)$$

where one stream of fully-connected layers output a scalar $V(s;\theta,\beta)$, and the other stream output an $|\mathcal{A}|$ dimensional vector $A(s,a;\theta,\alpha)$. Here, α and β are the parameters of the two streams of fully-connected layers. Eq. (5) is unidentifiable in the sense that given Q we cannot recover V and A uniquely. To address this issue of identifiability, the advantage function estimator can be forced to have zero advantage at the chosen action:

$$\begin{aligned}Q(s,a;\theta,\alpha,\beta)&=V(s;\theta,\beta)\\&\quad+\left(A(s,a;\theta,\alpha)-\frac{1}{|\mathcal{A}|}\sum_{a'}A\left(s,a';\theta,\alpha\right)\right). \quad (6)\end{aligned}$$

The idea behind the PER is that some experiences may be more important than others for training. We can take in priority experience that has a big error between deep Q-network and target netwark instead of selecting the experiences randomly. To generate the probability of being chosen for a replay, we have

$$P(i)=\frac{p_i^\alpha}{\sum_k p_k^\alpha}, \quad and \quad p_i=|\delta_i|+e, \quad (7)$$

where $|\delta_i|$ is the absolute Temporal Difference Error. Small constant e assures that no experience has 0 probability to be taken. If $\alpha=1$, it selects the experiences with the highest priorities while $\alpha=0$ for pure uniform randomness. Since we use priority sampling, which leads to the bias toward high-priority samples. To correct this bias, importance sampling weights (ISW) is used to adjust the updating by reducing the weights of the often seen samples [26],

$$\omega_i=\left(\frac{1}{N}\cdot\frac{1}{P(i)}\right)^\beta, \quad (8)$$

where β controls how much the ISW affect learning, and N is the size of replay buffer.

TABLE I
Training of Level-k Agent

```
Algorithm 1 Obtaining the Level-k Policy
Load the trained level-(k-1) (or rule-based level-0) policy, $\pi_{k-1}$;
Set the reasoning levels of all opponents in the environment,
  $p_i$, to level-(k-1): $\pi_{p_i}=\pi^{k-1}, i=2,3,\ldots,n_d$;
Initialize the ego driver's policy to a uniform action
  probability distribution over all states: $\pi_{p_1}=\pi^{uniform}$;
Train the ego driver using D3QN PER algorithm;
At the end of the training, ego driver learns to best respond
  to $\pi^{k-1}$,therefore the resulting policy is the level-k policy, $\pi^k$.
```

C. Combining Level-k Reasoning With D3QN PER

To generate vehicles with different levels of reasoning, we run the D3QN PER RL algorithm in our simulator, where the ego vehicle is the level-k learner. According to level-k theory, we assign our trained level-(k-1) policies (or predefined level-0 behavior) to the rest of the vehicles which constitute the environment.

In the proposed approach, the predetermined, non-strategic level-0 policy is the anchoring policy from which all the higher levels are derived using D3QN PER. To obtain the level-1 policy, a traffic scenario is created where all drivers are level-0 agents except the ego car that is to be trained to best respond to the level-0 policy. There are 5 predefined velocities (0.3 m/s, 0.6 m/s, 0.9 m/s,1.2 m/s, and 1.5 m/s) that opponents can choose randomly. At level k = 0 of reasoning, opponent vehicles travel at the selected constant speed without considering the motions of others. To avoid collisions between opponents affecting the training process of the ego vehicle, we set priority for speed selection of each opponent. For example, car 2 can select a speed first, and then car 4 will randomly pick a speed from our pre-defined "safe speed set" to avoid collisions with car 2 at collision area 2 (see Fig. 1). Finally, car 3 will randomly select a safe speed to avoid collision with car 4. Once the training is completed, the ego becomes a level-1 driver. The procedure for obtaining the level-k policy through the proposed combination of level-k reasoning and D3QN PER is explained in Table I, where n_d is the number of drivers.

Now, we have trained level-1 and level-2 policies in turn, but the problem is that these trained models are based on the assumption that all opponents are playing level-(k-1). If the true strategies chosen by opponents do not meet this assumption, the conflict between them will not be well resolved. The test results in Section IV can well reflect this issue. To solve the problem, all opponents will choose policy among trained models following uniform distribution, and the ego car explores the adaptive strategy in this mixed environment through D3QN PER. Since each vehicle cannot access the driving policies of others, all vehicles can observe only a partial state of the traffic via the Lidar sensor.

To alleviate the hidden state problem, an LSTM recurrent neural network is used in conjunction with the D3QN PER algorithm to resolve the hidden state by making the chosen action that depends not only on the current observation but also on a fixed number of the most recent observations which is a black

TABLE II
TRAINING OF ADAPTIVE AGENT

```
Algorithm 2 Obtaining the Adaptive Policy
1: Load the previously obtained level-1 and level-2 policy randomly,
   $\pi^1$ and $\pi^2$ (see Algorithm 1);
2: Set the agents in the environment, $p_i$, as level-1 agents or
   level-2 agents: $\pi_{p_i} = \pi^1$ or $\pi_{p_i} = \pi^2, i = 2, 3, \ldots, n_d$;
3: Initialize the ego driver's policy to a uniform action probability
   distribution over all states: $\pi_{p_1} = \pi^{uniform}$;
4: Train the ego driver using D3QN PER algorithm;
5: At the end of the training, ego driver learns to best respond to $\pi^1$
   and $\pi^2$. Thus, the resulting policy is the adaptive policy, $\pi^{adap}$.
```

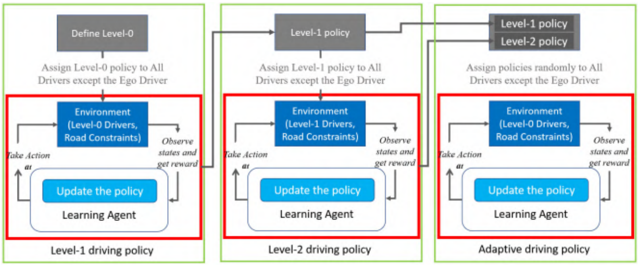

Define Level-0
Assign Level-0 policy to All Drivers except the Ego Driver
Environment (Level-0 Drivers, Road Constraints)
Take Action a_t
Observe states and get reward
Update the policy
Learning Agent
Level-1 driving policy
Level-1 policy
Assign Level-1 policy to All Drivers except the Ego Driver
Environment (Level-1 Drivers, Road Constraints)
Take Action a_t
Observe states and get reward
Update the policy
Learning Agent
Level-2 driving policy
Level-1 policy
Level-2 policy
Assign policies randomly to All Drivers except the Ego Driver
Environment (Level-0 Drivers, Road Constraints)
Take Action a_t
Observe states and get reward
Update the policy
Learning Agent
Adaptive driving policy

Fig. 3. Combination of level-k theory and RL.

box way to learn the pattern of various driving policy instead of using Bayesian-based method to estimate the belief of driver model of other vehicles. Training process of adaptive policy is described in Table II.

It is noted that the hierarchical learning process explained above decreases the computational cost since at each stage of learning, the agents other than the ego agent use previously trained policies and hence become parts of the environment. This helps to obtain traffic scenarios, containing a mixture of different levels, where all the agents are simultaneously making strategic decisions. This sharply contrasts conventional decision making approaches, in crowded traffic, where one driver is strategic decision maker and the rest are assigned predefined policies that satisfy certain kinematic constraints. A visual representation of the process of combining level-k reasoning and D3QN PER is given in Fig. 3.

D. Setting of Algorithm Parameters and Reward Function

Table III shows the parameter settings of the D3QN PER algorithm. To speed up the training process, each sequential training after obtaining the level-1 policy was done by loading the previous model to the ego vehicle and assigning the starting value of ϵ to 0.5 instead of 1.

In order to avoid insufficient exploration and to accelerate convergence, the parameter of the ϵ-greedy method decrease from 1 linearly according to the training steps, as shown in (9), and remain unchanged until it equals ending value.

$$\epsilon = \epsilon - \frac{1.0}{N_{training}}. \tag{9}$$

Parameter e in the PER algorithm is used to prevent the saved experience from not being replay after TD-error equals 0. The exponent α determines how much prioritization is used, with $\alpha = 0$ corresponding to the uniform case. Parameter β fully compensates for the non-uniform probabilities $P(i)$ if $\beta = 1$. It's increased linearly according to (10).

$$\beta = \beta + \frac{1.0}{N_{training}}. \tag{10}$$

TABLE III
PARAMETER SETTINGS

Parameters	Values
Discount factor (γ)	0.95
Learning rate	0.001
Starting (ending) value of ϵ greedy policy (ϵ)	1 (0.01)
Number of actions	5
Size of replay memory	6000
Maximum value of training steps	150
Number of steps to update the target network	500
Mini-batch size	32
Training steps ($N_{training}$)	15000
e	0.00001
Priority (α)	0.6
Adjusting the deviation (β)	0.4

TABLE IV
SETTING OF REWARD FUNCTION

Conditions	Values
Collision status	-5
Dangerous zone status and car approaching	-0.1*$\Delta(t)$
Changes in acceleration	-0.05*($\|a_t - a_{t-1}\|$)
Reach destination	2
Reach each checkpoint	0.1

The reward function shown in Table IV is used to evaluate the performance of the ego vehicle, which encourages ego vehicles to learn efficient human-driving behaviors. A reward function was designed to penalize collisions or being in dangerous states, and reward efficient behaviors, such as reaching the destination or progressing.

According to the setting of reward function, the ego car will receive a reward of 0.1 for reaching each checkpoint. If ego successfully reaches the destination, when going straight, it receives a reward of 2 points for passing the 20 checkpoints. When a collision occurs, 5 points are deducted. When ego vehicles and opponents cars are in a deadlock, 0.1 point will be deducted for each time-step, which will be a total of -1.5 points for the maximum number of steps each episode. When an ego vehicle reaches the destination, the score will be rewarded 2 points.

IV. SIMULATION AND EXPERIMENTAL RESULTS

A. Comparison of Different Algorithms

To compare learning efficiency, all algorithms were trained 8,000 episodes in an environment where all opponents were level-0 reasoning. All curves in Fig. 4 are smoothed with a moving average over 300 episodes. We can find that the prioritized reply and dueling structure are the two most crucial components of the D3QN PER algorithm, in that removing either component caused a large drop in learning performance. Nature DQN and

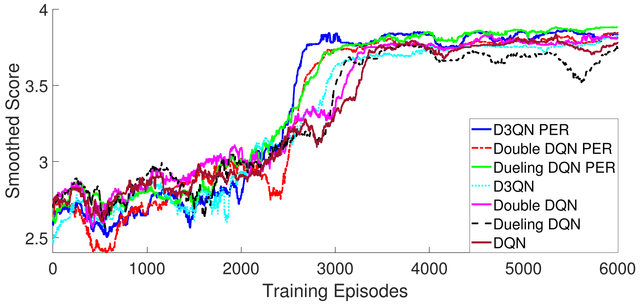


Fig. 4. Comparison of various DQN algorithms.

TABLE V
COMPARISON WITH EXISTING WORK

No.	Interaction type	GTDRL	AV controller [10]
1	L1 vs. L0	96.2%	**99%**
2	L2 vs. L1	**99.6%**	95%
3	L1 vs. L1	**94.9%**	84%
4	L2 vs. L2	**85.9%**	57%
5	L2 vs. L0	**83.1%**	41%
6	Adapt. vs. Mixed Env.	**96.2%**	94-95%

Dueling DQN perform worst during training, which could be caused by the overestimation mentioned in Section III.

B. Simulation Results

?enlrg -6pt?>According to the scheme proposed in Section III, we have successfully trained level-1, level-2, and adaptive driving strategies, respectively [1]. After obtaining the trained polices, we first tested the two scenarios of ego vehicles being of level-k and opponents' vehicles being level-(k-1) each for 1000 episodes, when k = 1, 2. They all have a success rate of over 96%, which meets expectations of level-k theory. Then, the trained level-1 policy and level-2 policy were tested on level-k versus level-k scenarios for 1000 episodes each, k = 1, 2. These scenarios have a much lower success rate, which is reasonable because all the cars have wrong assumptions about the driving policy of others. And the level-1 policies often result in deadlock due to conservative driving behaviors, and level-2 policies often result in collisions due to aggressive driving behaviors (see Table V).

We also compared the performance of our algorithm with the autonomous vehicle controller which is based on the driver interaction models and online model estimation proposed in [10], and their results are shown in the third column of Table V named AV controller. We named our algorithm GTDRL since it is based on GT and DRL. The comparison shows that our end-to-end scheme integrating perception, decision, and control has better performance in terms of the success rate than the results shown in [10], which focuses on a simpler two-car interaction without considering the noise existing observation information.

Finally, we tested the level-1, level-2, and adaptive policy 1000 times in the 81 scenarios following the distribution mentioned in Section II. As shown in the Table VI, the adaptive policy has the highest success rate in all three policies, which measures the ego vehicle's ability to pass through the intersection without collision and deadlock. The deadlock rate is highest for level-1 policy because it's conservative driving behaviour that tends to react by decelerating to a halt. Level-2 has the highest collision rate because it models an aggressive driving behaviour that tends to collide with other vehicles. The adaptive policy successfully combines the advantage of both level-1 and level-2 policies to reduce deadlock and collision when interacting in the mixed scenario.

TABLE VI
COMPARISON OF POLICIES IN MIXED ENVIRONMENT

Interaction type	Deadlock (%)	Collision (%)	Success (%)
L1 vs. Mixed	4.3%	1.7%	94.0%
L2 vs. Mixed	2.3%	9.3%	88.4%
Adapt. vs. Mixed	**3.5%**	**0.3%**	**96.2%**

C. Hardware Implementation

To show the performance of the trained model, owing to space constraints, we select four scenarios with four cars to show the interactions among level-k vehicles at unsignalized four-way intersection (see Fig. 1 left). We let four vehicles to be controlled by different level-k policies and show how each traffic scenario evolves depending on the different combinations of level-k policies. It can be observed from Fig. 5 that when level-1 (l1) and level-2 (l2) vehicles interact with each other, the conflicts between them can be resolved. This is expected since level-1 vehicles, representing cautious drivers, will yield the right of way and level-2 vehicles, representing aggressive drivers, will proceed ahead.

Columns (a)–(d) show five subsequent steps in a hardware testing where each vehicle can be controlled by level-1 or level-2 policies that are pretrained in our simulator. The bottom panels show the corresponding time histories of the four vehicles' motion state (see Fig. 6). All paths are divided into 200 waypoints. For each point the car reaches, the number of passed waypoints increases by one, e.g., the number of passed waypoints is 1 when the car is in its initial position and it's 200 when the car reaches its destination.

All vehicles are located outside the intersection at t = 0 s. Column (a) shows the interactions of car 1 controlled by the level-1 policy (conservative) with three cars controlled by the level-2 policy (aggressive). Because car 2, 3 and 4 all use the level-2 policy, they usually choose to pass the intersection as quickly as possible. By observing the motion state of each vehicle in Fig. 6, we can find that car 4 pass the intersection first and does not take any deceleration action. Although car 2 and car 3 also adopt level-2 policy, since car 4 enter the collision area first, car 2 and car 3 chose to adopt deceleration actions of different degrees at around t = 4 s based on their observation. Car 3 is in front of car 1 at around t = 6 s, therefore, car 1 takes deceleration action at around t = 5 s, and car 2 chooses to wait for car 1 to pass the collision area again. Finally, all four cars safely pass through the intersection in turn. Similarly, column (b) shows car 1 and car 4 controlled by level-1 policy interact with car 2

[1] Video for both training and testing is available at https://youtu.be/mPtoojXh2-s

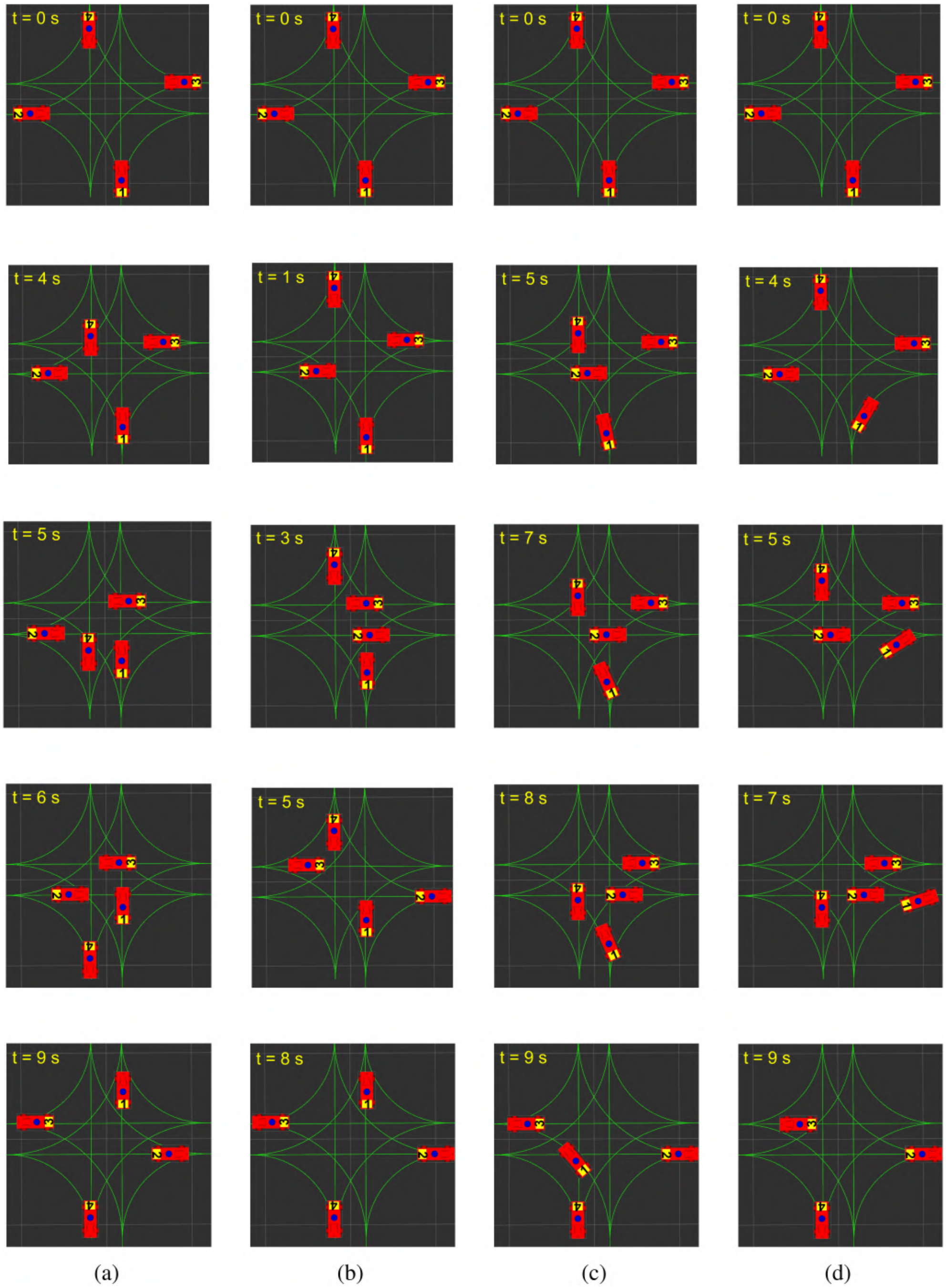


Fig. 5. A ten second sequence (see along the column) showing the interaction performed by the physical autonomous cars under (a) [l1-l2-l2-l2; car 1 going straight]. (b) [l1-l2-l2-l1; car 1 going straight]. (c) [l1-l2-l1-l1; car 1 turning left]. (d) [l2-l1-l1-l1; car 1 turning right] policy settings in order of car 1–4 with all opponents going straight. The numbers indicate the car IDs

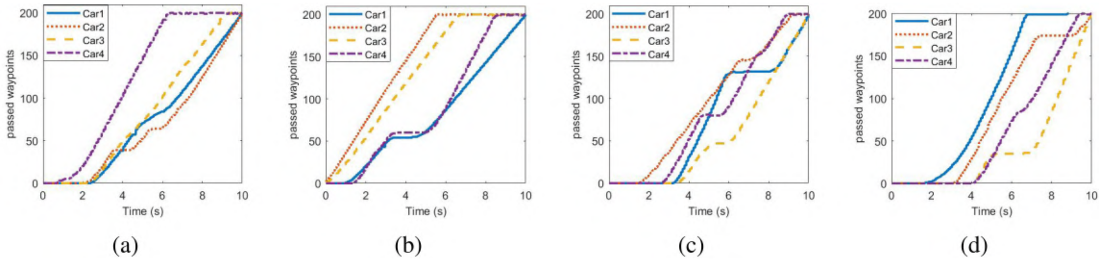


Fig. 6. Time histories of the four vehicles' motion states: (a) [l1-l2-l2-l2]. (b) [l1-l2-l2-l1]. (c) [l1-l2-l1-l1]. (d) [l2-l1-l1-l1].

and car 3 controlled by level-2 policy. Car 2 and car 3 enter the collision area first, and car 1 and car 4 take deceleration action to wait them to pass at around t = 3 s. Column (c) shows car 2 controlled by level-2 interacts with others controlled by level-1 with car 1 turning left. Column (d) shows car 1 turning right controlled by level-2 interacts with others controlled by level-1. The last two situations are similar: the car adopting level-2 strategy will pass through the intersection first, and the other vehicles with level-1 policy will pass through the intersection subsequently.

The examples above show that trained models obtained from simulator can deal with complex interaction scenarios without knowing the policies of others, which verifies the feasibility of our method for modeling different driving behaviors proposed in this letter.

V. Conclusion

An adaptive game-theoretic decision making strategy with DRL has been proposed for the ADVs sharing the road with other drivers in a multi-agent traffic scenario. The interactions between vehicles are modeled using a level-k game-theoretic framework. The ego estimates the driver model of opponents at each time step based on real sensor data and is shown to use it to adapt its behavior in both simulation and hardware implementation.

Both simulation results and hardware tests were reported and showed that the vehicle interaction model exhibited reasonable behavior expected in traffic. The performance of the model was then evaluated based on several ways, including the success rate, collision, deadlock, and snapshot of hardware testing. It was shown that the adaptive model had reasonably high rates of success in resolving traffic conflicts matching the expected behavior of each reasoning levels.

The framework proposed in this letter for modeling multi-vehicle interactions can be used as simulation tool for calibration, validation and verification of autonomous driving systems. In addition, it may also be used in high-level decision-making algorithms of ADVs, and to support intersection automation/autonomous intersection management. Moreover, vehicle interactions in some other traffic scenarios, such as highway merging and driving in parking lots, can be modeled based on the proposed framework with modified road layouts and geometries.

References

[1] A. Efrati, "Waymo riders describe experiences on the road," Accessed: Sep. 22, 2020. [Online]. Available: https://www. theinformation.com/articles/waymo-riders-describe-experiences-on-the-road

[2] N. Kalra, "With driverless cars, how safe is safe enough?," RAND Center Decis. Mak. Under Uncertain., Santa Monica, CA, USA, Tech. Rep., 2016, Accessed: Mar, 30, 2019. [Online]. Available: http://www.rand.org/blog/2016/02/with-driverless-cars-how-safe-is-safe-enough.html.

[3] U. D. Gupta, E. Talvitie, and M. Bowling, "Policy tree: Adaptive representation for policy gradient," in *Proc. AAAI Conf. Artif. Intell.*, vol. 29, no. 1, 2015, pp. 2547–2553.

[4] L. Claussmann, A. Carvalho, and G. Schildbach, "A path planner for autonomous driving on highways using a human mimicry approach with binary decision diagrams," in *Proc. Eur. Control Conf.*, 2015, pp. 2976–2982.

[5] J. F. Kooij, F. Flohr, E. A. Pool, and D. M. Gavrila, "Context-based path prediction for targets with switching dynamics," *Int. J. Comput. Vis.*, vol. 127, no. 3, pp. 239–262, 2019.

[6] T. N. Hoang and K. H. Low, "Interactive pomdp lite: Towards practical planning to predict and exploit intentions for interacting with self-interested agents," in *Proc. 23rd Int. Joint Conf. Artif. Intell.*, 2013, pp. 2298–2305.

[7] M. Babu *et al.*, "Model predictive control for autonomous driving considering actuator dynamics," in *Proc. Amer. Control Conf.*, 2019, pp. 1983–1989.

[8] C. Vallon, Z. Ercan, A. Carvalho, and F. Borrelli, "A machine learning approach for personalized autonomous lane change initiation and control," in *Proc. IEEE Intell. Veh. Symp.*, 2017, pp. 1590–1595.

[9] M. Garzón and A. Spalanzani, "Game theoretic decision making based on real sensor data for autonomous vehicles' maneuvers in high traffic," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 5378–5384.

[10] N. Li, I. Kolmanovsky, A. Girard, and Y. Yildiz, "Game theoretic modeling of vehicle interactions at unsignalized intersections and application to autonomous vehicle control," in *Proc. Annu. Amer. Control Conf.*, 2018, pp. 3215–3220.

[11] N. Li, Y. Yao, I. Kolmanovsky, E. Atkins, and A. R. Girard, "Game-theoretic modeling of multi-vehicle interactions at uncontrolled intersections," *IEEE Trans. Intell. Transp. Syst.*, to be published, doi: 10.1109/TITS.2020.3026160.

[12] R. Tian, S. Li, N. Li, I. Kolmanovsky, A. Girard, and Y. Yildiz, "Adaptive game-theoretic decision making for autonomous vehicle control at roundabouts," in *Proc. IEEE Conf. Decis. Control*, 2018, pp. 321–326.

[13] G. S. Sankar and K. Han, "Adaptive robust game-theoretic decision making strategy for autonomous vehicles in highway," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 14484–14493, Dec. 2020.

[14] A. Khan *et al.*, "Learning safe unlabeled multi-robot planning with motion constraints," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 7558–7565.

[15] G. Qu, A. Wierman, and N. Li, "Scalable reinforcement learning of localized policies for multi-agent networked systems," in *Proc. Learn. Dyn. Control*, 2020, pp. 256–266.

[16] M. Kuderer, S. Gulati, and W. Burgard, "Learning driving styles for autonomous vehicles from demonstration," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 2641–2646.

[17] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics*. Cham: Springer, 2018, pp. 621–635.

[18] Y. Chebotar *et al.*, "Closing the sim-to-real loop: Adapting simulation randomization with real world experience," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 8973–8979.

[19] Z. Dai, Y. Chen, B. K. H. Low, P. Jaillet, and T.-H. Ho, "R2-B2: Recursive reasoning-based bayesian optimization for no-regret learning in games," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 2291–2301.

[20] B. M. Albaba and Y. Yildiz, "Modeling cyber-physical human systems via an interplay between reinforcement learning and game theory," *Annu. Rev. Control*, vol. 48, pp. 1–21, 2019.

[21] N. Li, D. W. Oyler, M. Zhang, Y. Yildiz, I. Kolmanovsky, and A. R. Girard, "Game theoretic modeling of driver and vehicle interactions for verification and validation of autonomous vehicle control systems," *IEEE Trans. Control Syst. Technol.*, vol. 26, no. 5, pp. 1782–1797, Sep. 2018.

[22] G. Gigerenzer and R. Selten, *Bounded Rationality: The Adaptive Toolbox*. Cambridge, MA: MIT Press, pp. 37–49, 2001.

[23] Y. Wen, Y. Yang, and J. Wang, "Modelling bounded rationality in multi-agent interactions by generalized recursive reasoning," in *Proc. 29th Int. Joint Conf. Artif. Intell., IJCAI-20, C. Bessiere, Ed. Int. Joint Conf. Artif. Intell. Org.*, 2020, pp. 414–421.

[24] M. Hessel *et al.*, "Rainbow: Combining improvements in deep reinforcement learning," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 3215–3222.

[25] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[26] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," in *Int. Conf. Learn. Representations(ICLR)*, 2015, *arXiv:1511.05952*.

From Naturalistic Traffic Data to Learning-based Driving Policy: A Sim-to-Real Study

Mingfeng Yuan, *Student Member, IEEE,* Jinjun Shan, *Senior Member, IEEE,* and Kevin Mi

***Abstract*—Reinforcement learning (RL) is a promising way to achieve human-like autonomous driving (HAD) in complex and dynamic traffic, but faces challenges such as low sample efficiency, partial observability, and sim2real transfer. In light of this, a comprehensive solution for RL-driven HAD is established. First, an efficient training scheme called Deep Recurrent Q-learning from demonstration algorithm (DRQfD) is proposed for lane-changing decision-making to address the low sample efficiency in RL and the poor generalization capability in Imitation Learning (IL). The inherent LSTM structure potentially learns to predict future states of surrounding vehicles, helping to address the partially observable problem in autonomous driving (AD). Second, to reduce the sim2real gap, a twin high-fidelity simulator is built based on ROS-Gazebo for simulating LiDAR sensing, model training, and evaluations. Domain randomization is used to improve the robustness and generalization ability, making it easier for the model to be transferred to real-world scenarios. In addition, for the multi-objective optimization and imbalanced data issues in this scenario, a hierarchical decision-making framework is proposed to decompose the complex decision-making problem into several subtasks, making the driving policies easier to converge. To avoid the excessive dependence of the decision-making module on the output of perception module in modular systems, we train each modularized skill in an end-to-end manner. Moreover, we compare our method with a vanilla RL method to show improvement in learning efficiency. Comparisons between RL-based model and IL baseline in terms of safety, travel efficiency, and human-likeness are also given. To further validate the generalization ability of our model, we test the model on real traffic dataset. Finally, we implement the RL model on physical cars to demonstrate the performance of sim2real transfer.**

***Index Terms*—Human-like driving behavior modeling, reinforcement learning, Imitation learning, sim2real.**

I. INTRODUCTION

A Highly intelligent and robust autonomous driving system (ADS) is of great importance in improving driving safety and travel efficiency. In recent years, learning-based autonomous driving technology has become a research hotspot in both academia and industry. Compared to imitation learning (IL), optimizing long-term goals and policy exploration give RL a great potential advantage in achieving human-like driving behavior, that can cover extreme driving conditions. However, applying RL to the decision-making of AD still faces several challenges that need to be addressed, including i) low sample efficiency, requiring a massive amount of interactions; ii) incomplete observation information, leading to unstable training; iii) gap in sim2real, leading to difficulties in transferring models from simulation to real-world applications.

Instead of learning from scratch, there are three categories of methods that can help RL methods to speed up training process: (1) rule-based guidance [1], aiming to reduce unreasonable exploration behavior; (2) human-based guidance [2], incorporating human intervention during the training process; (3) IL based guidance [3], [4], utilizing human demonstration to pre-train and/or train RL network. The method proposed in this paper belongs to the third category. We use limited expert demonstration to improve the learning efficiency of Dueling Double Deep Recurrent Q-learning with Prioritized Experience Replay algorithm (D3RQN PER) to model the decision-making in complex lane-changing scenarios as partially observable Markov decision process (POMDP) problem [5]. The small amount of expert demonstration in this study serves two main purposes: i) initializing the parameters of the RL network before interacting with the environment, and ii) increasing the probability of the vehicle taking the correct actions during early exploration through the guidance of IL.

Lane-changing decision-making is a typical multi-objective problem, including adaptive cruise control (ACC), switching lanes, and merging. An effective way to solve such problems is to design each modularized skill through a hierarchical decision-making framework. The high-level strategic module is mainly responsible for macro-level decision-making with optimization goals including travel efficiency and safety. It will trigger a specific task that needs to be executed by technical level submodules in real-time based on the environment states. When the ACC module is activated, it will control the longitudinal motion of the autonomous vehicle (AV) with optimization goals including comfort, driving speed, and safety. When the lane-changing task is triggered, it will plan a collision-free and smooth trajectory to switch lanes. Currently, the industry commonly adopts a hierarchical decision-making framework for modular ADS consisting of perception, decision-making, and control. Although rule-based methods can create policies for each submodule effectively, they face challenges when scaling to complex scenarios, exhibiting overly conservative driving behavior. To address this problem, some works have attempted to use the RL-based method to learn each modularized skill in a hierarchical decision-making framework.

The main difficulties of decision-making for AVs in complex traffic scenes are twofold: 1) the behavioral patterns

This work was supported in part by NSERC Alliance Program under Grant ALLRP 555847-20, and in part by Mitacs Accelerate Program under Grant IT26108.

Mingfeng Yuan and Jinjun Shan are with the Department of Earth and Space Science, Lassonde school of Engineering, York University, ON M3J 1P3, Canada. {mfyuan, jjshan}@yorku.ca

Kevin Mi is with the Division of Engineering Science, University of Toronto, ON M5S 1A1, Canada. kevin.mi@mail.utoronto.ca

and intents of other vehicles are complex and cannot be directly observed, therefore, the capability of interaction with surrounding vehicles is needed and 2) the perception of the AVs is uncertain due to noise and occlusions. Given difficulties mentioned above, the decision-making process of AVs is a POMDP, which can cause instability during training and prevent the policy from converging. In this work, we consider the multi-vehicle interaction scenarios, where the driving behavior of AVs is influenced by surrounding vehicles, and vice versa. Second, we employ an LSTM network to stabilize the training process of DRL and address the POMDP problem. The LSTM network implicitly learns and predicts the driving behavior of surrounding vehicles by inputting past multiple frames of point cloud data as state input for DRL, which help reduce the ambiguity of LiDAR-based end2end AD given unknown behavioral patterns and intentions of surrounding vehicles.

RL training requires policy exploration, which is not feasible in safety-critical applications such as AD. A promising solution is to learn policies through high-fidelity simulators and transfer the model to real-world application scenarios. The most challenging problem lies in the sim2real gap, which comes from three aspects: (i) the difference in perception level; real traffic scenarios are more complex than the simulated environment. (ii) The raw data generated by the sensor hardware is noisy. (iii) The difference between the interaction features in the simulated environment and the real scene. We reduce the gap of sim2real in three main ways. First, to support the end2end training, we developed a high-fidelity simulator based on ROS-gazebo that can simulate sensor data and vehicle dynamics. Second, thanks to the URDF parameter format setting supported by ROS, we can keep the dynamics and sensor data of the virtual vehicle as close as possible to the physical vehicle. Third, to learn human-like driving behavior, we adopt reality-guided domain randomization, and we match the simulator with the real world by referring to the statistical distribution of real traffic data (e.g., action distribution, distance distribution). Finally, in terms of sensor selection, compared with vision-based solutions, LiDAR is suitable for different lighting conditions, which facilitates the migration of models to hardware.

The main contributions of this paper are listed as follows.

- A learning-efficient DRQfD framework is proposed to model lane-changing decisions as a POMDP, in which a small amount of expert data is employed to pre-train the RL network parameters and train the IL policy to guide early policy exploration for the vehicle.
- End-to-End modularized skill-based decision-making framework with two layers of hierarchy (strategic and tactical planner) is proposed to address the multi-objective problem in complex lane-changing scenarios.
- A comprehensive test is carried out in both randomly generated scenarios and real traffic data. Qualitative and quantitative comparisons between our method and IL baseline are also given to show the driving performance in terms of safety, travel efficiency, and human likeness.
- Sim2real transfer is successfully achieved by building a high-fidelity simulator and Domain Randomization. Experimental verification is carried out. The RL model can be directly applied on the hardware platform, exhibiting driving behavior consistent with it in the simulator.

The rest of the paper is organized as follows. Section II introduces some related works. Section III defines the research problem and describes the proposed DQRfD algorithm. Section IV introduces the trajectory generation and low-level controllers in both longitudinal direction and lateral directions. Section V describes the human driving data set and the implementation details. Section VI provides the experimental results. Section VII concludes this paper.

II. Related Works

A. *Human-Knowledge-Based Learning Method*

Some researchers have proposed using prior knowledge from humans to guide the interaction of RL in a training environment, as opposed to training policies from scratch. To prevent unsafe exploration, [1], [6] suggested the addition of a rule-based safety check module to the RL-based control system to achieve fast training. In [7], a human-guidance-based learning method allowing human experts to intervene in the interaction process in real-time was proposed to speed up the training of RL agents. However, this approach comes at the cost of increasing human workload. To solve this problem, Hug-DRL [2] was proposed with the aim of reducing the human workload and enhancing the performance of RL for training and testing on AD by utilizing intermittent guidance. Another promising method is to use limited demonstration data to pre-train an expert policy that can achieve a reasonable level of performance. IL is an effective method to mimic expert behavior. However, it performs poorly in some scenarios where the training data is not covered, especially in some extreme conditions (e.g., collision, near-collision). Therefore, a reasonable idea is to combine the strengths of IL and RL to efficiently train a robust model. In [3], [4], [8], the authors employed an imitative expert policy to aid in the learning process of the actor-critic-based RL agent for different traffic scenarios. Another approach is to design a loss function that enables the IL algorithm to learn the value function of actions in certain states, thereby achieving pre-training of the RL network [9], rather than learning the expert behavior policy based on a classification task.

B. *Hierarchical Decision-Making*

There have been several studies addressing the decision making problem for AD through a hierarchical decision framework. In [10], the authors used the DQN algorithm to learn the longitudinal control policy in a highway case with some assumptions about the high-level policy. In [11], the authors proposed an effective state-action abstraction and a hierarchical training framework for RL to achieve multi-lane cruising, and demonstrated that the model trained in a non-dynamic simulation environment has good transferability in a more realistic simulator. However, a challenge faced by above methods is that the decision module overly relies on the performance of the perception module. The failure of the perception module may result in fatal traffic accidents [12]. The hierarchical decision-making framework used in this work is different from previous research. High-level policy and ACC are trained respectively in an end-to-end manner with LiDAR-based observation. These modules directly learn features that affect decision-making from the raw sensor data, thereby preventing an excessive dependence on the performance of the perception module.

C. End2End Scheme and Sim2Real Transfer

ALVINN is the first work of IL for an end-to-end AV [13], [14]. Then, more complex and successful end-to-end driving systems were developed in [15]–[17], which utilized multiple cameras, enabling the system to extract distance information and learn to control the lateral motion of vehicle in an end-to-end fashion. Recently, various deep reinforcement learning (DRL) methods have been used to train LiDAR information-based end2end AD and navigation policies. In [18], the authors utilize unsupervised contrastive learning to differentiate between similar and dissimilar pairs of high-dimensional LiDAR data to learn representations of environments. While, in [19], the author used a CNN network to learn environmental features by converting LiDAR point clouds into gray images, achieving motion control in a static environment. To train the local navigation policy, in [20], a single frame of laser scan is combined with polar coordinates of waypoints to achieve collision-free exploration tasks.

Sim2real transfer is a class of methods to bridge the reality gap, connecting and integrating digital entities in simulations with their physical counterparts in the real world. Currently, the work on sim2real transfer can be divided into two categories, namely Domain Adaptation (DA) and Domain Randomization (DR). GAN is a popular technique in the field of DA for transforming synthetic images to resemble those captured from the Target Domain [21], [22]. While DR is a simple yet effective concept that operates by randomizing the dynamic properties of the Source Domain while undergoing training [22]–[24]. Recently, some researchers also start focusing on MARL sim2real transfer. They use domain randomization to develop their multiple autonomous vehicles and multiple unmanned aerial vehicles for different application backgrounds [25], [26].

III. METHODOLOGY

A. Problem Formulation

In lane-changing scenarios, vehicles must be able to adjust their actions to fit into the dynamic traffic environment safely and efficiently. Given the unknown driving behavior and intentions of the surrounding vehicles, the decision-making problem of AV in lane-changing scenarios can be modeled as a POMDP [27]. To train driving policies, this paper proposes a hierarchical decision-making architecture, as shown in Fig. 1, which is mainly divided into two levels. The first is the high-level decision-making module achieved by Global DQN Network, which is about transitioning between discrete actions including car-following, changing the lane to the left, and changing the lane to the right subject to the following conditions:

- Safety - releasing restrictions on lane-changing action under the condition that there will be no collision with surrounding vehicles.
- Travel efficiency - navigating the ego vehicle to the target lane where the car can drive faster.

In other words, this module should be able to give AV a strategic task to perform at the current stage after observing the surrounding environment.

Next, once the car following command is generated from the high-level decision-making block, Local DQN Network will serve as a tactical module to maintain a safe distance and speed with the vehicle in front of AV while considering travel speed, comfort, and reducing energy consumption. Trajectory generation is a non-learning-based tactical module. It will generate a smooth and comfortable trajectory using a fifth-order polynomial according to its current lane and speed, as well as the target lane and target speed when getting a switching lane command from the high-level module. However, the outputs from the Local DQN Network and trajectory generation block are still at the command level, such as the desired acceleration, velocity, position, and steering angle. Ideal driving behavior requires precise control performance, which can be guaranteed by implementing a low-level control module consisting of throttle control and steering control. In the following sections, we will introduce how to implement the above modules in detail.

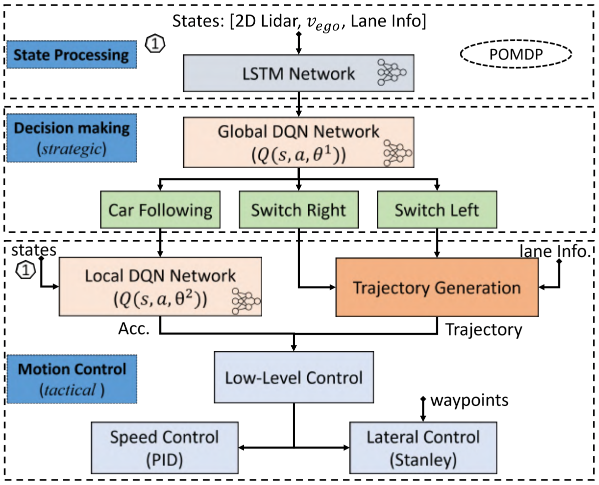


Fig. 1: Hierarchical decision making framework.

B. Observation Information

When driving a car, we make decisions primarily based on the information captured by our eyes. However, the driving policies of the surrounding vehicles are unknown to us, and we have to observe them for a period of time to approximately estimate their future motions. Therefore, in our study, we prohibited the use of accurate speed and position information from other vehicles. All observations were obtained from onboard 2D LiDAR plus the ego vehicle speed. It is worth noting that the method proposed in this paper can also be extended to 3D LiDAR, but requires more computational power.

In this work, the LSTM network is used to implicitly learn and predict the driving behavior of surrounding vehicles by feeding the past multiple frames of point clouds concatenated with ego speed and lane information to the DRL. It can help reduce the ambiguity of LiDAR-based end-to-end AD to stable training processing of DRL. Since we are using a 2D LiDAR which is mounted on the top of the car, to enhance the point cloud data from the LiDAR detection of surrounding cars, we mounted an isosceles triangle frame behind LiDAR. The scanning region of LiDAR is [135°, -135°] and the scanning range is set to [0.1, 2.5] m. The scanning frequency is 12 Hz with 1160 points. The left image in Fig. 2 shows a bird's-eye view of the simulated environment, displaying the LiDAR scan of surrounding vehicles from the perspective of the ego vehicle (denoted as ”1”), with other numbers representing surrounding vehicles. The right image shows the corresponding hardware testing environment, where the ego vehicle is in the middle lane with a blue triangular frame. The point clouds scanned by the ego LiDAR hardware is provided in the right figure. We distinguish surrounding vehicles with dashed circles in different colors.

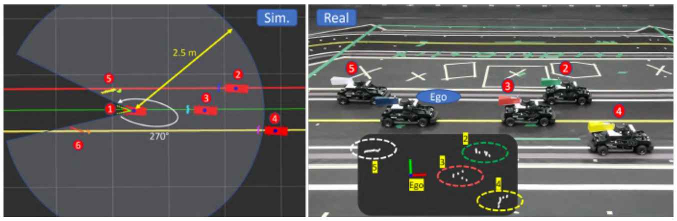


Fig. 2: Simulation and hardware test environments.

C. Reward Function and Action Space

To avoid manually designing a large number of predefined driving behaviors like rule-based methods, we adopt a coarse-grained reward function to encourage RL to learn near-optimal driving behaviors through exploration. A punishment mechanism is added to prevent the car from learning unsafe driving behaviors based on human common knowledge.

1) Car Following Policy: According to the US101 data [28], the intervehicle distance is mainly maintained between 11 m and 25 m above about 50% of the time. The distribution of intervehicle distance can be found in section IV. Because the size ratio of the car used in our study and a sedan car is 1:10. Therefore, the values for both the distance and the acceleration will be divided by 10 to define our reward function and action space.

The stage reward function for car following is defined as

$$R = w_1\phi_1 + w_2\phi_2 + w_3\phi_3 + w_4\phi_4 + w_5\phi_5 \qquad (1)$$

where ϕ_i represent indicator variables and w_i are the weight variables where $i \in [1, 5]$. The values of weights corresponding to each factor are shown in Table I. If a collision occurs, indicator variable ϕ_1 equals to 1, otherwise 0. According to the distance (D) between the ego vehicle and the car in front of ego car (to simplify the description, we denote this vehicle as the leader vehicle), three zones are defined: unsafe zone ($D \in [0.5, 1.1]$ m), interaction zone ($D \in [1.1, 2.5]$ m), and safe zone ($D \in [2.5, +\infty]$ m). When ϕ_2 =1, it indicates that the leader vehicle is in its unsafe zone, and is 0 otherwise. When ϕ_3=1, which indicates that the leader car is in its interaction area. Ego car is encouraged to maintain a relative velocity of 0 m/s to the leader vehicle. When ϕ_4 = 1, the leader car is outside the interaction area, and we encourage the ego vehicle to keep the speed above the average speed. The fifth term, ϕ_5 is introduced to consider energy consumption (or unnecessary driving actions) and equals to 2 if the action is a hard acceleration or a hard deceleration, 1 if the action is acceleration or deceleration, and 0 if the action is maintain. The action space of the car following policy consists of five actions: maintaining the current speed (0 m/s^2); hard acceleration (1.2 m/s^2); hard deceleration (-1.2 m/s^2); acceleration (0.6 m/s^2); deceleration (-0.6 m/s^2). This discrete representation of the action space is based on the distribution of vehicle accelerations obtained by processing the real traffic data [27], [29], which is recognized as a reasonable approximation to the set of human drivers' actions in highway traffic. It should be pointed out that we set speed saturation during training and evaluation, thus, the AV is only allowed to drive within the maximum speed. When the AV reaches the maximum speed, it cannot further obtain additional rewards through increasing speed. Therefore, the AV will finally learn to maintain its highest speed (by choosing 0 m/s^2) when there is no leader vehicle in the current lane by the trade-off between travel efficiency and energy consumption factors.

TABLE I: Setting of Reward Function

Conditions	Values
Collision violation (w_1)	-1000
Unsafe zone violation (w_2)	$-(\frac{1.1}{D})^3$
Relative speed (w_3)	$-2 \cdot \lvert v_{ego} - v_{front} \rvert$
Travel efficiency (w_4)	$v_{ego} - (v_{max} + v_{min})/2$
Energy consumption (w_5)	-0.6

Note: $v_{max} = 1.5$ m/s, and $v_{min} = 0.0$ m/s.

2) Lane-Changing Policy: For the high-level module, we expect the ego car to learn a more efficient driving behavior similar to that of an experienced driver. That is, the ego car will take any opportunity to navigate to the target lane where it can gain a higher speed under the premise that it will be collision-free. Therefore, we list some unsafe driving behaviors and encouraged the car to change lanes without triggering penalties. To obtain a positive reward during the training process, the ego car needs to avoid the following unsafe lane-changing behaviors:

- The penalty is -1 for lane-changing, if there's no leader vehicle in the current lane within the LiDAR detection range.
- The penalty is -1.5, if the car in front in the current lane is farther than the car ahead in the target lane.
- The penalty is -5, if a collision occurs during lane change.

The action space of the high-level module consists of {switching left, switching right, car following}. To simplify the training scenario, we assume that the speed of the ego vehicle before and after the lane change remains the same.

The logic behind this assumption is that the ego vehicle first generates a trajectory to a target lane where it can gain higher speed, and then accomplishes the acceleration goal in the target lane by using a trained car-following policy. Trajectories are generated based on the fifth-order polynomial under the following boundary conditions: (1) the target speed of the trajectory is the 'same' as the current speed, and (2) the target position of the trajectory is the location in the target lane right 'behind' its leading vehicle in the original lane to achieve a safe lane change.

It shold be notied that to obtain more flexible trajectories, we can modify the boundary conditions of the fifth-order polynomial by expanding the action space of the high-level module such as {switching left faster, switching right faster, switching left, switching right, Car following}, where the constraints of polynomial trajectory for switching lanes faster are (1) the target speed is 'higher' than the current speed, and (2) the target position of the trajectory is the location in the target lane 'parallel' to the leading vehicle in the original lane.

D. Deep Recurrent Q-learning from Demonstrations

Given the high-fidelity simulator, we can manually collect human demonstrations and automatically score the expert's action at a certain state according to the pre-designed reward function. To enhance the training efficiency of RL, we propose a DRQfD framework, which involves using a small set of demonstrations to pre-train RL network, followed by imitation learning (IL) to guide its early exploration. Given the LiDAR-based perception scheme and the POMDP, the Dueling Double Deep Recurrent Q-learning with Prioritized Experience Replay algorithm (D3RQN PER) [30], [31] becomes an ideal choice for our RL algorithm. The parameter setting of algorithm is shown in Table II.

1) Pretrained RL Network: The pre-training phase aims to teach the agent to imitate the demonstrator while also satisfying the Bellman equation for its value function, which can then be updated through TD error during interaction with the environment. To get pretrained RL Network, the learning vehicle updates the network by sampling mini-batches from demonstration data and applying four losses. These include 1-step and n-step double Q-learning losses to ensure the network satisfies the Bellman equation, large margin classification loss to enforce the value of the demonstrator's action, and L2 regularization loss to prevent over-fitting on the small demonstration dataset. The details of the loss function implementation can be found in [9].

2) IL-Based Guidance Policy: After completing the pre-training phase, a hybrid policy combining ϵ-greedy exploration and Guidance policy, is used to increase the probability of the vehicle taking the correct actions during early exploration through the IL-Based Guidance policy. During the ϵ-greedy exploration process, learning vehicle has a certain probability of taking actions generated by IL, which needs to be adjusted to balance exploration and exploitation to mitigate the bias introduced by using an IL policy on the replay buffer. It should be noted that both RL and IL use LSTM network to process a fixed number of past sequential observations as input.

3) Experience Replay: During the RL training, the learning vehicle interacts with surrounding vehicles in the simulator, generating its own data and adding it to the replay buffer $\mathcal{D}^{\text{replay}}$. The expert demonstration data will be permanently stored in the experience replay buffer $\mathcal{D}^{\text{expert}}$ and assigned a constant value in PER to ensure that the data is sampled during the training process.

4) IL Training: In order to avoid increasing human workload, both pretrained RL network and IL Guidance policy mentioned earlier are trained on a small amount of demonstration data collected by a human player using our simulator. To ensure a fair comparison of the performance between RL policy and IL Guidance policy in Section V, we increase the amount of demonstration data to the same number as RL training episodes. For both the IL Guidance policy and the IL Baseline policy, we performed supervised classification of the demonstrator's actions using a cross-entropy loss, with the same network architecture used by RL. Additionally, we still used L2 regularization loss to prevent overfitting of the model.

TABLE II: Parameter Settings

Parameters	**Values**
Discount factor (γ)	0.95
Learning rate	0.001
Starting (ending) value of ϵ greedy policy	1 (0.01)
Number of car-following actions	5
Number of high-level module actions	3
Size of replay memory ($\mathcal{D}^{\text{replay}}$)	10000
Size of expert memory ($\mathcal{D}^{\text{expert}}$)	2000
Number of steps to update the target network	100
Mini-batch size	32
e	0.00001
Priority (α)	0.6
Adjusting the deviation (β)	0.4
N-step returns (steps)	10

E. Simulation to Real World

For RL training, a high-fidelity simulator is developed based on ROS-Gazebo. To reduce the sim2real gap, Domain Randomization is adopted to improve the robustness and generalization ability of RL policies by randomizing the dynamic properties of the Source Domain. In this work, key factors for closing the sim2real gap include perception, environment complexity, vehicle dynamics, and interaction features. Regarding the perception, we adopt a LiDAR-based end-to-end scheme, since the LiDAR is insensitive to different lighting conditions. Thanks to the URDF setting supported by ROS, we can keep the dynamics and sensor data of the virtual car as close as possible to the physical car. We set our simulator parameters based on the data obtained from system identification of physical cars, such as the physical properties (e.g. mass, inertia, geometry) and LiDAR parameters (e.g. sampling frequency, noise, detection range). Real-world scenarios are more complex than the simulated environment, which could also affect the performance of RL models. As we conduct hardware tests indoors, in order to avoid the influence of irrelevant indoor obstacles on the model, we limit the maximum detection distance of LiDAR to 2.7 m and apply a mask function to set the point cloud data beyond the detection range to 0. In addition, during the training and hardware evaluation phases, point cloud data and ego speed are normalized based on their maximum value settings before feeding into the RL algorithm. Finally, to make the trained model generalize well in the test scenarios, we encourage the vehicle to experience as many interaction scenarios as possible during training by randomizing dynamic properties including the position and velocity of surrounding vehicles. Training scenarios and reward functions can be effectively designed by referring to the distribution of human driving characteristics (e.g., inter-vehicle distance and driving action) in real traffic data, as described in Section II-C and IV-A.

IV. Low Level Control

In the previous section, we introduced the idea of car following and lane-changing tasks using the "End-to-End" DRL method. However, once the car gets the acceleration command and steering command from the high-level module, the control performance needs to be guaranteed by implementing controllers. Therefore, at the Low-Level module, the PID controller was used to realize longitudinal control, and the Stanley controller was used to achieve lateral control.

A. Trajectory Generation Module

In real driving scenarios, we prefer a smooth trajectory. Therefore, we choose to use a fifth-order polynomial [32] to generate a trajectory for lane-changing, which allows us to specify six boundary conditions $(x(t), \dot{x}(t), \ddot{x}(t), y(t), \dot{y}(t), \ddot{y}(t))$ (position, velocity, acceleration in both longitudinal and lateral direction) at both $t_i = 0$ and $t_f = T$, where T is the terminal time for finishing lane-changing. For simplicity, we assume that the initial and final accelerations are 0, and the final velocity equals to the initial velocity. The reference trajectory can be expressed as Eq. (2). Each equation has 6 coefficients.

$$\begin{aligned}\mathrm{x(t)} &= a_5t^5 + a_4t^4 + a_3t^3 + a_2t^2 + a_1t + a_0 \\ \mathrm{y(t)} &= b_5t^5 + b_4t^4 + b_3t^3 + b_2t^2 + b_1t + b_0\end{aligned} \tag{2}$$

Then the time-dependent parameter matrix can be defined as Eq. (3):

$$M_{6\times6} = \begin{bmatrix} t_i^5 & t_i^4 & t_i^3 & t_i^2 & t_i & 1 \\ 5t_i^4 & 4t_i^3 & 3t_i^2 & 2t_i & 1 & 0 \\ 20t_i^3 & 12t_i^2 & 6t_i^1 & 2 & 0 & 0 \\ t_f^5 & t_f^4 & t_f^3 & t_f^2 & t_f^1 & 1 \\ 5t_f^4 & 4t_f^3 & 3t_f^2 & 2t_f^1 & 1 & 0 \\ 20t_f^3 & 12t_f^2 & 6t_f^1 & 2 & 0 & 0 \end{bmatrix} \tag{3}$$

B. Lateral Control

The Stanley controller achieves lateral control primarily by eliminating heading error $\psi(\mathrm{t})$ and cross-track error $e(t)$. $\psi(\mathrm{t})$ is defined by the angle between the trajectory heading and the car heading. $e(t)$ is the shortest distance between the center reference point of the front wheels (x_c, y_c) and the path (x_t, y_t) at current time t. $v(t)$ represents the linear speed of the front wheels, and the steering angle is denoted as $\delta(t)$. Considering the angle constraint of the vehicle, $\delta(t) \in [\delta\min, \delta\max]$, the controller can be expressed as Eq. (4) [33].

$$\delta(t) = \psi(t) + \tan^{-1}\left(\frac{ke(t)}{k_s + v(t)}\right) \tag{4}$$

where a softening constant, k_s, is added to ensure the denominator is non-zero. The parameter k is a constant.

C. Longitudinal Control

A PID controller is used to compensate the error in the speed. It looks at the current vehicle speed and adjusts the throttle to match the desired speed from the high-level module. The controller can be expressed as Eq. (5):

$$u = K_P\left(v_d - v\right) + K_I\int_0^t\left(v_d - v\right)dt + K_D\frac{d\left(v_d - v\right)}{dt} \tag{5}$$

$$T_h = k * v_d + u \tag{6}$$

Therefore, the throttle command T_h can be represented by the control law in Eq. (6) [34].

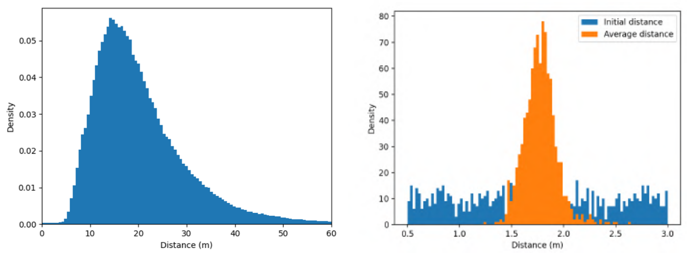
V. Experimental Validation

A. Implementation in Car-Following Scenarios

The distribution of the distance to the leader vehicle in car-following scenarios maintained by human drivers is shown in Fig. 3a. Since the size ratio of our scaled model to a real vehicle is 1:10, the distance value considered in both training and testing environment is 10 times smaller than that in the real world. The simulated environment consists of six scaled cars and three 100-m road segments. One car is the ego vehicle, and the others are surrounding vehicles that adopt rule-based driving policy. To train and evaluate the ACC using the scaled model, we set the initial intervehicle distance to obey the uniform distribution between 0.5 m and 3.0 m with maximum LiDAR detection of 2.7 m. The maximum speed limit of surrounding vehicles in each episode obeys uniform distribution between 0.5 m/s and 1.5 m/s. The maximum speed limit of the ego vehicle is 1.5 m/s. The trained model was tested on 1000 randomly generated scenarios. Testing results indicate that the ego vehicle can always maintain a safer driving distance with the leader car when performing the car following task. The distance between the ego vehicle and the leader vehicle follows a normal distribution with $\mu = 1.85$ and $\sigma^2 = 0.8$ m, as shown in Fig. 3b. In Fig. 3b, the blue bars represent the distribution of initial distance between the ego vehicle and the leader car, and the orange bars represent the distribution of average intervehicle distance in each episode, which is similar to the distribution of intervehicle distance in the real traffic data, especially in the detection range, as shown in the Fig. 3a.

B. Pretraining and Baseline Comparison

To benchmark the performance of the RL-based high-level policy, human player policy and IL-based policies are considered in this work as baselines. All policies use our trained car following policy to achieve the ACC on the current



(a) US-101 data set

(b) Testing results

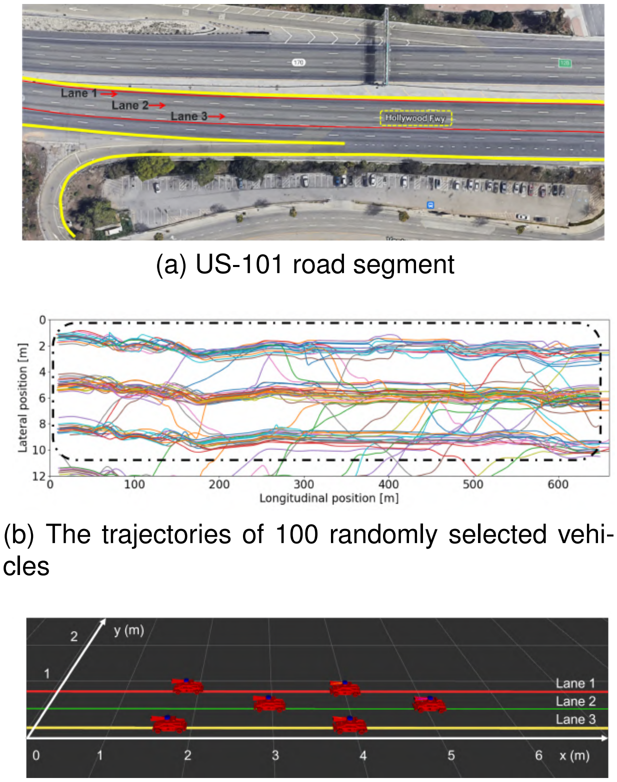
Fig. 3: Distribution of distance to the car in front.

lane. We randomly generated 100 evaluation scenarios, and the tested vehicle randomly selected a lane to start the test (0: left lane; 1: middle lane; and 2: right lane). The maximum speed of the ego vehicle is set to 1.5 m/s, while the speed limit for other vehicles in each episode is uniformly distributed between 0.5 to 0.8 m/s. In addition, the longitudinal distance between surrounding vehicles and the ego vehicle follows a uniform distribution between 1.0 to 3.0 m. All vehicles are located in front of the tested vehicle and each lane is guaranteed to have at least one vehicle.

We had a human player drive the tested vehicle in simulated environments 100 times using joystick. Each episode was played either until the driving task terminated or exceeded 50 seconds. During the human player driving, we collected the vehicle's laser scan concatenated with ego speed and lane number, actions, rewards, and terminations. This data serves two purposes. First, pretrained RL network and the IL Guidance policy are trained on this small dataset. Second, IL Guidance policy will also be regarded as one of baselines for later comparison with DQRfD policy. We noticed that the data related to the car following scenarios is about 20 times that of the data for changing the lane to left or right. Therefore, the pre-trained model is more easily to converge to a local optimal policy (i.e., following strategy). To ensure a fair comparison between the RL policy and the IL Baseline policy, we increased the demonstration data to 700 episodes (equivalent to 28,000 steps, close to the total training steps of the RL algorithm). To avoid overfitting the model, we balanced the number of three types of scenarios in the training dataset.

C. Naturalistic Human Driving Data Set

To further evaluate the generalization ability of the RL policy, we test the RL-based high-level model on real traffic data. We randomly selected 100 vehicles with lane-changing behaviors from real traffic data containing 3000 vehicles for comparison. The data was recorded by eight cameras for 10 minutes of all vehicle information in the US101 road segment with a length of 640 m. The data includes vehicle number, global time, position, speed, and lane number. A total of 25 types of information were recorded [28]. The road consists of six lanes, each with a lane width of 3.66 m. The sixth lane is the on-ramp, the off-ramp, and the auxiliary lane between them. The actual road structure is shown in Fig. 4a. Since some vehicles need to leave the expressway via the off-ramp, their lane-changing policies are different from that of other vehicles running on the first three lanes. Therefore, in this study, we only focus on the first three lanes only, and all selected trajectories are shown in Fig. 4b. To train the lane-changing policy, we reconstructed the road section in the simulator as shown in Fig. 4c, each with a lane width of 0.366 m.



(a) US-101 road segment

(b) The trajectories of 100 randomly selected vehicles

(c) Simulated environment

Fig. 4: Illustration of US-101 road segment, data set and reconstructed road structure.

Since our scaled car is 10 times smaller than real vehicles, we need to scale the traffic data proportionally in order to replay real traffic data for testing in the simulation environment. Considering that the speed limit of our scaled car during training was set to 1.5 m/s (equivalent to 15 m/s in real scenario), therefore, we remove all cases with vehicle speeds above 15 m/s from the real traffic data. During the testing, if the trained policy selects the same lane-changing action as the human driver in a given scenario, we consider that the driving behavior of the current case is successfully modeled. Due to the presence of noise in the original data, the trajectories of surrounding vehicles are fitted by a fifth-order polynomial, as introduced in section III. To obtain the boundary conditions for generating the trajectories of surrounding vehicles, we can directly scale the speed and distance in real traffic data by the ratio of the vehicle size λ (λ = 10 in our case). In this work, the duration required for a vehicle to complete a lane change in the scenario remains consistent with real traffic data. Assuming that the initial and final time of the lane-changing in the real traffic data are t_0 and t_n, the testing duration for each scenario is denoted by

$$\Delta t = t_n - t_0 \tag{7}$$

The position and speed information of vehicles in real traffic dataset can be expressed as $\tilde{\boldsymbol{p}}^i(t) = \left(\tilde{p}_x^i(t), \tilde{p}_y^i(t)\right)$ and $\tilde{\boldsymbol{v}}^i(t) = \left(\tilde{v}_x^i(t), \tilde{v}_y^i(t)\right)$. i is the vehicle number. $i = 0$ indicates a lane-changing vehicle. The boundary conditions including position and speed of vehiles in simulated environment can be calculated by

$$\left(p_x^i(t), p_y^i(t)\right) = \frac{1}{\lambda}\left(\tilde{p}_x^i(t), \tilde{p}_y^i(t)\right) \tag{8}$$

and

$$\left(v_x^i(t), v_y^i(t)\right) = \frac{1}{\lambda}\left(\tilde{v}_x^i(t), \tilde{v}_y^i(t)\right) \tag{9}$$

VI. Implementation Results

A. Robustness Analysis of Car-Following Policy

To evaluate the performance of the car-following policy, we compared it with 3 human players. In each group of comparisons, human players control the speed of the car using a joystick, and the trained policy was also tested in the same traffic scenarios. All human players and the trained policy use the same observation information to ensure the fairness of the comparison. Human drivers watch the speed information of the controlled car as well as the point cloud data detected by the LiDAR to obtain the position information of the vehicle in front (For example, if the intervehicle distance is out of the maximum detection range, players cannot observe the point cloud information of the front vehicle in the simulator). In each group of tests, we considered 30 cases where the leading vehicle, in the beginning, is located in the unsafe zone (gap<1.5 m), the interaction zone (1.5 m< gap <2.7 m), and the safe zone (gap>2.7 m) of the controlled vehicle respectively. The maximum speed limit of the leading vehicle is set to change within the range of [0.5, 1.5] m/s. In each episode, the simulator will randomly select a pair of parameters (initial distance, max speed) from list D = [0.8, 1.0, 1.25, 1.4, 1.5, 2, 2.25, 2.7, 2.9, 3.0] m and list *v*= [1.0, 0.75, 1.2] m/s. The evaluation includes safety, comfort, and energy consumption. The last two factors are reflected by the change of the vehicle's acceleration. To save fuel consumption, drivers usually try to use minimum effort to achieve the desired behaviors. In other words, the less often the driver uses deceleration actions, the better. The order of each action with respect to comfort is as follows: maintain the current speed>acceleration (or deceleration)>hard acceleration (or hard deceleration). In order to improve safety, the car following policy needs to be able to maintain a stable distance from the leading vehicle and maintain a relative speed of around 0 with it.

During the test, we use the reward function defined in section II to score the driving behaviors. The action frequency is 16 Hz, and the maximum number of steps is 300. We counted the average score for each group of tests, the probability of violating the unsafe zone, the distance between the ego car and the leader, and the speed difference separately. Table III shows that the trained car-following policy outperforms human players in the above aspects. To understand why the trained policy scored high, we analyzed the recorded data further. We selected three representative sets of comparative data respectively, as shown in Fig. 5. In Fig. 5a, the initial distance is 1.0 m, and the maximum speed of the leader is 1.25 m/s. Both the RL policy and player 1 choose to maintain the current speed of 0 to avoid collision with the leader. Compared to the performance of player 1, the trained policy can adjust actions quicker to maintain a stable distance and to get a smaller speed difference with the leader. From the speed curve, we can find that player 1 focuses more on comfort, and the RL policy is more on travel efficiency. In Fig. 5b, the leader is out of the LiDAR detection range at the beginning, and both the RL policy and player 2 choose to accelerate. As the distance decreases, the leader begins to enter the detection range of the LiDAR. However, player 2 cannot take actions as quickly and effectively as the RL policy to maintain a safe interactive distance with the leader. Finally, the car controlled by player 2 enters the unsafe area, thereby increasing the risk of collision with the leader. According to Fig. 5c, we can see that both the RL policy and player 3 can effectively take actions to maintain a safe interaction distance with the leader, and take as few 'deceleration' actions as possible to save energy consumption. To compare the robustness of different driving behaviors during testing, we performed a statistical analysis of the recorded data. From Fig. 6a, we can see that compared to the other three human players, the RL policy uses more "maintain" actions and uses less "deceleration" action to reduce energy consumption. Fig. 6b shows that the average distance between the ego vehicle and the leader is 2.06 m and the fluctuation range is 1.2 m smaller than other three groups of tests. According to Fig. 6c, the average score of the RL policy in the 30 groups of tests is -235.54, and the fluctuation of the scores is significantly smaller than that of human players which proves that the trained car-follow policy is more robust. Although the discrete actions adopted in this paper is different from the continuous action space in real world, we can conclude from the above comparison that the learning-based driving policy can perform the same or even better than human drivers on some specific driving tasks. Unlike human drivers, the machine will not be fatigued due to repetitive tasks which leads to unsafe-driving in humans.

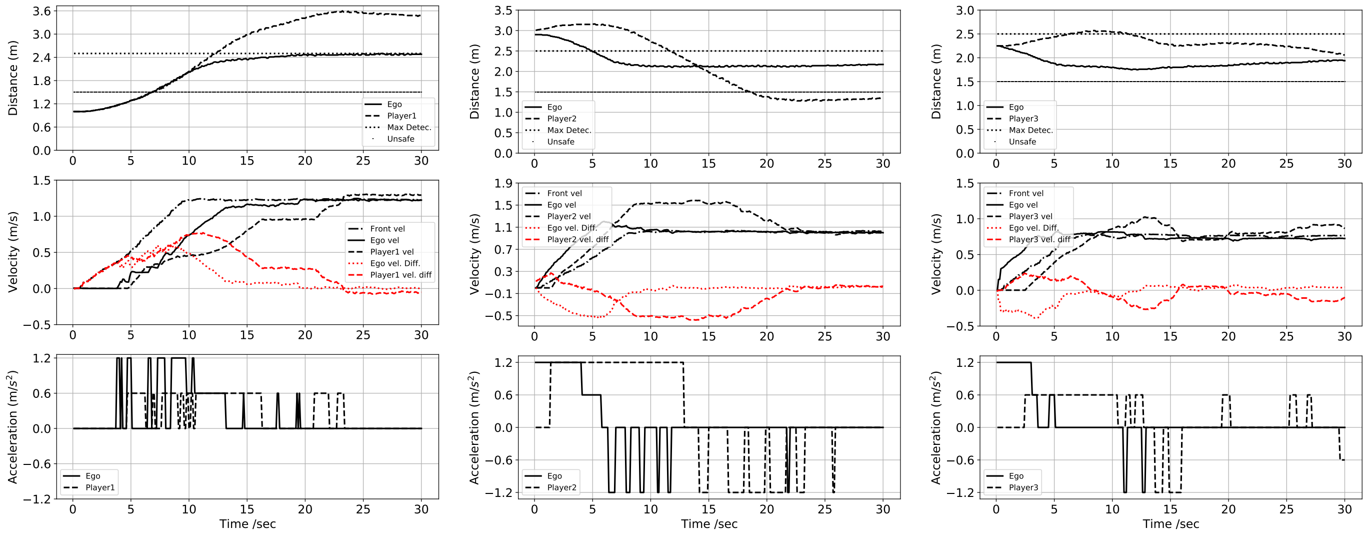
TABLE III: Comparison with Human Players

Policy	**Rewards**	**Unsafe rate**	Δd **(m)**	Δv **(m/s)**
RL policy	**-235.54**	**0.0%**	2.06	**0.105**
Player 1	-427.08	6.7%	2.51	0.248
Player 2	-488.31	63.3%	1.82	0.238
Player 3	-408.13	3.3%	2.23	0.257

B. Training Results

To maintain the conciseness of the paper, we demonstrate the results of utilizing our DRQfD framework to enhance the training efficiency of the RL algorithm for the high-level policy. we also compare it with vanilla D3QN PER and IL Guidance policy, as shown in Fig. 7. The red dashed line in

This article has been accepted for publication in IEEE Transactions on Vehicular Technology. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVT.2023.3307409

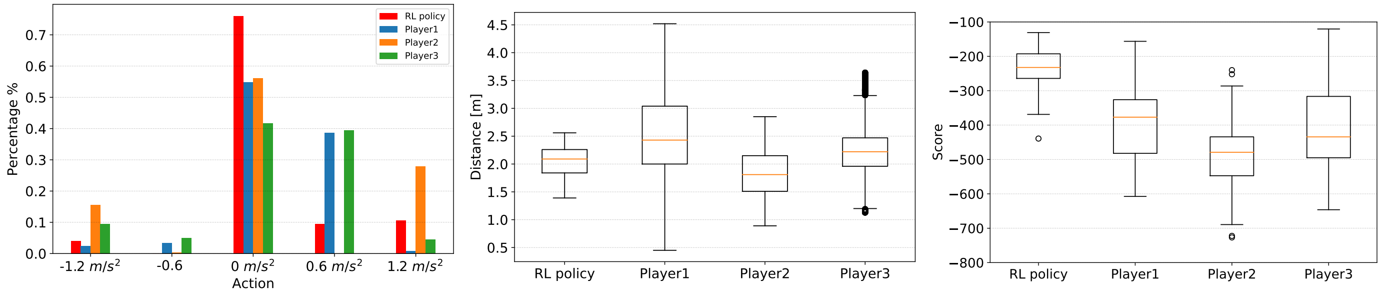


(a)

(b)

(c)

Fig. 5: Comparison of learned policy with human players on features of intervehicle distance, relative speed and actions: (a) learned policy vs. player 1 (unsafe zone) with 1.25 m/s speed limit of the leader car; (b) learned policy vs. player 2 (safe zone) with 1.0 m/s speed limit of the leader car; (c) learned policy vs. player 3 (interaction zone) with 0.75 m/s speed limit of the leader.



(a) Distribution of actions

(b) Distribution of distances to leader

(c) Distribution of scores

Fig. 6: Comparison of driving preference and robustness of different policies (a) action distribution of different policies; (b) Head-way space distribution of different policies; (c) Score distribution of different policies.

the figure represents the performance of the Guidance policy trained by the IL algorithm on a small dataset, which is used to guide RL training. Due to the small data size, Guidance policy falls into a local minimum of choosing the car following policy most of the time. On average, IL Guidance policy obtains an accumulated reward of around 2.75/episode, demonstrating a lower travel efficiency. The blue curve represents the training process using the DRQfD framework. We can see that the proposed framework can effectively help the RL algorithm to converge quickly to the optimal policy level, saving about 30% of training episodes. The yellow line represents the training curve of the vanilla D3RQN PER algorithm. We cut off the training curve at the termination of the DRQfD training. However, as the number of training episodes increases, we found that the vanilla D3RQN PER algorithm can also converge to the level of the DRQfD algorithm. Therefore, we can conclude that the pre-trained RL network and the IL Guidance policy trained on a small amount of expert demonstration can effectively help to improve the learning efficiency of RL. In the next section, we will compare the RL policy with the IL and human player policies in randomly generated scenarios.

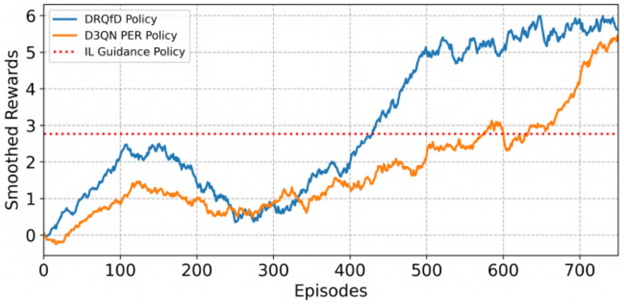


Fig. 7: Training processes of different learning methods in the lane change scenarios.

TABLE IV: Baseline Comparison Results

Methods	**Collision**	Δd	v_{avg} **(m/s)**	**Invalid *a***
Human Player	**5%**	42.5	0.85	**0.03%**
DRQfD	7%	**44.5**	**0.89**	0.27%
IL Baseline	20%	38.5	0.77	1.06%
IL Guidance	13%	36.0	0.72	0.88%

C. Comparison with Imitation Learning

Since the vanilla D3QN PER and our DRQfD algorithms eventually converge to the same level, for the sake of simplicity, we only select one model to represent the RL policy. We compare the performance of RL policy with a human player, IL Guidance policy, and IL Baseline policy in the same scenarios. We evaluate the driving safety of each policy based on the collision rate. The driving distance and average speed are used to evaluate the traffic efficiency. Invalid action rate is the proportion of the total number of actions taken by the ego vehicle during the entire testing, in which either the ego changed lanes despite no vehicles being present in front of it, or the lane-changing action caused the ego to leave the road. To provide consistent observation information for the human player and other trained policies, similar to the car-following scenario introduced earlier, we only provided the human player with the visualized point cloud of surrounding vehicles through RViz, without displaying information about vehicles out of the LiDAR detection range. The test results are shown in Table IV. The human player has lower collision rate and invalid action rate than all trained policies, indicating ideal driving behavior. Although the RL policy has slightly higher values than the human driving policy in these two metrics, it is significantly better than the IL policy. This result is reasonable, as even though we provided the IL policy with the same amount of data as the RL training, the demonstration data only includes successful driving scenarios. It may not generalize well to new and unseen scenarios during testing. We also analyzed the reasons why human players had collisions. This was mainly due to the fact that in order to be compatible with the size of the indoor validation environment, we set the detection range of LiDAR to 2.7m. Therefore, all collision scenarios of human players occurred when there were no displayed vehicles in front of the target lane before the lane change. However, when the human drivers choose to change lanes to get a higher speed, vehicles suddenly appeared in front of the ego in the target lane, leading to collisions. The above reasons also apply to other learning-based policies that experienced collisions in the same scenarios. This problem can be solved by increasing the maximum detection range of the LiDAR. Regarding the evaluation of driving efficiency, we found that the RL policy performed the best in terms of driving efficiency, with an average driving speed of 0.89 m/s, higher than the other driving policies. Additionally, this speed was higher than the maximum driving speed setting of surrounding vehicles (0.8 m/s), indicating that the RL policy can effectively take lane-changing actions to improve driving speed. To evaluate the similarity between different learning-based policies and human driving behavior, Fig. 8 shows the action distribution of each driving policy in the same 100 test scenarios. Overall, the RL policy is closest to human driving behavior with a slightly higher proportion of lane-changing actions than human players, which explains why RL achieves higher travel efficiency. However, based on Fig. 8 and Table IV, we can see that the generalization ability of the IL Baseline policy is unsatisfactory, showing a higher collision rate and invalid action rate. The red bar represents the IL Guidance policy used to guide RL early exploration. Due to the small amount of data, IL Guidance policy falls into a local minimum of choosing car-following policy most of the time. The proportion of lane-changing actions taken by IL Guidance policy is significantly lower than that of the other policies. We can conclude that our RL-driven policy outperforms IL-based policy in terms of driving safety, travel efficiency, and human likeness.

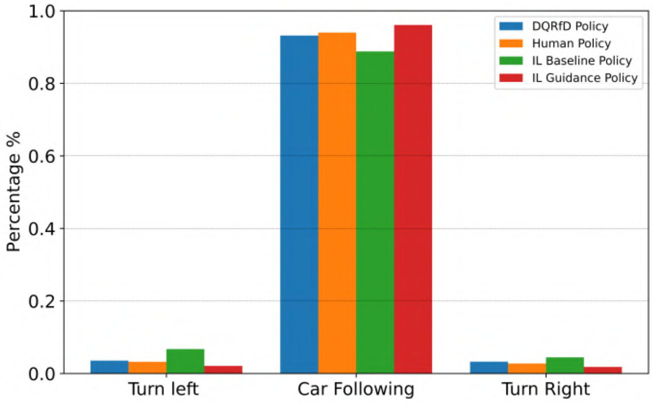


Fig. 8: Distribution of high-level actions.

D. Test Results in Real Traffic Data

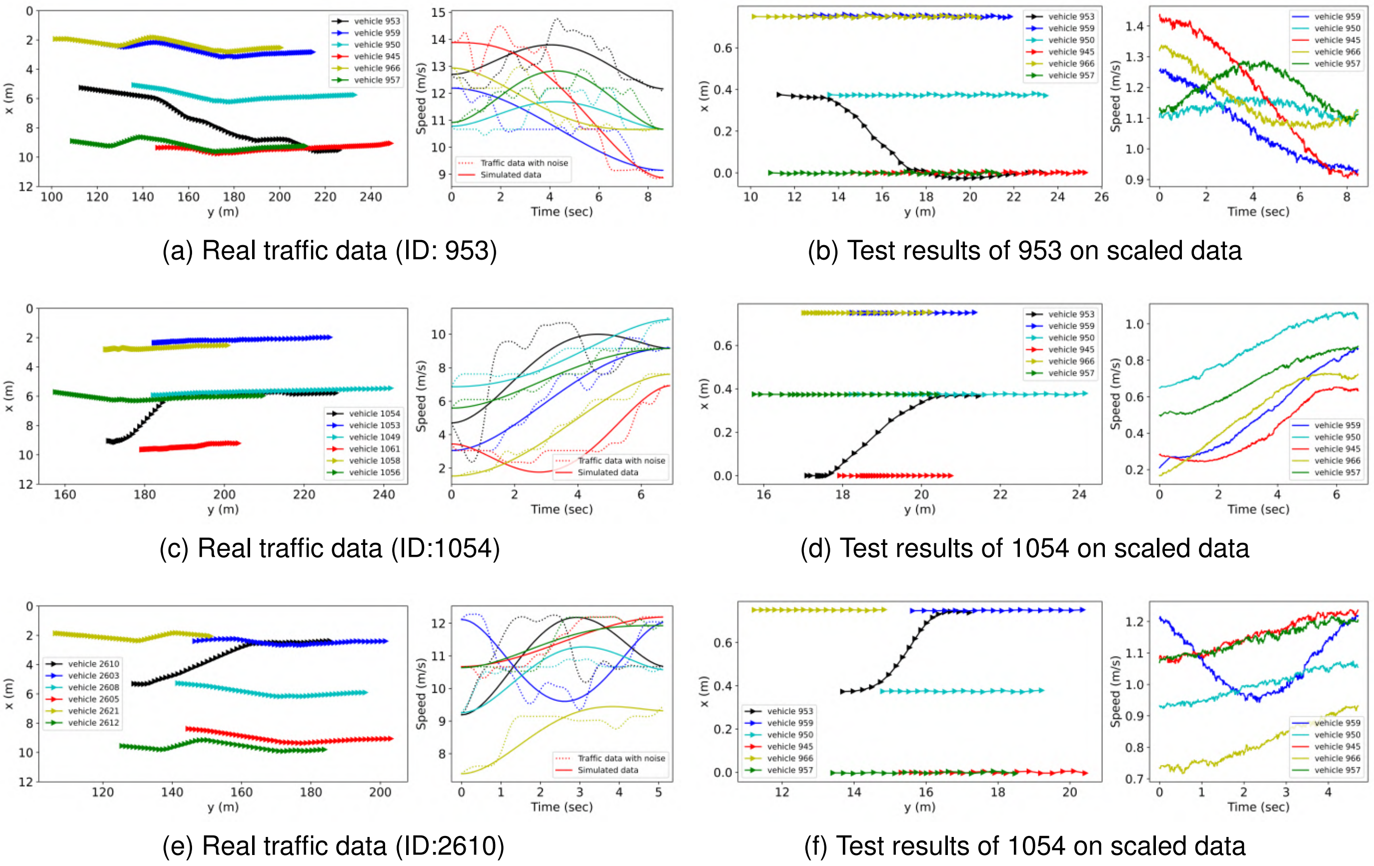
According to the previous section, we demonstrated that the RL policy outperforms the IL policy. In this section, we aim to validate whether the RL policy can choose lane-changing actions consistent with human drivers in the given scenarios. The primary factors considered by a vehicle during lane-changing decision-making involve relational attributes such as position and speed relative to other vehicles in the surrounding environment. As explained in Section IV-C, we scaled the boundary conditions of surrounding vehicles based on real traffic data to generate trajectories for replaying in the testing environments. After testing 100 lane-changing scenarios, the modeling success rate under the scaled speed settings is above 81%. The remaining cases, which involve collision or not selecting a lane-changing action, are considered as modeling failures. Here, we take three cases from the 100 testing scenarios as examples to demonstrate the driving performance of the trained policy. The position and speed information of surrounding vehicles as well as the testing duration for each case are shown in Table V.

Taking vehicle 953 as first example, the first two values of each set of data in the second row of Table V indicate the initial and final scale positions of surrounding vehicles. From Fig. 9a, we can see that 959 (blue), 950 (cyan), 945 (red) are

TABLE V: Boundary Conditions of Surrounding Vehicles

$(p_y^1, p_y^{'1}, v_y^1, v_y^{'1})$	$(p_y^2, p_y^{'2}, v_y^2, v_y^{'2})$	$(p_y^3, p_y^{'3}, v_y^3, v_y^{'3})$	$(p_y^4, p_y^{'4}, v_y^4, v_y^{'4})$	$(p_y^5, p_y^{'5}, v_y^5, v_y^{'5})$	Δt
Ego IDs = 953, (p_y^0, v_y^0) = (11.31, 1.25); Surroundings = 959, 950, 945, 966, 957					
(12.27, 21.45, 1.22, 0.91)	(13.62, 23.29, 1.07, 1.07)	(14.67, 24.87, 1.39, 0.90)	(10.18, 20.05, 1.24, 1.06)	(10.93, 21.14, 1.11, 1.11)	8.6
Ego IDs = 1054, (p_y^0, v_y^0) = (17.10, 0.47); Surroundings =1053, 1049, 1061, 1058, 1056					
(18.24, 22.64, 0.31, 0.92)	(18.22, 24.17, 0.69, 1.09)	(17.93, 20.35, 0.35, 0.69)	(17.02, 20.08, 0.15, 0.76)	(15.77, 20.97, 0.56, 0.91)	6.9
Ego IDs = 2610, (p_y^0, v_y^0) = (12.90, 0.92); Surroundings = 2603, 2608, 2605, 2621, 2612					
(14.66, 20.15, 1.21, 1.20)	(14.17, 19.57, 0.93, 1.06)	(14.47, 20.30, 1.07, 1.22)	(10.64, 15.11, 0.74, 0.93)	(12.56, 18.39, 1.06, 1.19)	5.2

Note: boundary conditions of vehicle i: $(p_y^i, p_y^{'i}, v_y^i, v_y^{'i})$ = (Init. pos., Final pos., Init. speed, Final speed); Testing duration: Δt.



(a) Real traffic data (ID: 953)

(b) Test results of 953 on scaled data

(c) Real traffic data (ID:1054)

(d) Test results of 1054 on scaled data

(e) Real traffic data (ID:2610)

(f) Test results of 1054 on scaled data

Fig. 9: Comparing driving behavior of RL policy with human driver policy under simulated environment: (a), (c), (d) The trajectory and velocity information extracted from US101 with lane-changing vehicle of 953, 1054, and 2610; (b), (d), (f) are the testing results of RL policy with replaying the trajectories of surrounding vehicles on scaled data.

the vehicles in front of the tested vehicle (953, black) at the very beginning, of which the 945 (red) is the farthest from the tested vehicle (black). The first values of the fourth and fifth set of data in the second row of the table are smaller than initial position of ego car (p_y^0 = 11.31 m), indicating that 966 (yellow) and 957 (green) are the vehicles behind the tested vehicle (black). The last two values of each set of data in the table are the initial and final scale speeds of surrounding vehicles. We can see that the initial speed of the vehicle 950 (cyan) ahead of tested vehicle in the current lane and the rear vehicle 957 (green) in the target lane are smaller than the initial speed of tested vehicle (v_y^0 = 1.25), indicating that their current speeds are lower than the ego vehicle speed while the current speed of vehicle 945 (red) in the target lane is faster than the tested vehicle. The trajectories and speeds of surrounding vehicles extracted from real traffic data for case 1 are shown in Fig. 9a. Fig. 9b shows the testing result in the simulated environment, and we can find that speed profiles of surrounding vehicles in the simulated environment have the same trend as the real traffic data. The ego vehicle chooses to change the lane to the right similar to the decision made by human driver. Therefore, the driving behavior in this scene is considered to be successfully modeled. Similarly, the other two examples are about real scenarios where the tested vehicles change lane to the left from the middle lane and the bottom lane respectively as show in Fig. 9c and 9e. Tested

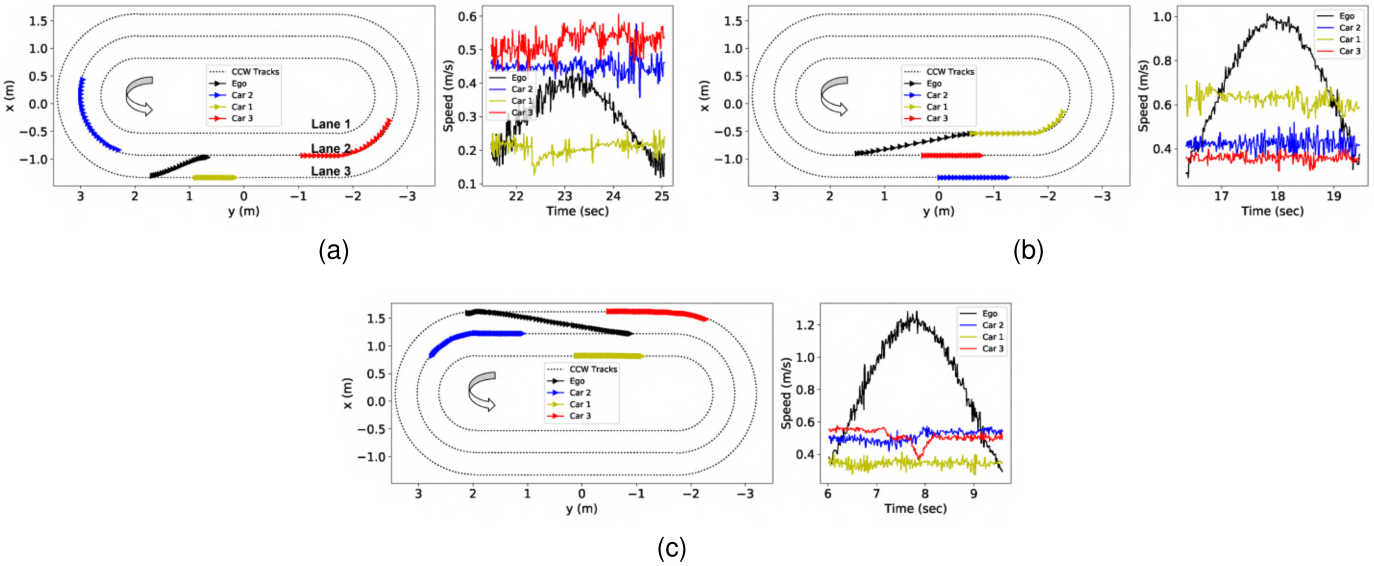


Fig. 10: Testing results of the hardware implement in different scenarios (a) Changing the lane to the left from lane 3; (b) Changing the lane to the left from lane 2; (c) Changing the lane to the right from lane 2.

vehicles all made decisions to change the lane same as human drivers as shown in Fig. 9d and Fig. 9f. It should be noted that the reward function designed for high-level policy focuses more on driving safety and travel efficiency under different driving conditions rather than fitting the true trajectories of lane-changing vehicles, therefore, we have excluded the speed curve of tested vehicles from test results to avoid confusion. More details about tests can be found in video [1].

E. Hardware Implementation

To verify the performance of sim2real transfer, in this section, we select three scenarios with four cars to show the lane-changing behavior in real world. Both the car-following model and the lane-changing model are loaded to the cars and conducted continuous testing for 5 hours. The size of the physical car and the virtual car in the simulator are completely identical. Each car is equipped with an Nvidia Jetson-TX2 GPU and a 2D LiDAR. The decision frequency is 15 Hz and the control frequency is 100 Hz. Although there are only four physical cars available for use, we select the testing scenarios where the surrounding vehicles can directly influence decision-making made by the ego car. The surrounding vehicles are set to execute the trained car-following policy only during testing, and the ego car adopts the complete policy proposed in this paper. To facilitate continuous testing, we designed an elliptical three-lane scene as shown in Fig. 10 where the outer lane is lane 3 and the inner lane is lane 1. All cars move counter-clockwise. We use motion capture system to get the ground truth of positions and velocities. From Fig. 10a, we can see that the ego car is currently in the lane 3 where the speed of car 1 (yellow) in front is lower than the speed of car 3 (red) in the target lane. And the gap between car 2 (blue) and car 3 (red) in the target lane is safe enough for the ego car to complete lane-changing. According to Fig. 10b, the ego car is in the lane 2 while the speed of car 3 (red) in the current lane is the slowest. In addition, car 1 (yellow) in the left lane is faster and farther away from the ego car than car 2 (blue) in the lane 3, therefore, the ego car chooses to overtake car 3 (red) from lane 1 in this case. From Fig. 10c, the ego car is in the lane 2, and the gap between car 3 (red) in the lane 1 and car 2 (blue) in the current lane of the ego car is smaller than that of car 1 (yellow) in the lane 3. Therefore, the ego car chooses to overtake the car 2 (blue) in front from lane 3. Check out the video for more test scenarios.

From the above examples, we can see that the policies learned from simulator can be directly used on the hardware. The performance of the ego car in the real scene is consistent with the driving behavior in the simulated environment, which proves the feasibility of the method proposed in this paper.

VII. Conclusions

In this work, we systematically studied the application of a DRL method in lane-changing scenarios from three aspects: learning efficiency, partial observability, and sim2real transfer. Specifically, we first propose a new DQRfD algorithm, which has three advantages: (i) improving the learning efficiency of RL; (ii) improving the generalization ability of IL; (iii) solving the POMDP problem in decision-making of AD through the LSTM network, thereby stabilizing the training process. Secondly, a hierarchical decision-making framework training car-following policy and high-level policy separately in an end-to-end manner is proposed to address the multi-objective problem in lane-changing scenarios. Third, to validate the effectiveness of our method, we build a high-fidelity simulation platform based on ROS-Gazebo for training and evaluating of different driving policies. According to the testing results in car following scenarios, the RL-driven policy is capable of performing as

[1] Video of our experiment results can be viewed at https://youtu.be/Svp2S1OaSB8

well as or even better than human drivers in safety and energy consumption. Regarding the high-level lane-changing policy, we compared our DQRfD method with a pure RL method showing about 30% improvement in learning efficiency. Qualitative and quantitative comparisons between our model and IL baseline are also conducted. The experimental results show that our proposed method outperforms IL in terms of safety, travel efficiency, and human likeness. To further validate the generalization ability of our model, we test the model on real traffic data, demonstrating a successful modeling rate of 81%. Finally, we load the trained model onto our hardware platform for evaluation, which exhibits consistent behaviors with the simulation. As a result, the overall decision-making framework proposed in this work exhibits great potential to enhance the practical application of RL-driven HAD.

Acknowledgment

The authors would like to thank Dr. Hassan Alkomy, Hunter Schofield, and Dr. Xiaoyu Zhu for insightful discussions. We would also like to thank Yiquan Ma for processing the data.

References

[1] D. Chen, L. Jiang, Y. Wang, and Z. Li, "Autonomous driving using safe reinforcement learning by incorporating a regret-based human lane-changing decision model," in *2020 American Control Conference (ACC)*. IEEE, 2020, pp. 4355–4361.

[2] J. Wu, Z. Huang, Z. Hu, and C. Lv, "Toward human-in-the-loop ai: Enhancing deep reinforcement learning via real-time human guidance for autonomous driving," *Engineering*, vol. 21, pp. 75–91, 2023.

[3] Z. Huang, J. Wu, and C. Lv, "Efficient deep reinforcement learning with imitative expert priors for autonomous driving," *IEEE Trans Neural Netw Learn Syst*, 2022.

[4] J. Wu, Z. Huang, W. Huang, and C. Lv, "Prioritized experience-based reinforcement learning with human guidance for autonomous driving," *IEEE Trans Neural Netw Learn Syst*, 2022.

[5] M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps," in *2015 AAAI Fall Symposium Series*, 2015.

[6] H. Krasowski, X. Wang, and M. Althoff, "Safe reinforcement learning for autonomous lane changing using set-based prediction," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–7.

[7] J. Wu, Z. Huang, C. Huang, Z. Hu, P. Hang, Y. Xing, and C. Lv, "Human-in-the-loop deep reinforcement learning with application to autonomous driving," *arXiv preprint arXiv:2104.07246*, 2021.

[8] H. Liu, Z. Huang, J. Wu, and C. Lv, "Improved deep reinforcement learning with expert demonstrations for urban autonomous driving," in *2022 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2022, pp. 921–928.

[9] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, I. Osband *et al.*, "Deep q-learning from demonstrations," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[10] T. Shi, P. Wang, X. Cheng, C.-Y. Chan, and D. Huang, "Driving decision and control for automated lane change behavior based on deep reinforcement learning," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 2895–2900.

[11] K. Rezaee, P. Yadmellat, M. S. Nosrati, E. A. Abolfathi, M. Elmahgiubi, and J. Luo, "Multi-lane cruising using hierarchical planning and reinforcement learning," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 1800–1806.

[12] V. A. Banks, K. L. Plant, and N. A. Stanton, "Driver error or designer error: Using the perceptual cycle model to explore the circumstances surrounding the fatal tesla crash on 7th may 2016," *Safety Science*, vol. 108, pp. 278–285, 2018.

[13] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," *Adv. Neural Inf. Process.*, vol. 1, 1988.

[14] L. Le Mero, D. Yi, M. Dianati, and A. Mouzakitis, "A survey on imitation learning techniques for end-to-end autonomous vehicles," *IEEE Trans. Intell. Transp. Syst.*, 2022.

[15] U. Muller, J. Ben, E. Cosatto, B. Flepp, and Y. Cun, "Off-road obstacle avoidance through end-to-end learning," *Adv. Neural Inf. Process.*, vol. 18, 2005.

[16] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[17] J. Zhou, R. Wang, X. Liu, Y. Jiang, S. Jiang, J. Tao, J. Miao, and S. Song, "Exploring imitation learning for autonomous driving with feedback synthesizer and differentiable rasterization," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 1450–1457.

[18] P. Cai, S. Wang, H. Wang, and M. Liu, "Carl-lead: Lidar-based end-to-end autonomous driving with contrastive deep reinforcement learning," *arXiv preprint arXiv:2109.08473*, 2021.

[19] S. Chen, M. Wang, W. Song, Y. Yang, Y. Li, and M. Fu, "Stabilization approaches for reinforcement learning-based end-to-end autonomous driving," *IEEE Trans. Veh. Technol.*, vol. 69, no. 5, pp. 4740–4750, 2020.

[20] R. Cimurs, I. H. Suh, and J. H. Lee, "Goal-driven autonomous exploration through deep reinforcement learning," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 730–737, 2021.

[21] K. Arndt, M. Hazara, A. Ghadirzadeh, and V. Kyrki, "Meta reinforcement learning for sim-to-real domain adaptation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 2725–2731.

[22] A. Loquercio, E. Kaufmann, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza, "Deep drone racing: From simulation to reality with domain randomization," *IEEE Trans. Robot.*, vol. 36, no. 1, pp. 1–14, 2019.

[23] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2017, pp. 23–30.

[24] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield, "Training deep networks with synthetic data: Bridging the reality gap by domain randomization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 969–977.

[25] E. Candela, L. Parada, L. Marques, T.-A. Georgescu, Y. Demiris, and P. Angeloudis, "Transferring multi-agent reinforcement learning policies for autonomous driving using sim-to-real," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 8814–8820.

[26] H. Shi, G. Liu, K. Zhang, Z. Zhou, and J. Wang, "Marl sim2real transfer: Merging physical reality with digital virtuality in metaverse," *IEEE Trans. Syst. Man Cybern. Syst.*, 2022.

[27] N. Li, D. W. Oyler, M. Zhang, Y. Yildiz, I. Kolmanovsky, and A. R. Girard, "Game theoretic modeling of driver and vehicle interactions for verification and validation of autonomous vehicle control systems," *IEEE Trans Control Syst Technol.*, vol. 26, no. 5, pp. 1782–1797, 2017.

[28] V. Alexiadis, J. Colyar, J. Halkias, R. Hranac, and G. McHale, "The next generation simulation program," *ITE J. (Inst. Transp. Eng.)*, vol. 74, no. 8, p. 22, 2004.

[29] B. M. Albaba and Y. Yildiz, "Modeling cyber-physical human systems via an interplay between reinforcement learning and game theory," *Annu. Rev. Control.*, vol. 48, pp. 1–21, 2019.

[30] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[31] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, 2015.

[32] H. Alkomy and J. Shan, "Vibration reduction of a quadrotor with a cable-suspended payload using polynomial trajectories," *Nonlinear Dyn.*, vol. 104, no. 4, pp. 3713–3735, 2021.

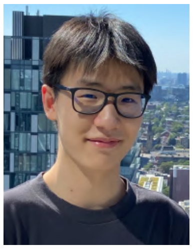
[33] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann *et al.*, "Stanley: The robot that won the darpa grand challenge," *J. Field Robot.*, vol. 23, no. 9, pp. 661–692, 2006.

[34] D. E. Rivera, M. Morari, and S. Skogestad, "Internal model control: Pid controller design," *Ind. Eng. Chem. Process.*, vol. 25, no. 1, pp. 252–265, 1986.

This article has been accepted for publication in IEEE Transactions on Vehicular Technology. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVT.2023.3307409

Mingfeng Yuan received his M.S. (2019) in control engineering from Tianjin University of Technology, China. He is currently pursuing his Ph.D degree with the Department of Earth and Space Science and Engineering, York University, Toronto, ON, Canada. His research interests include machine learning for prediction and decision-making in automated driving, game theory, deep reinforcement learning, and nonlinear system modeling.

Jinjun Shan (SM'08) received the Ph.D. degree in spacecraft design from the Harbin Institute of Technology, Harbin, China, in 2002. He is currently a Full Professor of Space Engineering and Chair of the Department of Earth and Space Science and Engineering, York University, Toronto, ON, Canada. His research interests include dynamics, control, and navigation. Dr. Shan is a Fellow of Canadian Academy of Engineering (CAE), a Fellow of Engineering Institute of Canada (EIC) and a Fellow of American Astronautical Society (AAS). He was the recipient of the Alexander von Humboldt Research Fellowship and JSPS Invitation Fellowship in 2012. Since 2007, he has been a Professional Engineer in Ontario.



Kevin Mi is currently pursuing a Bachelor's degree of Engineering Science specializing in Machine Intelligence at the University of Toronto, Toronto, ON, Canada. His research interests include deep reinforcement learning, machine learning for autonomous driving, and pedestrian behavior prediction.

Scalable Game-Theoretic Decision-Making for Self-Driving Cars at Unsignalized Intersections

Mingfeng Yuan, *Student Membership*, Jinjun Shan, *Senior Member, IEEE*, and Hunter Schofield

***Abstract*—Sharing the road with human drivers requires autonomous vehicles to account for interactions between them. To resolve traffic conflicts in unsignalized intersections, a robust adaptive game-theoretic decision-making algorithm with scalability is proposed based on the receding horizon optimization, level-k game theory, and switching directed graph. A mismatch between the inherent (k-1) assumption of level-k theory and actual driver type may lead to unsafe action selection and reduce driving safety. To handle this problem, in this work, an autonomous vehicle would predict the driver types of surrounding vehicles based on historical interactive behaviors between them and utilize its trust in the driver types to achieve an adaptive driving strategy. Besides, switching interaction graph is incorporated into an adaptive level-k framework for the first time, so as to cut off the connection between ego vehicle and nearby vehicles that do not affect driving behavior of the former, contributing to reducing the computing complexity. The feasibility, effectiveness, and real-time implementation of the proposed method are validated on both hardware and ROS-Gazebo platform.**

***Index Terms*—Scalable adaptive control, level-k game theory, driving aggressiveness, multi-vehicle interaction.**

I. Introduction

THE decision-making module is crucial for safe and efficient driving in autonomous vehicles (AVs). However, AVs face significant challenges in coexisting with human-driven vehicles (HDVs) and making fast and optimal driving decisions in complex and unknown traffic environments with only partial observations [1]. Designing robust, optimal, and computationally efficient decision-making algorithms has become a research hotspot, particularly for unsignalized intersections, which are more challenging for both HDVs and AVs due to complex vehicle interactions.

The most crucial task of advanced decision-making systems is to avoid collisions. There are four main categories of collision avoidance methods (CA): motion planning-based, risk assessment-based, game-theoretic-based, and learning-based (supervised and reinforcement learning). Studies have been conducted in each category [2]. Motion planning methods are commonly used to solve CA problems [3], but their application can be limited because the CA constraints they rely on are typically non-linear and non-convex. This can make the problem NP-hard [4].

Two methods for risk assessment have been developed, namely the deterministic approach and the probabilistic approach. The deterministic approach predicts whether a collision will occur or not by using a pre-determined threshold of specific indicators such as time headway or time to collision [5]. Although this approach has a low computational burden, it is not effective in more complex scenarios and does not model input data uncertainties. On the other hand, the probabilistic approach [6] uses empirical criteria as safety thresholds, limiting its adaptiveness in different traffic scenarios and leading to over-conservative actions. This study introduced a driver-type assessment model based on game theory and integrated it into the decision-making framework for AVs on structured roads. It overcomes the above-mentioned problems by considering multiple safety metrics and drivers' driving style preferences for effective and efficient CA.

Game theory is a promising method to make strategic decisions for AVs in mixed traffic scenarios. Previous studies have used Nash equilibrium [7], [8], Stackelberg game, and differential game approaches [9] to model driving conflicts, controller design [10], and interaction behaviors of vehicles at different traffic scenes. However, in this study, the level-k game theory approach is used to formulate vehicle interactions as a dynamic game. This method considers the observation of surrounding cars, predicts their actions, and finds the optimal response, which differs from previous works as it breaks down the Nash-equilibrium rational-expectations logic and assumes that drivers regard others as less sophisticated than themselves. There have been numerous previous works in the field of AV control at intersections, which have utilized a combination of level-k game theory and receding horizon control. For instance, in [11], a multi-vehicle interaction model was proposed to address driving conflicts at unsignalized intersections, but only two-car interactions were considered due to computational limitations. To mitigate the computational burden associated with estimating driver types and predicting future actions of other vehicles, pure learning based approaches [6] and a combination of level-k theory and learning-based methods [12]–[14] have been proposed.

These approaches move works of driver-type estimation and behavior prediction to offline training and use a trained model online to reduce running time of algorithms. In [13], an explicit online implementation scheme was proposed that

Manuscript received Month xx, 2xxx; revised Month xx, 2xxx and Month xx, 2xxx; accepted Month xx, 2xxx. Date of publication Month xx, 2xxx; date of current version Month xx, 2xxx.

This work was supported in part by NSERC Alliance Program under Grant ALLRP 555847-20, and in part by Mitacs Accelerate Program under Grant IT26108.

Mingfeng Yuan, Jinjun Shan, and Hunter Schofield are with the Department of Earth and Space Science, Lassonde school of Engineering, York University, ON M3J 1P3, Canada. {mfyuan, jjshan, hunterls}@yorku.ca.

uses function approximation techniques to avoid optimization problems in real-time. In [14], the algorithm was extended to different intersection shapes, and an imitation learning-based algorithm for level-k control policies was proposed. End-to-end algorithms were also designed to map relationships between observations and vehicle operations [12], [15], showing improvements in real-time performance and flexibility for multi-vehicle scenarios. However, uncertainties from simplified kinematic models and unknown driving preferences were not considered in these works, potentially reducing AV safety. [16] proposes a method to accurately predict lane-changing behaviors in complex transportation environments by integrating driving environments and drivers' cognitive processes using a fuzzy inference system and long short-term memory neural network. But, performance is limited by the quality of training data, and the learning-based method lacks interpretability. In this work, uncertainties are treated as disturbances and included in the driving model. The AV assumes that surrounding interactive vehicles are aggressive drivers, resulting in more conservative driving behavior in the absence of interaction data. As the AV's trust in the driving type of surrounding vehicles increases during interactions, the driving strategy is constantly adjusted. Previous research has considered AV driving uncertainty in highway situations [17], but few studies have explored intersections without traffic lights. To the best of our knowledge, there has been insufficient research on reducing the computational complexity of decision-making algorithms while maintaining interpretability, based on level-k game theory.

Our approach differs from existing works in the following ways: (1) An adaptive decision-making algorithm is designed using receding horizon optimization, level-k game theory, and directed switching graph to address interactions between AV and vehicles with varying driving preferences in complex unsignalized intersections. (2) By utilizing the switching interaction graph, AVs can establish instantaneous interactive connections with other vehicles, allowing them to adapt their decision-making strategies to complex traffic situations, which effectively addresses the challenges of computational complexity and scalability faced by previous level-k based algorithms. (3) The proposed adaptive strategy adjusts the size of anticipated disturbances based on the aggressiveness of other interacting vehicles, which provides a 'balanced' control action for the AV that is safer than aggressive strategies while also being more efficient than conservative strategies; (4) Compared to previous studies, we have created a high-fidelity simulator using ROS-gazebo for unsignalized intersections. This simulation environment is capable of evaluating decision-making algorithms in terms of scalability, multi-vehicle interaction, interpretability, and real-time implementation.

The rest of this paper is organized as follows. In Section II, the problem formulation is presented. Section III introduces kinematic model with pure pursuit controller. Section IV describes the details about a scalable adaptive game-theoretic decision-making framework. Hardware and simulation results are provided in Section V to show the effectiveness of proposed method. Section VI concludes this paper.

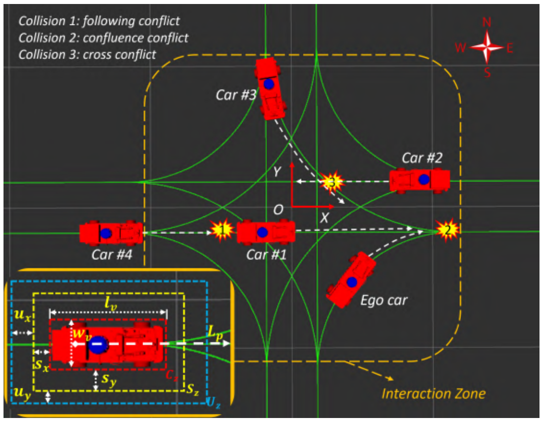


Fig. 1: Unsignalized Intersection Scenario

II. PROBLEM FORMULATION

There are 12 possible paths for AVs at a four-way single-lane unsignalized intersection as shown in Fig. 1. One vehicle is chosen as the ego vehicle and the others are classified as opponent vehicles with different reasoning levels. Following [9], we define the roles of AVs and potential traffic conflicts. Each vehicle is seen as an independent decision-maker and can be divided into four categories based on their potential conflicts: Host vehicles (HV), leading vehicle (LV), interactive vehicle (IV), and other vehicles (OV). At an unsignalized intersection, there are three types of driving conflicts determined by the moving trajectories and speeds of AVs.

- **Following Conflict** arises when a HV is traveling on the same path as a LV at a higher speed.
- **Confluence Conflict** occurs when two vehicles traveling on different paths merge into the same lane. If an IV passes the collision point before HV, the Confluence Conflict becomes a Following Conflict.
- **Cross Conflict** happens when two vehicles traveling on different paths with an intersection point are heading in different directions

The collision points highlighted in Fig. 1 demonstrate traffic conflicts faced by AVs at intersections. Interactions among AVs are depicted through a switching directed graph, which will be further explained later. To define the Laplacian matrix of the graph, we refer to the AV experiencing Confluence Conflicts or Cross Conflicts with the HV as an IV. Vehicles that do not impact the actions taken by HVs are considered as OVs. The designations of LV, IV, and OV are subject to change according to the traffic states of AVs in real time.

To resolve the traffic conflicts of AVs at unsignalized intersections, we propose a scalable adaptive game-theoretic decision-making framework, which consists of two modules, i.e., modeling, and decision-making, as illustrated in Fig. 2. First, the interaction topology of HV is established according to the traffic states obtained from perception system. Since the driving aggressiveness of IV has significant effects on their driving behaviors, HV must account for the driver type of IV during the decision-making process. Specifically, HV uses the kinematic model to predict the future actions of IV based on the level-k game theory following a receding horizon strategy to find its own optimal actions. Then, HV's belief on the driver type of IV is updated by comparing the actual actions taken by IV and corresponding predictions made by HV. Finally, the generated speed command will be executed by the controllers. In this work, PID and pure pursuit controllers are used to achieve the longitudinal and lateral control, respectively.

III. VEHICLE DYNAMIC MODEL

Kinematic bicycle model is commonly used to design decision-making algorithms for AVs [17] denoted by Eq. (1):

$$x(t+1) = x(t) + v(t)\cos(\psi(t)+\beta(t))\Delta t + u_x(t) \tag{1a}$$

$$y(t+1) = y(t) + v(t)\sin(\psi(t)+\beta(t))\Delta t + u_y(t) \tag{1b}$$

$$\psi(t+1) = \psi(t) + \frac{v(t)}{l_r}\sin(\beta(t))\Delta t \tag{1c}$$

$$v(t+1) = v(t) + a(t)\Delta t \tag{1d}$$

$$\beta(t) = \arctan\left(\frac{l_r}{l_r + l_f}\tan\left(\delta(t)\right)\right) \tag{1e}$$

$$\delta(t) = \arctan\left(\frac{2L\sin\alpha(t)}{ld}\right) \tag{1f}$$

The pair of $(x(t), y(t))$ represents the center of gravity's coordinate position at time t, while $v(t)$, $\psi(t)$, and $\beta(t)$ denote the longitudinal speed, yaw angle, and angle of the speed relative to the vehicle's longitudinal axis, respectively. $a(t)$ represents the longitudinal acceleration. $\delta(t)$ denotes the steering angle. The distances of vehicle's center of mass to the front and rear axles are denoted by l_f and l_r. To account for uncertainties resulting from a simplified model and unknown driving models, we use $u_x(t)$ and $u_y(t)$ to represent the position uncertainties in both longitudinal and lateral directions. Δt represents the sampling period. Since we focused on optimizing acceleration, control of lateral movements have delegated to the pure pursuit controller by incorporating Eq. (1f). The look-ahead distance, denoted as ld, represents the distance between the vehicle's rear axles and a specified point on a desired path, which is calculated using the parameter $k_v v(t)$. The angle between the vehicle's body heading and the look-ahead line, as defined by the center of the rear axles and the target point, is known as α.

IV. SCALABLE DECISION-MAKING ALGORITHM WITH LEVEL-K THEORY

A. Graph Theory Notions

The interactions among vehicles in the intersections are denoted by switching directed graphs. Let $\{\mathcal{G}_k = (\mathcal{V}, \mathcal{E}_k, \mathcal{W}_k = [w_{ij}^k]) \mid k \in \mathcal{P}\}$ be the set of all possible graphs with $\mathcal{P} = \{1, \cdots, Q\}$ where $Q > 1$ is an integer. $\mathcal{V} = \{Car1, Ego, Car2, \cdots, CarN\}$ is a set of $N+1$ nodes in $\mathcal{G}_k$ and $\mathcal{E}_k \subset \mathcal{V} \times \mathcal{V}$ represents the set of edges. $\mathcal{W}_k = [w_{ij}^k]$ denotes the weighted adjacency matrix, where w_{ij}^k is the weight of the directed edge (j, i) and $w_{ij}^k > 0$ if $(j, i) \in \mathcal{E}_k$; $w_{ij}^k = 0$ otherwise. Let $w_{ii}^k \equiv 0, \forall i \in \mathcal{V}$. The Laplacian matrix of $\mathcal{G}_k$ is defined as $\mathbf{L}_k = \operatorname{diag}\{\Delta_1^k, \cdots, \Delta_N^k\} - \mathcal{W}_k$, where $\Delta_i^k = \sum_{j=1}^N w_{ij}^k$ is the in-degree of node $i, i = 1, \cdots, N$ [2]. Given any graph $\mathcal{G}$, $\mathcal{V}(\mathcal{G}), \mathcal{E}(\mathcal{G})$, and $\mathbf{L}(\mathcal{G})$ are represented its node set, edge set, and Laplacian matrix, respectively [18], [19].

The topology of the interaction graph is updated in real time according to the traffic conflicts defined in section II. A sequence of waypoints representing the future path of HV at time t intersects with that of another vehicle leading to a Cross Conflict or a Confluence Conflict, such vehicle is classified as an IV. The length of the future path of vehicle l is defined by

$$\mathcal{L}\left(p_t[l]\right) = \sum_{j=0}^{\hat{N}-1} |\hat{p}_{t+j+1} - \hat{p}_{t+j}|, \tag{2}$$

where $\hat{p}_{t+j}$ represents the coordinates of the next jth waypoint starting from the center point of its rear axle. $\hat{N}$ is the total number of waypoints in its future path, $p_t[l]$, of vehicle l.

Then the edge between the HV and IV will be denoted by a double arrow, which represents an interaction among them. For the Following Conflict, the edge between HV and the other vehicle will be denoted by a single arrow pointing to the HV, which represents a collision avoidance task for HV and no interaction among them. Taking Fig. 1 as an example, in traditional level-k based decision-making algorithms, the interaction topology between vehicles can be represented by Fig. 3(a), which could limit the scalability and real-time implementation of algorithms due to its strongly connected property. However, some vehicles that do not affect the actions taken by HV should be removed from the interaction graph for reducing computational burden purposes. Switching directed graph can help to effectively simplify the interactions between vehicles as shown in Fig. 3(b) with solid edges. Therefore, the ego only needs to account for Car 1 instead of all of them in this case.

Remark 1: Set $\hat{D}[l]$ represents the group of vehicles for which HV needs to account for their potential actions when making decisions, while set $\hat{O}[l]$ includes the vehicles that pose collision risk or are in a Following Conflict situation with HV.

B. Action Set and Running Reward for Decision-Making

To resolve driving conflicts, we assume that a vehicle has a finite set of acceleration levels to choose to adjust its speeds along the desired path at each time step, i.e., $a(t) \in A = \{a^1, \cdots, a^{\mathcal{M}}\}, \forall t$. The acceleration to be applied to the vehicle at each step is decided according to the optimization of a reward function described as follows.

The cumulative reward is given by

$$\mathcal{R}\left(\gamma_t\right) = \sum_{j=0}^{N-1} \lambda^j R_{t+j}. \tag{3}$$

An optimal action sequence of HV with prediction horizon N, $\gamma_t^* = \{a_t^*, a_{t+1}^*, \cdots, a_{t+N-1}^*\}$, can be obtained by maximizing above cumulative reward given by Eq. (3) following a receding horizon strategy. Since the HV only executes the first element of the action sequence at each time step, and repeats for every control cycle, it provides a certain degree of inherent robustness to uncertainties because of the feedback loop [20].

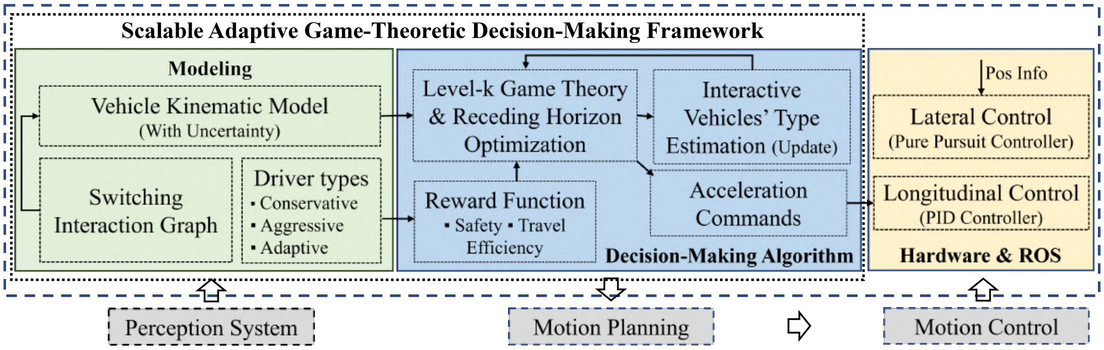

Scalable Adaptive Game-Theoretic Decision-Making Framework
Modeling
Vehicle Kinematic Model
(With Uncertainty)
Switching Interaction Graph
Driver types
▪ Conservative
▪ Aggressive
▪ Adaptive
Level-k Game Theory & Receding Horizon Optimization
Interactive Vehicles' Type Estimation (Update)
Reward Function
▪ Safety ▪ Travel Efficiency
Acceleration Commands
Decision-Making Algorithm
Pos Info
Lateral Control
(Pure Pursuit Controller)
Longitudinal Control
(PID Controller)
Hardware & ROS
Perception System
Motion Planning
Motion Control

Fig. 2: Game-Theoretic Decision-Making Framework

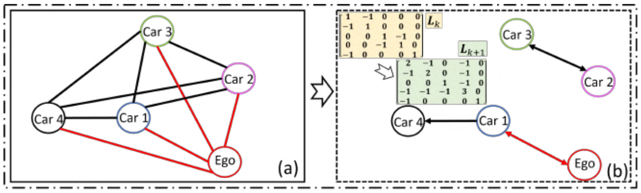


Fig. 3: Interaction Graph: (a) Undirected Graph (Strongly connected); (b) Directed Graph

The stage reward is defined as:

$$R_{t+j} = w_1\phi_{t+j}^{(1)} + w_2\phi_{t+j}^{(2)} + w_3\phi_{t+j}^{(3)} + w_4\phi_{t+j}^{(4)} + w_5\phi_{t+j}^{(5)} \quad (4)$$

where $\phi_{t+j}^{(k)}$ is k^{th} indicator variable for a specific driving feature at prediction step j. $\omega_k > 0$ is the weight of the corresponding factor. More features could be added to Eq. (4) for more driving preferences of HV.

There are four approximations of vehicle perceptions, i.e., Collision zone (C_z), Safe zone (S_z), Uncertainty zone (U_z), and the length of future path ($\mathcal{L}(p)$), which will be used to define the reward function. C_z is red dashed rectangular with the length of l_v m and the width of w_v m; the S_z is yellow dashed rectangular with the length of (l_v+2s_x) m and the width of (w_v+2s_y) m; U_z is blue dashed rectangular with the length of (l_v+2s_x+2u_x) m and the width of (w_v+2s_y+2u_y) m, in which $s_x, s_y, u_x, u_y \geq 0$, as shown in Fig. 1.

According to the observations defined above, driving features of the reward function are characterized by:

- **Interaction Status** determined by the Cross/Confluence Conflict between HV and IV. If intersection of their future paths is detected $\phi_t^{(1)} = J_{ttc}$; and 0 otherwise.

$$\phi_t^{(1)} = \sum_{i=1}^{n} \begin{cases} -J_{ttc}, & p_t[l] \cap p_t[i], \forall i \in \hat{D}[l] \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

$$J_{ttc} = 1/\left(\left(\Delta T_{(i,l)}\right)^2 + \varepsilon\right); \Delta T_{(i,l)} = T_{\hat{c}}[l] - T_{\hat{c}}[i]$$
$$T_{\hat{c}}[l] = \Delta s^{\hat{c}}[l]/v_l; T_{\hat{c}}[i] = \Delta s^{\hat{c}}[i]/v_i \quad (6)$$

where $p_t[i]$ represents the sequence of waypoints of the ith vehicle at time t. And the HV is denoted by l. We assume that the cross point between future paths is $\hat{c}$. The time for vehicle l and vehicle i to reach the cross point $\hat{c}$ at current velocity from their current positions can be expressed as $T_{\hat{c}}[l]$ and $T_{\hat{c}}[i]$, respectively. The closer $\Delta T_{(i,l)}$ is to zero, the greater the risk of collision between them.

- **Collision status**: If an overlap, representing a vehicle collision, between the C_z of HV and that of any other cars is detected then $\phi_t^{(2)}$ = -1; and 0 otherwise.

$$\phi_t^{(2)} = \sum_{i=1}^{n} \begin{cases} -1, & \mathcal{C}_t[l] \cap \mathcal{C}_t[i], \forall i \in \hat{O}[l] \\ 0, & \text{otherwise} \end{cases} \quad (7)$$

- **Safe zone violation status**: If an overlap between the S_z of HV and that of any other cars is detected then $\phi_t^{(3)}$ = -1; and 0 otherwise.

$$\phi_t^{(3)} = \sum_{i=1}^{n} \begin{cases} -1, & \mathcal{S}_t[l] \cap \mathcal{S}_t[i], \forall i \in \hat{O}[l] \\ 0, & \text{otherwise} \end{cases} \quad (8)$$

- **Uncertainty zone violation status**: If an overlap between the U_z of HV and that of any other cars is detected then $\phi_t^{(4)}$ = -1; and 0 otherwise.

$$\phi_t^{(4)} = \sum_{i=1}^{n} \begin{cases} -1, & \mathcal{U}_t[l] \cap \mathcal{U}_t[i], \forall i \in \hat{O}[l] \\ 0, & \text{otherwise} \end{cases} \quad (9)$$

- **Travel efficiency**: $\phi_t^{(5)}$ is to encourage vehicles to pass the intersection efficiently, which is described by

$$\phi_t^{(5)} = -\left|v_t - v^{\text{ref}}\right| \quad (10)$$

where reference speed v^{ref} is typically chosen as the legislated speed limit of the traffic scenario.

The calibration of model-based controllers is challenging [17], however, intuitively, we prioritize driving safely over travel efficiency when tuning the weights of these factors in the reward function. Therefore, the tuning parameters are chosen as,

$$w_2 > w_3, w_4 > w_1 > w_5 \quad (11)$$

C. Robust Adaptive Game-Theoretic Decision-Making

1) Level-k Decision-Making: To model multi-vehicle interactions, all driving features except $\phi^{(5)}$ in the stage reward function (4) are utilized to reveal the interactive behaviors, depending on the states of vehicles. A sequence of action of a HV is represented by $\gamma_t[l]$. The action sequence of i^{th} IVs with reasoning deepth 'k' predicted by HV is denoted by $\gamma_t^{(k)}[i]$. These actions are used to calculate the cumulated reward in Eq.

(4) according to the corresponding traffic states at prediction steps $j \in \mathbb{Z}_{[0,N-1]}$, denoted by

$$\begin{aligned} s_{t+j} =&[x_{t+j}[1], y_{t+j}[1], \psi_{t+j}[1], v_{t+j}[1], \\ &\cdots, x_{t+j}[n], y_{t+j}[n], \psi_{t+j}[n], v_{t+j}[n]]^T . \end{aligned} \tag{12}$$

Remark 2: It should be noted that s_{t+j} is the state information of selected vehicles, represented by the $\mathbf{L}_{\sigma(t)}$ of interaction graph in the set of $\mathcal{O}[l] = \hat{D}[l] \oplus \hat{O}[l]$, that affect the action of HV instead of accounting for interactions among all vehicles in the unsignalized intersections.

Inspired by skilled human drivers (HDs), they could navigate complex unsignalized intersections effectively due to: (1) driving difficulty being dependent on the number of interacting vehicles rather than the total number present; (2) prioritization of driving tasks based on conflict types with surrounding vehicles; and (3) personalized driving preferences that experienced HDs utilize to estimate and predict others' behaviors, leading to optimal actions for safety and efficiency.

The level-k theory adopted in this paper exactly works in the way mentioned above. Specifically, level-'k' represents the reasoning level of decision-makers starting from non-strategic policy, level-0, that usually takes action to achieve its goal without accounting for the interactions between itself and other agents. However, decision-makers above this level of reasoning will assume that all the other agents are level-(k-1). They will predict the future actions taken by IV based on such an assumption and take their own optimal actions accordingly.

In this paper, level-0 HV, representing an aggressive driver, mainly behaves as a collision avoidance strategy in static environments with a shorter length of future path and zero uncertainty to others. A level-1 HV will assume that all IVs are drivers with level-0 reasoning, therefore, responds to them cautiously. The length of its future path and the size of the uncertainty zone around IVs will always be the maximum values from the perspective of HV. Similarly, more higher levels can be defined. In this work, only level-0 and level-1 drivers are considered due to the similarity between level-0 and level-2 [11], [17]. However, this algorithm can be extended to higher levels at the expense of increased computational complexity. Once the level-0 is defined, the action calculation for finding level-k action, $k \geq 1$, in the case of N-vehicle interactions is represented by the following form:

$$\begin{aligned} R_{t+j}^{(k)}[l] = R_{t+j}\Big(&\gamma_{t+j}^{(k)}[l] \mid s_0, \gamma_t^{(k)}[l], \gamma_{t+1}^{(k)}[l], \cdots, \gamma_{t+j-1}^{(k)}[l], \\ &\gamma_t^{(k-1)}[i], \gamma_{t+1}^{(k-1)}[i], \cdots, \gamma_{t+j-1}^{(k-1)}[i]\Big)\; i \in \hat{D}[l], \end{aligned} \tag{13}$$

and its cumulative reward is

$$\mathcal{R}^{(k)}\left(\gamma_t^{(k)}[l]\right) = \sum_{j=0}^{N-1} \lambda^j R_{t+j}^{(k)}[l] \tag{14}$$

2) Robust Adaptive Decison-Making: The assumption that level-k HV always interacts with IVs with level-(k-1) reasoning level is unrealistic. Mismatch between the (k-1) assumption and actual driver type may lead to unsafe action selection and reduce the driving safety. Intuitively, the interactions of level-k versus level-k could lead to unexpected driving behavior, i.e., congestion or collision [12]. Inspired by HDs, HV should also be capable to assess the driver type of each IV via interactions. The *trust* (or belief) of HV on the driving model of each IV should also be updated in real time by comparing the actual action applied by each IV, $\gamma[i](t)$, and corresponding predictions for a level-k driver, $\gamma_t^{(k)}[i]$, made by HV, which can be represented as a continuous parameter between level-0 and level-1. The HV's *trust*, $T_{HV}^{K=k^*}[i](t)$, can be represented by a probability that the i-th IVs can be modeled as model k driver, given by

$$k^* = \arg\min_{k\in\{0,1\}} \left\| \gamma[i](t) - \gamma_t^{(k)}[i] \right\| \tag{15a}$$

$$\tilde{T}_{HV}^{(K=k^*)}[i]_l(t) = T_{HV}^{(K=k^*)}[i](t-1) + \Delta P \tag{15b}$$

$$T_{HV}^{(K=k)}[i](t) = \frac{\tilde{T}_{HV}^{(K=k)}[i](t)}{\sum_{k'=0}^{1} \tilde{T}_{HV}^{(K=k')}[i](t)}, \forall k \in \{0,1\}, \tag{15c}$$

where the rate of increment of the *trust* is denoted by ΔP, which is a positive constant. Since there may exist some scenarios in which the actions taken by drivers are the same regardless of the driver types, the probability of each driver type remains the same. Otherwise, the HV's *trust* in driver model k^* that matches the actual action best increases by ΔP. Then, the probability distribution is normalized by Eq. (15c). An algorithm flowchart is provided to help illustrate the decision-making process of above level-0, level-1, and adaptive policies, see Fig. 4.

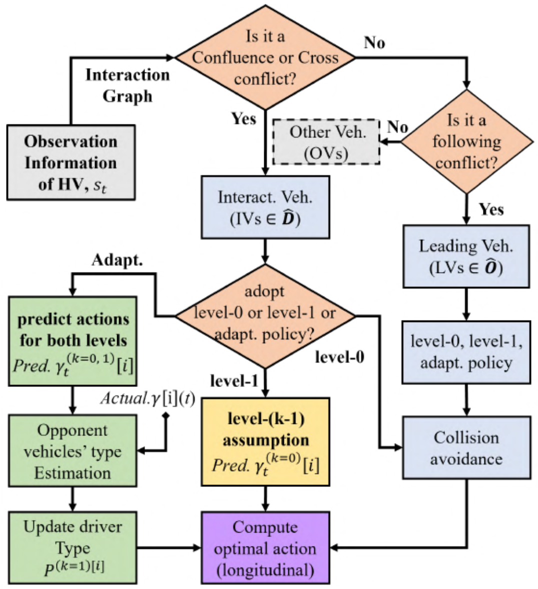

Observation Information of HV, s_t
Interaction Graph
Is it a Confluence or Cross conflict?
No
Yes
Other Veh. (OVs)
No
Is it a following conflict?
Yes
Interact. Veh. (IVs ∈ $\hat{D}$)
Leading Veh. (LVs ∈ $\hat{O}$)
Adapt.
adopt level-0 or level-1 or adapt. policy?
level-0
level-1
predict actions for both levels Pred. $\gamma_t^{(k=0,1)}[i]$
level-0, level-1, adapt. policy
Actual.γ[i](t)
Opponent vehicles' type Estimation
level-(k-1) assumption Pred. $\gamma_t^{(k=0)}[i]$
Collision avoidance
Update driver Type $p^{(k=1)[i]}$
Compute optimal action (longitudinal)

Fig. 4: Algorithm Flowchart

The adaptive decision-making approach proposed in this paper is based on the multi-model strategy, it finds an optimal action sequence for HV at each time step according to the estimated driver model of each IV. The expected accumulated reward of an action sequence is calculated by

$$\mathcal{R}_P\left(\gamma_t[l]\right) = \sum_{k=0}^{1} T_{HV}^{(K=k)}[i](t)\mathcal{R}^{(k)}\left(\gamma_t^{(k)}[l]\right), \forall i \in \mathcal{D}[l] \tag{16}$$

To improve the robustness of the algorithm, two sources of modeling errors are considered during the decision-making process of HV. Specifically, a simplified kinematic model could introduce the model uncertainty to the decision-making process, which is denoted by $u^{(m)} = (u_x^{(m)}, u_y^{(m)}) \in U_m$, and unknown driver type could lead to an interaction uncertainty denoted by $u^{(d)} = (u_x^{(d)}, u_y^{(d)}) \in U_d$. To deal with the uncertainties from the simplified kinematic model, the safe margin of the safety zone is used to compensate for the model uncertainty, $u^{(m)}$, which is a mismatch in the position between the actual position of IV and the corresponding prediction by HV, the values of which will not be changed all the time. The uncertainty zone of a vehicle is also a rectangle area that subsumes the safe zone of the car with margins on both longitudinal sides and lateral sides to handle the interaction uncertainty. To realize the adaptive scheme, the size of it is modified according to the HV's trust in the driver type of each IV. The uncertainty set could be denoted by

$$\mathcal{U}_{HV}[i](t) = \mathcal{U}_m \oplus T_{HV}^{(K=0)}[i](t)\mathcal{U}_d \tag{17}$$

To ensure safety, the probability that the driver type of IV can be modeled as level-0 is set to 1 initially in Eq.(15), which is reasonable since HV should behave cautiously when it does not have too much interaction data with IV at the beginning.

In addition, unlike the highway scene where cars are only moving in the longitudinal direction, HV should be capable to resolve lateral conflicts including Cross Conflict and Confluence Conflict in unsignalized intersections, therefore, the length of its future path should be modified according to the value of Trust in the model of each IV as well, given by

$$\hat{\mathcal{L}}\left(p_t[l]\right) = \mathcal{L}\left(p_t[l]\right) T_{HV}^{(K=0)}[i](t) \tag{18}$$

The driving feature about interaction conflict status will be calculated based on the length of it. To find the optimal action sequence, $\hat{\gamma}[l]$, it essentially solves an optimization problem by maximizing the expected cumulative reward 16, given by

$$\begin{aligned} \hat{\gamma}_t[l] = \arg \max_{\gamma_t[l] \in \mathcal{A}} \min_{u_{t+j} \in \mathcal{U}_t[i](t)} \mathcal{R}_P\left(\gamma_t[l]\right) \\ \text{s. t. } \forall j \in \mathbb{Z}_{0,N-1}, [1a-1f], \forall i \in \mathcal{O} \end{aligned} \tag{19}$$

A decision tree approach is used for searching an optimal action sequence at each time step by enumerating all possible combinations of discrete actions. Then, the first element of $\hat{\gamma}_t[l]$ is applied to the vehicle, and the cycle continues.

V. EXPERIMENTAL VALIDATION

To evaluate the performance of our algorithm in terms of adaptability, computational complexity, real-time performance, and scalability, we analyze test results on both hardware platform and high-fidelity simulator, including switching interaction graph, travel efficiency, drivers' type estimation, computational load and its comparison with a traditional method.

A. Performance of Adaptive Policy on Hardware

To experimentally validate the effectiveness of the scalable adaptive game-theoretic decision-making algorithm (Adpt.), four AVs in indoor environment unsignalized intersection are utilized, see Fig. 5. Scaled model vehicles receive control commands via Wi-Fi from a workstation with Intel(R) Core(TM) i7-7700 CPU. An OptiTrack system consisting of 16 Flex 13 cameras is used to capture motion states of all cars and feed them back to the workstation via USB cables.

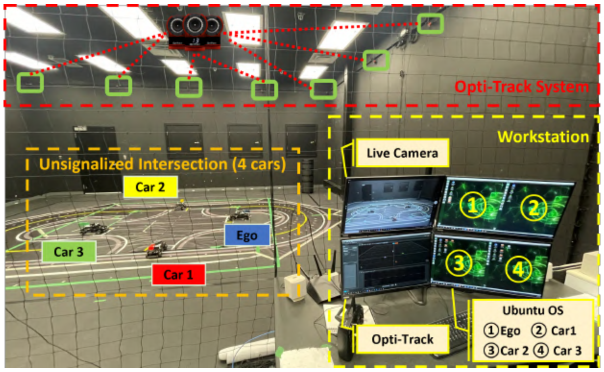


Fig. 5: Hardware Platform

TABLE I: Policy Setting at Unprotected Left Turn Scenarios

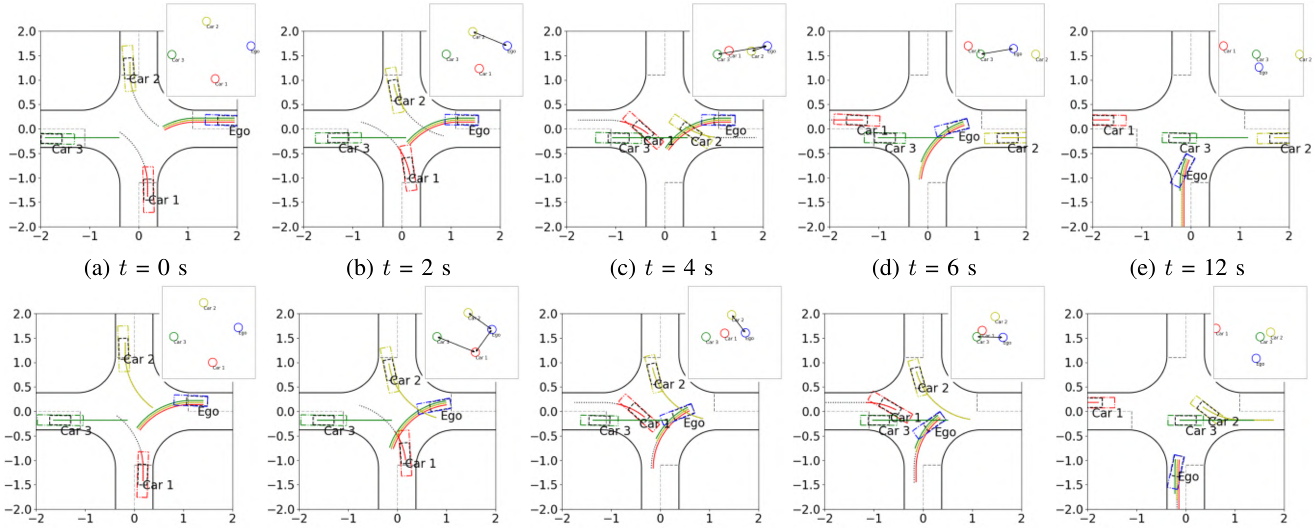
Case	Car1	Ego	Car2	Car3	Car4	Car5	Car6
1	L0(l)	Adpt.(l)	L0(l)	L1(s)	∅	∅	∅
2	L0(l)	Adpt.(l)	L1(l)	L1(s)	∅	∅	∅
3	L0(l)	Adpt.(l)	L1(l)	L1(s)	L0(l)	L1(r)	∅
4	L0(l)	Adpt.(l)	L1(l)	L0(r)	L0(l)	L1(r)	L1(s)

Note: L0: level0; L1: level1; Turn left (l), right (r); straight (s).

Due to space limitations, unprotected left turn is chosen as the primary testing scenario since it is the most challenging case among intersection-related problems [21], where all vehicles navigate intersections based on their local observation. In each test, different driving strategies were assigned to vehicles as shown in Table I, where the level-0 policy can be regarded as an aggressive driver (Aggr.) while level-1 is a conservative driver (Consrv.). These driving policies are unknown to each other. Ego adopts the proposed adaptive driving policy that allows it to navigate through an unsignalized intersection where surrounding vehicles exhibit varying degrees of aggressive driving behavior safely and efficiently via interaction. [1].

The decision-making result of Case 1 is shown in the first row of Fig. 6, where the dotted line in grey with fixed length indicates the future path of each vehicle, which is used to define the driving conflicts, i.e., Cross, Confluence, and Following conflicts. Solid lines in different colors in front of all vehicles except ego can be regarded as the ground truth of the driver type of each vehicle, which is used to calculate stage rewards at each time step. The length of which will not be changed until the next round of tests. The length of the line segment represents the aggressiveness of a driver. The more conservative the driving behavior, the longer the line segment. The maximum length is equal to that of the gray dotted line. For ego car, there are multiple line segments in front of it. And the number of lines depends on the number of IVs. The bounding box of each vehicle in different colors represents

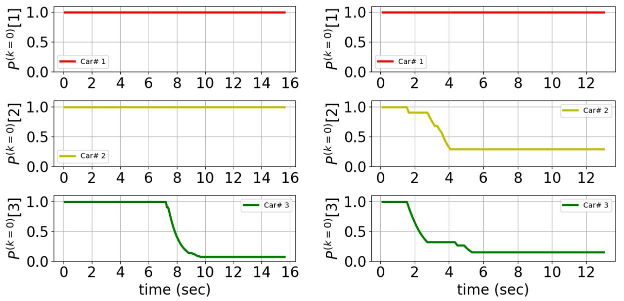
[1] More testing scenarios, algorithm parameters, and experimental settings can be found at https://youtu.be/q6vKrjqHD54



(a) t = 0 s (b) t = 2 s (c) t = 4 s (d) t = 6 s (e) t = 12 s

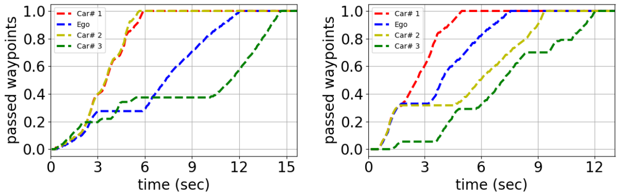
(f) t = 0 s (g) t = 1 s (h) t = 3 s (i) t = 4 s (j) t = 7 s

Fig. 6: Adaptive Decision-Making Results of Ego Vehicle in Unprotected Left Turn Scenarios (a-e) Aggress: car 1, car 2; Consrv.: car 3; (f-j) Aggress: car 1; Consrv.: car 2, car 3



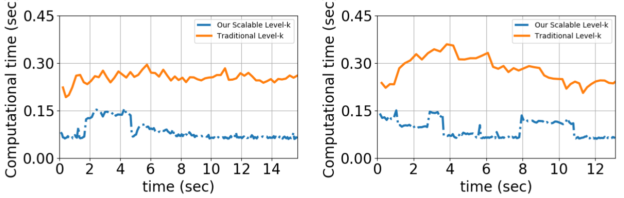
(a) 2 Aggr. & 1 Consrv. Drivers (b) 1 Aggr. & 2 Consrv. Drivers

Fig. 7: Driver Models Identification History of Ego Vehicle



(a) 2 Aggr. & 1 Consrv. Drivers (b) 1 Aggr. & 2 Consrv. Drivers

Fig. 8: Travel Efficiency of Four Vehicles



(a) 2 Aggr. & 1 Consrv. drivers (b) 1 Aggr. & 2 Consrv. drivers

Fig. 9: Computational Time

the uncertainty zone from the perspective of ego. Both the length of each line segment in front of the ego car and the size of the bounding box of each IV have a linear dependence relation with the probability that a specific car can be modeled as level-0 driver. Intuitively, the higher the confidence that a certain vehicle can be modeled as an aggressive driver, the larger the reserved safety interaction space would be. In the beginning, the length of all line segments of ego and the size of the bounding boxes of IVs are the maximum. This is because ego initially assumes that the driving types of surrounding vehicles are all level-0. Ego would behave cautiously due to the lack of interactive data. From 0 s to 4 s, car 1 and car 2 start to accelerate and choose to pass through the intersection first while car 3 and ego choose to yield the right of the way to them. At t = 6 s, ego car is interacting with car 3. The length of the green line of ego indicating the probability that car 3 can be modeled as level-0 starts reducing. This is because car 3 chooses to continue to yield the right of way to the ego, therefore, car 3 finally is regarded as a level-1 driver after interactions instead of level-0. However, the length of the solid red line and yellow line in front of ego car does not change during the interactions, which means that the driver types of car 1 and car 2 identified by ego are level-0. The driver model identification history of ego car in case 1 can be also found in Fig. 7a.

The travel efficiency of vehicles at the intersection can be represented by Fig. 8a, in which the number of waypoints for each path is normalized. The y-axis represents the completion progress of vehicles on their path, and the time corresponding to the progress of 1 can be used to indicate the order in which each vehicle exits the intersection. In Case 1, aggressive vehicles, car 1 and car 2, are the first to pass through the intersection, followed by ego vehicle. And the conservative vehicle of car 3 is the last to exit the intersection.

In Case 2, level-0 policy is assigned to car 1 while level-1 policy is assigned to car 2 and car 3. According to the second row of Fig. 6, all cars choose to stop and let car 1 pass the intersection before t = 3 s. Then car 2 and car 3 continue to wait until ego passed the potential collision points. After that car 2 choose to move since it is closer to the exit point of the intersection. Two level-1 drivers are successfully identified by ego according to Fig. 7b. The order of exiting the intersection is car 1, ego, car 2, and car 3, respectively as shown in Fig. 8b. Due to space limitations, more test scenarios and model parameter settings could be found in the video.

The computational load mainly comes from solving the optimal actions for the HV following receding horizon optimization while considering all possible driving behaviors of the IVs, which is mainly affected by the number of IVs. The switching interaction graphs of the ego vehicle are generated according to its local observation and corresponding trust as shown in Fig. 6. In such four-vehicle intersection scenarios, the proposed algorithm effectively simplifies the interaction relationship between the ego car and surrounding vehicles, thereby reducing the computational cost of game-theoretic decision-making algorithm. Compared with the traditional level-k decision-making algorithm [17], the computational time of our algorithm is reduced by around 50% on average as shown in Fig. 9. The advantage of the proposed algorithm in reducing computational complexity is more obvious in more complex multi-vehicle scenarios, which will be discussed in the next section using our high-fidelity platform.

B. Scalability and Computational Complexity

Developing a high-fidelity simulator for verifying the AV system is of great significance for improving the driving safety of AVs and saving costs of road tests. In this work, we developed a high-fidelity testing platform for the unsignalized intersection scenarios with multiple vehicles using ROS-Gazebo, which supports dynamic simulation, sensor data acquisition, and customized traffic environments helping to narrow the gap between the simulation and the real world, see Fig. 10. The Gazebo environment is developed on Ubuntu 18.04 workstation with Intel Xeon W-1290P CPU.

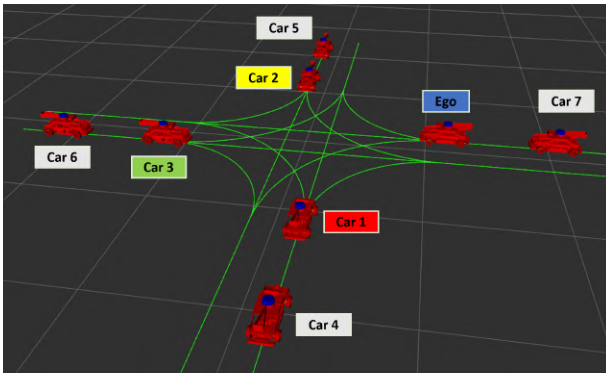
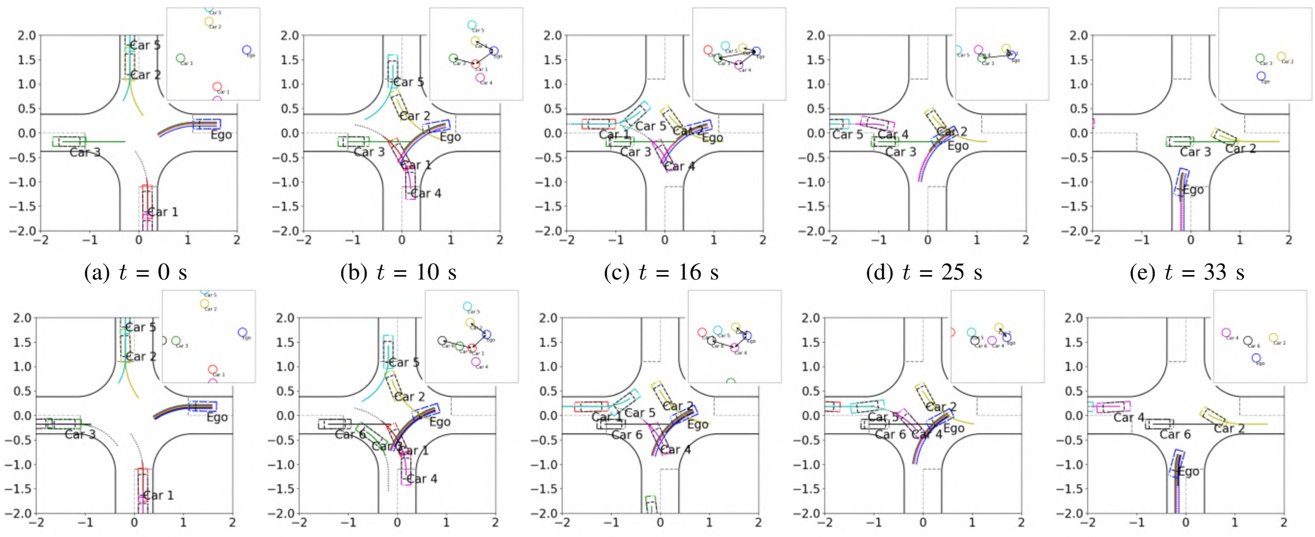


Fig. 10: Simulation Environment

To validate the scalability and computational efficiency of the proposed algorithm, more complex scenarios are considered in this section, where 6 or 7 vehicles are navigating the intersection at the same time with different combinations of aggressiveness as shown in Table I. Case 3 is a six-car scenario, where the ego car adopts an adaptive policy while the rest vehicles use the level-1 strategy except car 1 and car 4 using level-0. In Case 4, an additional car 6 is added behind car 3 and goes straight with a level-1 driving strategy. The other vehicles are basically the same as Case 3 except the car 3 turning right with a level-0 strategy. According to Fig. 11, the ego car passed the intersections successfully without any collisions with other cars. The orders of exiting the intersections for these two cases are shown in Fig. 12. Similar to the previous four-car scenarios, vehicles with aggressive policy pass through the intersection before vehicles with conservative policy. And ego car adjusted its driving strategy adaptively based on the aggressiveness of IVs while guaranteeing both traffic efficiency and driving safety. According to Fig. 13, ego has successfully identified that level-1 policy is assigned to car 2 and car 3 in Case 3 and assigned to car 2 and car 6 in Case 4. It should be noted that although car 5 adopts a level-1 policy in both scenarios, ego car does not update its trust level for car 5 as a level-0 driver. This is because car 5 turns right in both cases, according to the switching interaction graph provided in Fig. 11, there is no interaction between the ego car and car 5. Therefore, the ego's trust level for car 5 as a level-0 driver remains at its initial value.

The reduction in computational complexity in this work is mainly attributed to the introduction of a switching interaction topology mechanism. In the following, we will use the seven-vehicle scenario to demonstrate how this algorithm can effectively reduce computation time. At t=0 s, ego's future path has no intersections with that of surrounding vehicles, resulting in no interaction behavior and allowing the algorithm to run at its fastest state, close to 0.03 s. However, at t=7 s, ego vehicle models car 1 and car 2 as interactive vehicles because their future paths intersected. Additionally, car 6 was also considered an interactive vehicle because the driving behavior of car 1 is going to be influenced by the car 6, indirectly affecting ego's decision-making. The interaction topology between ego and the surrounding vehicles can be seen in the upper right corner of each figure, see Fig. 11 (g). Therefore, the computation time rapidly increases in Fig. 14 (b), peaking at around 0.17 seconds. As the vehicles proceed, the interaction topology changes, and by t=17 s, only car 2 is affecting ego's actions, leading to a decrease in computation time. After t=27 s, ego successfully passes through the intersection, terminating all interaction behavior and returning to its fastest-running state. In contrast, for the traditional algorithm, ego needs to consider the behavior of all vehicles, resulting in a computation time of approximately 0.4 s.

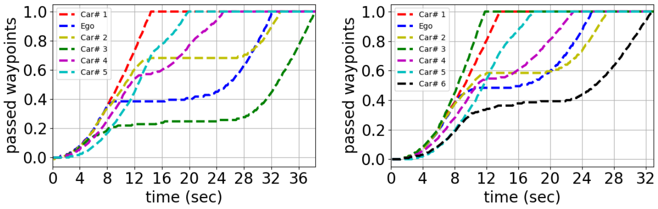
In summary, the computational time of our algorithm does not increase with the number of vehicles compared with the traditional algorithm, as shown in Fig. 14. The computational load is reduced by 55% in six-car scenario and reduced by 63% in seven-car scenario on average, where the blue dotted line represents the computational efficiency of the traditional level-k algorithm and the yellow line represents that of our algorithm. This is because the proposed algorithm can effectively cut off the relationship between ego and other vehicles that do not affect its decision-making directly, thereby improving the scalability and performance in real-time implementation. We remark that more vehicles could be added to the simulator at the expense of increasing the computational burden due to the physics calculation of the Gazebo.



(a) t = 0 s (b) t = 10 s (c) t = 16 s (d) t = 25 s (e) t = 33 s

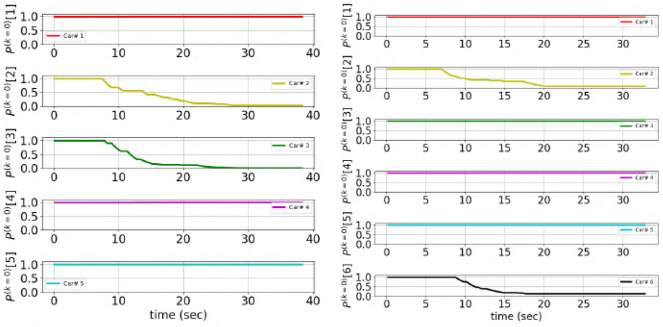
(f) t = 0 s (g) t = 7 s (h) t = 15 s (i) t = 17 s (j) t = 27 s

Fig. 11: Adaptive Decision-making Results of Ego Vehicle in Unprotected Left Turn Scenarios (a-e) Consrv.: car 2, car 3, and car 5; Aggr.: car 1 and car 4; (f-j) Consrv.: car 2, car 5, and car 6; Aggr.: car 1, car 3, and, car 4



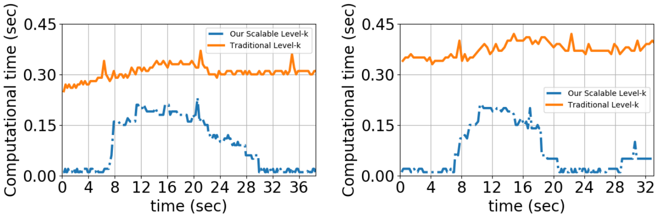
(a) Six-car Scenario (b) Seven-car Scenario

Fig. 12: Travel Efficiency



(a) 2 Aggr. & 3 Consrv. Drivers (b) 3 Aggr. & 3 Consrv. Drivers

Fig. 13: Driver Models Identification History of Ego Vehicle



(a) Six-car Scenario (b) Seven-car Scenario

Fig. 14: Computational Time

VI. CONCLUSIONS AND FUTURE WORK

Based on game-theory and interaction graph, a novel scalable decision-making framework for AVs is proposed for resolving driving conflicts at unsignalized intersections. The aggressiveness of drivers and uncertainties arising due to the simplified model are taken into account in the decision-making framework. In the payoff function design of decision-making, multiple driving features are considered including driving safety, travel efficiency, and driving aggressiveness. To reduce the inherent computational complexity of level-k game theory, the concept of switching directed graphs is incorporated into the adaptive decision-making framework. Finally, the algorithm is verified on both self-driving car hardware and a high-fidelity simulator with multiple vehicles. According to the testing results, it can be conducted that the proposed algorithm makes robust adaptive decisions for AVs, meanwhile, the performance of the algorithm in terms of interpretability, computational efficiency, and scalability can be guaranteed. Our future work will focus on the real time implementation of the proposed method in continuous action space. Game Theory-Model Predicted Control-Deep Reinforcement Learning hybrid approach could further boost the performance of proposed algorithm in computational complexity and safety.

ACKNOWLEDGMENT

The authors would like to thank Mr.Hao Wang, Dr.Junjie Kang, and Dr.Shengwen Yang for insightful discussions.

REFERENCES

[1] Z. Tian, T. Feng, H. J. Timmermans, and B. Yao, "Using autonomous vehicles or shared cars? results of a stated choice experiment," *Transp. Res. Part C Emerg. Technol.*, vol. 128, p. 103117, 2021.

[2] G. Li, Y. Yang *et al.*, "Risk assessment based collision avoidance decision-making for autonomous vehicles in multi-scenarios," *Transp. Res. Part C Emerg. Technol.*, vol. 122, p. 102820, 2021.

[3] G. Li *et al.*, "Deep learning approaches on pedestrian detection in hazy weather," *IEEE Trans. Ind. Electron.*, vol. 67, no. 10, p. 8889, 2019.

[4] H. Wang, Y. Huang, A. Khajepour, Y. Zhang, Y. Rasekhipour, and D. Cao, "Crash mitigation in motion planning for autonomous vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 9, pp. 3313–3323, 2019.

[5] M. Bosnak and I. Skrjanc, "Efficient time-to-collision estimation for a braking supervision system with lidar," in *2017 3rd IEEE International Conference on Cybernetics (CYBCONF)*, DOI 10.1109/CYBConf.2017.7985775, pp. 1–6, 2017.

[6] S. Noh, "Decision-making framework for autonomous driving at road intersections: Safeguarding against collision, overly conservative behavior, and violation vehicles," *IEEE Trans. Ind. Electron.*, vol. 66, no. 4, pp. 3275–3286, 2018.

[7] P. Hang, C. Lv, Y. Xing, C. Huang *et al.*, "Human-like decision making for autonomous driving: A noncooperative game theoretic approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 4, pp. 2076–2087, 2020.

[8] H. Qi *et al.*, "Online inference of lane changing events for connected and automated vehicle applications with analytical logistic diffusion stochastic differential equation," *Transp. Res. Part C Emerg. Technol.*, vol. 144, p. 103874, 2022.

[9] P. Hang, C. Huang, Z. Hu, and C. Lv, "Driving conflict resolution of autonomous vehicles at unsignalized intersections: A differential game approach," *IEEE/ASME Trans. Mechatron.*, vol. 27, no. 6, pp. 5136–5146, 2022.

[10] X. Ji, K. Yang, X. Na, C. Lv, and Y. Liu, "Shared steering torque control for lane change assistance: A stochastic game-theoretic approach," *IEEE Trans. Ind. Electron.*, vol. 66, no. 4, pp. 3093–3105, 2018.

[11] N. Li, I. Kolmanovsky, A. Girard, and Y. Yildiz, "Game theoretic modeling of vehicle interactions at unsignalized intersections and application to autonomous vehicle control," in *2018 Annual American Control Conference (ACC)*, DOI 10.23919/ACC.2018.8430842, pp. 3215–3220, 2018.

[12] M. Yuan, J. Shan, and K. Mi, "Deep reinforcement learning based game-theoretic decision-making for autonomous vehicles," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 818–825, 2021.

[13] R. Tian, S. Li *et al.*, "Adaptive game-theoretic decision making for autonomous vehicle control at roundabouts," in *2018 IEEE Conference on Decision and Control (CDC)*, DOI 10.1109/CDC.2018.8619275, pp. 321–326. IEEE, 2018.

[14] R. Tian, N. Li, I. Kolmanovsky, Y. Yildiz, and A. R. Girard, "Game-theoretic modeling of traffic in unsignalized intersection network for autonomous vehicle control verification and validation," *IEEE trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 2211–2226, 2022.

[15] G. Li, S. Li *et al.*, "Continuous decision-making for autonomous driving at intersections using deep deterministic policy gradient," *IET Intell. Transp. Syst.*, vol. 16, no. 12, pp. 1669–1681, 2022.

[16] W. Wang *et al.*, "An intelligent lane-changing behavior prediction and decision-making strategy for an autonomous vehicle," *IEEE Trans. Ind. Electron.*, vol. 69, no. 3, pp. 2927–2937, 2021.

[17] G. S. Sankar and K. Han, "Adaptive robust game-theoretic decision making strategy for autonomous vehicles in highway," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 14 484–14 493, 2020.

[18] J. Qin, Q. Ma, Yu *et al.*, "On synchronization of dynamical systems over directed switching topologies: An algebraic and geometric perspective," *IEEE Trans. Automat. Contr.*, vol. 65, no. 12, pp. 5083–5098, 2020.

[19] H. Wang and J. Shan, "Fully distributed event-triggered formation control for multiple quadrotors," *IEEE Trans. Ind. Electron.*, vol. 70, DOI 10.1109/TIE.2023.3239870, no. 12, pp. 12 566–12 575, 2023.

[20] P. O. Scokaert and D. Q. Mayne, "Min-max feedback model predictive control for constrained linear systems," *IEEE Trans. Automat. Contr.*, vol. 43, no. 8, pp. 1136–1142, 1998.

[21] Z. Huang, J. Wu, and C. Lv, "Efficient deep reinforcement learning with imitative expert priors for autonomous driving," *IEEE Trans. Neural. Netw. Learn. Syst.*, DOI 10.1109/TNNLS.2022.3142822, pp. 1–13, 2022.

Mingfeng Yuan received his M.S. (2019) in control engineering from Tianjin University of Technology, China. He is currently pursuing his Ph.D degree with the Department of Earth and Space Science and Engineering, York University, Toronto, ON, Canada. His research interests include machine learning for prediction and decision-making in automated driving, game theory, deep reinforcement learning, and nonlinear system modeling.

Jinjun Shan (SM'08) received the Ph.D. degree in Spacecraft Design from the Harbin Institute of Technology, Harbin, China, in 2002. He is currently a Full Professor of Space Engineering at York University, Toronto, Canada. His research interests include dynamics, control, and navigation of autonomous systems. Dr. Shan is a Fellow of Canadian Academy of Engineering (CAE), a Fellow of Engineering Institute of Canada (EIC) and a Fellow of American Astronautical Society (AAS). Since 2007, he has been a Professional Engineer in Ontario.

Hunter Schofield received his B.Eng (hons., 2020) and M.S. (2022) in space engineering from York University, Toronto, Canada. His research interests include autonomous vehicle perception and control, multiple object tracking, and world models for autonomous vehicle reinforcement learning.