Safety21

INNOVATING FOR SAFETY

US DOT National
University Transportation Center for Safety

Carnegie Mellon University

Penn
University of Pennsylvania

The Ohio State University

Morgan State University

CCAC Community College of Allegheny County

The University of Texas Rio Grande Valley

Avoiding Collisions in Connected and Autonomous Driving Using Safe Deep Reinforcement Learning Exploration through Control Barrier/Lyapunov Functions

Bilin Aksun-Guvenc (https://orcid.org/0000-0002-6619-2178)
Levent Guvenc (https://orcid.org/0000-0001-8823-1820)

Final Report – 07/31/2026

DISCLAIMER

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated in the interest of information exchange. The report is funded, partially or entirely, under [grant number 69A3552344811 / 69A3552348316] from the U.S. Department of Transportation's University Transportation Centers Program. The U.S. Government assumes no liability for the contents or use thereof.

Technical Report Documentation Page

<table>
<tr><td colspan="2">1. Report No.

573</td><td>2. Government Accession No.</td><td>3. Recipient’s Catalog No.</td></tr>
<tr><td colspan="3">4. Title and Subtitle

Avoiding Collisions in Connected and Autonomous Driving Using Safe Deep Reinforcement Learning Exploration through Control Barrier/Lyapunov Functions</td><td>5. Report Date

July 31, 2026

6. Performing Organization Code

N/A</td></tr>
<tr><td colspan="3">7. Author(s)

Haochong Chen (https://orcid.org/0009-0000-5461-0822)

Xincheng Cao (https://orcid.org/0009-0008-2525-9031)

Levent Guvenc (https://orcid.org/0000-0001-8823-1820)

Bilin Aksun-Guvenc (https://orcid.org/0000-0002-6619-2178)</td><td>8. Performing Organization Report No.

N/A</td></tr>
<tr><td colspan="3">9. Performing Organization Name and Address

Automated Driving Lab, The Ohio State University

1320 Kinnear Road, Columbus, OH, 43212</td><td>10. Work Unit No.

N/A

11. Contract or Grant No.

Federal Grant # 69A3552344811 / 69A3552348316</td></tr>
<tr><td colspan="3">12. Sponsoring Agency Name and Address

Safety21 University Transportation Center

Carnegie Mellon University

5000 Forbes Avenue

Pittsburgh, PA 15213</td><td>13. Type of Report and Period Covered

Final Report (July 1, 2025 – June 30, 2026)

14. Sponsoring Agency Code

USDOT</td></tr>
</table>

15. Supplementary Notes

Conducted in cooperation with the U.S. Department of Transportation, Federal Highway Administration.

16. Abstract

This research aims to address several key challenges in road traffic safety including vulnerable road users (VRUs) by improving decision-making, vehicle control, and system evaluation of autonomous driving. The main contributions of this research are summarized as follows:
(1) An HOCLF-HOCBF-QP-based low-level control strategy is proposed to simultaneously ensure accurate trajectory tracking and collision avoidance. HOBCF acts as a safety filter for deep reinforcement learning based higher level decision making and HOBCF guarantees stable response. The proposed controller improves both the safety and robustness of ego-vehicle control in complex traffic environments.
(2) A hierarchical autonomous driving framework is developed by integrating a deep reinforcement learning (DRL)-based high-level decision-making agent with the proposed HOCLF-HOCBF-QP low-level controller. The framework enables safe, smooth, and efficient navigation of ego-vehicles. In addition, different DRL algorithms including SAC-D, DDQN, etc. are systematically compared and evaluated in this project using highway driving, intersection management and VRU interaction scenarios.
(3) Hardware-in-the-Loop (HIL) and Vehicle-in-Virtual-Environment (VVE) approaches are used to evaluate autonomous driving systems under complex and high-risk traffic scenarios involving VRUs. VVE-based real-vehicle experiments are performed to demonstrate its effectiveness for realistic ADAS and AV function evaluation and validation.

17. Key Words	**18. Distribution Statement**
safety of road traffic, VRU safety, crash avoidance systems, autonomous driving, deep reinforcement learning, Control Lyapunov Function, Control Barrier Function, Vehicle-in-Virtual-Environment, real vehicle testing	No restrictions. This document is available through the National Technical Information Service, Springfield, VA 22161.

19. Security Classif. (of this report)	**20. Security Classif. (of this page)**	**21. No. of Pages**	**22. Price**
Unclassified	Unclassified	93	

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized

Contents

Chapter 1: Introduction, Overview and Dissemination and Outreach Summary

1.1 Introduction and Overview

Rapid urbanization and the continuous increase in privately owned vehicles have created many challenges for modern transportation systems like traffic congestion, road safety risks, unsafe interactions among vehicles, and unsafe interactions between vehicles and Vulnerable Road Users (VRUs) [1], [2], [3]. Although Advanced Driver Assistance Systems (ADAS) and Autonomous Vehicle (AV) driving functions provide promising solutions for reducing human-error-related crashes, ensuring safe and reliable operation in complex traffic environments still remains a major challenge. According to recent statistics in the U.S., close to 6 million motor vehicle accidents were reported in 2022 with more than 42,000 fatalities and more than 1.6 million injuries. In 2021, vulnerable road users accounted for about one-third of all traffic fatalities including 7,388 pedestrians, 966 cyclists, and 5,932 motorcyclists. In 2021, about 116,000 pedestrians, 105,000 cyclists, and 201,000 motorcyclists were treated in hospitals after collisions. Motor vehicle accidents are the second leading cause of deaths resulting from unintentional injury in the U.S. Connected and autonomous driving with carefully developed crash mitigation methods will help in the reduction of road traffic accidents and the fatalities and injuries they cause, including the ones with vulnerable road users. Based on these facts, this project focused on developing data driven collision avoidance algorithms that learn to avoid collisions with nearby vehicles and nearby vulnerable road users, building upon our approach to pedestrian and bicyclist safety in our Year 1 and Year 2 Safety 21 projects where a deep reinforcement learning safety algorithm with safe exploration was developed and evaluated using realistic hardware-in-the-loop (HIL) and vehicle-in-virtual-environment (VVE) platforms [4], [5].

The current state-of-the-art crash mitigation methods use a catalog of possible collision interaction scenarios using model-in-the-loop simulations for development and evaluation [6], [7], [8], [9], [10]. We have also used that approach partially in our previous projects by focusing on FARS cases [4], [5]. While this approach will be successful in mitigating collisions for the scenarios in the catalog being used, it is not generic in nature and will not necessarily mitigate crashes in the many other possible dangerous interactions between the road actors. The motivation for the use of the data driven approach in this project is this limitation in scope and the difficulty of accurately modeling the interaction dynamics between vehicles and between vehicles and vulnerable road users in dangerous encounters. Our deep reinforcement learning based collision avoidance algorithm learns from interactions in complex traffic environments, adapts to various traffic scenarios, and satisfies real-time performance requirements, making it a more robust and advanced solution. We have used a hybrid approach in that the trained deep reinforcement learning algorithm is only activated to work during dangerous and close interactions with other road users.

Training is also based on such interactions using traffic microsimulation to generate realistic training interactions automatically as compared to the fixed nature of the scenario driven approach.

Existing autonomous driving safety approaches often rely on predefined catalogs of possible collision interaction scenarios and design rule-based strategies to mitigate the accident [11], [12], [13]. While these methods are useful for evaluating known traffic scenarios, they may not fully capture the wide range of rare, complex, and safety-critical interactions that occur in real traffic [14], [15], [16], [17]. In addition, purely learning-based autonomous driving algorithms can adapt to complex environments but often lack explicit safety guarantees, especially during training or under previously unseen traffic scenarios. On the other hand, traditional optimization-based control methods provide stronger interpretability and constraint satisfaction but may struggle to handle high-level decision-making in highly dynamic multi-agent traffic environments [18]. Since reinforcement learning algorithms suffer from the absence of safety guarantees during the learning process [19], [20], we have integrated our deep reinforcement learning algorithm with Control Lyapunov and Control Barrier Functions to limit its training to feasible areas of interaction with other road actors within the road barriers, with realistic vehicle dynamic model constraints as compared to the oversimplified ones used in the literature. Efficacy of the proposed method is demonstrated using Hardware-in-the-Loop (HIL) simulations and with a real vehicle using VVE.

This research developed an integrated autonomous driving safety framework that combines perception, learning-based decision-making, safety-critical control, and realistic evaluation and testing. The overall objective was to improve the safety of road traffic including VRU safety by enabling autonomous vehicles to detect surrounding road actors, make safe and efficient driving decisions, execute those decisions through constraint-based control, and validate the resulting system under realistic and repeatable testing conditions. In this way, the project establishes a complete research pipeline for autonomous driving safety while combining sensing, decision-making, control, and validation. Based on this pipeline, we have systematically developed and evaluated autonomous driving technologies that can reduce the risk of road traffic incidents. Specifically, this research made the following major contributions:

(1) An HOCLF-HOCBF-QP-based low-level control strategy was developed to simultaneously ensure accurate trajectory tracking and collision avoidance. This control strategy improved both the safety and robustness of ego-vehicle control in complex road traffic environments.

(2) A hierarchical autonomous driving framework was developed by integrating a deep reinforcement learning (DRL)-based high-level decision-making agent with an HOCLF-HOCBF-QP low-level controller. The framework enables safe, smooth, and efficient navigation of ego-vehicles. In addition, different DRL algorithms including SAC-D, DDQN, etc. were systematically compared and evaluated in this project.

(3) The HIL and VVE approaches were used to evaluate autonomous driving systems under complex and high-risk traffic scenarios involving other road actors and VRUs. VVE-based real-vehicle experiments were performed to demonstrate its effectiveness for realistic ADAS and AV function evaluation and validation.

This final research progress report is organized as follows. The rest of Chapter 1 lists the dissemination and outreach activities of the project after providing an overview of the research background, motivation, objectives, and major contributions above. Chapter 2 reviews and investigates other road actor and VRU detection and trajectory prediction methods. The corresponding test results are also presented. Chapter 3 introduces a High-Order Control Lyapunov Function and High-Order Control Barrier Function Quadratic Programming (HOCLF-HOCBF-QP) based low-level path tracking controller. Chapter 4 presents a hierarchical collision avoidance framework that integrates deep reinforcement learning (DRL)-based high-level decision-making with a low-level controller. DDQN and SAC-D agents were trained and compared to evaluating their decision-making performance. Chapter 5 introduces the Vehicle-in-Virtual-Environment (VVE) approach for autonomous driving function evaluation. It further presents VVE synchronization results, V2P pedestrian motion synchronization results, and real-vehicle longitudinal and lateral testing results for model validation and controller design in VVE-based experiments. Finally, Chapter 6 summarizes the major findings and contributions of this research and discusses possible directions for future work.

1.2 Dissemination and Outreach Summary

The resulting publications of this project are listed below.

Journal Papers:

1. Chen, H., Aksun-Guvenc, B, 2025, “Hierarchical Deep Reinforcement Learning-Based Path Planning with Underlying High-Order Control Lyapunov Function—Control Barrier Function—Quadratic Programming Collision Avoidance Path Tracking Control of Lane-Changing Maneuvers for Autonomous Vehicles,” *Electronics*, Vol. 14, pp. 2776, https://doi.org/10.3390/electronics14142776.
2. Cao, X., Chen, H., Guvenc, L., Aksun-Guvenc, B., 2025, “Communication Disturbance Observer Based Delay-Tolerant Control for Autonomous Driving Systems,” *Sensors*, Vol. 25, pp. 6381, https://doi.org/10.3390/s25206381.
3. Chen, H., Aksun-Guvenc, B., 2026, “Safe Hierarchical Autonomous Driving Framework: Integrating SAC-D Planning with HOCLF-HOCBF-QP Constrained Low-Level Control,” *IEEE Open Journal of Vehicular Technology*, vol. 7, pp. 1474-1489, https://doi.org/10.1109/OJVT.2026.3691614

Conference Paper:

4. Chen, H., Cao, X., Guvenc, L., and Aksun Guvenc, B., 2026, “High Order Control Lyapunov Function - Control Barrier Function - Quadratic Programming Based Autonomous Driving Controller for Bicyclist Safety,” SAE Technical Paper 2026-01-0034, https://doi.org/10.4271/2026-01-0034.

B.S. Honor’s Thesis:

5. R. Rao, 2026, “Building a Virtual Testing Framework for Pedestrian-AV interaction," Honors Thesis, Department of Mechanical and Aerospace Engineering, The Ohio State University, Columbus, OH, USA.

The project outreach activities are listed below.

Safety 21 Faculty Meeting presentation:

6. L. Guvenc with B. Aksun-Guvenc, Avoiding Collisions in Connected and Autonomous Driving Using Safe Deep Reinforcement Learning Exploration through Control Barrier/Lyapunov Functions, Invited Talk, Safety 21 UTC Faculty Meeting, Carnegie Mellon University, Pittsburgh, Pennsylvania, December 4, 2025.

Posters:

7. Aksun-Guvenc, B., Guvenc, L., Bicyclist Safety System Using HOCLF/HOCBF Under DRL Decision Making for AV, Safety 21 Deployment Partner Consortium Symposium, November 13, 2025.

8. Chen, H., Cao, X., Zhang, F., Tian, D., Bicyclist Safety System Using HOCLF/HOCBF Under DRL Decision Making for AV, Ohio State University AI Research Summit, Poster Session and Presentation, March 4, 2026.

9. Chen, H., Cao, X., Tian, D., Automated Driving Lab Safety 21 UTC Research, Ohio State University MAE Graduate Research Showcase, October 31, 2025.

10. Zhang, F., Automated Driving Lab Safety 21 UTC Research, Ohio State University College of Engineering Welcome Back Exhibition Day, August 28, 2025.

The project also contributed to workforce development and education and to the pool of trained transportation professionals. Three doctoral students (Mechanical and Aerospace Engineering) were trained during the project. One M.S. Plan B student (Electrical and Computer Engineering) graduated in December 2025 and has already joined the Transportation Workforce. Our undergraduate B.S. Honor’s Thesis student (Mechanical and Aerospace Engineering) graduated in May 2026 and started working as a test engineer at Tesla. One M.S. Plan A student

(Electrical and Computer Engineering) will graduate in December 2026 and will continue as a doctoral student in our lab. We also recruited two new undergraduate students (Computer Science, Computer Engineering) in Spring 2026 who are currently being trained on autonomous driving technologies in our lab. The ME 8322 Vehicle System Dynamics and Control and ECE 5553 Autonomy in Vehicles courses, both taught during Spring 2026, helped with the educational outreach goals of the project. Both courses ended with final project presentations related to transportation topics. The online section of ECE 5553 that ran in parallel with the regular course was attended by three GM engineers. ECE 5553 has seen considerable revision during the Spring 2026 semester with significantly revised lecture notes supported by many examples with their Matlab m-files for more hands-on training on autonomous driving. We also informed our deployment partner and industry contacts regularly of our research results and their potential uses.

Chapter 2: Vulnerable Road User (VRU) Detection

2.1 Camera + LiDAR Based VRU Detection Literature Review

Designing robust path planning and collision avoidance functions begins with accurate perception of the surrounding traffic environment [21], [22]. For autonomous driving systems, the reliable detection of surrounding traffic and in particular the detection of vulnerable road users (VRUs), such as pedestrians and bicyclists, is especially important because their motion pattern is highly randomized and their physical protection is limited. Therefore, the perception module must accurately detect VRUs, estimate their positions and orientations, and provide useful motion information for subsequent decision-making and control. This chapter focuses on VRU detection as nearby vehicle detection methods are already widely available in the literature.

Camera-based object detection has been widely studied for VRU perception because cameras provide visual information for object classification and localization. Existing image-based detection methods can generally be divided into two categories: two-stage detectors and one-stage detectors. Two-stage methods, such as R-CNN, Fast R-CNN, Faster R-CNN, and Sparse R-CNN, first generate region proposals and then classify and refine the detected objects using convolutional neural networks [23], [24], [25], [26]. These methods usually provide high detection accuracy, but their multi-step structure can introduce considerable computational cost, limiting their real-time performance in autonomous driving applications.

To improve detection speed, one-stage detectors directly predict object categories and bounding boxes from the full image in a single network pass. YOLO is one of the most representative one-stage detection frameworks. It divides an image into grid regions and simultaneously predicts bounding boxes, object confidence scores, and class probabilities [27]. Compared with two-stage methods, YOLO-based architectures are generally more suitable for real-time autonomous driving applications because they provide faster inference while maintaining competitive detection accuracy [28].

Although cameras are effective for object recognition, they are sensitive to lighting conditions, occlusions, and depth ambiguity. LiDAR sensors can complement cameras by providing accurate three-dimensional spatial information [29], [30]. By measuring reflected laser signals, LiDAR can generate 3D point clouds of the surrounding environment and support more reliable estimation of object position, distance, and orientation. This makes LiDAR particularly useful for detecting and tracking VRUs in complex driving scenarios.

Recent studies have further improved VRU detection by combining camera images with LiDAR point clouds. Camera-LiDAR fusion methods take advantage of the semantic richness of visual images and the geometric accuracy of LiDAR measurements. For example, neural-network-based fusion frameworks have been proposed to integrate image features with LiDAR point cloud

information for 3D object detection [31]. Other studies have used LiDAR-based models such as PointPillars or intensity-aware voxel encoders to improve the detection of vehicles, pedestrians, and bicyclists in outdoor driving environments [32], [33], [34]. Overall, camera-LiDAR-based perception provides a promising solution for robust VRU detection by combining accurate classification with reliable spatial localization.

2.2 Camera + Lidar Based VRU Detection Approach

Based on the literature review presented in the previous section, camera-based object detection provides an effective approach for identifying vulnerable road users (VRUs), such as pedestrians and bicyclists. In this study, YOLOv11 is adopted as the image-based detection model because of its high detection efficiency, simple end-to-end structure, and strong performance in real-time object detection tasks [35]. Unlike two-stage detectors that rely on separate proposal generation and object classification steps, YOLO directly processes RGB images and predicts object classes and bounding boxes in a single network pass, making it suitable for real-time autonomous driving applications.

However, a model trained primarily on real-world image datasets may not always perform reliably in virtual environments generated by Unreal Engine 4, which are used in the Vehicle-in-Virtual-Environment (VVE) testing framework. Since VVE evaluation depends on accurate perception of virtual pedestrians, bicyclists, and surrounding vehicles, the detection model must be adapted to this environment. To address this issue, transfer learning is applied to the pre-trained YOLOv11 model using the CARLA Object Detection Dataset [36]. Because CARLA is built on Unreal Engine 4, this dataset provides labeled images that are visually consistent with the virtual environment used in VVE testing. By fine-tuning YOLOv11 with CARLA-generated images, the detection model is expected to improve its ability to recognize VRUs and vehicles in virtual traffic scenes.

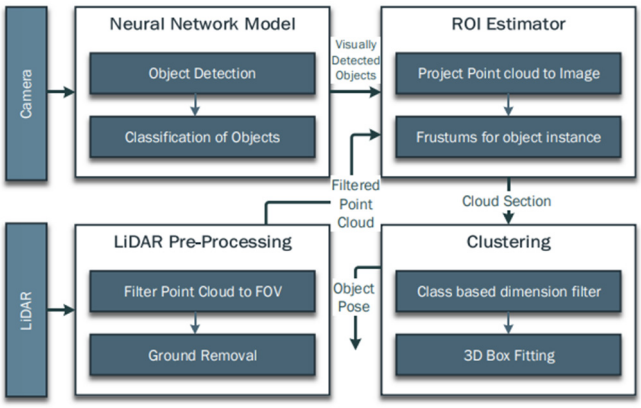

Figure 2.1. Camera and LiDAR based VRUs detection [31].

After VRUs are detected from RGB images, LiDAR point cloud data are used to provide spatial information for estimating their positions and orientations. Figure 2.1 illustrates the camera-

LiDAR-based VRU detection pipeline [31]. First, LiDAR preprocessing is performed to filter the raw point cloud by retaining points within the camera field of view and removing ground points. This step reduces irrelevant data and improves the efficiency of subsequent object localization. Then, the bounding boxes generated by the camera-based detector are projected onto the LiDAR point cloud to associate visual detections with corresponding 3D spatial points. Based on the projected regions of interest, clustering and class-based dimension filtering are applied to extract object-level point clusters. Finally, 3D bounding box fitting is performed to estimate the location, size, and orientation of each detected VRU.

By combining camera-based semantic detection with LiDAR-based spatial localization, the proposed perception pipeline can provide both object category information and accurate 3D position estimates. The camera detector identifies whether an object is a pedestrian, bicyclist, or vehicle, while the LiDAR data provides depth, distance, and orientation information. This multi-sensor fusion strategy improves the reliability of VRU perception in complex traffic environments and provides essential input information for downstream decision-making and collision avoidance modules.

After obtaining the estimated positions and orientations of VRUs, trajectory prediction is performed to estimate their future motion. In this study, the Spatio-Temporal Graph Transformer Network (STGFNet) framework is adopted for pedestrian trajectory prediction [37]. The historical position sequence of each VRU is used as the input, and the model predicts future trajectories by capturing both temporal motion patterns and spatial interactions among surrounding road users. The predicted VRU trajectories can then be used by the autonomous driving system to anticipate potential conflicts and support safer planning and control decisions.

2.3 Test Results

Figure 2.2 shows the raw sensor data used for VRU detection, including the RGB image and the corresponding LiDAR point cloud. The RGB image provides visual information for object recognition, while the LiDAR point cloud provides three-dimensional spatial information of the surrounding environment. These two types of sensor data serve as the inputs for the proposed camera-LiDAR-based VRU detection framework. Figure 2.3 illustrates the Camera and LiDAR-based object detection system architecture. The LiDAR point cloud is first processed by the PointPillars network. The network generates a set of LiDAR-based 3D object proposals containing the estimated object position, dimensions, orientation, and detection confidence. In parallel, the RGB image is processed by the YOLO network. The YOLO image detection network produces 2D object bounding boxes, object classification, and probability. Then, the processed LiDAR data are subsequently associated with the YOLO detections through a matching module. For each matched LiDAR–camera detection pair, the LiDAR spatial information and the YOLO visual information are combined and passed through a fully connected layer for sensor fusion. The fused

output retains the object classification information obtained from YOLO while incorporating the relative position information estimated from the LiDAR data.

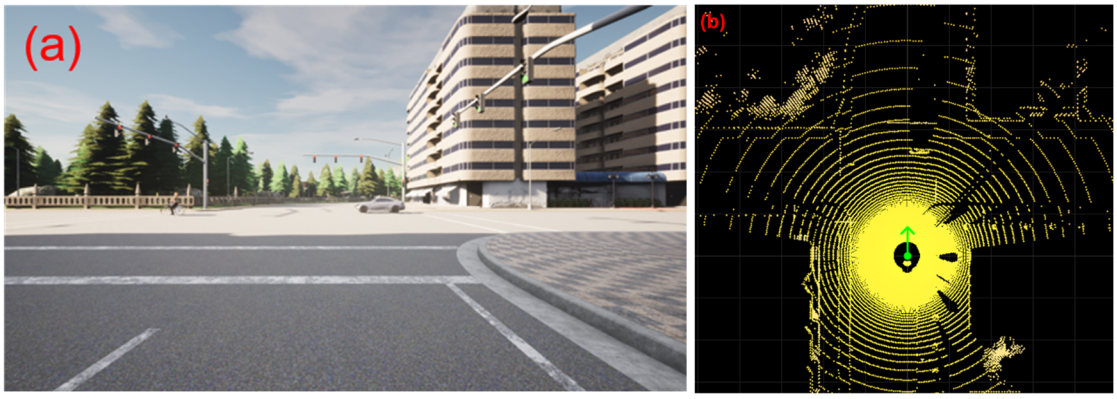

Figure 2.2. Raw Data: (a) RGB Image; (b) Lidar Point Cloud.

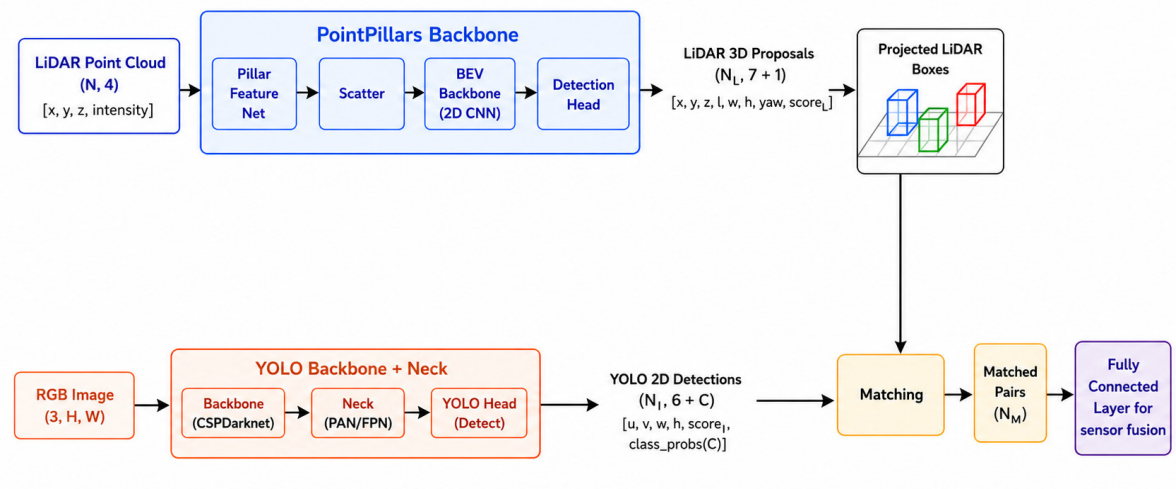

Figure 2.3. Camera and Lidar-based object detection architecture.

Figure 2.4 presents the object detection results using the architecture presented above. The detected objects are marked with bounding boxes, and each bounding box includes the predicted object category, object relative location, and confidence score. As shown in Figure 2.4, both the vehicle and the bicyclist-related objects are successfully detected. The car is identified as a vehicle with its corresponding bounding box and predicted position. For the bicyclist, the detection model separates the rider and the bicycle into two different objects: the rider is classified as a pedestrian, while the bicycle body is detected as a separate object. This result indicates that the model is able to recognize important VRU-related components in the scene, but it also reveals a limitation in the current detection pipeline. Specifically, the bicyclist is not treated as one unified traffic participant. Instead, the human rider and the bicycle are detected independently. This may affect subsequent LiDAR association, object-level localization, and trajectory prediction, since the decision-making modules may need to determine whether the detected pedestrian and bicycle belong to the same

bicyclist. Therefore, further improvement is needed in future work, such as fine-tuning the detection model with additional bicyclist data, introducing post-processing logic to associate riders with bicycles, or defining a specific bicyclist class for more consistent VRU representation. Overall, the preliminary results demonstrate the feasibility of using the camera-based detection model to identify VRUs and surrounding vehicles, while also highlighting the need for further refinement before integration into the complete autonomous driving safety pipeline.

Figure 2.4. Camera and Lidar-based object detection results.

In addition, several representative traffic scenarios were developed and simulated to evaluate the capability and robustness of the proposed VRU detection architecture under different traffic conditions. These scenarios include the normal driving environment shown in Figure 2.4, as well as two more complex traffic environments: an intersection scenario and a roundabout scenario. The complex scenarios introduce denser traffic interactions, varying object positions, and more challenging geometric relationships between the ego vehicle and surrounding road users. Demonstration video links for these simulation tests are also provided here to further illustrate the performance of the proposed fusion framework. Demo Video Link: 1. Normal Traffic Environment: video link. 2. Intersection: video link. 3. Roundabout: video link.

Chapter 3: High Order Control Lyapunov Function - Control Barrier Function - Quadratic Programming Based Low-Level Path Tracking Controller

3.1 Introduction

Low-level actuator control is a critical component of a modular autonomous driving system. It receives trajectory or maneuver commands generated by the high-level decision-making module and converts them into executable steering and acceleration commands while ensuring accurate vehicle motion. Since the controller directly interacts with the physical vehicle, its performance has a significant impact on both driving safety and tracking accuracy. Even if the high-level planner generates a feasible driving strategy, inadequate low-level control strategy may lead to tracking errors or unsafe maneuvers, particularly in complex dynamic traffic environments involving VRUs.

Traditional low-level controllers such as pure-pursuit [29], [30], Stanley [31], disturbance observer (DOB) [32], parameter-space multi-objective PID [33], model predictive control (MPC) [18], [43], [44], [45], [46] and other optimization based controls [47], [48], [49], [50], [51], have demonstrated strong path-tracking performance under nominal driving conditions. However, these approaches are primarily designed to minimize tracking errors and generally assume that the reference trajectory generated by the planner is already safe. As a result, they often lack the capability to actively prevent collisions when unexpected obstacles appear or when the high-level planner generates incorrect decisions. This limitation becomes particularly critical in safety-critical traffic scenarios involving pedestrians and bicyclists, where an unsafe or imperfect decision generated by the high-level planner may directly result in collisions if the low-level controller is unable to intervene and enforce safety constraints.

To improve both tracking performance and collision avoidance capability, this research uses a low-level controller based on the High-Order Control Lyapunov Function–High-Order Control Barrier Function–Quadratic Programming (HOCLF-HOCBF-QP) framework. Within this optimization-based controller, High-Order Control Lyapunov Functions (HOCLFs) enforce system stability and accurate trajectory tracking, while High-Order Control Barrier Functions (HOCBFs) introduce formal safety constraints that prevent the vehicle from entering unsafe regions around surrounding obstacles. By solving a quadratic programming (QP) problem at every control step, the controller generates control commands that simultaneously satisfy stability and safety requirements.

The proposed controller is designed to enhance the robustness of autonomous vehicle control in complex traffic environments involving vulnerable road users. To comprehensively evaluate its performance, a systematic validation framework is adopted. First, Model-in-the-Loop (MIL) simulations are conducted to verify the controller's trajectory tracking accuracy and

collision avoidance capability under various traffic scenarios. Next, Hardware-in-the-Loop (HIL) experiments are performed to assess the controller's computational efficiency and real-time capability. Finally, representative Fatality Analysis Reporting System (FARS) bicyclist crash cases are reconstructed and employed to evaluate the controller under realistic high-risk traffic scenarios. These multi-level evaluations demonstrate that the proposed HOCLF-HOCBF-QP controller provides reliable trajectory tracking, effective collision avoidance, and robust real-time performance in challenging autonomous driving environments.

3.2 Methodology

3.2.1 Vehicle model

In this section, a linear single-track lateral vehicle dynamic model is utilized for controller design and vehicle simulation. As shown in Figure 3.1, the model describes planar lateral motion of the vehicle at constant speed, with wheel side slip angles included for realism. The complete derivation of this vehicle model can be found in Chapter 2 of [18]. Equations 3.1 define the state-space equations of the vehicle dynamics where the state variables include the vehicle side slip angle (β) and yaw rate (r), while the inputs include the front and rear steering angles (δ_f and δ_r). A yaw moment disturbance (M_{zd}) is also incorporated as an external disturbance. Equations 3.2 and 3.3 are used to calculate the vehicle's position based on vehicle speed (V) and the yaw angle (ψ). Table 3.1 summarizes the model parameters, which are adopted from [20].

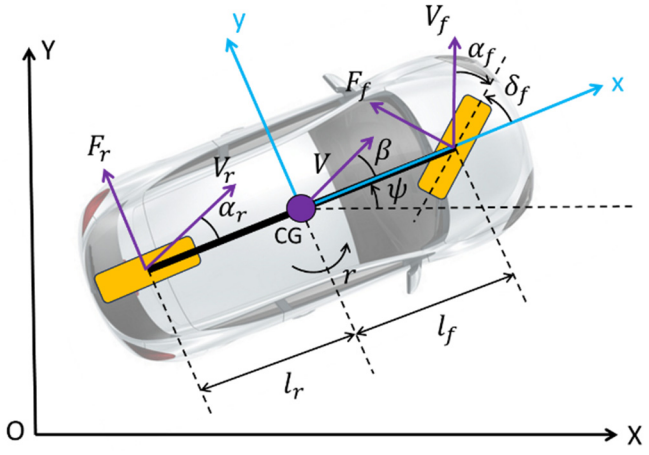

Figure 3.1. Linear single-track lateral vehicle dynamic model [54].

$$\begin{bmatrix}\beta\\ r\end{bmatrix}=\begin{bmatrix}\frac{-C_f-C_r}{mV} & -1+\frac{C_r l_r-C_f l_f}{mV^2}\\ \frac{C_r l_r-C_f l_f}{I_z} & \frac{-C_f {l_f}^2-C_r {l_r}^2}{I_z V}\end{bmatrix}\begin{bmatrix}\beta\\ r\end{bmatrix}+\begin{bmatrix}\frac{C_f}{mV} & \frac{C_r}{mV}\\ \frac{C_f l_f}{I_z} & \frac{C_r l_r}{I_z}\end{bmatrix}\begin{bmatrix}\delta_f\\ \delta_r\end{bmatrix}+\begin{bmatrix}0\\ \frac{1}{I_z}\end{bmatrix}M_{zd} \tag{3.1}$$

$$\Delta x=\int_0^{t_f} v\cos(\beta+\psi)\,dt \tag{3.2}$$

$$\Delta y=\int_0^{t_f} v\sin(\beta+\psi)\,dt \tag{3.3}$$

Table 3.1. Lateral model parameters [20].

Symbol	Parameter
X, Y	Earth-fixed frame coordinate
x, y	Vehicle-fixed frame coordinate
V	Center-of-gravity (CG) velocity
m	Mass
I_z	Yaw moment of inertia
β	Side-slip angle
ψ	Yaw angle
r	Yaw rate
M_{zd}	Yaw disturbance moment
δ_f, δ_r	Front & rear wheel steer angle
α_f, α_r	Front & rear tire slip angle
C_f, C_r	Front & rear tire cornering stiffness
l_f, l_r	Distance between CG and front & rear axle
V_f, V_r	Front & rear axle velocity
F_f, F_r	Front & rear lateral tire force

In this model, several assumptions are made including front-steering vehicle ($\delta_r = 0$) and zero yaw moment disturbances ($M_{zd} = 0$). After making these assumptions, the system can be represented by a five-degree-of-freedom (5-DOF) lateral vehicle dynamic model. The state-space equation for the proposed 5-DOF vehicle lateral dynamic model is given by

$$\begin{bmatrix}\dot{\beta}\\ \dot{r}\\ \dot{x}\\ \dot{y}\\ \dot{\psi}\end{bmatrix}=\begin{bmatrix}A_{11}*\beta+A_{12}*r\\ A_{21}*\beta+A_{22}*r\\ v*\cos(\beta+\psi)\\ v*\sin(\beta+\psi)\\ r\end{bmatrix}+\begin{bmatrix}B_1\\ B_2\\ 0\\ 0\\ 0\end{bmatrix}\delta_f. \tag{3.4}$$

where $A_{11}=\frac{-C_f-C_r}{mv}$, $A_{12}=-1+\left(\frac{C_r l_r-C_f l_f}{mv^2}\right)$, $A_{21}=\frac{C_r l_r-C_f l_f}{I_z}$, $A_{22}=\frac{-C_f l_f^2-C_r l_r^2}{I_z V}$, $B_1=\frac{C_f}{mv}$ and $B_2=\frac{C_f l_f}{I_z}$. Unlike simplified lateral dynamic models as in Equation 3.1, which consider only internal vehicle states like sideslip angle and yaw rate, the proposed model incorporates the vehicle's position and heading (x, y, ψ). This augmentation provides a more intuitive geometric understanding of the vehicle's motion, which is especially useful for developing HOCLF–

HOCBF–based controllers, where reference tracking and obstacle avoidance are formulated in the global coordinate frame.

3.2.2 Low-level CLF-CBF Based Controller Design

3.2.2.1 Preliminary

In this section, we first introduce CLF and CBF. For a control-affine system, the dynamics can be written as:

$$\dot{x} = f(x) + g(x)u \tag{3.5}$$

where $x \in \mathbb{R}^n$ is the system state, $u \in \mathbb{R}^m$ is the control input, and $f(x)$, $g(x)$ are locally Lipschitz functions. $f(x)$ represents the system dynamics, while $g(x)$ describes how control inputs affect the system.

The control goal is to let the vehicle navigate to the desired destination, which requires stabilizing the system around an equilibrium point. A common approach is to use a CLF as a constraint in an optimization-based controller, ensuring Lyapunov stability. Specifically, CLF is a continuously differentiable function $V(x)$: $R^n \to R$, that satisfies certain inequalities with positive constants $c_1 > 0$, $c_2 > 0$, $c_3 > 0$ such that for $\forall x \in \mathbb{R}^n$,

$$c_1\|x\|^2 \leq V(x) \leq c_2\|x\|^2 \tag{3.6}$$

and also,

$$\inf_{u\epsilon U}[L_f V(x) + L_g V(x)u + c_3 V(x)] \leq 0 \tag{3.7}$$

where L_f and L_g represents Lie derivatives along $f(x)$ and $g(x)$. If these conditions hold, then $V(x)$ is a CLF for the system ensuring global and exponential stabilization.

Besides stability, another key control objective is collision avoidance when obstacles are present. To ensure safety, CBFs are used. A CBF defines a safe set of states and enforces the system to keep staying inside this set over time. Like CLFs, CBFs can be added as constraints in an optimization-based framework, such as quadratic programming (QP). Formally, a CBF is defined by a continuously differentiable function $h(x)$: $R^n \to R$ that specifies the safe set as:

$$C = \{x \epsilon R^n : h(x) \geq 0\} \tag{3.8}$$

The function $h(x)$ is considered a CBF if there exists a control input $u \epsilon R^n$ such that the following condition holds for $\forall x \epsilon C$:

$$\sup_{u\epsilon U}[L_f h(x) + L_g h(x)u + \alpha(h(x))] \geq 0 \tag{3.9}$$

where α is a class-κ function. The detailed derivation and proof of CLF and CBF can be found in [49].

3.2.2.2 HOCLF and HOCBF Definition

In more complicated systems, where safety constraints involve higher-order derivatives of the system states, traditional first order CLFs and CBFs are no longer sufficient. To address this, High-Order CLFs (HOCLFs) and High-Order CBFs (HOCBFs) are introduced. The relative degree of a HOCLF or HOCBF is defined as the number of times the function must be differentiated along the system dynamics before the control input u explicitly appears.

Specifically, let HOCLF be a dth-order continuously differentiable function $V(x)$: $R^n \rightarrow R$. Then, define $\phi_0(x) = V(x)$ and construct a sequence of functions $\phi_i(x)$: $R^n \rightarrow R$, $i \in \{1, \dots, d\}$ such that:

$$\phi_i(x) = \dot{\phi}_{i-1}(x) + \alpha_i(\phi_{i-1}(x)), i \in \{1, \dots, d\} \tag{3.10}$$

where each α_i is a class-κ function. If there exist α_d such that $\forall x \neq \mathbf{0}_n$

$$\inf_{u \epsilon U}[L_f^d V(x) + L_g L_f^{d-1} V(x) u + S(h(x)) + \alpha_d(\phi_{d-1}(x))] \leq 0 \tag{3.11}$$

where L_f and L_g represent Lie derivatives along $f(x)$ and $g(x)$, then $V(x)$ is a HOCLF for the system, guaranteeing global and exponential stabilization.

Similarly, HOCBF is an rth-order continuously differentiable function $h(x)$: $R^n \rightarrow R$. Then, define $\Psi_0(x) = h(x)$ and a sequence of functions $\Psi_i(x)$: $R^n \rightarrow R$, $i \in \{1, \dots, r\}$:

$$\Psi_i(x) = \dot{\Psi}_{i-1}(x) + \beta_i(\Psi_{i-1}(x)), i \in \{1, \dots, r\} \tag{3.12}$$

where each β_i is a class-κ function. Correspondingly, define a sequence of sets C_i, $i \in \{1, \dots, r\}$:

$$C_i(x) = \{x \in \mathbb{R}^n : \Psi_{i-1}(x) \geq 0\}, i \in \{1, \dots, r\} \tag{3.13}$$

If there exist β_r and a control input $u \epsilon R^n$ such that the following condition holds for $\forall x \epsilon C_1 \cap C_2 \cap \dots \cap C_i$

$$\sup_{u \epsilon U}\left[L_f^m h(x) + L_g L_f^{m-1} h(x) u + S(h(x)) + \alpha_i(\Psi_{i-1}(x))\right] \geq 0 \tag{3.14}$$

where L_f and L_g denote Lie derivatives along $f(x)$ and $g(x)$, then $h(x)$ is a HOCBF for the system which can guarantee safety. The detailed derivation and proof of HOCLF and HOCBF can be found in [40], [44].

3.2.2.3 HOCLF-HOCBF-QP Controller Design for Vehicle Dynamic Model

The primary objective for the low-level path tracking controller design is to compute safe and effective control inputs that enable the vehicle to follow a desired path from the start to the destination while avoiding collisions. As the first step, we develop the path-following component based on the aforementioned HOCLF framework.

$$V(x) = \left(x - x_g\right)^2 + \left(y - y_g\right)^2 \tag{3.15}$$

$$\begin{aligned} \dot{V}(x,u) &= \mathcal{L}_f V(x) + \mathcal{L}_g V(x) u \\ &= 2v\cos(\beta+\psi)\left(x - x_g\right) + 2v\sin(\beta+\psi)\left(y - y_g\right) \end{aligned} \tag{3.16}$$

$$\begin{aligned} \mathcal{L}_f^2 V(x) = \nabla\left(\mathcal{L}_f V(x)\right)^T f(x) \\ = -2v\sin(\beta+\psi)\left(x - x_g\right)(A_{11}\beta + A_{12}r) + \\ 2v\cos(\beta+\psi)\left(y - y_g\right)(A_{11}\beta + A_{12}r) + \\ 2v^2 - 2v\sin(\beta+\psi)\left(x - x_g\right)r + 2v\cos(\beta+\psi)\left(y - y_g\right)r \end{aligned} \tag{3.17}$$

$$\begin{aligned} \mathcal{L}_g\mathcal{L}_f V(x)u = \nabla\left(\mathcal{L}_f V(x)\right)^T g(x)u = -2v\sin(\beta+\psi)\left(x - x_g\right)B_1\delta_f + \\ 2v\cos(\beta+\psi)\left(y - y_g\right)B_1\delta_f \end{aligned} \tag{3.18}$$

Equations 3.15–3.18 describe the HOCLF formulation and its derivatives, where $[x, y]$ denote the vehicle's current coordinates and $[x_g, y_g]$ represent the destination. The coefficients A_{11}, A_{12}, A_{21}, A_{22}, B_1, B_2 are derived from the previously defined vehicle dynamics model. HOCLF is used to ensure system stability, which can force the vehicle to navigate towards the target waypoint on the desired path. The concept is straightforward: the vehicle's position should gradually converge to the goal coordinates. To achieve this, we construct a Lyapunov candidate function $V(x)$ that measures the squared distance between the current position and the goal. Equation 3.19, then, shows the HOCLF constraint.

$$\mathcal{L}_f^2 V(x) + \mathcal{L}_g\mathcal{L}_f V(x)u + \alpha_1\left(\dot{V}(x,u)\right) + \alpha_2\left(V(x)\right) \leq \delta \tag{3.19}$$

Here, α_1 and α_2 are class-κ functions and δ is a slack variable, which are implemented as positive constant gains. This constraint ensures that the control input u consistently drives the system towards the goal in a stable and controlled manner.

Similarly, we design the collision-avoidance component using HOCBFs. Define the barrier candidate function:

$$h(x) = (x - x_o)^2 + (y - y_o)^2 - r_o^2 \tag{3.20}$$

$$\dot{h}(x,u) = \mathcal{L}_f h(x) + \mathcal{L}_g h(x)u = \mathcal{L}_f h(x) \\ = 2v\cos(\beta+\psi)(x-x_o) + 2v\sin(\beta+\psi)(y-y_o) \tag{3.21}$$

$$\mathcal{L}_f^2 h(x) = \nabla\left(\mathcal{L}_f h(x)\right)^T f(x) \\ = -2v\sin(\beta+\psi)(x-x_o)(A_{11}\beta + A_{12}r) + \\ 2v\cos(\beta+\psi)(y-y_o)(A_{11}\beta + A_{12}r) + \\ 2v^2 - 2v\sin(\beta+\psi)(x-x_o)r + 2v\cos(\beta+\psi)(y-y_o)r \tag{3.22}$$

$$\mathcal{L}_g\mathcal{L}_f h(x)u = \nabla\left(\mathcal{L}_f h(x)\right)^T g(x)u = -2v\sin(\beta+\psi)(x-x_o)B_1\delta_f + \\ 2v\cos(\beta+\psi)(y-y_o)B_1\delta_f \tag{3.23}$$

Equations 3.20–3.23 describe the HOCBF formulation and its derivatives, where $[x, y]$ denote the vehicle's current coordinates and $[x_o, y_o]$ represent the coordinates of the obstacles. The coefficients A_{11}, A_{12}, A_{21}, A_{22}, B_1, B_2 were defined in the previously defined vehicle dynamics model. HOCBF is used to ensure system safety, in order to keep the vehicle away from potential collision with nearby obstacles. The concept is straightforward: the vehicle's position should not enter a circular danger zone of radius r_o centered at $[x_o, y_o]$. To achieve this, we construct a barrier candidate function $h(x)$ which quantifies whether the center of the vehicle is entering the dangerous zone or not. For each obstacle, a corresponding HOCBF is needed. The HOCBF constraint is formulated as:

$$\mathcal{L}_f^2 h(x) + \mathcal{L}_g\mathcal{L}_f h(x)u + \alpha_3\left(\mathcal{L}_f h(x)\right) + \alpha_4\big(h(x)\big) \geq 0 \tag{3.24}$$

Here, α_3 and α_4 are class-κ functions, which are implemented as positive constant gains. This constraint ensures that the control input u can keep the system away from the unsafe region with obstacles.

In this HOCBF design, although representing bicyclists as circles is a simplification given that bicyclists and other VRUs often have high length-to-width ratios, the circular model offers several advantages for real-time control. First, it leads to a simpler analytical form of HOCBFs, resulting in improved computational efficiency, which is critical for real-time implementation. Second, the circular radius can be readily inflated to account for modeling uncertainties, such as sudden posture changes or pose-estimation errors of VRUs.

In practical implementation, several strategies can be used to mitigate the potential limitations of circular approximations. For example, each obstacle can be represented using multiple overlapping circles to better capture the bicyclist's geometric shape. Alternatively, the proposed HOCBF formulation can be extended to many other non-circular obstacle representations, such as ellipses or super-ellipses.

3.2.2.4 HOCLF-HOCBF-QP Formulation

To integrate the previously designed HOCLF and HOCBF into a unified control framework, we formulate a QP that incorporates both HOCLF and HOCBF as inequality constraints. This optimization-based approach allows the controller to compute control inputs that not only guide the system towards the desired destination but also to ensure safety by preventing vehicle entry into unsafe regions around obstacles. The QP is defined as follows:

$$u^* = arg\min_{u,} \|u - u_{ref}\|^2 + q\delta^2 \tag{3.25}$$

$$s.t \quad \mathcal{L}_f^2 V(x) + \mathcal{L}_g \mathcal{L}_f V(x) u + \alpha_1\left(\dot{V}(x,u)\right) + \alpha_2\left(V(x)\right) \leq \delta$$

and

$$\mathcal{L}_f^2 h(x) + \mathcal{L}_g \mathcal{L}_f h(x) u + \alpha_3\left(\mathcal{L}_f h(x)\right) + \alpha_4\left(h(x)\right) \geq 0, for\ each\ obstacle$$

Here, u is the control input and u_{ref} is a nominal reference input, which is usually set to zero to minimize control effort. The slack variable δ provides flexibility by allowing temporary relaxation of the HOCLF constraint in cases of conflict with HOCBF. The penalty weight $q > 0$ determines the trade-off between control performance and constraint violation.

The performance of the proposed controller is largely influenced by the selection of the CLF and CBF constraint coefficients. The CLF coefficients primarily determine the convergence behavior of the tracking controller. Increasing these coefficients generally improves convergence speed and tracking accuracy whereas excessively large values may result in overly aggressive control actions and impose stringent convergence requirements, potentially leading to an infeasible optimization problem. In contrast, the CBF coefficients regulate the conservativeness of the safety constraints. Larger CBF coefficients relax the safety constraints, allowing the vehicle to follow more efficient trajectories at the expense of a reduced safety margin. Smaller coefficients enforce stricter safety requirements, causing the vehicle to maintain greater separation from surrounding obstacles. Overall, appropriate parameter tuning is essential for balancing trajectory tracking performance, collision avoidance capability, and optimization feasibility. Among all tuning parameters, the CLF constraint coefficients have the most direct and significant influence on the feasibility and overall performance of the controller.

3.3 Results

3.3.1 HOCLF-HOCBF-QP Controller Simulation Test

The proposed low-level controller is evaluated through a comprehensive validation framework consisting of Model-in-the-Loop (MIL) simulations, Hardware-in-the-Loop (HIL)

experiments, and representative real-world traffic accident scenarios. First, MIL simulations are conducted to verify the path-tracking performance of the HOCLF controller under obstacle-free conditions using a single lane-change maneuver. The collision avoidance capability of the complete HOCLF-HOCBF controller is then evaluated in dynamic traffic scenarios involving surrounding obstacles. To further assess the practicality of the proposed controller, HIL experiments are performed using an automotive-grade electronic control unit. These experiments demonstrate that the controller can be deployed in real-time autonomous driving applications while also evaluating its computational efficiency and real-time execution performance. Finally, representative bicyclist crash cases reconstructed from the Fatality Analysis Reporting System (FARS) are employed to systematically evaluate the controller under realistic high-risk traffic scenarios. These case studies provide a comprehensive assessment of the controller's ability to achieve accurate trajectory tracking while maintaining safe and collision-free vehicle operation in challenging interactions with vulnerable road users.

3.3.1.1 Path Tracking Performance Test

The blue dashed line in Figure 3.2 represents the desired reference path of a single lane change maneuver, while the red solid line illustrates the actual trajectory of the ego vehicle under the HOCLF-QP control. As observed in the figure, the tracking performance is highly accurate throughout the entire path.

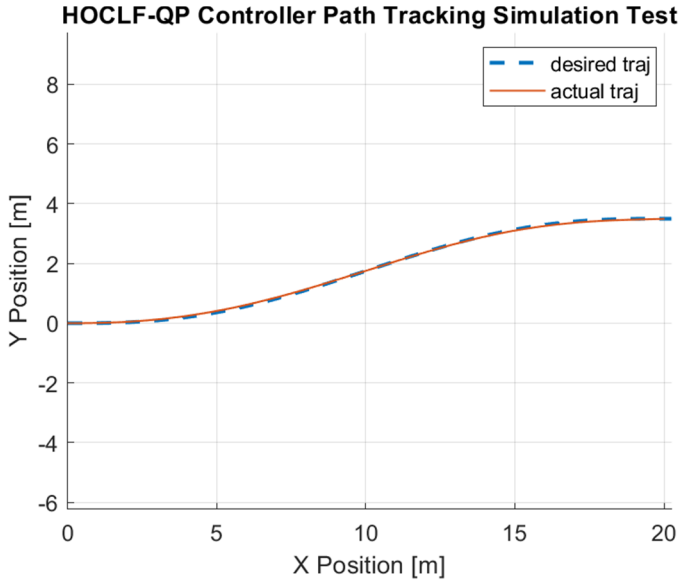

Figure 3.2. Single-lane change trajectory tracking simulation performance [54].

Figure 3.3 presents the corresponding path-tracking error of the proposed HOCLF-QP controller. The root-mean-square-error (RMSE) is 0.009 m, and the maximum tracking error is

0.017 m. These results indicate that the controller enables the vehicle to follow the reference path with minimal lateral path deviation. The small tracking errors further demonstrate the effectiveness of the HOCLF formulation in guaranteeing system stability. This result validates the controller's ability to serve as a reliable low-level trajectory tracking controller for use in the proposed hierarchical control framework.

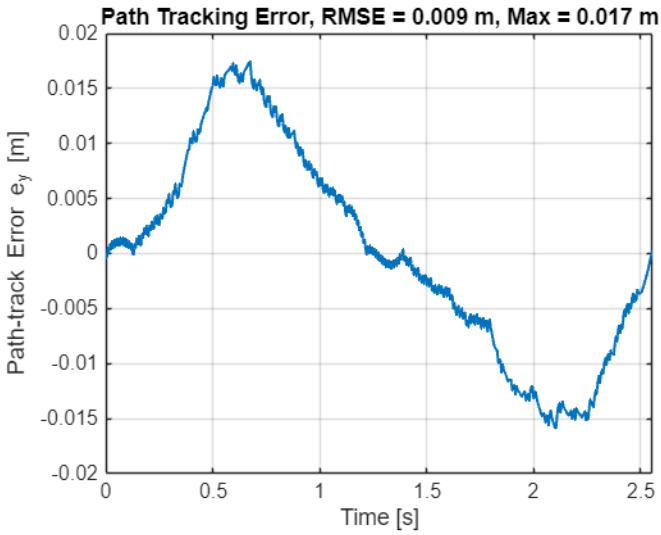

Figure 3.3. HOCBF-QP simulation path tracking error plot [54].

3.3.1.2 Collision Avoidance Performance Test with Static Obstacle

Figure 3.4 illustrates the path tracking performance of the HOCLF-HOCBF-QP controller in a traffic environment with a static obstacle. The red dashed line shows the predefined reference path for the single-lane-change maneuver, while the orange circle marks a static obstacle intentionally placed close to this path. The blue solid line shows the actual trajectory of the ego-vehicle under the HOCLF-HOCBF-QP controller. In contrast to the HOCLF-QP control discussed in the previous section, which only performs path tracking, the inclusion of the HOCBF enables the vehicle to modify its trajectory to avoid a potential collision while still maintaining adherence to the reference path after bypassing the obstacle. The HOCBF serves as a safety constraint, ensuring that the system state remains within a safe set even when the original reference path would have resulted in a collision. The temporary deviation from the reference trajectory observed near the obstacle reflects the collision avoidance capability introduced by the HOCBF. Once the obstacle is cleared, the vehicle smoothly converges back to the reference path, demonstrating the controller’s effectiveness in balancing safety with path tracking stability.

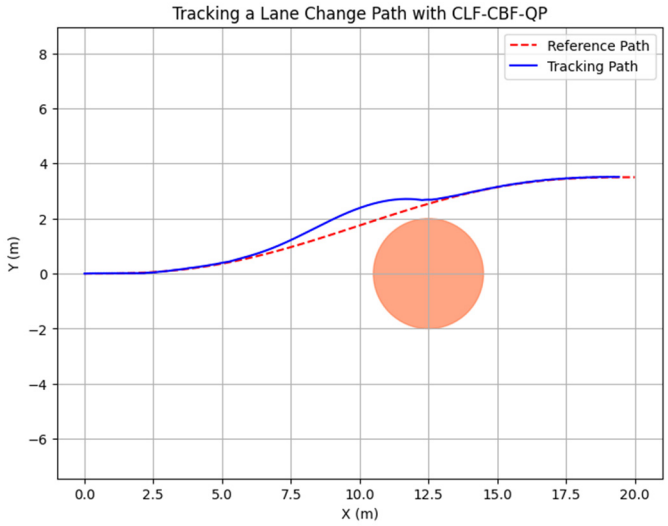

Figure 3.4. Reference trajectory tracking test simulation results [54].

Figure 3.5 presents the path tracking simulation results of proposed HOCLF–HOCBF–QP controller in an environment containing multiple static obstacles. The ego vehicle begins from an arbitrary initial position and aims to reach the target destination, which is represented by green dots, while avoiding collisions with circular obstacles that are marked by the color orange. The dashed curve demonstrates the actual motion of the ego-vehicle under the control of the HOCLF–HOCBF–QP controller. In contrast to the previous case, which primarily evaluated the controller's ability to continuously track a sequence of waypoints (similar to pure pursuit), this scenario focuses on testing the controller's capability to track a reference path while ensuring safe obstacle avoidance and successful arrival at the destination.

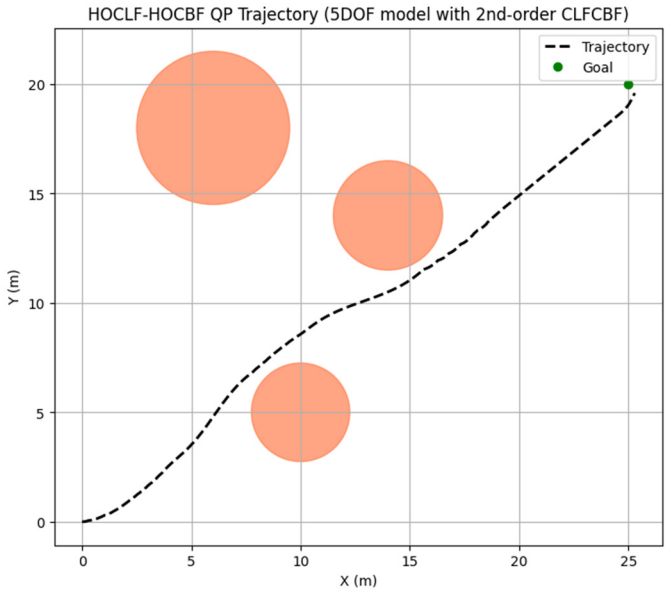

Figure 3.5. Reference point tracking test simulation results [54].

It is observed from the figure that the vehicle can successfully reach the goal while smoothly evading all three obstacles. This behavior demonstrates that the HOCBF constraints effectively enforce safety by keeping the system state away from unsafe regions, while the HOCLF ensures stable tracking of the reference target. The results highlight the controller's capability to achieve both collision avoidance and accurate goal tracking in complex driving environments. Overall, these two cases confirm the effectiveness of the HOCLF-HOCBF-QP controller in static obstacle traffic scenarios, showing robust performance in reference path tracking and highlighting its suitability as a low-level module within an autonomous driving system.

3.3.1.3 Collision Avoidance Performance Test with Dynamic Obstacle

To further test the collision-avoidance performance of the proposed low-level HOCLF–HOCBF–QP controller, a traffic environment containing a dynamic obstacle with the AV programmed to follow a predefined path is constructed for testing. As shown in Figure 3.6, the ego-vehicle can follow its reference path both before and after interacting with the obstacle, while executing a collision avoidance maneuver when encountering the obstacle. The vehicle temporarily deviates from the reference trajectory to maintain safety and then smoothly converges back after evading the obstacle. This demonstrates the controller's ability to ensure safe navigation in the presence of moving obstacles while preserving accurate path tracking.

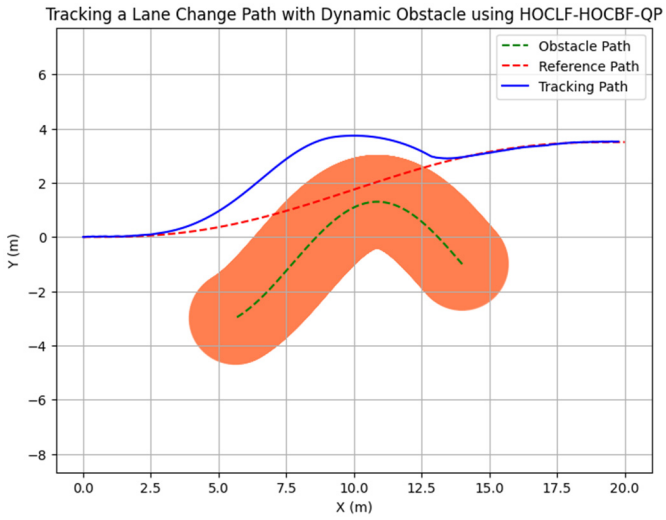

Figure 3.6. Reference trajectory tracking test simulation results with dynamic obstacles [54].

Figure 3.7 presents the time plot of the distance between the ego-vehicle and the dynamic obstacle. The minimum separation occurs at approximately 2.0 seconds, corresponding to the point of closest interaction between ego-vehicle and obstacle. Importantly, this minimum distance remains above the predefined 2-meter safety threshold, which is generally regarded as a socially acceptable lower bound for safe operation in typical low-speed, urban driving conditions. These

results confirm that the HOCLF-HOCBF-QP controller not only avoids collisions with static obstacles but can also maintain safe distance when encountering dynamic obstacles.

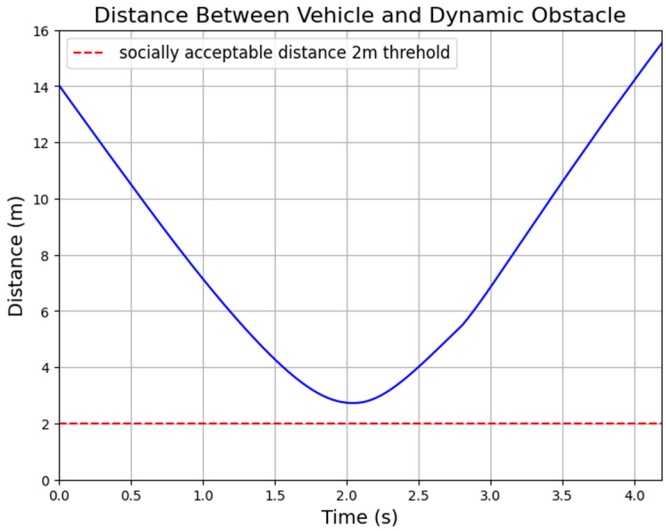

Figure 3.7. Distance between ego-vehicle and dynamic obstacles [54].

The simulation results presented here provide a comprehensive evaluation of the proposed HOCLF-HOCBF-QP controller. Starting from pure path tracking, the controller was shown to achieve stable and accurate path following capability. When static obstacles were introduced, the controller successfully balanced safety and path tracking. Extending to dynamic obstacles, the controller maintained safe distances while performing effective avoidance maneuvers and returning back to the desired path. These results reflect the controller's capability to handle emergency conditions in complex driving environments. In the following sections, we further validate its performance by applying it to representative FARS bicyclist crash scenarios.

3.3.2 HOCLF-HOCBF-QP Controller Hardware-In-Loop Test

3.3.2.1 Hardware-In-Loop Setups

The proposed low-level HOCLF-QP controller may demand substantial computational resources when deployed in highly complex environments. This raises concerns about its real- time capability in autonomous driving systems. To address this challenge, Hardware-in-the-Loop (HIL) simulation is performed to test the real time capability of the proposed controller. Figure 3.8(a) illustrates the overall HIL architecture used and its information flow, and Figure 3.8(b) presents the laboratory layout of the HIL experimental platform. Detailed configurations and implementation settings of the HIL system can be found in [53].

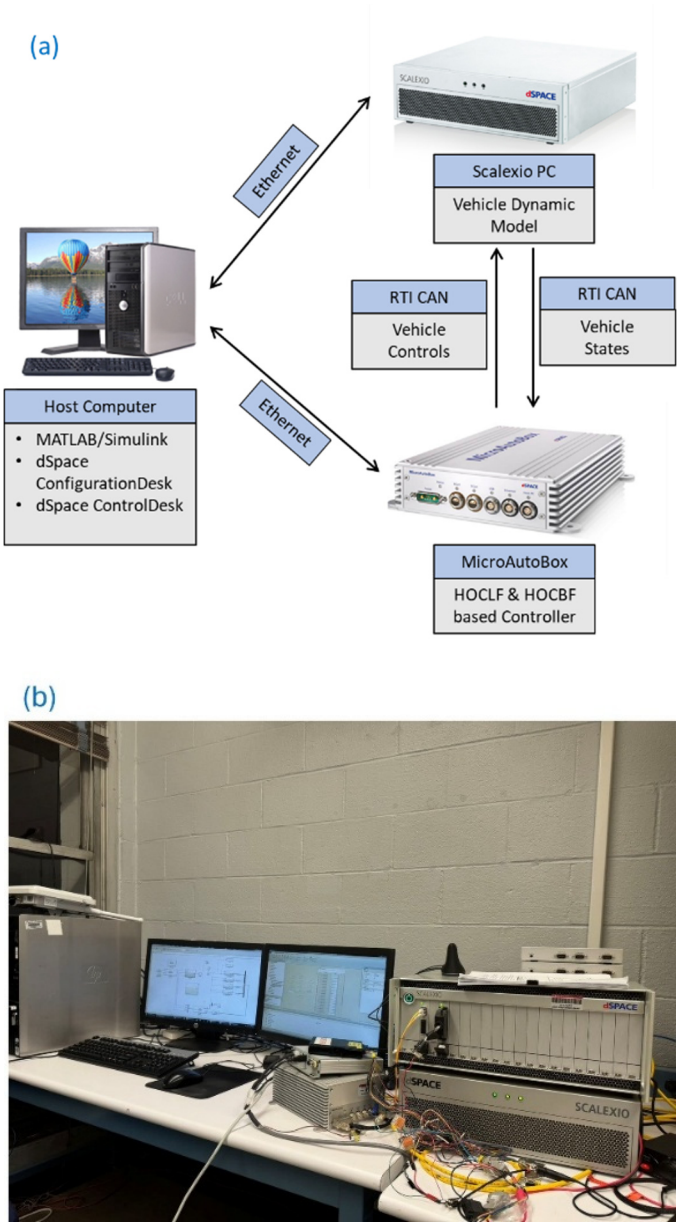

Figure 3.8. HIL setup: (a) HIL architecture and information flowchart; (b) HIL simulator layout [54].

3.3.2.2 Path Tracking Performance Test

Figure 3.9 shows the path tracking performance during the HIL test. Similar to the model-in-the-loop (MIL) simulation results shown in Figure 3.2, the blue dashed line represents the desired reference path, while the red solid line denotes the actual path of the ego vehicle controlled by the HOCLF-QP controller implemented in the HIL platform. The vehicle is still able to closely follow the reference path throughout the entire lane-change maneuver, and the MIL and HIL simulation paths are similar.

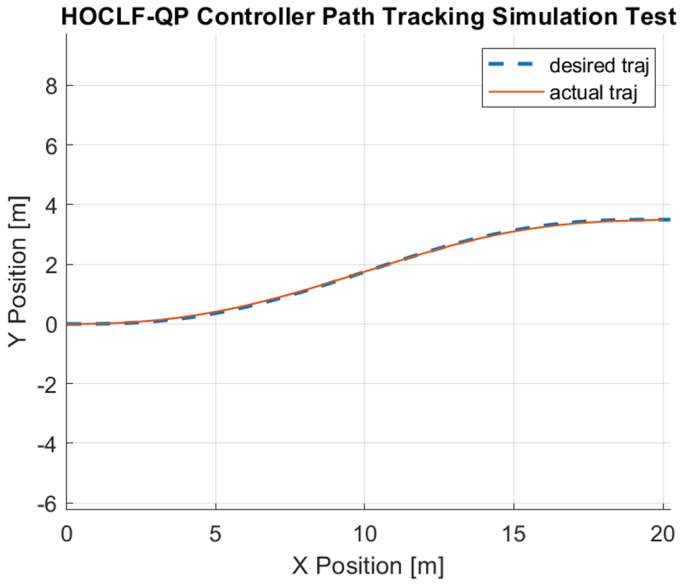

Figure 3.9. Single-lane change trajectory tracking simulation performance [54].

Figure 3.10 presents the corresponding path-tracking error obtained from the HIL experiment. The root-mean-square-error (RMSE) is 0.014 m, and the maximum tracking error reaches 0.047 m. Although the tracking errors are slightly larger compared to the MIL simulation results, they remain within a very small range and demonstrate satisfactory tracking accuracy. The slight increase in error is due to CAN network delays, controller calculation frequency settings, and hardware-related uncertainties. Nevertheless, the overall tracking performance remains highly accurate, indicating the robustness of the proposed low-level controller in real-time implementation. Also, the detailed tracking results of the proposed low-level controller under static or dynamic obstacles including HIL test can be found in our previous work in reference [54].

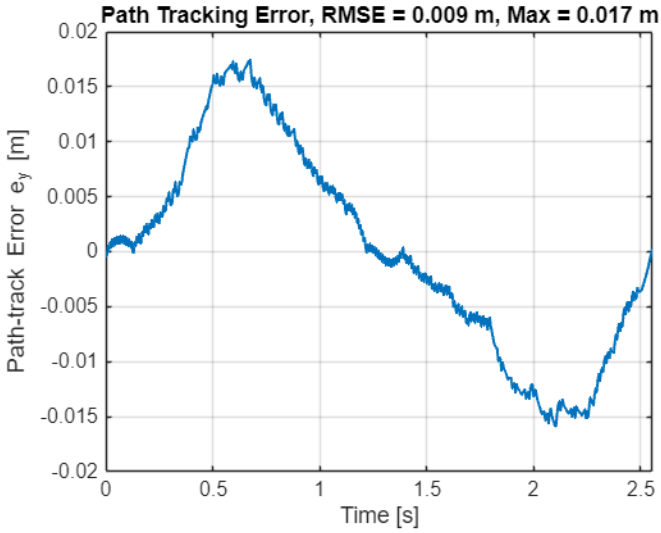

Figure 3.10. Single-lane change trajectory tracking simulation performance [54].

3.3.2.3 HIL Computational Efficiency Evaluation

Computational-efficiency evaluations were conducted at a control frequency of 100 Hz (10 ms per control step). The proposed HOCLF-HOCBF-QP controller requires, on average, 0.618 ms per control step, with a maximum computation time of 1.242 ms and a standard deviation of 0.125 ms. The timing is measured in a single-thread Google Colab CPU environment. This corresponds to approximately 6.2% of the available control cycle, demonstrating that the controller is well suited for real-time operation with sufficient computational margin.

3.3.3 FARS Bicyclist Crash Cases Study

To further evaluate the proposed low-level HOCLF-HOCBF-QP controller's collision avoidance capability, we replicate realistic FARS bicyclist crash scenarios, which capture severe and representative real-world bicyclist accidents. By testing the controller in these scenarios, we can comprehensively evaluate its capability to prevent such accidents in real life.

3.3.3.1 FARS 210: Motorist left turn, bicyclist failed to yield sign

We first replicate realistic FARS210 bicyclist crash scenarios, which involve a motorist attempting a left turn while a bicyclist simultaneously ignores a yield sign and proceeds forward. Although such events occur only occasionally, they represent critical traffic situations where bicyclists violate traffic regulations. At the same time, motorists making left turns may fail to properly account for approaching bicyclists from the side, substantially increasing the risk of severe collisions. Figure 3.11 demonstrates the simulation results of the FARS210 scenario under the control of the proposed HOCLF-HOCBF-QP controller, which enables the ego-vehicle to perform effective path-following and collision avoidance maneuvers. In the figure, the blue dashed line and green dashed line represent the reference paths for the vehicle and the bicyclist, respectively. The red dashed line shows the actual trajectory of the ego-vehicle, while the pink dashed line corresponds to the bicyclist's motion. From the figure we can observe that if the ego-vehicle were to strictly follow its original reference path, it would inevitably collide with the bicyclist approaching from the side, leading to a severe accident. However, the additional HOCBF safety guarantees constraint actively enforces collision avoidance by forcing the ego-vehicle to deviate from its original trajectory. Meanwhile, the HOCLF path tracking constraint ensures that this deviation remains minimal, maintaining accurate path tracking while guaranteeing safety. The snapshots of the vehicle clearly demonstrate how it smoothly adjusts its left-turn maneuver to avoid the bicyclist and then gradually return toward a safe path without collision. This example highlights the effectiveness of integrating HOCLF (for accurate path tracking) with HOCBF (for safety guarantees) in the QP framework, showing that the controller can autonomously override potentially unsafe paths and ensure safe maneuvering in real-world crash scenarios.

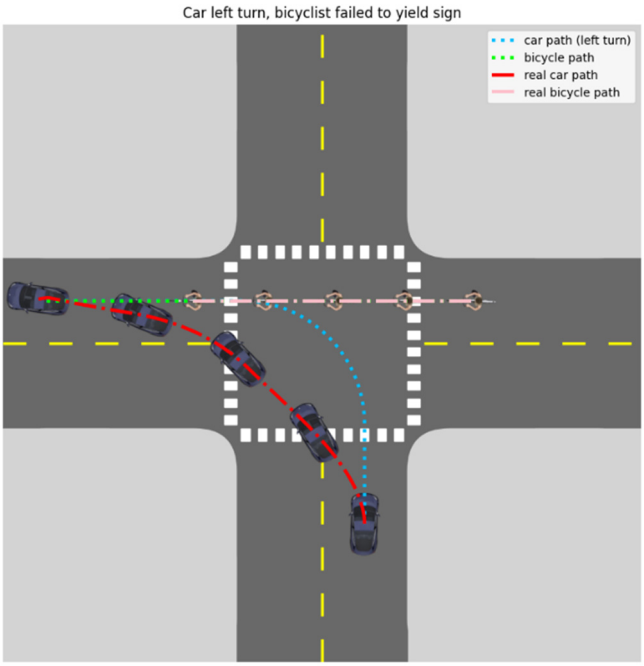

Figure 3.11. Snapshot of FARS210 Simulation with HOCLF-HOCBF-QP Collision Avoidance Controller [54].

3.3.3.2 FARS 220: Motorist Left Turn / Merge

Next, we replicate the realistic FARS220 bicyclist crash scenario, in which the ego-vehicle continues to move straight while the bicyclist suddenly changes lanes and attempts to merge in front of the vehicle. Compared with the FARS210 left-turn case, this scenario occurs more frequently in practice, as bicyclists typically lack rear-view mirrors and may forget to check for approaching vehicles before starting a lane change. Such inattentive maneuvers frequently lead to conflicts with following vehicles and can cause severe collisions if the vehicle cannot properly manage to brake. Figure 3.12 demonstrates the simulation results of the FARS220 scenario under the control of HOCLF-HOCBF-QP low-level controller, which allows the ego-vehicle to perform path-following and collision avoidance simultaneously. In the figure, the blue dashed line and green dashed line represent the reference paths for the vehicle and the bicyclist, respectively. The red solid line shows the actual trajectory of the ego-vehicle, while the yellow solid line corresponds to the bicyclist's motion.

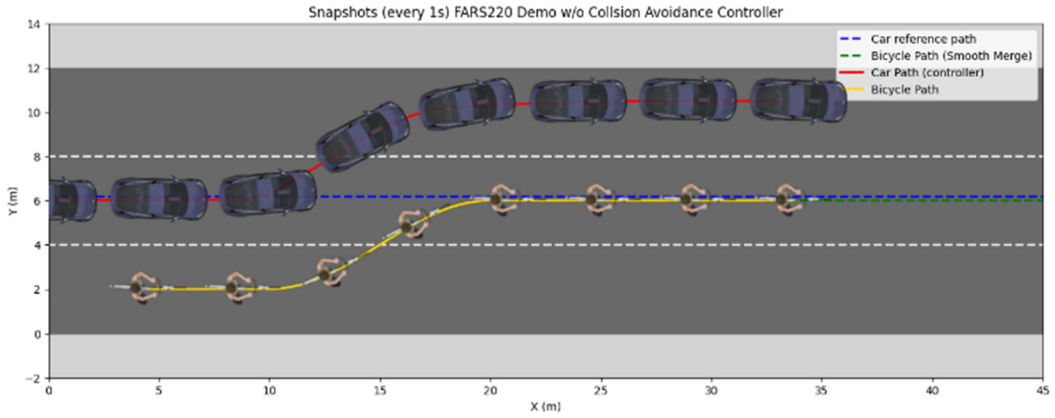

Figure 3.12. Snapshot of FARS220 Simulation with HOCLF-HOCBF-QP Collision Avoidance Controller [54].

In this case, the bicyclist suddenly changes lanes and cuts in front of the ego-vehicle. If the ego-vehicle were to strictly follow its original reference path or fail to slow down, it would inevitably collide with the bicyclist, leading to a severe crash. Such collisions are particularly dangerous because the impact from behind may cause the bicyclist to suddenly accelerate forward, lose balance, or, in the worst case, be run over by the vehicle. However, the HOCBF safety constraint forces the ego-vehicle to deviate from its original lane and perform a single-lane-change maneuver to avoid collision. Meanwhile, the HOCLF stability constraint ensures that this deviation remains minimal, thereby maintaining path-tracking performance while guaranteeing safety. The snapshots clearly demonstrate how the ego-vehicle dynamically adjusts its path to avoid the bicyclist and subsequently converges in a stable manner back into its pre-determined path afterwards.

3.3.3.3 FARS 310: Bicyclists Failed to Yield, Midblock

Finally, we replicate the realistic FARS310 bicyclist crash scenario, in which the ego-vehicle continues straight while a bicyclist suddenly appears from the roadside and enters the traffic lane. Compared with the previous two FARS cases, this scenario is both more common and considerably more challenging to avoid. This type of case can occur in many different settings, such as bicyclists entering from driveways, parking lots, or sidewalks and often without checking for oncoming vehicles. Such cases require that drivers maintain continuous awareness of sidewalk and roadside conditions. When such an emergency situation unfolds, drivers must perform immediate emergency braking and/or lane changing to prevent a collision. This case offers a comprehensive evaluation of the controller's capability to handle sudden emergencies in a highly complicated and dynamic traffic environment.

Figure 3.13 demonstrates the simulation results of the FARS310 scenario using the HOCLF-HOCBF-QP low-level controller, which enables both path-following and collision avoidance. In the figure, the blue dashed line and green dashed line represent the reference paths

for the vehicle and the bicyclist, respectively. The red dashed line shows the actual path of the ego-vehicle, while the pink dashed line corresponds to the bicyclist's motion.

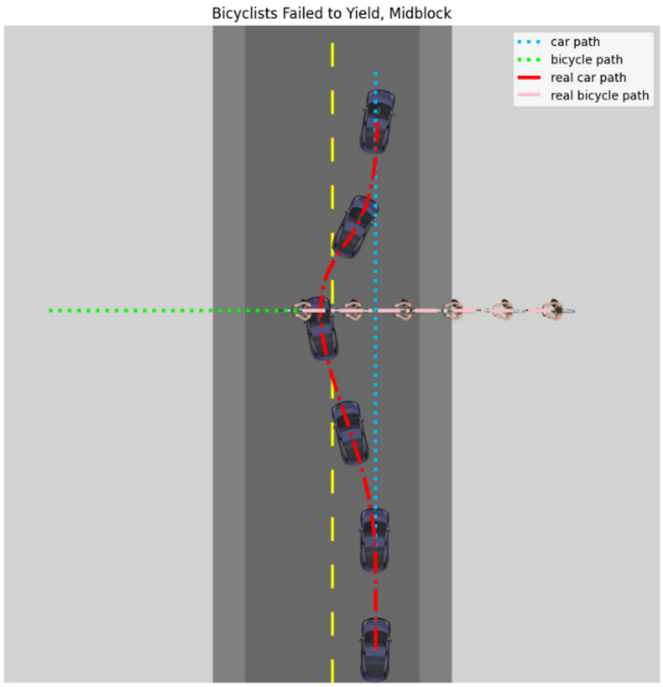

Figure 3.13. Snapshot of FARS220 Simulation with HOCLF-HOCBF-QP Collision Avoidance Controller [54].

In this case, the bicyclist suddenly enters the roadway from the sidewalk, cutting directly across the ego-vehicle's path. If the vehicle had continued along its original route or failed to brake in time, a side collision would be unavoidable. Such crashes are especially dangerous, as the impact could cause the bicyclist to overturn and potentially be run over. To mitigate this risk, the HOCBF forces the vehicle to leave its lane and execute an emergency evasive maneuver, while the HOCLF limits the deviation to preserve path-following performance. The snapshots show how the ego-vehicle smoothly adjusts its trajectory to avoid the bicyclist and then stabilizes back onto a safe path.

The most appropriate response to this emergency scenario in practice would be to stop, as emergency braking is the most reliable way to handle a bicyclist suddenly entering from the roadside. However, our focus here is on evaluating the low-level controller. In a full autonomous driving framework, the low-level controller typically operates in conjunction with a high-level decision-making module. This means that even if the high-level controller makes a suboptimal or unsafe decision, such as continuing forward despite the bicyclist's intrusion, the additional safety layer introduced by the HOCBF constraint ensures that the low-level controller can still enforce collision avoidance and guarantee safety, which makes this method suitable for use in an autonomous driving framework.

3.4 Chapter Conclusions and Future Work

Bicyclists are highly vulnerable road users due to their small size, lower visibility, and their frequent presence in vehicle blind spots. Therefore, ensuring bicyclist safety remains a fundamental challenge in autonomous driving system development. To address this challenge, we proposed an HOCLF-HOCBF-QP framework. Designed as a low-level controller, it can be integrated with high-level planning modules to provide both path tracking and safety guarantees. Unlike traditional path-tracking approaches that primarily enforce path following, the HOCLF-HOCBF-QP controller incorporates an additional safety constraint through the HOCBF, ensuring the simultaneous achievement of path tracking and safety.

The controller's effectiveness was validated through a series of progressively complex evaluation experiments. Starting with basic path-tracking tasks, we demonstrated stable path following capability. Extending to static and dynamic obstacle scenarios, the controller successfully balanced reference tracking and safety. Finally, by replicating and testing our controller in three representative FARS bicyclist crash scenarios, we showed that the controller can autonomously track the desired path and perform collision avoidance maneuvers in a dynamic traffic environment to avoid collisions with bicyclists during emergency situations. Together, these results confirm that the HOCLF-HOCBF-QP controller provides a reliable low-level safety mechanism that complements high-level decision-making, significantly enhancing the safety of bicyclists and other vulnerable road users in complex traffic environments in their interactions with AVs.

Despite the advancements presented in this study, the proposed HOCLF-HOCBF-QP low-level controller still has several limitations that need further investigation and adjustment. First, representing bicyclists as circles is a simplification, and many other road users may also have a high length-to-width ratio or irregular shape. Therefore, our current approach and formulation can be further extended to other safety constraint shapes such as ellipses or near-rectangular super-ellipses. These extended geometric HOCBFs have analytical expressions similar to circular HOCBFs and, therefore, can be introduced in future work to better capture actual geometry of the obstacles with irregular shapes without significantly increasing computational load. Also, the effectiveness of the controller depends heavily on the choice of hyperparameters. In this section, hyperparameters were selected based on empirical knowledge and simulation study, which proved sufficient for the presented scenarios. However, identifying the most robust hyperparameter set across varying conditions remains an open problem. Developing algorithms that are capable of dynamically tuning hyperparameters in response to real-world conditions will be an important direction for future research.

Chapter 4: Hierarchical Collision Avoidance Framework

4.1 Introduction

While autonomous driving technologies offer a promising solution to human-related errors and traffic inefficiencies [55], developing engineering systems capable of navigating complex, dynamic urban environments remain a formidable challenge. Extensive research has already been conducted in the autonomous driving field to help vehicles navigate safely and efficiently [56], [57], [58], [59], [60], [61], [62]. Current approaches can be broadly categorized into two types. The first is end-to-end, model-free learning, where an autonomous driving agent directly maps sensory inputs to control actions. Bojarski et al. were among the first to propose a convolutional neural network based end-to-end framework for autonomous driving systems (ADS) [11]. Kendall et al. then applied deep reinforcement learning (DRL) methods to training an autonomous driving agent [12]. Building on this, subsequent research has expanded end-to-end applications in ADS with several innovations [65], [66], [67], including uncertainty-weighted decision transformer architecture for complex roundabout scenarios [16] and hierarchical reinforcement learning (H-REIL) to better balance safety and efficiency in near-accident scenarios [17]. Compared to traditional modular based approaches, the end-to-end method allows for optimization across the entire processing pipeline and offers the potential for improved overall performance. Nevertheless, the lack of explicitly defined safety constraints can be problematic, as such systems might perform unsafe behaviors in rare or safety-critical scenarios.

The second category is modular, model-based design, which remains the mainstream in autonomous driving systems development [70], [71], [72], [73], [74]. In this paradigm, path planning and collision avoidance are treated as dedicated modules, often formulated as optimization problems or addressed using machine learning based method to generate safe and feasible paths. Ames et al. formulated path following and collision avoidance as constraints using CLF-CBF-QP to ensure both efficient and safe navigation [75], [76], [77], [78]. Nageshrao et al. proposed to apply short-horizon safety mechanisms for highway autonomous driving [27]. Zhang et al. introduced a bootstrapping based demonstration policy to accelerate training of high-level autonomous driving decision-making agents [28]. Compared with end-to-end learning approaches, modular autonomous driving frameworks generally provide greater robustness, interpretability, and reliability by explicitly separating perception, decision-making, planning, and control into independent functional modules. This modular architecture enables individual components to be developed, analyzed, and optimized separately, facilitating systematic performance evaluation, simpler ablation studies, and algorithm improvements. As a result, modular approaches remain the dominant paradigm in both academic research and industrial autonomous driving systems.

Motivated by these advantages, this research adopts a hierarchical decision-making and control framework that integrates a DRL-based high-level planner with a HOCLF-HOCBF-QP based low-level path tracking controller. Figure 4.1 shows the example control flow chart of the

proposed DRL and HOCLF-HOCBF-QP Based hierarchical autonomous driving control framework. Within this framework, the high-level planning agent operates in a discrete action space and generates lane-level maneuver commands, such as lane keeping, left lane changes, and right lane changes, by learning an optimal decision policy through interactions with diverse traffic scenarios. The low-level controller then executes these maneuver commands while ensuring both tracking accuracy and vehicle safety. Specifically, the HOCLFs enforce system stability and accurate trajectory tracking, whereas the HOCBFs impose formal safety constraints that prevent collisions with surrounding road users. The final control commands are obtained by solving a QP problem that simultaneously satisfies both stability and safety requirements.

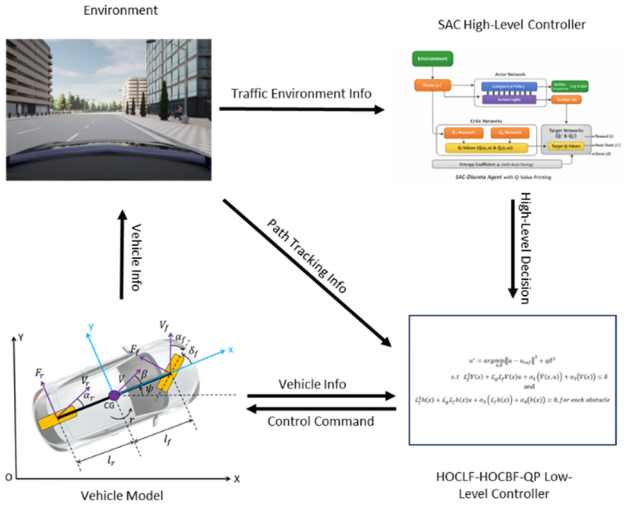

Figure 4.1. Proposed Hierarchical DRL and HOCLF-HOCBF-QP Based Control Flow Chart [115].

For high-level decision-making, numerous deep reinforcement learning (DRL) algorithms have been successfully applied to autonomous driving in recent years. Representative approaches include Deep Deterministic Policy Gradient (DDPG) [81], Proximal Policy Optimization (PPO) [82], and Soft Actor-Critic (SAC) [83], [84]. Among existing DRL methods, value-based and policy-based algorithms have emerged as the two dominant approaches for autonomous driving decision-making. To identify a robust and effective high-level planner within the proposed hierarchical framework, this study investigates representative algorithms from both categories. Specifically, Double Deep Q-Network (DDQN) is selected as the representative value-based algorithm, while Soft Actor-Critic Discrete (SAC-D) is adopted as the representative policy-based algorithm. Their enhanced variants are further developed and systematically compared to evaluate their decision-making performance under complex traffic scenarios.

To develop and evaluate the proposed high-level decision-making agent, the Highway-env simulation platform is adopted throughout this study [85]. Compared with high-fidelity simulators such as CARLA [86], [87] and TORCS [88], Highway-env provides a lightweight yet flexible environment that contains the essential traffic scenarios required for autonomous driving research.

More importantly, its modular architecture naturally separates high-level decision-making from low-level vehicle control, making it particularly suitable for training, testing, and comparing different decision-making algorithms within the proposed hierarchical framework. This separation also significantly improves training efficiency and facilitates systematic algorithm evaluation.

The proposed high-level planner is evaluated through a progressive set of traffic environments with increasing levels of complexity. The evaluation begins with a multi-lane highway scenario, where the agent learns fundamental driving behaviors such as lane keeping, lane changing, and obstacle avoidance in relatively structured traffic conditions. This scenario serves as a baseline for verifying the effectiveness of the decision-making algorithms before introducing more challenging urban environments.

After validating the basic driving capability, the evaluation is extended to roundabout [89] and intersection scenarios [90], [91], which are widely recognized as two of the most challenging urban driving environments for autonomous vehicles. These scenarios involve complex traffic interactions, irregular road geometries, dynamic right-of-way negotiations, and frequent conflicts with surrounding vehicles, substantially increasing both decision-making difficulty and collision risk [92], [93]. Consequently, they provide rigorous benchmarks for assessing the robustness, safety, and generalization capability of autonomous driving algorithms under realistic traffic conditions.

4.2 Methodology

4.2.1 DDQN High-Level Decision-Making Agent for Vehicle Dynamic Model

As the representative value-based reinforcement learning algorithm, DDQN is first employed to develop the high-level decision-making agent. DDQN extends the original Deep Q-Network (DQN) by decoupling action selection from target Q-value estimation, thereby effectively reducing the overestimation bias commonly observed in DQN and improving training stability. Similar to other deep reinforcement learning methods, DDQN learns directly through interactions with the driving environment without requiring manually labeled data, making it well suited for autonomous driving tasks with high-dimensional state representations such as occupancy grids.

Compared with many policy-based reinforcement learning algorithms, DDQN offers several advantages for lane-level autonomous driving decision-making. As an off-policy value-based algorithm, DDQN can efficiently reuse previously collected experience through experience replay, leading to improved sample efficiency and more stable training. In addition, the discrete action space adopted in this study, including lane keeping, left lane change, and right lane change, naturally aligns with the action representation of DDQN, allowing the agent to learn effective driving policies with relatively low implementation complexity.

For these reasons, DDQN is selected as the baseline value-based reinforcement learning algorithm for the proposed hierarchical autonomous driving framework. Its performance is subsequently evaluated and compared with the policy-based Soft Actor-Critic Discrete (SAC-D) algorithm to investigate the advantages and limitations of the two reinforcement learning paradigms in complex traffic environments.

Human drivers naturally perform lane-changing maneuvers to avoid obstacles, such as stationary vehicles, slow-moving traffic, or unexpected hazards. Inspired by this common driving strategy, the proposed high-level decision-making agent is designed to learn lane-level maneuver planning in multi-lane traffic environments. By selecting appropriate maneuvers, including lane keeping and lane changes, the agent can safely navigate around obstacles while maintaining efficient vehicle operation. Although the primary experiments focus on obstacle avoidance in highway traffic, the proposed decision-making strategy can be naturally extended to scenarios involving vulnerable road users (VRUs), such as pedestrians and bicyclists.

The DDQN agent is trained by minimizing the temporal-difference (TD) loss defined as

$$L_i(\theta) = \mathbb{E}_{(s,a,r)}[\Big(r + \gamma \max_{a_{t+1}} Q_{\theta_i^-}\Big(s_{t+1}, arg\max_{a_{t+1}} Q_{\theta_i}(s_{t+1}, a_{t+1})\Big) - Q_{\theta_i}(s_t, a_t)\Big)^2] \tag{4.1}$$

Unlike the original DQN, which uses the same network for both action selection and target value estimation, DDQN employs separate online and target networks during Q-value updates. Specifically, the online network selects the action with the highest estimated value ($arg\max_{a_{t+1}} Q_{\theta_i}(s_{t+1}, a_{t+1})$, while the target network evaluates the selected action ($\max_{a_{t+1}} Q_{\theta_i^-}\Big(s_{t+1}, arg\max_{a_{t+1}} Q_{\theta_i}(s_{t+1}, a_{t+1})\Big)$. This decoupled update mechanism effectively reduces the overestimation bias commonly observed in DQN, leading to improved training stability and more reliable value estimation. Table 4.1 summarizes the parameters used in the DDQN loss function. Although DDQN is adopted as the baseline value-based algorithm in this study because of its simplicity and stable learning characteristics, the proposed hierarchical framework is algorithm-independent. More advanced reinforcement learning algorithms, such as SAC-D (which is introduced in the next section) and their enhanced variants, can be readily incorporated into the same framework for more complex autonomous driving tasks.

Table 4.1. DDQN loss function parameters [74].

Symbol	Parameter
s	State
a	Action
r	Immediate reward
θ_i^-	Target-network's parameter
θ_i	Online-network's parameter
γ	Discount for future reward

Figure 4.2 illustrates the neural network structure used in the DDQN framework and Algorithm 4.1 demonstrates the pseudocode of hierarchical DDQN implementation. The DDQN employs a fully connected feedforward neural network consisting of two hidden layers, each with 128 neurons using ReLU as the activation function. The input layer has 25 units (flattened from 5×5 including ego vehicle's information and traffic environment information), and the output layer has 3 units corresponding to three lane-level actions. The training process uses a standard replay buffer with a size of 100,000 and a mini-batch size of 64. The Q-network is updated using the mean squared error (MSE) loss between the predicted Q-values and the target Q-values. The Adam optimizer is used with a learning rate of 0.001. To stabilize training and mitigate overestimation bias, a target network is maintained and synchronized with the main Q-network every 100 learning steps. An ε-greedy exploration strategy is adopted, where ε linearly decays from 1.0 to 0.05 over 200,000 steps. This linear decay strategy allows the agent to fully explore the environment instead of converging to a local optimal policy.

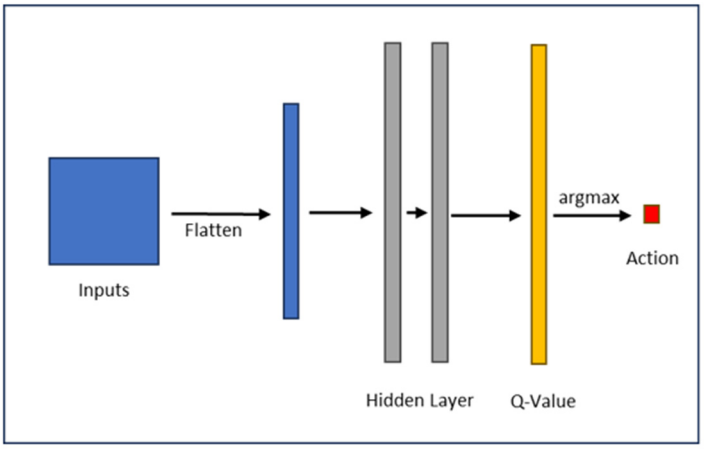

Figure 4.2. DDQN framework neural network structure [74].

Algorithm 4.1. Hierarchical DDQN algorithm flowchart [74].

Algorithm 4.1
1: Initialize high-level DDQN agent
2: Initialize low-level HOCLF-HOCBF-QP controller
3: **for** each episode **do**
4: Initialize and reset traffic environment
5: **for** t = 1 to T **do**
6: With probability ϵ select a random high-level action a_t
7: Otherwise select high-level action $a_t = \max_a Q^*(s_t, a; \theta)$
8: **for** each low-level control steps **do**
9: Reset target tracking point according to current states and a_t
10: Reset obstacle list according to current states
11: Calculate optimal control u^* using HOCLF-HOCBF-QP
12: Execute u^* for a control step
13: **end for**

14: Calculate reward r_t and record next state s_{t+1}
15: Store transition (s_t, a_t, r_t, s_{t+1}) in replay buffer D
16: **if** t mod training frequency == 0 **then**
17: Sample random minibatch of transitions (s_j, a_j, r_j, s_{j+1})) from D
18: Set $y_j = r_j + \gamma \max_{a_{j+1}} \hat{Q}(s_{j+1}, \text{argmax}_{a_{j+1}} Q(s_j, a_{j+1}; \theta); \theta)$
19: for non-terminal s_{j+1}
20: or $y_j = r_j$ for terminal s_{j+1}
21: Perform a gradient descent step to update θ
22: Every N steps reset $\hat{Q} = Q$
23: **end if**
24: Set $s_{t+1} = s_t$
25: **end for**
26: **end for**

4.2.2 SAC-D High-Level Decision-Making Agent for Vehicle Dynamic Model

Algorithm 4.2 displays the pseudocode of the SAC-D algorithm. Compared to other DRL algorithms, SAC-D offers distinct advantages, particularly in discrete decision-making tasks where traditional value-based methods, such as Deep Q-Networks (DQN), struggle with efficient exploration. SAC-D extends the entropy-regularized reinforcement learning framework to discrete action spaces, allowing the agent to learn stochastic policies that balance exploration and exploitation more effectively than standard Q-learning approaches. Moreover, SAC-D takes advantage of both policy-based and value-based methods, utilizing an actor network to generate action probabilities and a critic network to estimate action-value functions. This dual-network approach enhances learning efficiency and stability, making it particularly well-suited for autonomous driving, adaptive control systems, and decision-making in dynamic traffic environments.

Algorithm 4.2. Hierarchical SAC-D algorithm flowchart [74].

Algorithm 4.2

1: Initialize Q-network parameters θ_1, θ_2 (Critics)
2: Initialize target Q-networks parameters $\theta_1' \leftarrow \theta_1$, $\theta_2' \leftarrow \theta_2$
3: Initialize policy network (Actor) parameters ϕ
4: Initialize entropy temperature α
5: Initialize replay buffer $\mathcal{D}$
6: **for** each episode **do**
7: Observe initial state s
8: **for** each time step t **do**
9: Select action $a \sim \pi_\phi(a|s)$ from policy
10: Execute action a, observe next state s', reward r
11: Store transition in replay buffer: $\mathcal{D} \leftarrow \mathcal{D} \cup (s, a, r, s')$
12: Execute action a, observe next state s', reward r
13: **Compute target Q-value using double Q-learning**
14: Sample next action $a_i' \sim \pi_\phi(a'|s_i')$

Compute log probability of action $\log \pi(a'|s_i')$

$Q_{target} = r_i + \gamma \left[\min\left(Q_{\theta_1'}(s_i', a_i'), Q_{\theta_2'}(s_i', a_i')\right) - \alpha \log \pi(a_i'|s_i')\right]$

Compute Q-value loss

$L_Q(\theta) = MSE(Q_{\theta_1}(s_i, a_i) - Q_{target}) + MSE(Q_{\theta_2}(s_i, a_i) - Q_{target})$

Update Critic networks

$\theta_1 \leftarrow \theta_1 - \lambda_Q \nabla_{\theta_1} L_Q(\theta)$

$\theta_2 \leftarrow \theta_2 - \lambda_Q \nabla_{\theta_2} L_Q(\theta)$

Compute policy loss using entropy-regularized objective

$L_\pi(\phi) = E[\log \pi(a_i|s_i)\,(\alpha \log \pi(a_i|s_i) - \min\left(Q_{\theta_1}(s_i, a_i), Q_2(s_i, a_i)\right))]$

Update Actor network

$\phi \leftarrow \phi - \lambda_\pi \nabla_\phi L_\pi(\phi)$

Adjust temperature α to balance exploration and exploitation

$L_\alpha = E\left[-\alpha\left(\log \pi(a_i|s_i) + target_{entropuy}\right)\right]$

$\alpha \leftarrow \alpha - \lambda_\alpha \nabla_\alpha L_\alpha$

Soft update target Q-networks

$\theta_1' \leftarrow \tau\theta_1 + (1-\tau)\theta_1'$

$\theta_2' \leftarrow \tau\theta_2 + (1-\tau)\theta_2'$

$s \leftarrow s'$

end for

end for

One of the key strengths of SAC-D lies in its off-policy learning with experience replay, which improves sample efficiency by enabling the agent to learn from past experiences rather than relying solely on real-time interactions. Additionally, the use of two Q-networks mitigates overestimation bias, leading to more stable training and improved policy robustness. Unlike traditional deterministic methods, SAC-D incorporates entropy maximization, leading to better exploration while preventing premature convergence to suboptimal policies. This is significant in complex traffic scenarios where decision-making must account for multiple interacting agents, including other vehicles and VRUs.

In addition, SAC-D does not require manually tuned noise. SAC's inherent entropy-based exploration enables the policy to learn a well-balanced tradeoff between safety and efficiency in autonomous vehicle navigation. Therefore, given its ability to handle discrete action spaces, multi-agent interactions, and safety-critical decisions, SAC-D is applied to design a high-level decision-making agent in this paper. The autonomous navigation problem is modeled as a Markov Decision Process (MDP), defined as follows:

- State Space (S): The state space S contains all possible environment states $s \in S$ which are represented as a fixed-size array that contains the kinematic information of the ego vehicle and surrounding vehicles including position (x, y), heading (ψ) and velocity (v_x, v_y) with $s = [x, y, v_x, v_y, \psi]$.

- Action Space (A): The action space A is a discrete set of five discrete high-level actions $a \in A$ comprising of: keep lane, lane change left, lane change right, accelerate and decelerate.

- Reward Design (R): The reward function R is designed to promote safe, goal-directed, and efficient driving behavior and is composed of four components: collision penalty, task completion reward, lane-deviation penalty, and speed-based reward and is given by

$$r = \begin{cases} r_{task\ completion}, & if\ reach\ goal \\ r_{collision}, & if\ collide \\ r_{lane\ keeping} + r_{speed}, & else \end{cases} \tag{4.2}$$

- Transition Model (P): A traffic simulation model that predicts the probabilities of the next states given a particular action and the current state. The transition model depends on the ego vehicle's dynamics and surrounding vehicles' traffic models.

The optimal policy π^* requires maximum cumulative return and entropy encouraging sustained exploration and robust policy learning as

$$\pi^* = argmax_\pi \mathbb{E}_{(s_t,a_t)\sim\tau_\pi}[\sum_{t=0}^{T} \gamma^t R(s_t, a_t) + \alpha\mathcal{H}(\pi(\cdot\,|s_t))] \tag{4.3}$$

This encourages sustained exploration and robust policy learning where $\gamma \in [0, 1]$ is the discount factor, $R(s_t, a_t)$ is the immediate reward function of choosing action a_t at state s_t. α is the temperature parameter and $\mathcal{H}(\pi(\cdot\,|s_t))$ is the entropy function. The corresponding soft state value function is

$$V_{soft}(s_t) = \sum_{a_t} \pi(a_t|s_t)[Q(s_t, a_t) - \alpha log\pi(a_t|s_t)] \tag{4.4}$$

where $Q(s_t, a_t)$ is the action value function which is an estimate of the cumulative return of choosing action a_t at state s_t. In order to obtain the optimal policy, SAC-D optimizes an equivalent stochastic policy objective derived from the maximum-entropy formulation as

$$\mathcal{J}_\pi = \mathbb{E}_{s_0\sim D_0} \sum_a \pi(a|s)[Q(s, a) - \alpha log\pi(a|s)]) \tag{4.5}$$

for the policy iteration of SAC-D. Since the action value function $Q(s, a)$ is used to estimate the cumulative return in this process, a policy evaluation step is required to estimate the soft action-value function. As mentioned before, SAC-D uses a dual Q-network to mitigate the over-estimation issue in its critic network. Therefore, the corresponding soft action value function is given by

$$Q_{soft}(s_t, a_t) = R(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim P} V_{soft}(s_{t+1}) \tag{4.6}$$

The corresponding cost function based on the soft Bellman residual should be minimized to get better action function

$$J_Q = \mathbb{E}_{(s_t, a_t) \sim D} \left[\sum_{i=1,2} \frac{1}{2} \left(Q_{\theta_i}(s_t, a_t) - \hat{Q}(s_t, a_t) \right)^2 \right] \tag{4.7}$$

$$\begin{aligned} \hat{Q}(s_t, a_t) = R(s_t, a_t) + \gamma \sum_{a'} \pi(a'|s_{t+1}) \left[\min_i Q_{\theta'_i}(s_{t+1}, a') \right. \\ \left. - \alpha log \pi(a'|s_{t+1}) \right] \end{aligned} \tag{4.8}$$

where θ is the online network weight and θ' is the target network weight. For better balance exploration and exploitation, the temperature hyperparameter α can be automatically adjusted by using the following objective functions with target entropy constant $\bar{\mathcal{H}}$:

$$J_\alpha = \sum_{a_t} \pi(a_t|s_t)[-\alpha log \pi(a_t|s_t) + \bar{\mathcal{H}}]) \tag{4.9}$$

Figure 4.3 illustrates the neural network structure used in the SAC-D framework and consists of a categorical policy network as actor, a double Q-network as critic, and corresponding target networks. The state is flattened and fed into both the actor and critic. The actor network is implemented as a two-layer MLP with hidden dimensions of 256 and 256 units, each followed by ReLU activations, and a final linear output layer that produces scores for all discrete actions. In this paper, the action space contains the five lane-level actions of: lane keep, lane change left, lane change right, accelerate and decelerate. Then, a softmax operation is applied to obtain the action probabilities. During interaction with the environment, actions are sampled from the categorical distribution in training and selected greedily during evaluation.

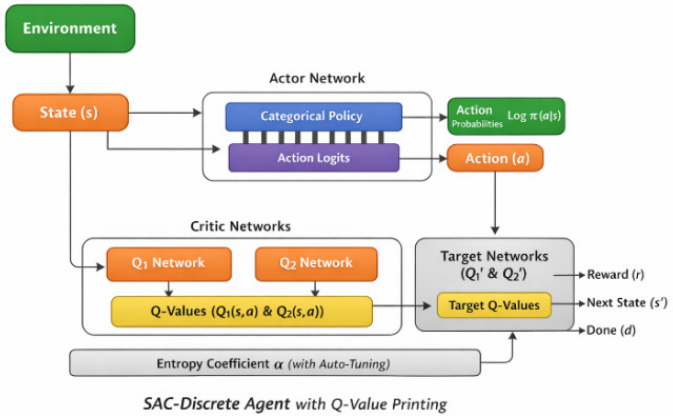

Figure 4.3. SAC-D framework neural network structure [115].

The critic network applies a double Q-learning structure to mitigate overestimation bias, where two independent Q-networks share the same architecture as the actor backbone: a two-layer MLP with hidden dimensions of 256 and 256 units followed by a linear layer that outputs Q-values for all discrete actions. Target Q-networks are maintained using soft updates to stabilize learning. The actor and critic interact through an entropy-regularized Bellman backup, in which the target state value is computed as the expectation over the policy of the minimum target Q-value minus an entropy term scaled by the temperature parameter. The policy is optimized to maximize expected returns while encouraging exploration via entropy regularization. The entropy temperature is automatically tuned during training to match a target entropy proportional to the size of the discrete action space, enabling adaptive exploration throughout learning. Table 4.2 presents the SAC-D high-level decision-making agent's hyperparameter settings.

Table 4.2. SAC-D high-level decision-making agent hyperparameter setting [115].

Hyperparameter	Value
Discount Factor	0.99
Soft Update Coefficient	0.005
Actor Learning Rate	0.0001
Critic Learning Rate	0.0003
Alpha Learning Rate	0.0003
Replay Buffer Size	1,000,000
Batch Size	64

4.3 Results

4.3.1 Simulation Results of the DDQN Agent in a Multi-Lane Highway Environment

Before evaluating the proposed framework in more complex urban traffic scenarios, a set of multi-lane highway experiments is first conducted to verify the feasibility of the overall hierarchical decision-making and control framework. At this stage, the DDQN algorithm is adopted as the high-level decision-making agent to validate the interaction between the high-level planner and the HOCLF-HOCBF-QP low-level controller.

Two highway scenarios with different levels of complexity are designed within the Highway-env simulator [85]. In both scenarios, the ego vehicle is required to travel from a predefined starting position to a designated destination while navigating through traffic containing multiple obstacles. To reach its destination safely, the vehicle must dynamically perform lane-changing maneuvers to avoid surrounding obstacles.

Unlike conventional trajectory-planning approaches, no predefined global reference path is provided. Instead, the DDQN-based high-level planner generates discrete lane-level maneuver commands, including lane keeping, left lane changes, and right lane changes, based on observations of the ego vehicle and the surrounding traffic environment. These high-level

decisions are then executed by the HOCLF-HOCBF-QP low-level controller, which guarantees accurate trajectory tracking while enforcing collision avoidance constraints throughout the maneuver.

In this experiment, the environment state is represented as a 5×5 array that encodes information about the ego vehicle and its surrounding obstacles. Each row corresponds to an entity (either the ego vehicle or an obstacle) and includes the following attributes: [presence, x, y, v_x, v_y], where presence is a binary indicator of whether the object exists in the current frame. The action space is discrete and consists of three possible maneuvers: idle (no lane change), left lane change, and right lane change. The reward function is designed to encourage goal-reaching behavior while penalizing unsafe or inefficient actions. A positive reward of +50 is assigned when the vehicle successfully reaches its destination. A large negative reward of −100 is applied in the event of a collision with any obstacle. Additionally, at each timestep, the agent receives a dense reward proportional to its forward progress (Δx), calculated as $r_{progress} = c * (x_{current} - x_{prev})$, where c is a positive coefficient, which encourages faster navigation. A small penalty of −0.5 is applied when the agent executes a lane change (either left or right), to discourage unnecessary lateral movement and to promote trajectory stability. Simulation results from both test cases show that the proposed framework can make the vehicle successfully navigate going from the start point to its destination while effectively avoiding all obstacles using the appropriate lane changing maneuver(s).

We first present the training progress of the high-level DDQN based decision-making agent in the proposed hierarchical control framework. Table 4.3 summarizes the key hyperparameters used in the DDQN training process and Figure 4.4 illustrates the training progress of the DDQN high-level decision-making agent in the three-lane complex autonomous driving environment. A low-pass filter is applied to smooth both the episode reward and step count curves for better interpretability. Initially, the agent exhibits poor performance due to high probability of random explorations, with total reward remaining negative and step count relatively low, indicating frequent collisions and early episode terminations. After approximately 2500 episodes, a significant improvement is observed: the total reward begins to rise rapidly, and the average number of steps per episode increases concurrently. This trend suggests that the agent gradually learns an effective lane-changing strategy to avoid obstacles and to extend its episode longevity. After about 3500 episodes, both rewards and steps per episode are maintained at a relatively high level, indicating convergence to a stable policy. Some fluctuations are still present in the reward curve, likely due to exploration behavior or occasional difficult scenarios.

Table 4.3. Hyperparameters and training settings [115].

Hyperparameter	Value
Replay buffer size	100,000
Mini-batch size	64
Learning rate	0.001

Target network update frequency	Every 100 steps
Exploration strategy (ε decay)	1.0 → 0.05 over 200,000 steps
Discount factor (γ)	0.99
Optimizer	Adam
Loss function	Mean Squared Error (MSE)

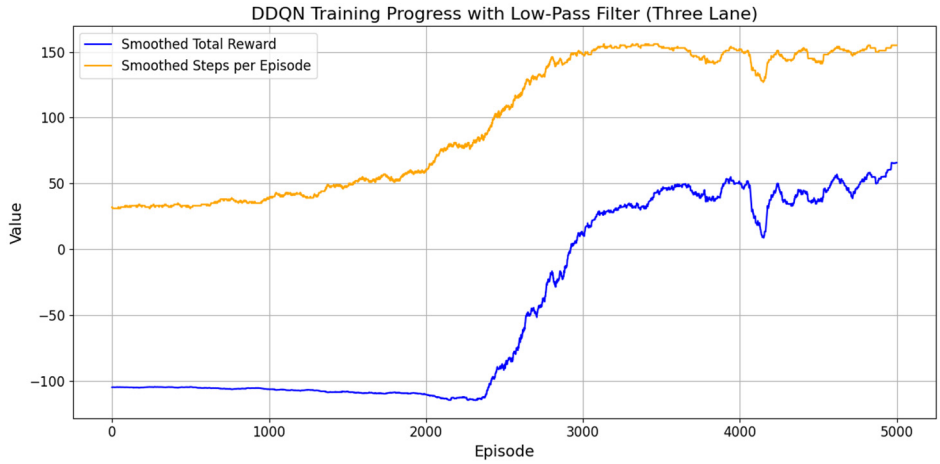

Figure 4.4. Deep reinforcement learning training progress [74].

Figure 4.5 illustrates the ego vehicle's trajectory in a two-lane highway test case using the proposed hierarchical controller. The road is divided into two lanes with clearly marked boundaries and centerline. The ego vehicle (represented by yellow rectangle) starts in the upper lane and initially follows a straight path before encountering an obstacle (represented by the red rectangle) positioned ahead in the same lane. The vehicle performs a lane-change maneuver in front of the obstacle by smoothly transitioning into the lower lane. After passing the obstacle, the vehicle performs another lane-change maneuver and returns to the upper lane, to avoid collision with the second obstacle.

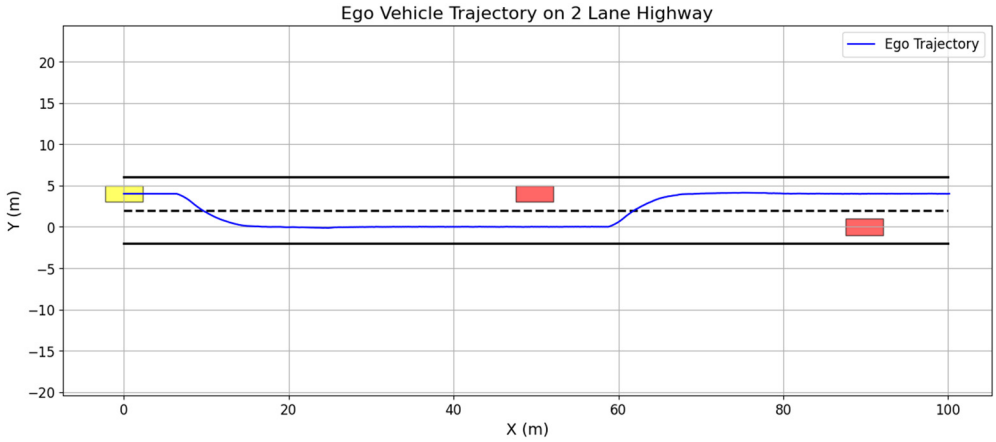

Figure 4.5. Hierarchical controller overall performance in simple test case [74].

Figure 4.6 illustrates the ego vehicle's trajectory in a complex three-lane highway test case using the proposed hierarchical controller. From the plot, we notice that this environment is much more complex compared to the simple two-lane test case demonstrated before. The ego vehicle (represented by the yellow rectangle) starts in the upper lane and initially follows a straight path before encountering an obstacle (represented by the red rectangle) positioned ahead in the same lane. The vehicle performs two consecutive lane-change maneuvers to transition smoothly into the lower lane in response to a series of obstacles. After passing the obstacles, the vehicle performs another set of two consecutive lane-change maneuvers and returns to the upper lane, to avoid collision with other obstacles.

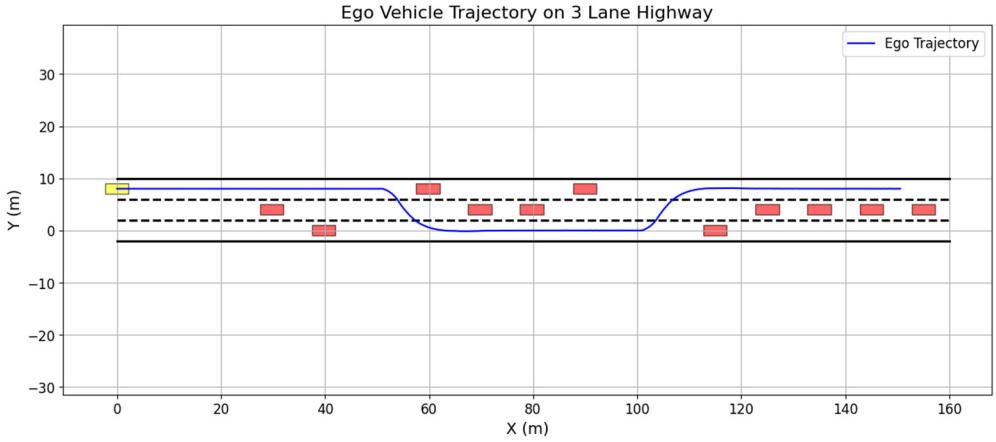

Figure 4.6. Hierarchical controller overall performance in complex test case [74].

The two path plots in Figures 4.5 and 4.6 demonstrate the effectiveness of the proposed hierarchical control system. The high-level DDQN agent can correctly generate lane-change decisions based on obstacle positions, while the low-level HOCLF-HOCBF-QP controller ensures safe and stable execution. The path remains continuous and collision-free throughout the simulation, indicating successful integration of decision-making and control components. Moreover, the vehicle consistently follows a straight path in obstacle-free regions, without performing unnecessary lane-change behaviors. This suggests that the penalty design for lane changes successfully discourages unnecessary lane-changing actions, which makes the decision-making process more efficient.

To evaluate the computational efficiency of the proposed hierarchical control framework, we conducted computational cost tests based on the aforementioned simulation test case. In complex test case settings, the low-level control simulation frequency is 100 Hz and high-level policy simulation frequency is 5 Hz. The low-level HOCLF-HOCBF-QP controller requires an average solve time of approximately 0.66 ms per step using the Gurobi optimizer while the high-level DDQN decision-making agent requires an average computational time of approximately 0.60 ms per step. These tests were performed in a Google Colab CPU environment which is using a

single-threaded Intel Xeon processor without GPU acceleration. Even under this relatively limited computational setting, the solve times accounts for only 6.6% of the low-level control cycle (10 ms) and 0.3% of the high-level decision-making cycle (200 ms), respectively. These results demonstrate that the proposed framework has good real-time capability. While it is expected that the computational time, particularly for the HOCLF-HOCBF-QP controller, will increase in more complex traffic scenarios with higher numbers of obstacles and constraints, our current results suggest that the proposed framework still provides sufficient real-time capability for typical autonomous driving tasks. In future work, we plan to further evaluate its effectiveness and real time capability by conducting tests using the Hardware-in-the-Loop (HIL) approach and Vehicle-in-Virtual-Environment (VVE) approach, to ensure its real-world feasibility and practical applicability in different driving scenarios.

The demo videos for the simulation result using the vehicle dynamics model are provided at: Two Lane Test Case Demo, Three Lane Test Case Demo, FARS230 Demo with HOCLF-HOCBF-QP Controller.

4.3.2 Roundabout and Intersection Simulation Results

4.3.2.1 Simulation Results

After the feasibility of the proposed hierarchical framework is verified in the multi-lane highway environment, the evaluation is further extended to more complex and safety-critical urban traffic scenarios. Compared with structured highway driving, roundabouts and intersections require the ego vehicle to handle more complex traffic interactions, including merging, yielding, lane selection, and exit planning. Therefore, these environments provide more rigorous test cases for evaluating the robustness and decision-making capability of the proposed framework. In this stage, SAC-D is adopted as the primary high-level decision-making algorithm due to its stable training performance and strong exploration capability, while DDQN is also trained in the same environments to provide a baseline comparison.

To evaluate the proposed framework under challenging roundabout traffic conditions, two customized roundabout simulation environments are developed based on the “Roundabout Env” provided by the Highway-env library [85]. Both scenarios represent typical roundabout driving tasks, where the ego vehicle must merge into circulating traffic, interact safely with surrounding vehicles, and exit through a designated branch. Several background vehicles are included in the environment and controlled by the Intelligent Driver Model (IDM) car-following model to emulate dynamic traffic interactions [94]. Each episode lasts up to 15 seconds and terminates when the ego vehicle reaches the target exit, collides with another vehicle, or departs from the roadway. The observation space consists of kinematic states, including positions, velocities, and headings of the ego vehicle and neighboring vehicles in the global coordinate frame, which are used as inputs to the high-level decision-making agent.

The ego vehicle is controlled through the proposed two-layer hierarchical architecture. The high-level SAC-D decision-making module is responsible for selecting discrete driving maneuvers, such as merging into the roundabout, maintaining the current lane, or choosing an appropriate lane sequence toward the target exit. Once a high-level action is selected, the low-level controller executes the decision by generating and tracking a look-ahead point along the centerline of the selected lane. This tracking point is determined based on the current vehicle speed and the selected high-level maneuver. The low-level controller then computes the required steering and acceleration commands while respecting vehicle dynamic constraints and physical actuator limits. In this way, the high-level planner focuses on strategic decision-making, while the low-level controller ensures safe and accurate maneuver execution.

Training the SAC-D agent requires a carefully designed reward function to guide the agent toward safe and efficient driving behavior. The reward function assigns a large penalty for collisions, penalizes lane deviation, and provides a positive reward when the ego vehicle successfully reaches the designated exit. Driving efficiency is encouraged through a piecewise speed reward that favors travel within a desired speed range while discouraging both overly slow and overly aggressive driving. In addition, a small penalty is applied to lane-change actions to reduce unnecessary lateral maneuvers, and a step penalty is included to encourage task completion within a shorter time. Overall, this reward design enables the agent to learn a policy that balances safety, efficiency, and smoothness in complex roundabout traffic environments.

Table 4.4 presents the collision rates under different reward configurations across two traffic scenarios, namely the roundabout and intersection environments. It compares the proposed reward function with two ablated versions, where either the speed-related term or the lane-change penalty term is removed. As shown in Table 4.4, the proposed reward function consistently achieves relatively lower collision rates in both scenarios compared to the ablated variants. This demonstrates that both components play a critical role in ensuring safe and stable driving behavior. The speed-related term encourages the ego vehicle to maintain its velocity within the desired range. Meanwhile, the lane-change penalty term discourages unnecessary or frequent lateral maneuvers. Overall, the combination of these two reward components can significantly improve success rates in simulated traffic environments.

Table 4.4. Reward Design Analysis (Collision Rate) [115].

Traffic Scenarios	Proposed Reward	No Speed Term	No Lane Change Term
Roundabout Complex	19.3%	30.7%	30.3%
Intersection Complex	27.7%	37.7%	37.0%

Figure 4.7 shows the training progress of the SAC-D agent in the simple roundabout traffic scenario. The blue curve shows the smoothed total episode reward, while the orange curve represents the smoothed number of steps per episode. A low-pass filter is applied to both metrics

to eliminate high-frequency noise and better understand the underlying learning trend. At the early stage of training, the agent exhibits relatively low rewards and short episode lengths, indicating frequent failures or early terminations. As training progresses, both the total reward and the episode length increase rapidly, demonstrating that the agent quickly learns good driving strategy and improves its ability to complete the navigation without premature termination. After approximately 700 episodes, the training process enters a more stable phase. The total reward fluctuates around a high value with moderate variance, while the episode length remains consistently close to its upper bound. This indicates that the learned policy has gradually converged to a stable and near-optimal driving strategy, enabling the ego vehicle to navigate the roundabout safely and efficiently for most episodes. Although occasional reward fluctuations are observed, they are expected in stochastic environments due to exploration and varying traffic interactions.

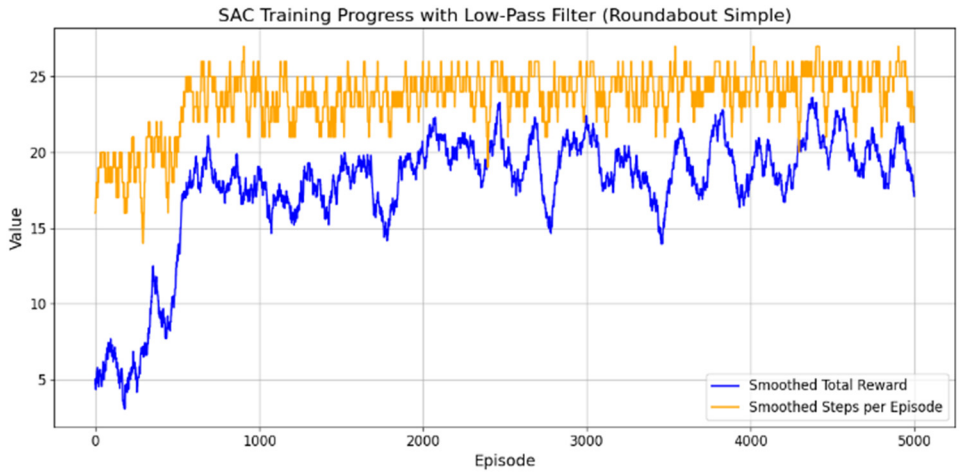

Figure 4.7. SAC-D training progress in simple roundabout environment [115].

Figure 4.8 illustrates the ego vehicle's path in the roundabout highway test case using the proposed hierarchical controller. Simulation results show that the ego vehicle can safely enter the roundabout traffic flow, avoid collisions while being surrounded by multiple vehicles, and consistently identify and travel toward the correct exit before the episode timeout. In addition, the vehicle maintains desired speed within the reward-optimal region for most of the route, while dynamically reducing speed only when yielding to other vehicles. The number of lane changes per episode remains minimal, confirming that the agent avoids unnecessary lane changing. It is also worth noting that the ego vehicle trajectories are smooth with no unnatural lateral jumps. Overall, simulation results demonstrate that the proposed hierarchical control framework enables the vehicle to learn how to navigate safely and efficiently in challenging roundabout environments, thereby validating the effectiveness of the proposed approach.

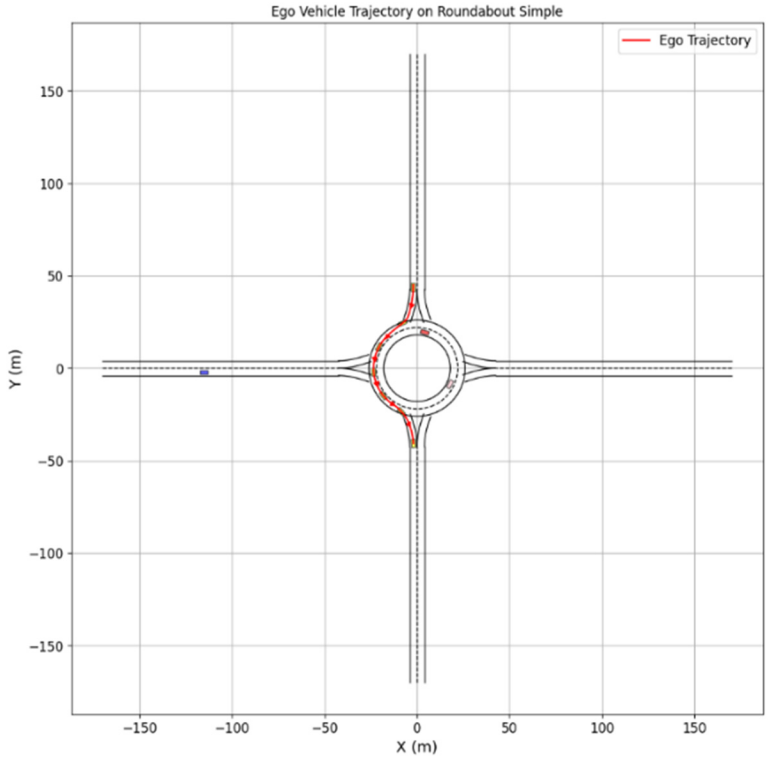

Figure 4.8. SAC-D On simple roundabout environment (Video: Roundabout Simple) [115].

A more complex roundabout environment is used to further evaluate the performance of the proposed control framework. Compared to the simpler configuration used earlier, this complex roundabout environment introduces a larger number of surrounding vehicles and additional entry and exit lanes. The other components of the control framework and experimental settings remain unchanged. During training, the hierarchical agent requires additional exploration due to the expanded decision complexity. Figure 4.9 shows the training progress of the SAC-D agent in the complex roundabout scenario. In contrast to the simple roundabout scenario, the agent exhibits a slower learning process during the early training phase. The total reward initially fluctuates at relatively low values, reflecting the increased task difficulty caused by more traffic interactions and more challenging decision-making requirements. As training progresses, the episode reward gradually increases and reaches a high-performance region while the episode length also improves and remains close to its maximum value for most episodes. However, compared with the simple scenario, the reward curve shows noticeably higher variance and intermittent performance drops, indicating that the agent must continuously adapt to more complex and stochastic traffic situations. Despite the increased training difficulty, the SAC-D agent eventually converges to a stable policy that consistently completes the roundabout navigation task.

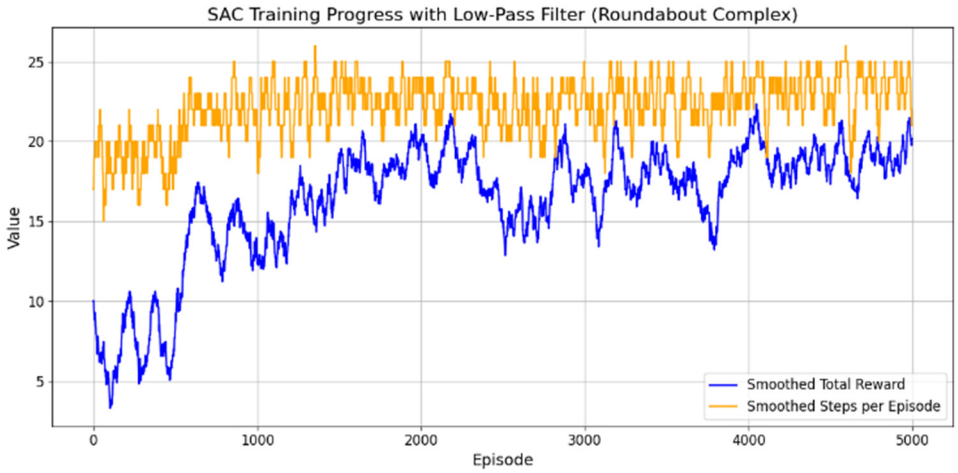

Figure 4.9. SAC-D training progresses in complex roundabout environment [115].

Figure 4.10 illustrates the ego vehicle's trajectory in the complex roundabout test case using the proposed hierarchical controller. Despite the increased complexity of the environment, the overall performance remains highly consistent with that observed in the simple roundabout. Simulation results show that the ego vehicle can safely merge into the roundabout traffic flow, avoid collisions even when surrounded by multiple vehicles, and consistently identify and navigate towards the correct exit before the episode timeout, without a significant change in performance. Similar to the simple roundabout case, the ego vehicle maintains its speed within the optimal range and its paths remain smooth with no unnatural lateral jumps. Overall, these results demonstrate that the proposed hierarchical control framework enables the ego vehicle to achieve safe and efficient navigation in both simple and complex roundabout environments.

To further evaluate the proposed hierarchical control framework beyond roundabout scenarios, we constructed two customized intersection environments based on the “Intersection Env” provided by the “highway-env library” [95]. These environments are designed to represent typical unsignalized urban intersections with increasing levels of traffic complexity. In both intersection scenarios, the ego vehicle is required to perform a left-turn maneuver, which is widely recognized as one of the most challenging tasks in urban driving due to potential conflicts with crossing traffic flows. The simple intersection case considers a structured and relatively sparse traffic setting, allowing the ego vehicle to complete the left turn with limited interactions. In contrast, the complex intersection case introduces more lanes, denser traffic and more frequent conflict with surrounding vehicles, requiring the ego vehicle to perform decision-making carefully to avoid the potential collision while reaching the destination.

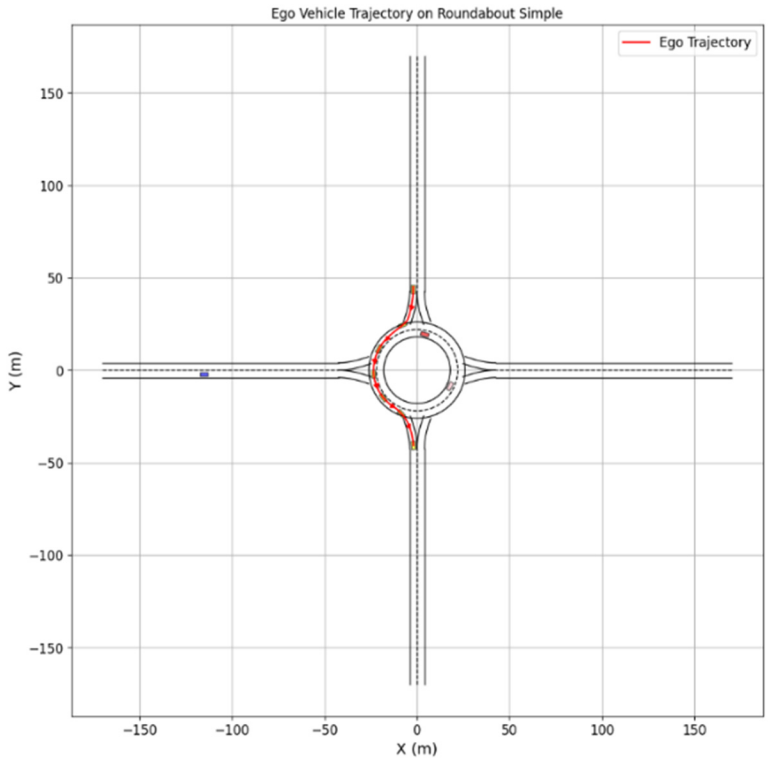

Figure 4.10. SAC-D On complex roundabout environment (Video: Roundabout Complex) [115].

Figure 4.11 illustrates the training progress of the SAC-D agent in the simple intersection traffic scenario. At the early stage of training, the agent achieves relatively low rewards and exhibits longer episode durations. This behavior is expected, as navigation in the single-lane intersection environment is relatively simple, allowing the ego vehicle to occasionally complete the task even under randomly selected actions. However, due to the lack of an efficient driving strategy, the agent typically requires more steps to reach the destination. As training progresses, the total episode reward increases rapidly, while the episode length gradually decreases and stabilizes at approximately 45 steps. This indicates that the agent quickly learns an effective driving strategy and significantly improves its efficiency in completing the navigation task. After approximately 400 episodes, the training process enters a more stable phase. The total reward fluctuates around a high value with moderate variance, and the episode length remains consistently close to a steady level. These results suggest that the learned policy has converged to a stable and near-optimal solution, enabling the ego vehicle to navigate the intersection safely and efficiently in most episodes. Occasional reward fluctuations are still observed, which are expected in a stochastic traffic environment due to exploration and varying interactions with surrounding vehicles.

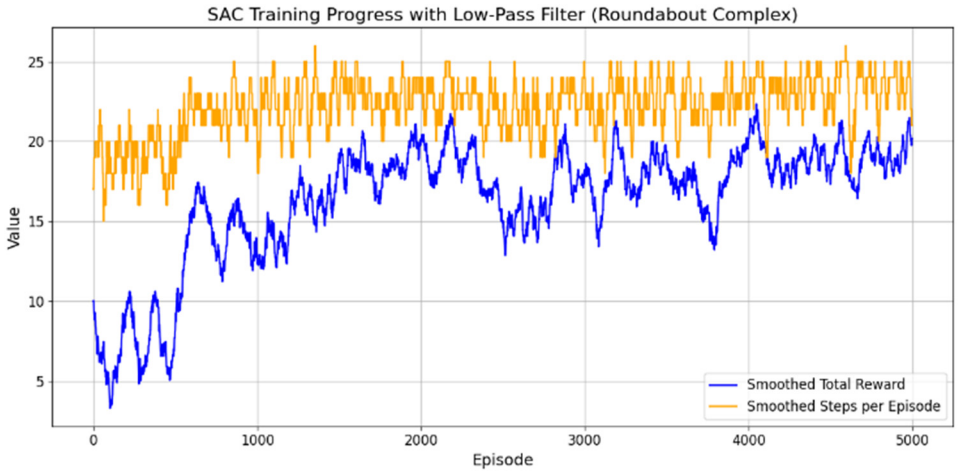

Figure 4.11. SAC-D training progresses in simple intersection environment [115].

Figure 4.12 illustrates the ego vehicle's trajectory in the simple intersection environment under the SAC-D policy. The resulting trajectory demonstrates smooth and consistent vehicle motion throughout the intersection. As shown in the figure, the ego vehicle is able to approach the intersection, execute the turning maneuver, and exit along the designated route without collisions. The learned policy enables the vehicle to maintain an appropriate speed profile for most of the trajectory, while adjusting its motion near the intersection center to ensure safe navigation. The absence of abrupt lateral deviations or oscillatory behavior indicates that the agent has learned a stable and efficient driving strategy. Overall, the observed trajectory confirms that the SAC-D agent can successfully complete the navigation task in the simple intersection scenario, providing evidence of effective policy convergence and reliable driving behavior in a structured traffic environment.

Figure 4.13 illustrates the training performance of the SAC agent in the complex intersection scenario. At the early stage of training, the agent exhibits relatively low rewards, indicating poor driving behavior under randomly initialized policies in the more challenging traffic environment. Compared to the simple intersection case, the complex intersection introduces denser traffic and more frequent conflicts with other vehicles, making it more difficult for the agent to complete the navigation task efficiently during early exploration. As training progresses, the total episode reward increases steadily and reaches a higher average level, while the episode length gradually stabilizes around 18–20 steps. This trend indicates that the agent successfully learns effective driving strategies in this challenging environment which allows it to pass the intersection more efficiently. After approximately 1000 episodes, the training process shows a relatively stable pattern. The total reward oscillates around a high value with moderate variance, and the episode length remains within a narrow range. These results indicate that the SAC agent converges to a stable and near-optimal policy which can handle complex traffic interactions. Occasional reward drops are still observed, which are expected due to the stochastic nature of traffic flow and exploration noise, but they do not significantly affect the overall convergence behavior.

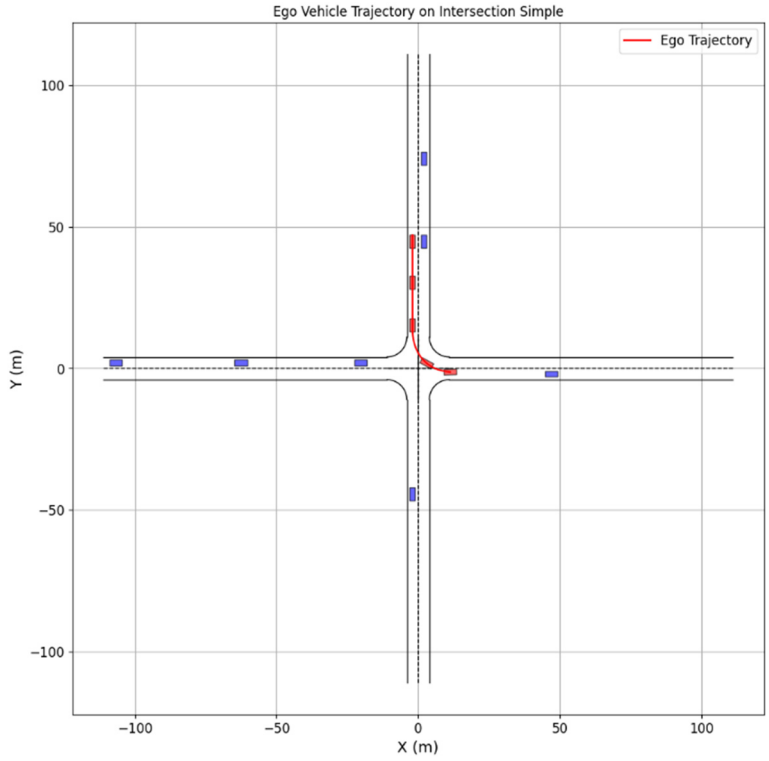

Figure 4.12. SAC-D On simple intersection environment (Video: Intersection Simple) [115].

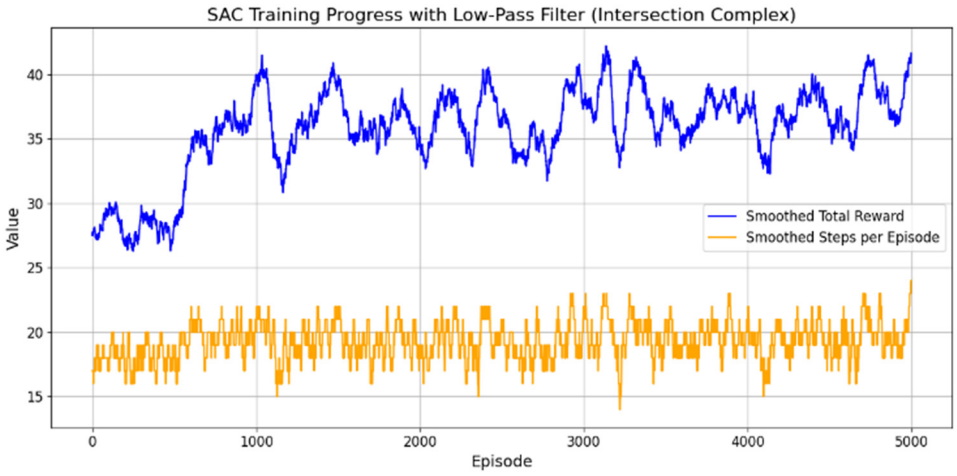

Figure 4.13. SAC-D training progress in complex intersection environment [115].

Figure 4.14 shows the ego vehicle trajectory in the complex intersection scenario under the learned SAC-D policy. The ego vehicle approaches the intersection from the north entry, executes the required left-turning maneuver, and reaches the target exit while maintaining a smooth path. Compared with the simple intersection case in Figure 4.10, the complex intersection introduces denser traffic flows. As a result, the ego vehicle in the complex setting requires more careful decision-making near the intersection center to ensure safe navigation. Despite the presence of multiple surrounding vehicles in conflicting streams, the ego vehicle successfully completes the

navigation task without collisions, indicating that the learned policy can handle interactive decision-making under complex traffic conditions. The path also remains well aligned with the lane centerline and exhibits no abrupt lateral oscillations. Overall, these results demonstrate that the proposed hierarchical control framework achieves robust and scalable performance, enabling the ego vehicle to navigate safely and efficiently in both simple and complex intersection environments.

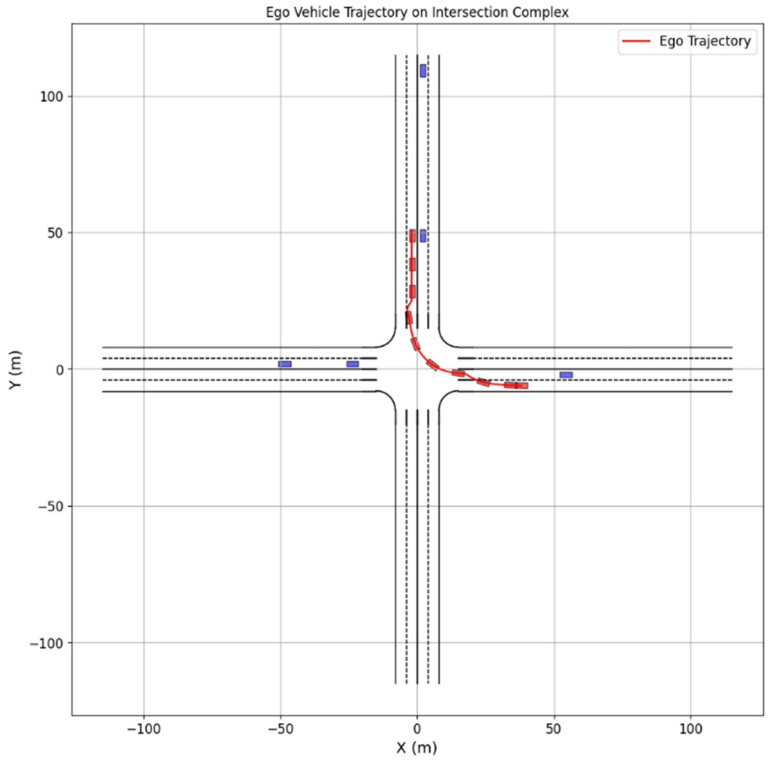

Figure 4.14. SAC-D On complex intersection environment (Video: Intersection Complex) [115].

To further evaluate the robustness of the proposed control framework under challenging traffic conditions, a dense traffic roundabout scenario is specifically constructed as a worst-case execution environment. This scenario introduces significantly increased traffic density and multi-directional vehicle interactions, resulting in frequent interaction with surrounding vehicles and limited maneuvering space. Figure 4.15 illustrates the training performance of the SAC agent in this dense roundabout environment. At the early stage of training, the agent exhibits relatively low rewards, indicating unstable and poor driving strategy under randomly initialized policies. Compared to simpler scenarios, the dense roundabout presents higher complexity due to continuous interactions with surrounding vehicles, making it difficult for the agent to complete the navigation task during initial exploration. As training progresses, the total episode reward increases steadily and reaches a higher average level, while the episode length gradually stabilizes. After approximately 1000 episodes, the training process becomes relatively stable. The total reward fluctuates around a high value with moderate variance, and the episode length remains within a

stable range. These results demonstrate that the SAC agent converges on a stable and near-optimal policy, capable of handling highly complex and interactive traffic conditions.

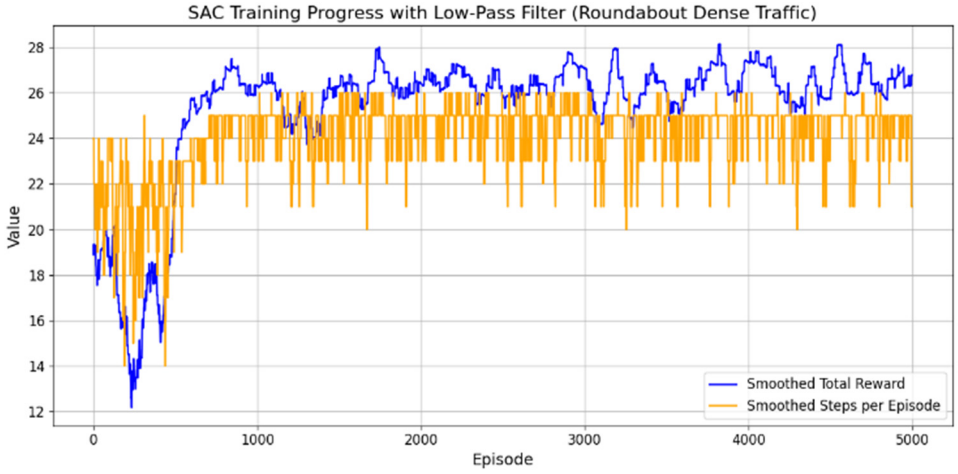

Figure 4.15. SAC-D training progress in roundabout with dense traffic [115].

Figure 4.16 shows an exemplary path of the ego vehicle in the dense traffic roundabout scenario. Despite the presence of multiple surrounding vehicles and frequent interactions, the ego vehicle is able to identify a feasible path in traffic and successfully navigate through the roundabout and reach the target exit. The resulting trajectory is smooth and collision-free, demonstrating effective decision-making and control under dense traffic conditions. This result indicates that the proposed framework can handle complex multi-vehicle interactions and can reliably guide the ego vehicle through dense traffic environments, further validating its effectiveness and robustness.

The complete demo video link with detailed explanation is provided here: Complete Demo Video.

4.3.2.2 HIL Setups and Testing Results

The HIL test is performed to further evaluate the real-time performance of the proposed control framework under challenging traffic conditions. Figure 4.17 illustrates the HIL setup for evaluating the proposed hierarchical control framework. In this configuration, the vehicle dynamic model implemented in Simulink runs in real time on the SCALEXIO simulation computer (bottom left). The proposed HOCLF-HOCBF-QP-based low-level controller is deployed on the MicroAutoBox (MABX, bottom right in the figure). Meanwhile, the virtual traffic environment and the SAC-based high-level decision-making agent are executed on a separate computer (top of the figure). These components communicate through the CAN bus and Ethernet using the UDP protocol, enabling comprehensive evaluation of the overall system within a real time traffic environment.

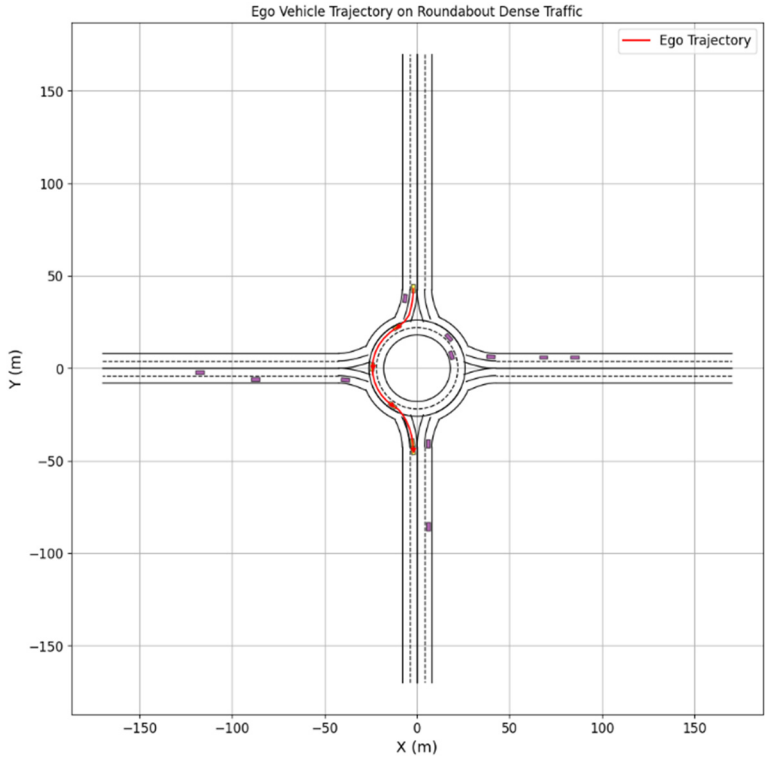

Figure 4.16. SAC-D On roundabout with dense traffic environment (Video: Roundabout Dense Traffic) [115].

Figure 4.17. HIL setup for evaluating the proposed hierarchical control framework [115].

Figure 4.18 illustrates the HIL test result of the proposed hierarchical control framework in the complex roundabout traffic environment. As shown in the figure, the experimental results are consistent with the simulation results. The ego vehicle is able to generate a smooth and collision-free path while interacting with surrounding traffic participants. Furthermore, the vehicle successfully navigates through the roundabout and reaches the designated destination without entering into unsafe regions with other road users. These results demonstrate the effectiveness of the proposed framework's real-time capability in handling complex traffic scenarios.

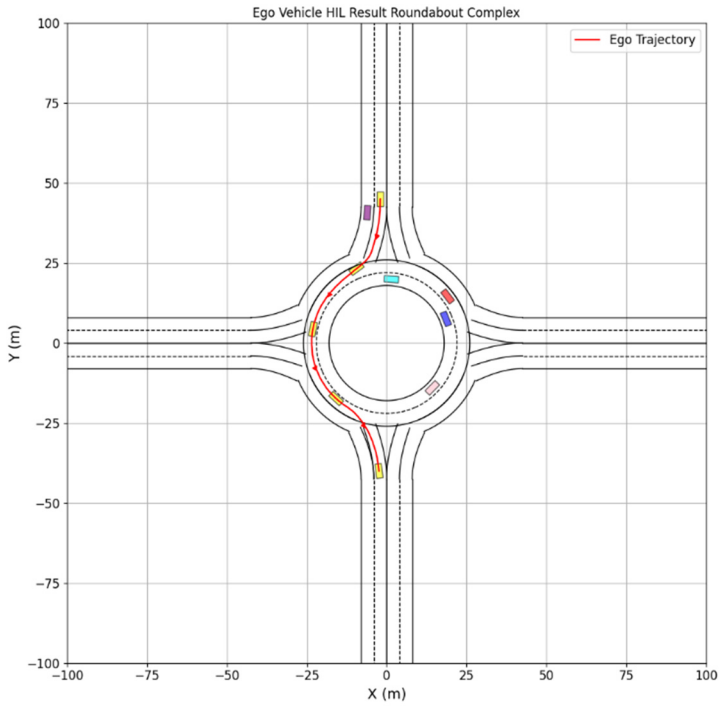

Figure 4.18. HIL test in complex roundabout traffic environment (Video: Roundabout Complex HIL) [115].

4.3.2.3 Comparison of Results

Table 4.5 summarizes the collision rates of different methods across various traffic scenarios, evaluated over 300 episodes. The results show that the proposed SAC+HOCLF-HOCBF-QP decision making consistently outperforms DDQN+PID and a random policy across all scenarios, achieving the lowest collision rates. In relatively simple scenarios, such as the roundabout simple case, both DDQN+PID and SAC+HOCLF-HOCBF-QP significantly improve over the random policy, with SAC+HOCLF-HOCBF-QP providing further performance gains. In more complex scenarios, such as intersections with dense traffic conditions, the performance gap becomes more pronounced. Also, in the intersection scenarios, DDQN+PID performs similarly to the random policy, indicating its limited capability in handling complex interactions. In contrast, SAC+HOCLF-HOCBF-QP maintains significantly lower collision rates, demonstrating better robustness and adaptability. These results suggest that SAC+HOCLF-HOCBF-QP is more effective in learning stable and safe policies in highly interactive environments, while DDQN+PID struggles to generalize under increased complexity.

Table 4.5. Collision Rate Table [115].

Traffic Scenarios	Random	DDQN+PID	SAC+HOCLF-HOCBF-QP
Roundabout Simple	44.3%	19.3%	13.7%
Roundabout Complex	54.0%	22.7%	19.3%
Intersection Simple	36.3%	36.7%	21.0%
Intersection Complex	56.3%	52.3%	27.7%

Table 4.6 presents the average task completion times of different methods across various traffic scenarios. It can be observed that the proposed method (SAC + HOCLF-HOCBF-QP) generally requires longer time to complete the task compared to SAC+PID and DDQN+PID. This behavior is mainly due to the safety constraints enforced by the HOCBF-based controller. In potentially unsafe situations, the controller modifies the control inputs to ensure constraint satisfaction, such as reducing speed, delaying lane changes, or maintaining safer distances from other vehicles. As a result, the vehicle may choose a more conservative driving strategy, leading to increased completion time. However, this reflects a trade-off between safety and efficiency, where the proposed method prioritizes safe and feasible behaviors over aggressive or risky actions.

Table 4.6. Task Completion Time [115].

Traffic Scenarios	SAC+PID	DDQN+PID	SAC+HOCLF-HOCBF-QP
Roundabout Simple	4.904 s	4.774 s	12.8 s
Roundabout Complex	4.636 s	4.946 s	7.8 s
Intersection Simple	4.014 s	4.012 s	5.2 s
Intersection Complex	6.820 s	6.854s	16.8 s

4.4 Chapter Conclusions and Future Work

This chapter presented a hierarchical autonomous driving decision-making and control framework that integrates deep reinforcement learning (DRL)-based high-level planning with a HOCLF-HOCBF-QP-based low-level controller. The proposed architecture separates strategic decision-making from safety-critical control execution. The high-level DRL agent generates discrete lane-level maneuver decisions, while the low-level controller ensures accurate trajectory tracking and collision avoidance through optimization-based stability and safety constraints.

The framework was first evaluated using a DDQN-based high-level decision-making agent in a multi-lane highway environment. This preliminary study verified the feasibility of the proposed hierarchical architecture, demonstrating that the high-level planner can generate appropriate lane-changing decisions for obstacle avoidance, while the low-level controller can safely and accurately execute these decisions. The DDQN-based results also confirmed the modularity of the framework, showing that the decision-making agent and low-level controller can be developed, tested, and improved independently.

After the feasibility of the hierarchical structure was validated, the study was further extended to more complex and safety-critical traffic environments, including roundabout and urban intersection scenarios. In these environments, SAC-D was adopted as the primary high-level decision-making algorithm, while DDQN was used as a baseline for performance comparison. Simulation results have demonstrated that the SAC-D-based planner, when integrated with the HOCLF-HOCBF-QP low-level controller, enables the ego vehicle to perform robust, efficient, and collision-free maneuvers in complex multi-agent traffic environments. Compared with the DDQN-

based agent, SAC-D provides stronger exploration capability and more stable policy learning, making it more suitable for challenging scenarios that require merging, yielding, lane selection, and exit planning.

The proposed hierarchical framework combines the advantages of learning-based decision-making and constraint-based control. Compared with traditional optimization-based methods such as MPC, the high-level DRL planner reduces the need to solve large-scale planning problems online and improves adaptability in dynamic traffic environments. Compared with end-to-end reinforcement learning approaches, the proposed framework provides better interpretability and safety assurance by explicitly separating decision-making from low-level control. The low-level HOCLF-HOCBF-QP controller acts as a safety-critical execution layer, ensuring that learned high-level decisions are implemented while satisfying stability and collision avoidance constraints.

Overall, the results confirm that the proposed DDQN/SAC-D + PID/HOCLF-HOCBF-QP hierarchical architecture provides a reliable, interpretable, and scalable solution for autonomous navigation in complex traffic environments. The DDQN-based experiments validate the basic feasibility and modularity of the framework, while the SAC-D-based experiments demonstrate improved decision-making performance in more challenging roundabout and intersection scenarios.

Future work will focus on more realistic and comprehensive validation of the proposed framework. First, Vehicle-in-Virtual-Environment (VVE)-based testing [96], [97] will be conducted to evaluate the framework under more realistic vehicle operation conditions. Second, high-fidelity nonlinear vehicle dynamics and nonlinear tire models will be incorporated into the simulation environment to further assess robustness under more complex driving conditions. Although the controller may still be designed based on a linearized vehicle model, its performance and safety will be evaluated under more realistic nonlinear dynamics. Third, the high-level action space will be further investigated, including refined or adaptive action discretization, to improve decision-making flexibility in complex traffic interactions. Finally, more sophisticated reward functions, sensitivity analyses, and ablation studies will be conducted to better understand the contribution of each module and further improve performance in dense and highly interactive traffic scenarios.

Chapter 5: Vehicle-in-Virtual-Environment (VVE) Approach for Autonomous Driving Function Evaluation

5.1 Introduction

The development and validation of advanced driver assistance systems (ADAS) and autonomous driving functions typically follow a progressive testing pipeline. In industry practice, this process usually begins with Model-in-the-Loop (MIL) simulations [98], followed by Hardware-in-the-Loop (HIL) experiments [99], [100], [101], [102], [103], proving ground testing, and finally public road testing. As the testing stage advances, the system gradually incorporates more physical hardware and operates under increasingly realistic driving conditions. MIL simulations provide a fully virtual environment for early-stage algorithm development, while HIL testing introduces physical electronic control units and real hardware components into the validation process that runs in real time. Both MIL and HIL methods rely on vehicle models with different levels of fidelity, including longitudinal, lateral, and vertical dynamics, to approximate the behavior of the real vehicle.

Although MIL and HIL testing are efficient and safe, their accuracy is inherently limited by the fidelity of the vehicle models and environmental assumptions used in simulation. In contrast, proving ground and public road testing provide higher realism because the actual vehicle is tested under physical driving conditions. However, these methods are costly, time-consuming, and difficult to use for evaluating rare or safety-critical traffic scenarios. Public road testing is especially inefficient for validating edge cases, since encountering rare hazardous events may require extensive driving over millions of miles. More importantly, public road testing of beta versions of ADAS or AV functions, especially as part of a series produced vehicle, may expose surrounding road users to unnecessary risk, raising significant safety and ethical concerns.

To address these limitations, this chapter introduces the Vehicle-in-Virtual-Environment (VVE) method as an intermediate validation approach between simulation-based testing and public road testing [96], [97], [104], [105], [106]. The key idea of VVE is to operate a real vehicle in a controlled physical environment while surrounding traffic participants, obstacles, and hazardous scenarios are generated virtually using a real timer operated edge or cloud-based game engine type simulator. It is also possible to incorporate real multiple actors (AVs, vehicles, pedestrians, bicyclists etc.) at separate locations safely into the shared virtual environment of VVE. In this way, VVE preserves the real vehicle dynamics and actuator responses that are difficult to fully capture in simulation, while avoiding the safety risks associated with testing dangerous scenarios on public roads. Compared with traditional MIL and HIL methods, VVE reduces the dependence on high-fidelity vehicle modeling because the actual vehicle is directly involved in the testing process. Compared with public road testing, VVE provides a safer, more repeatable, and more cost-effective way to evaluate autonomous driving functions under rare and high-risk traffic conditions.

Another important advantage of the VVE framework is its suitability for vehicle-to-everything (V2X) and vehicle-to-pedestrian (V2P) communication research. V2X includes all communication between the ego vehicle and the other road actors and includes V2P systems that are designed to improve vulnerable road user (VRU) safety by enabling information exchange between vehicles and pedestrians or bicyclists through mobile devices, wireless communication units, or other connected technologies. By integrating V2P communication into the VVE framework, real pedestrians or bicyclists can be introduced into controlled test scenarios using the synchronized motion of their virtual avatars that the real vehicle responds to. This enables safe and repeatable evaluation of V2P-enabled collision avoidance functions without placing real vulnerable road users in dangerous situations.

In addition to presenting the VVE testing framework, this chapter also discusses the simplified modeling of the real test vehicle for controller design. Although VVE testing uses the actual vehicle during validation, a simplified vehicle model is still required for designing and implementing model-based controllers. Therefore, this chapter introduces a simplified vehicle dynamic model that captures the essential longitudinal and lateral behaviors of the real vehicle while remaining tractable for controller development. Based on this model, the controller can be designed and then evaluated through VVE experiments using the actual vehicle in a controlled testing environment. This provides a practical bridge between model-based controller design and realistic vehicle-level validation.

5.2 Methodology

5.2.1 Vehicle-In-Virtual-Environment (VVE) Implementation

The overall VVE architecture is illustrated in Figure 5.1. In this framework, the real vehicle operates in a secure and open testing area, such as a parking lot, while its motion is synchronized with a corresponding virtual vehicle in a high-fidelity 3D virtual environment. The virtual environment can be flexibly modified to generate different road layouts, traffic participants, and safety-critical scenarios. Instead of relying on physical onboard perception sensors, the real vehicle receives virtual sensor information generated from the perspective of the synchronized virtual vehicle. Based on these virtual sensor inputs, the onboard control unit computes the required control commands. These commands are then executed by the real vehicle, and the resulting real-vehicle motion is fed back to update the virtual vehicle, thereby closing the real-virtual feedback loop.

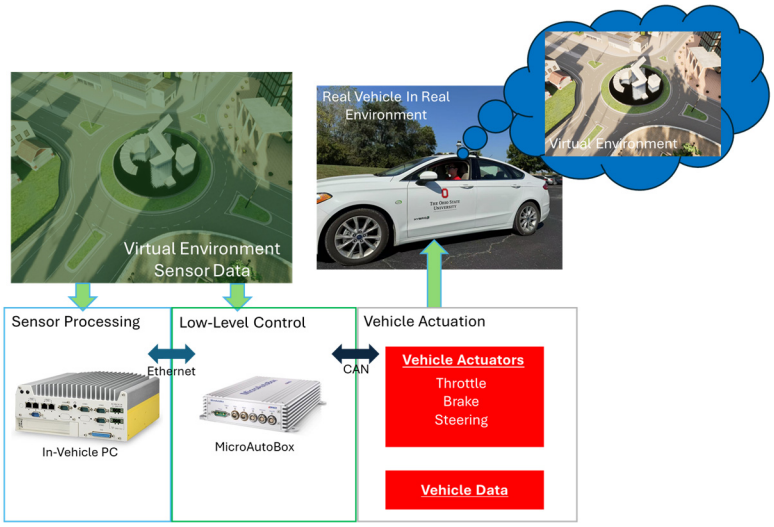

Figure 5.1. VVE Architecture [97].

The current implementation of the VVE system is shown in Figure 5.2, and the research autonomous driving vehicle used for VVE implementation are shown in Figure 5.3. The real vehicle is equipped with an RTK-GPS unit, OxTS xNAV500, together with dual antennas to provide accurate position and heading measurements. These measurements are received and processed by the onboard control unit, a dSPACE MicroAutoBox (MABX). The processed vehicle state information is then transmitted from the MABX to the onboard simulation computer through Ethernet UDP communication. The simulation computer runs a CARLA-based virtual environment. After receiving the real-vehicle GPS information, the virtual environment performs coordinate frame transformation and conversion to align the real vehicle motion with the virtual world coordinate system. Through this synchronization process, the virtual vehicle moves consistently with the real vehicle in the CARLA environment. Virtual perception sensors, such as radar and LiDAR, then generate measurements of surrounding virtual traffic participants and obstacles from the viewpoint of the virtual vehicle. The virtual sensor data generated are transmitted to the decision making computer which sends the low level actuator control commands to the MABX through Ethernet UDP communication. Different types of sensor information require different processing procedures before being used by the controller. Virtual perception measurements, such as radar and LiDAR data, can be directly sent to the decision-making computer. In contrast, virtual localization information, such as GPS-based position data, must first undergo inverse coordinate transformation and frame conversion before being transmitted back to the real vehicle control unit. This ensures that the localization information remains consistent between the real-world coordinate frame and the virtual environment coordinate frame. After receiving the virtual sensor information and synchronized vehicle states, the decision making computer makes the high level decisions and the MABX computes the actuator commands required by the autonomous driving controller. These commands are then sent to the real vehicle drive-by-wire system through the CAN bus to control vehicle motion. As the real vehicle moves, its updated position and heading are measured again by the RTK-GPS system and transmitted back

into the virtual environment, completing the closed-loop VVE testing process. Therefore, accurate real-virtual coordinate transformation is essential for maintaining synchronization between the real vehicle and the virtual vehicle. The details of the frame transformation and conversion procedures have already been presented in first and second year reports [4], [5].

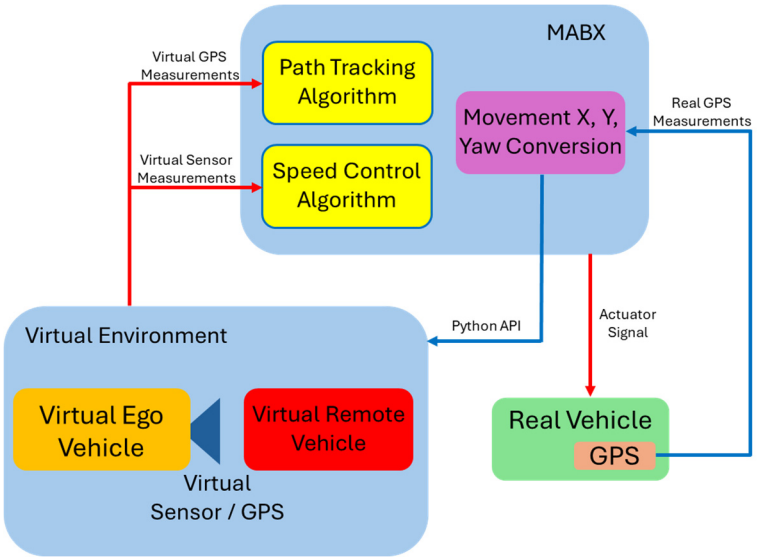

Figure 5.2. Implemented VVE Architecture [97].

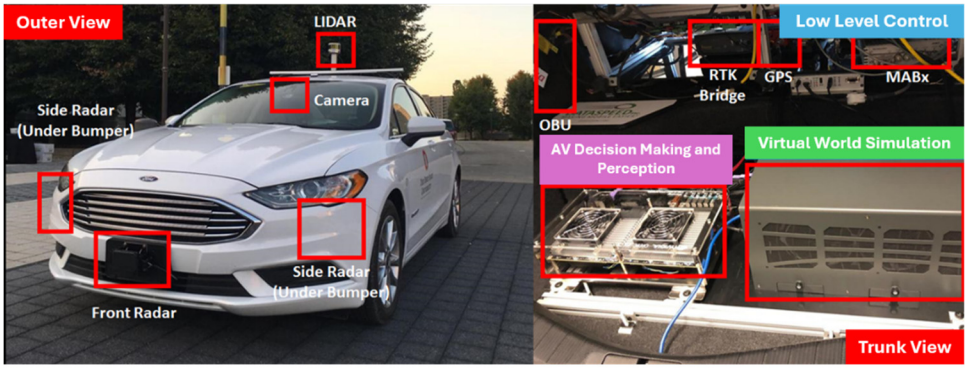

Figure 5.3. AV used for VVE implementation [97].

5.2.2 Vehicle-To-Pedestrian (V2P) Implementation

Vehicle-to-Pedestrian (V2P) communication is an important technology for improving vulnerable road user (VRU) safety, as it enables vehicles to obtain pedestrian-related information beyond the direct sensing range of onboard perception systems. When integrated with the Vehicle-in-Virtual-Environment (VVE) framework, V2P communication allows real pedestrian motion data to be introduced into the virtual environment. In this way, VRU-related safety functions can

be evaluated under controlled and repeatable conditions without exposing real pedestrians to dangerous traffic scenarios.

Figure 5.4 illustrates the basic concept of the integrated VVE-V2P testing framework. In the real world, the real vehicle and the real pedestrian operate in separate safe spaces. The real pedestrian carries a smartphone, which collects motion-related information such as GPS position and heading. Through frame transformation, the pedestrian's real-world motion is mapped into the virtual environment, where a corresponding virtual pedestrian is generated. At the same time, the real vehicle is synchronized with a virtual vehicle in the same virtual environment. As a result, although the real vehicle and real pedestrian remain physically separated, their virtual counterparts can interact within the same traffic scenario. This setup enables safe evaluation of pedestrian-aware autonomous driving functions and V2P-based collision avoidance strategies.

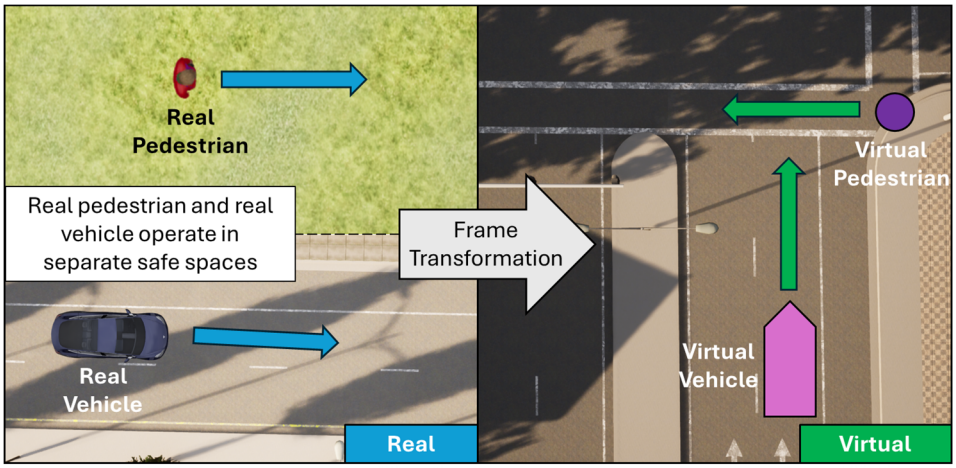

Figure 5.4. Conceptual Illustration of the Integrated VVE-V2P Testing Framework [97].

In our previous work, we used our own Android app and Bluetooth 5.x to communicate between smartphones. To make the approach more widespread and easier to develop and use for the students being trained, we decided to use the Matlab Mobile App as it allows us to easily program both iPhones and Android phones even though our previous Android app was much more capable. We also moved to an over-the-cloud server communication approach as compared to using Bluetooth 5.x, which considerably increases the location possibilities for the actors taking part in VVE. The low latency of this communication approach has been determined experimentally [107]. As shown in Figure 5.5, the proposed VVE-V2P system is mainly implemented through MATLAB-based tools. A commercial smartphone carried by the pedestrian is used as a mobile sensing device to collect real-time pedestrian information, including GPS position, yaw angle, and motion-related sensor measurements. These data are acquired through the MATLAB Mobile App and uploaded to MATLAB Cloud. The onboard computer connected to the VVE system then accesses the cloud data through the network and downloads the pedestrian information in real time.

In this way, the vehicle can obtain real-time pedestrian state information without relying solely on onboard perception sensors.

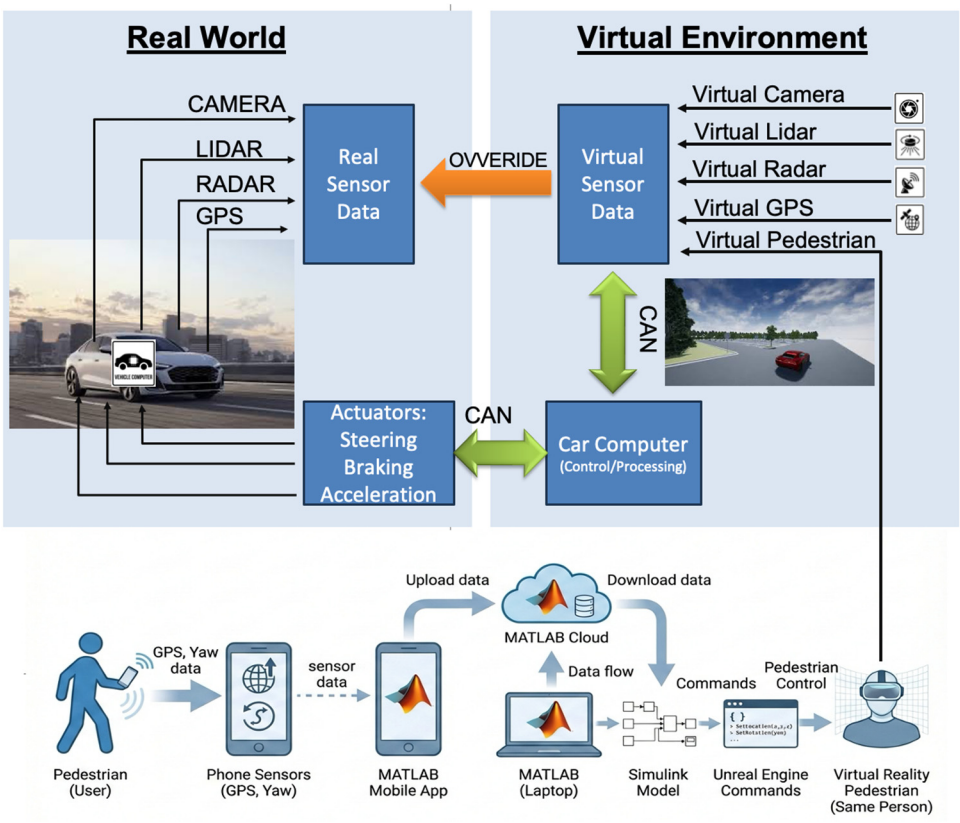

Figure 5.5. VVE-V2P Architecture Overview [107].

After pedestrian information is received by the onboard computer, it is transformed from the real-world geographic coordinate system into the virtual environment coordinate frame. The transformed pedestrian position and heading are then used to generate and control a virtual pedestrian in the VVE environment. Meanwhile, the real vehicle motion is synchronized with a virtual vehicle through the existing VVE real-virtual synchronization process. The virtual environment can then provide virtual sensor information, such as virtual camera, LiDAR, radar, GPS, and pedestrian state data, to the vehicle control system. Based on these inputs, the vehicle controller computes the corresponding control commands, which are executed by the real vehicle through its drive-by-wire actuator interface.

To obtain pedestrian motion information, the proposed V2P system utilizes the MATLAB Mobile application to access the internal sensors of a commercial smartphone carried by the pedestrian. As shown in Figure 5.6, MATLAB Mobile provides access to multiple sensor channels, including position, orientation, and inertial measurements. The position sensor provides the pedestrian's global location information, including latitude and longitude, together with the

horizontal accuracy radius. This accuracy value indicates the uncertainty of the GPS measurement and can be used to assess the reliability of the pedestrian position estimate.

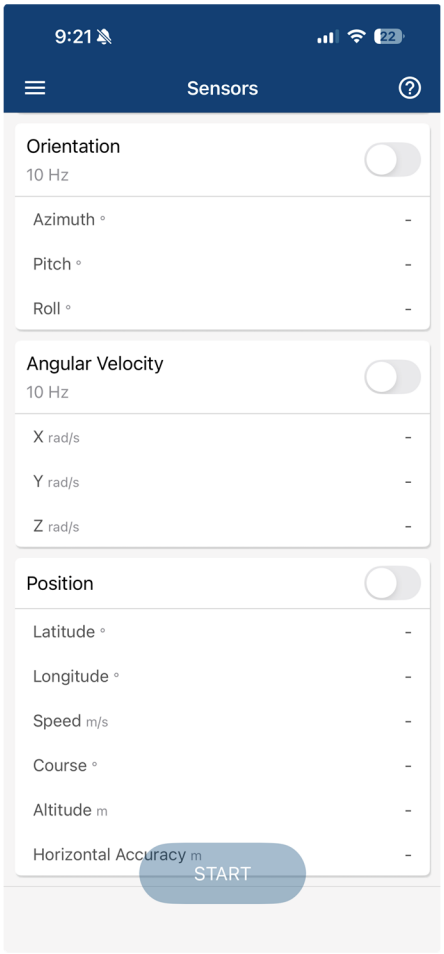

Figure 5.6. MATLAB Mobile App [107].

The orientation sensor provides the absolute heading of the smartphone relative to magnetic north, where the azimuth angle is used to approximate the pedestrian's facing direction. In addition, inertial measurements, such as acceleration along the X, Y, and Z axes, can be used to detect pedestrian motion patterns, identify walking steps, and estimate short-term velocity changes. Together, these sensor measurements provide the essential pedestrian state information required for V2P-based safety evaluation. Overall, the integrated VVE-V2P framework enables a real vehicle to interact with a virtual pedestrian whose motion is driven by real-world human movement data. This approach combines the realism of real pedestrian motion, the safety of virtual traffic interaction, and the repeatability of simulation-based testing. Therefore, it provides a practical platform for evaluating VRU related autonomous driving functions. It should be noted that the human actor motion in the Unreal Engine virtual reality environment had to be fine tuned in order to have realistic motion of the corresponding avatar and reduce jitter and floating feet. We are also working with using a camera and bones to relay human limb motion more realistically to the avatar.

5.3 Results

5.3.1 Vehicle-In-Virtual-Environment (VVE) Test Results

Figure 5.7 presents the synchronization result between the real vehicle and the virtual vehicle in the VVE framework. Figure 5.7(a) shows the vehicle trajectory obtained from the real-world RTK-GPS measurements after frame transformation, while Figure 5.7(b) shows the corresponding trajectory mapped into the virtual environment. The recorded trajectory is smooth and continuous, indicating that the vehicle position and heading information are consistently captured during the test. As shown in Figure 5.7(b), the transformed trajectory is successfully aligned with the road geometry in the virtual environment. The virtual vehicle follows the same overall motion pattern as the real vehicle, including the straight driving segment and the turning maneuver. The consistency between the real-world trajectory in Figure 5.7(a) and the virtual trajectory in Figure 5.7(b) demonstrates that the proposed frame transformation can accurately synchronize real vehicle motion with its virtual counterpart. The results demonstrate that satisfactory synchronization between the real-world and virtual-world motions has been achieved.

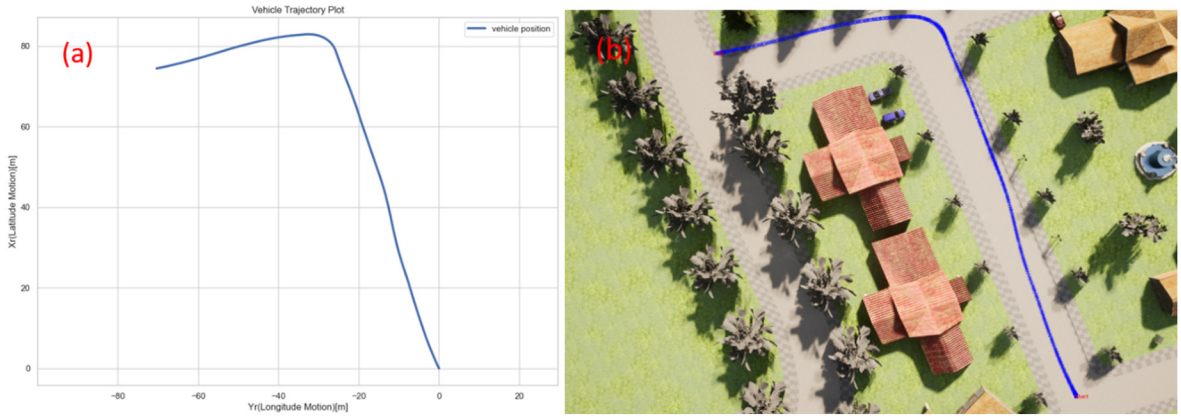

Figure 5.7. VVE synchronization test: (a) Motion trajectory in the real world; (b) Motion trajectory in the virtual world [97].

5.3.2 Vehicle-To-Pedestrian (V2P) Test Results

Figure 5.8 shows the pedestrian motion synchronization result between real world and virtual world. The red markers represent the raw GPS position updates obtained from the smartphone, while the blue curve denotes the smoothed pedestrian trajectory estimated by fusing the available sensor measurements through a Kalman filter–based approach. The green triangle indicates the initial pedestrian position. As shown in the figure, the raw GPS measurements exhibit noticeable scatter and noise, which is expected given the limited accuracy of smartphone sensors. If these raw measurements were used directly, the resulting pedestrian motion in the virtual

environment would be unstable and unrealistic. After applying the Kalman filter and combining multiple sensor sources, the estimated pedestrian trajectory becomes significantly smoother and more continuous, while still preserving the overall motion trend of the pedestrian.

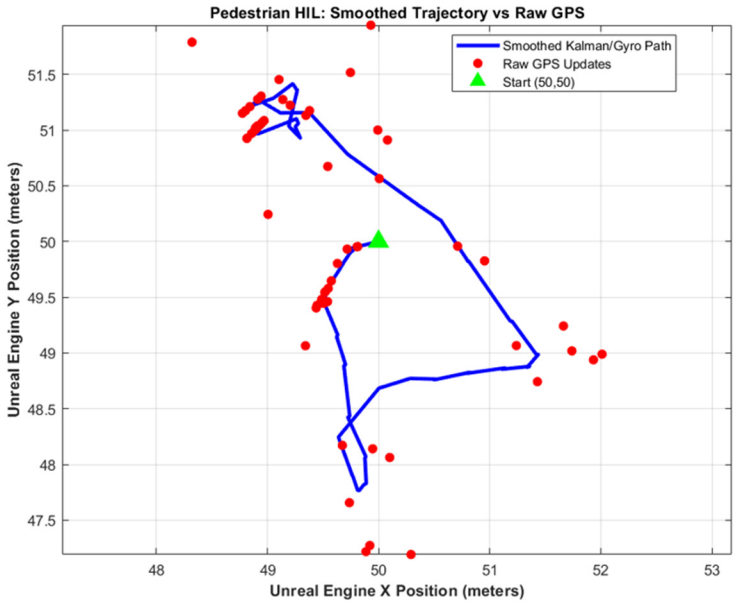

Figure 5.8. MATLAB Mobile App [107].

This result demonstrates that the proposed sensor fusion approach can effectively improve the quality of pedestrian localization for V2P applications. The filtered pedestrian position estimate is then transformed from the real-world coordinate frame into the Unreal Engine coordinate system through the frame transformation process. After this transformation, the estimated pedestrian motion can synchronize with virtual pedestrians. Therefore, the proposed framework enables the motion of a real pedestrian carrying a smartphone to be synchronized with a virtual pedestrian in the virtual environment, allowing safe interaction with the real vehicle within the VVE testing framework. Demo video link is provided at: V2P Synchronization Video.mp4.

5.3.3 Real Vehicle Longitudinal Testing

5.3.3.1 Real Vehicle Longitudinal Model Verification

To design a longitudinal speed tracking controller for the real testing vehicle used in the VVE framework, a simplified longitudinal vehicle model is first proposed and verified using experimental data. Although the VVE framework uses the actual vehicle during testing, a realistic and reliable vehicle model is still required for model-based controller development. Therefore, the longitudinal motion of the vehicle is approximated using a first-order system. The transfer function of the simplified longitudinal model is represented as:

$$G(s) = \frac{K}{\tau s + 1} \tag{5.1}$$

where K is the steady-state gain, and τ is the time constant of the system. In this study, a 10% throttle step response test is conducted while using the OxTS GPS system to record the real vehicle speed. Multiple experimental run data are collected to capture the vehicle's longitudinal response under the same throttle input. Figure 5.9 shows the recorded 10% throttle step response results and the fitted first-order model. The solid curves represent the measured vehicle speed profiles from different experimental runs, while the red dashed curve represents the fitted first-order response. Due to the uneven surface of the testing area, the experimental data exhibit two distinct response groups, corresponding approximately to slightly uphill and slightly downhill driving conditions. Based on the experimental data, the longitudinal model parameters are calculated using Equations 5.2 and 5.3:

$$K = \frac{v_{ss} - v_0}{u_{step}} \tag{5.2}$$

$$v(\tau) = v_0 + 0.632(v_{ss} - v_0) \tag{5.3}$$

As a result, the corresponding parameter is $K = 0.75$ and $\tau = 9$ s. Under a 10% throttle input, the identified model predicts a steady-state speed of approximately 7.5 m/s. As shown in the figure, the simplified first-order model captures the overall acceleration trend of the real vehicle. Although variations exist among different experimental runs due to road conditions, vehicle operating conditions, and measurement noise, the fitted model provides a reasonable approximation of the average longitudinal behavior. Therefore, the identified first-order model is suitable for controller design and preliminary validation within the VVE-based testing framework. A similar step response experiment is conducted to identify and verify the braking behavior of the real vehicle. In this test, a 25% brake command is applied while the vehicle is traveling at an initial speed of approximately 6.5 m/s. The vehicle speed is still recorded using the OxTS GPS system, and the collected data is used to fit a simplified first-order braking model.

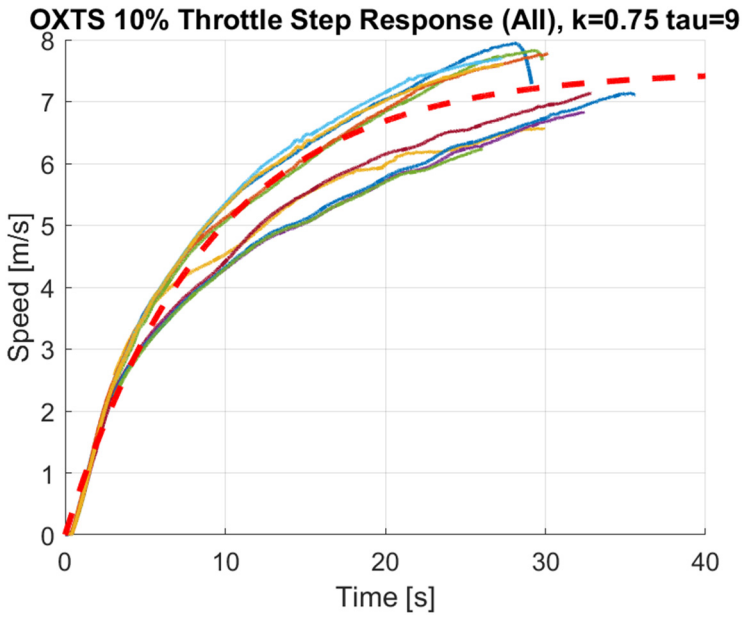

Figure 5.9. 10% throttle step response.

Figure 5.10 shows the recorded 25% brake step response results and the fitted first-order braking model. The solid curves represent the measured vehicle speed profiles from multiple experimental runs, while the red dashed curve represents the fitted first-order response. As shown in the figure, the vehicle speed decreases rapidly after the brake command is applied and then gradually approaches a low-speed steady-state region. Although variations exist among different experimental trials, the fitted first-order model captures the overall deceleration trend of the real vehicle. As a result, the corresponding parameter is $K = 0.23$ and $\tau = 10.82$ s.

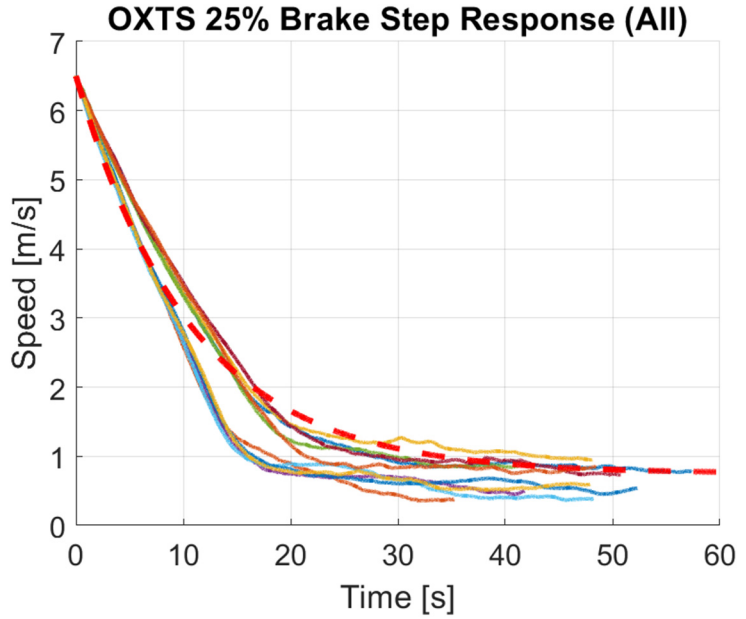

Figure 5.10. 25% brake step response.

Together with the 10% throttle step response test, the 25% brake step response experiment provides the essential longitudinal acceleration and deceleration pattern of the real vehicle. These first order simplified longitudinal models can be used to design longitudinal speed control and its preliminary MIL simulation validation before the controller is evaluated in the VVE testing framework.

5.3.3.1 Real Vehicle Longitudinal Controller Testing

Figures 5.11 and 5.12 show the step response results of the designed longitudinal speed controllers for throttle and braking, respectively. In Figure 5.11, the throttle controller is evaluated by commanding the vehicle speed to track a constant reference speed of 25 km/h (6.944 m/s). The response shows that the vehicle speed increases rapidly from rest, exhibits a moderate overshoot, and then gradually converges to the desired speed. The final response remains close to the reference value, indicating that the designed throttle controller can effectively regulate the vehicle speed with small steady-state errors. Figure 5.12 presents the braking controller response, where the vehicle speed is reduced from 25 km/h (6.944 m/s) to a lower reference speed of 15 km/h (4.167 m/s). The response decreases smoothly after the braking command is applied, with a slight undershoot before converging back to the desired speed. This behavior indicates that the braking controller can effectively reduce the vehicle speed and maintain stable tracking of the lower speed reference.

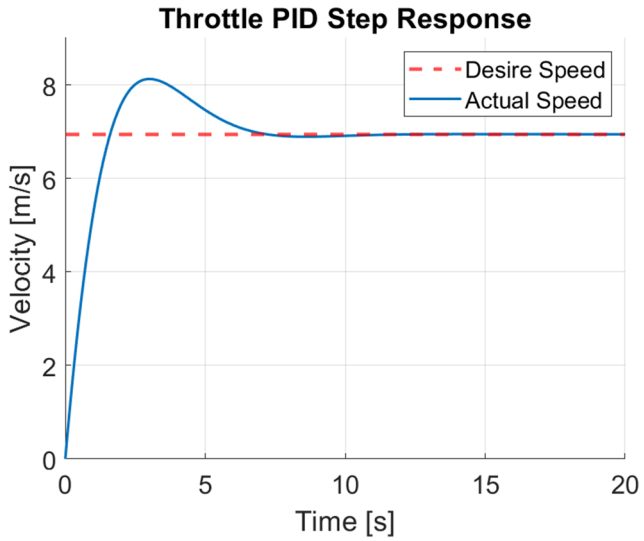

Figure 5.11. Throttle PID Step Response with $K_p = 7.5063 \times 1.6$ and $K_i = 7.5063$.

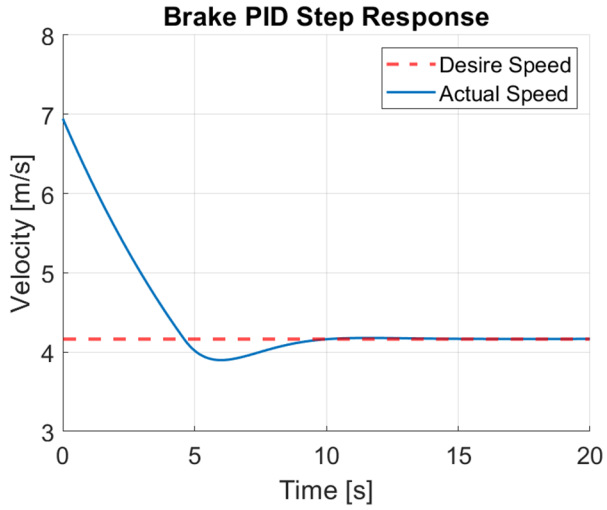

Figure 5.12. Brake PID Step Response with $K_p = 20.36 \times 1.9$ and $K_i = 20.36$.

Figure 5.13 presents the real-vehicle longitudinal PID speed tracking test results using a target speed of 15 km/h (4.17 m/s). The red dashed line represents the desired reference speed, while the solid curves represent the measured vehicle speed responses from multiple experimental runs. As shown in the figure, the vehicle initially accelerates rapidly from rest and reaches a peak speed above the reference value, indicating a noticeable overshoot during the transient response. After the overshoot, the controller gradually reduces the vehicle speed to desired value. Although small oscillations can be observed during the settling stage, the measured responses converge to the reference speed after approximately 20 seconds and remain close to the desired value for the rest of the experiment. The consistency among different experimental runs indicates that the designed longitudinal controller can help the real vehicle track desired speed. In general, the controller achieves small steady-state error and stable speed tracking after the transient phase, demonstrating its effectiveness for basic longitudinal control in the VVE testing framework.

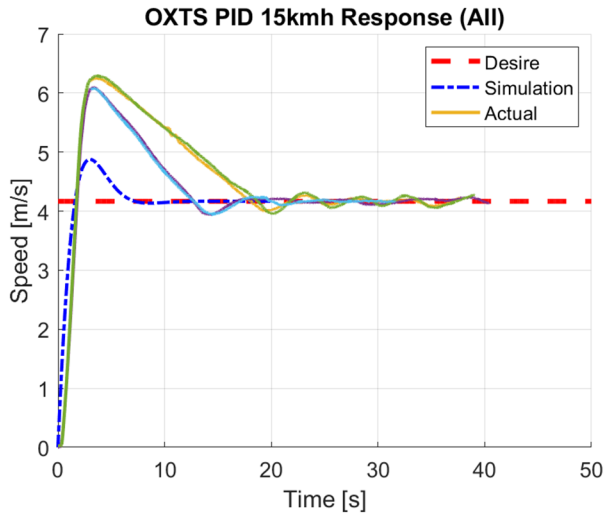

Figure 5.13. real-vehicle longitudinal PID testing.

5.3.4 Real Vehicle Lateral Testing

5.3.4.1 Real Vehicle Lateral Model Verification

In addition to the longitudinal vehicle model, a simplified lateral vehicle model is also required for lateral controller design in the VVE testing framework. Although the actual vehicle is used during VVE experiments, a tractable model is necessary for model-based controller development and preliminary validation. Therefore, a linear vehicle lateral dynamic model is adopted to describe the lateral motion and yaw dynamics of the real vehicle. The proposed model is described in chapter 3 Equation 3.1. The main vehicle parameters used in the lateral model are summarized in Table 5.1.

Table 5.1. Parameters of the linear vehicle lateral dynamic model used for controller design.

Symbol	Parameter	Value
V	Center-of-gravity (CG) velocity	4.17 m/s
m	Mass	1,998 kg
I_z	Yaw moment of inertia	3,728 kgm^2
C_f	Front tire cornering stiffness	189,000 N/rad
C_r	Rear tire cornering stiffness	167,000 N/rad
l_f	Distance between CG and front axle	1.30 m
l_r	Distance between CG and rear axle	1.55 m
W	Width of the vehicle	2.11 m
L_s	Look ahead distance	4 m

To verify the proposed lateral model, real-vehicle experimental data are collected. The measured yaw rate is selected as the primary validation signal because it directly reflects the vehicle's lateral response and turning behavior, and it can be reliably obtained from the OxTS GPS system. The steering input and longitudinal speed measured from the real vehicle are applied to the lateral vehicle model, and the simulated yaw rate response is compared with the measured yaw rate from the experiment. After tuning of the key lateral model parameters, the simulated yaw rate response shows good agreement with the measured yaw rate of the real vehicle as shown in Figure 5.14. The model provides a reasonable approximation of the real vehicle's lateral behavior for controller design. Based on this verification result, the proposed vehicle lateral dynamic model is used as the basis for subsequent lateral controller development. The controller is designed using this model and then evaluated within the VVE framework using the real vehicle, allowing the proposed control strategy to be tested under more realistic vehicle-level conditions.

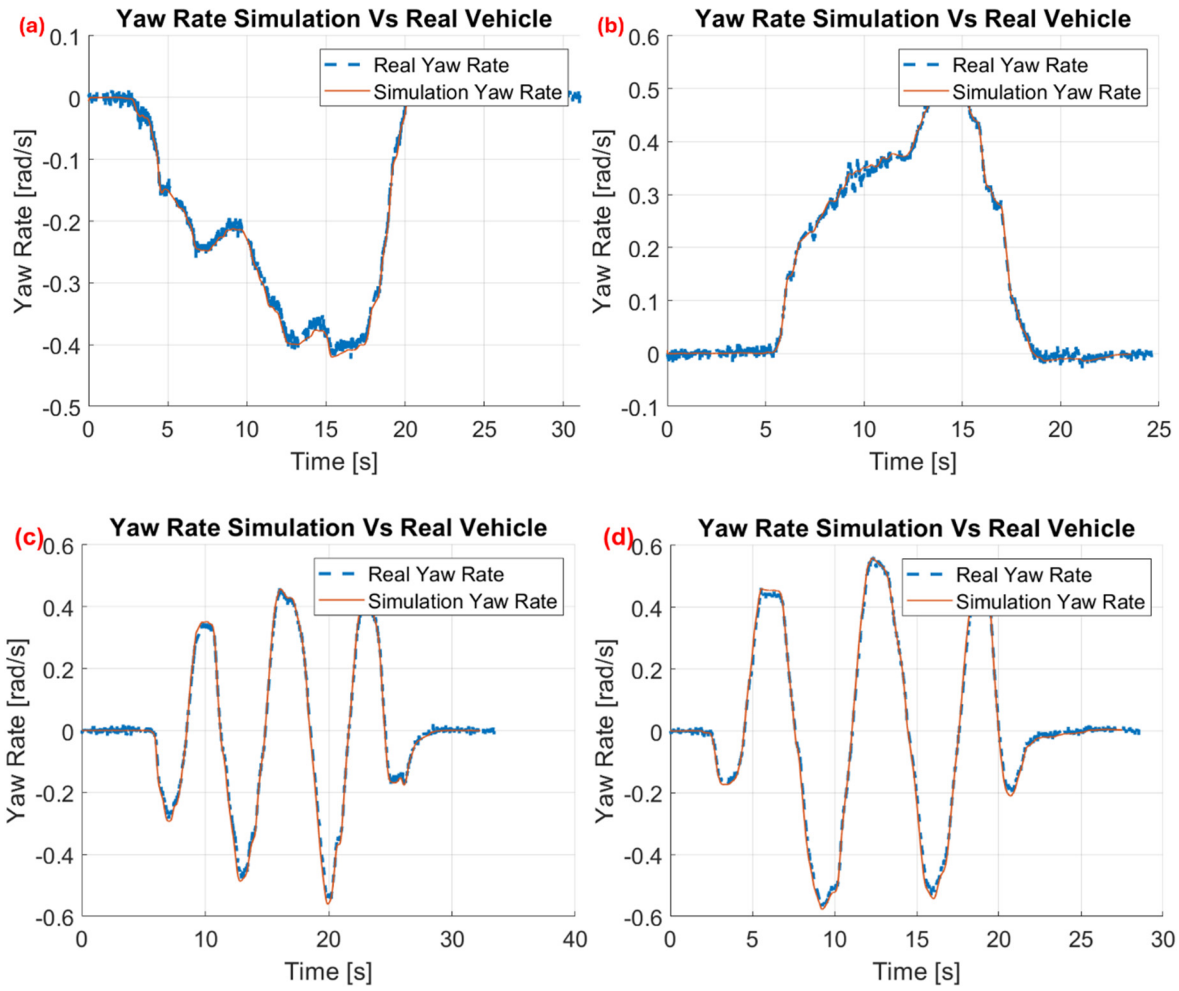

Figure 5.14. Yaw rate response comparison: (a) Right Turn; (b) Left Turn; (c) & (d) Slalom Motions.

5.3.4.2 Real Vehicle Lateral Controller Testing

In this project, the PD path tracking controller is designed using the parameter space approach, the details of which can be found in [108]. The parameter space approach is a systematic method employed in control design to understand how variations in control parameters impact system stability, performance, and robustness [109]. This approach is particularly useful when designing PID controllers. By adjusting the two control gains K_p and K_d, the poles of the system can be calculated and evaluated using D-stability which a subset of Hurwitz stability that requires poles to reside within a selected, bounded region in the left half of the plane. By utilizing the boundary crossing theorem, a set of control gains can be identified that ensure the closed-loop system meets D-stability criteria. Figure 5.15 shows the D stability region in the complex plane. There are five potential ways to cross the D-stability boundary. By plotting all five boundaries, an optimal region can be identified for selecting control gains.

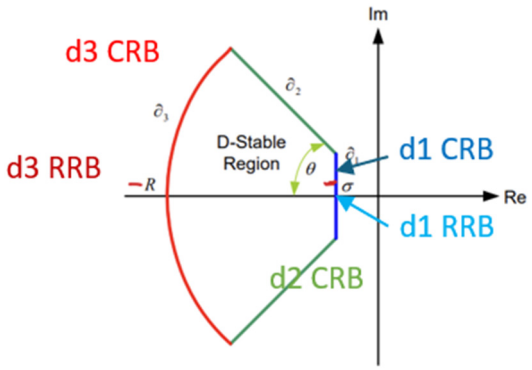

Figure 5.15. D-stability.

Figure 5.16 (a) displays the calculated optimal region satisfying D-stability using the parameter space approach for K_p and K_d. The control gains located within the D-stability boundary (upper right) ensure that the closed-loop system achieves D-stability. Based on this analysis, $K_p = 0.1$ and $K_d = 0.054$ are selected. After determining the controller gains K_p and K_d through the parameter-space method, the closed-loop poles corresponding to the selected gain were recalculated to verify D-stability of the system. As shown in Figure 5.16(b), the pair of system poles are located within the prescribed D-stability region, which indicates that the closed-loop system is asymptotically stable and satisfies the required constraints. Therefore, the pole-location analysis verifies that the K_p and K_d values selected using the parameter space approach can provide the expected closed-loop stability and transient performance.

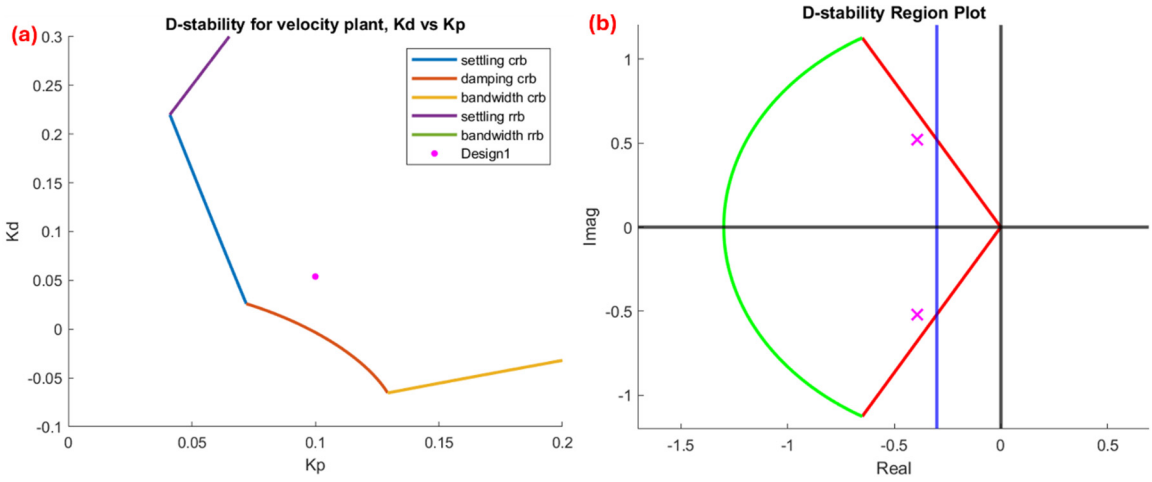

Figure 5.16. Parameter Space Design: (a) K_p and K_d Selection; (b) D-stability Check.

Figure 5.17 shows the lateral control experiment results using the designed PD controller. The blue dashed curve represents the reference trajectory, the red dashed curve represents the simulation result, and the solid curves represent the real-vehicle experimental data from multiple

trials. As shown in the figure, the real-vehicle trajectories show similar response patterns across different experimental runs, indicating good repeatability of the lateral control experiment. A small difference can still be observed between the simulation and real-vehicle results. Compared with the simulation response, the real-vehicle data exhibit a slight tracking delay, particularly during the lane-change maneuver, as well as a moderate lateral overshoot. The lateral position reaches a peak of approximately 5.5 m relative to the desired value of 4.5 m, corresponding to an overshoot of approximately 22%. Such discrepancies are expected in real-vehicle experiments because of actuator dynamics, communication and measurement delays [53], [110]. Also, the linear lateral dynamic model used for controller design and simulation cannot fully capture the real vehicle dynamics. Despite these differences, the real-vehicle trajectory remains close to the simulation result and gradually converges to the reference path after the transient response.

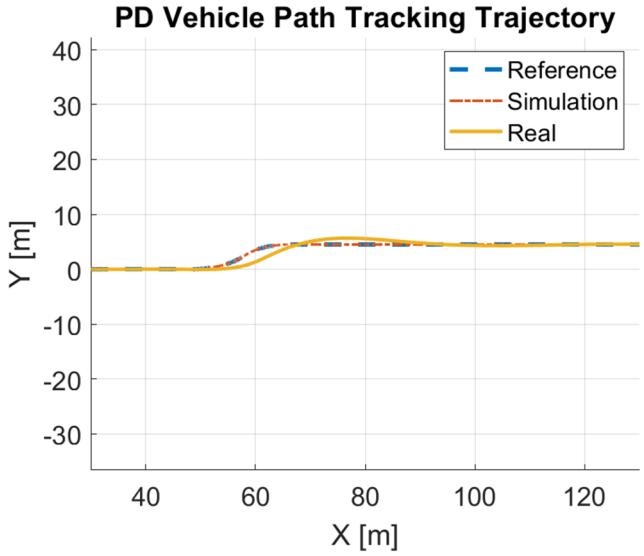

Figure 5.17. real-vehicle longitudinal PD testing.

Figure 5.18 presents the steering-wheel-angle signals recorded during the real-vehicle PD controller test. The yellow curve represents the steering angle calculated by the PD controller, the red curve represents the command sent to the steering actuator after applying the rate-limiting and saturation constraints, and the blue curve represents the actual steering angle measured from the actuator. The calculated steering input remains well within the specified upper and lower saturation limits throughout the experiment. The rate limiter effectively smooths rapid variations in the calculated input before it is sent to the actuator. Moreover, the measured steering angle closely follows the commanded signal, with only minor deviations and an estimated response delay of approximately 0.2 s. These results confirm that the calculated steering input is feasible for real-vehicle implementation and that the steering actuator provides satisfactory command-tracking performance.

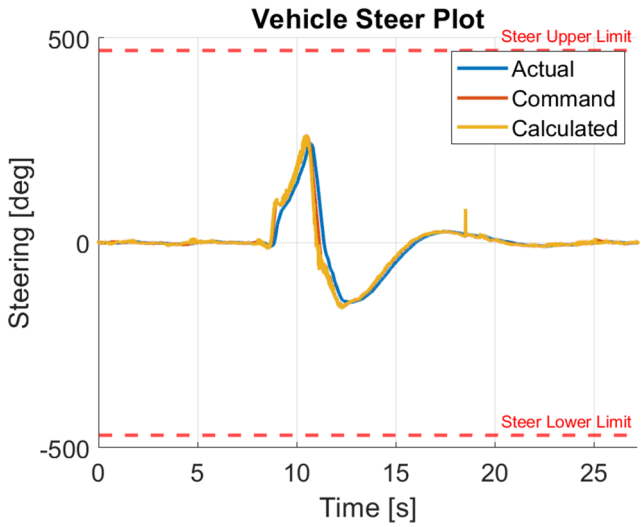

Figure 5.18. vehicle steering wheel plot.

Overall, these results demonstrate the feasibility of real-vehicle lateral control implementation, while also highlighting the need for further controller tuning and model refinement before applying the controller to more safety-critical VVE experiments. The comparison also confirms the importance of real-vehicle validation, since a controller that performs well in simulation may still exhibit different transient behavior on the actual vehicle.

5.3.5 VVE Collision Avoidance Algorithm Evaluation Demonstration

To demonstrate that the proposed virtual vehicle environment (VVE) has the capability to develop and evaluate autonomous driving functions, an additional VVE-based demonstration is introduced. In this demonstration, a stationary obstacle vehicle is placed in the virtual environment, while the ego vehicle performs a single lane-change maneuver to avoid a potential collision. The same maneuver is then conducted using the real vehicle, and the recorded vehicle states and tracking data are imported into the virtual environment for synchronized visualization and evaluation. A demonstration video is also provided to show the complete workflow and the interaction between the virtual traffic environment and the real vehicle. Video Link is provided here: video link.

5.4 Chapter Conclusions

This chapter presented the Vehicle-in-Virtual-Environment (VVE) framework for realistic, efficient and safe evaluation of autonomous driving functions. Compared with conventional testing methods such as Model-in-the-Loop (MIL), Hardware-in-the-Loop (HIL), proving ground testing, and public road testing, the VVE approach provides a practical intermediate solution that combines the realism of real-vehicle dynamics with the flexible virtual traffic environment. By operating the real vehicle in a large safe testing area while synchronizing its motion with a virtual vehicle in a

virtual environment, the VVE framework enables safety-critical scenarios to be evaluated without exposing real road users to unnecessary risk.

In addition, this chapter introduced the integration of Vehicle-to-Pedestrian (V2P) communication into the VVE framework for vulnerable road user safety evaluation. This integrated VVE-V2P framework allows pedestrian-aware autonomous driving functions to be tested safely and repeatedly without requiring a real pedestrian to be placed in dangerous vehicle interaction testing scenarios.

To design controller for real-vehicle implementation, simplified longitudinal and lateral vehicle dynamic models were also developed and verified in this chapter. For longitudinal dynamics, throttle and braking step response experiments were conducted, and simplified first-order models were adopted to capture the major acceleration and deceleration behavior of the real vehicle. Based on the first order longitudinal model, speed controllers were designed and tested through both simulation and real-vehicle experiments. The results showed that the controller can help ego-vehicle to track desire speed with acceptable steady-state tracking performance.

For lateral dynamics, a lateral vehicle dynamic model was proposed and verified using real-vehicle experimental data. The yaw rate response was selected as the primary validation signal because it directly reflects the vehicle's turning behavior. After tuning the key model parameters, the simulated yaw rate response showed reasonable agreement with the measured vehicle yaw rate, providing a basis for lateral controller design. A PD-based lateral controller was then evaluated through real-vehicle experiments. The results demonstrated the feasibility of real-vehicle lateral control implementation, although the observed overshoot and model mismatch also indicated the need for further controller tuning and model refinement [108], [109], [111], [112], [113], [114].

Chapter 6: Conclusions and Future Work

Ensuring the safety of vulnerable road users (VRUs), such as pedestrians and bicyclists, has already become a critical challenge that modern cities must confront. Autonomous driving systems provide a promising solution by reducing human-error-related accidents and improving the safety, efficiency, and reliability of transportation systems even in complex urban environments.

This research has aimed to address several key challenges in VRU safety by improving decision-making, vehicle control, and system evaluation. The main contributions of this research are summarized as follows:

(1) A HOCLF-HOCBF-QP-based low-level control strategy was proposed to simultaneously ensure accurate trajectory tracking and collision avoidance. The proposed controller improved both the safety and robustness of ego-vehicle control in complex traffic environments.

(2) A hierarchical autonomous driving framework was developed by integrating a deep reinforcement learning (DRL)-based high-level decision-making agent with the proposed HOCLF-HOCBF-QP low-level controller. The framework enabled safe, smooth, and efficient navigation of ego-vehicles. In addition, different DRL algorithms including SAC-D, DDQN, etc. were systematically compared and evaluated in this study.

(3) A Vehicle-in-Virtual-Environment (VVE) approach was introduced to evaluate autonomous driving systems under complex and high-risk traffic scenarios involving VRUs. VVE-based real-vehicle experiments are performed to demonstrate its effectiveness for realistic ADAS evaluation and validation.

In this project, the low-level controller was first developed based on the High-Order Control Lyapunov Function and High-Order Control Barrier Function Quadratic Programming framework. The HOCLF constraints ensured system stability and accurate path tracking, while the HOCBF constraints provided formal safety constraints to prevent the ego vehicle from entering unsafe regions around surrounding road users and obstacles. The controller was systematically evaluated through model-in-the-loop simulations, hardware-in-the-loop experiments, and representative FARS bicyclist crash cases simulations. These results demonstrated that the proposed controller could perform reliable and collision-free path tracking in challenging VRU-related traffic scenarios.

Building on the low-level controller, a hierarchical autonomous driving decision-making and control framework was developed. A DDQN-based high-level agent was first implemented in a multi-lane highway environment to verify the feasibility of hierarchical architecture and its ability to generate lane-level obstacle avoidance decisions. The study was then extended to more complex roundabout and intersection environments, where SAC-D was adopted as the primary

high-level decision-making algorithm and DDQN was used as a baseline for comparison. The results showed that the SAC-D-based planner, when integrated with the HOCLF-HOCBF-QP controller, can support safe, efficient, and robust autonomous navigation in complex multi-agent traffic environments.

This project also proposed the Vehicle-in-Virtual-Environment testing framework for realistic and repeatable evaluation of autonomous driving functions. The VVE system enabled a real vehicle operating in a controlled physical testing area to be synchronized with a virtual vehicle in virtual traffic environment. This allowed rare, complex, and safety-critical traffic scenarios to be tested without exposing real road users to unnecessary risk. In addition, a V2P testing framework was integrated into the VVE system. A pedestrian carrying a commercial smartphone provided real-time GPS, heading, and motion-related information through the MATLAB Mobile App and the MATLAB Cloud. After sensor fusion and coordinate frame transformation, the pedestrian motion was synchronized with the virtual pedestrian, enabling safe and repeatable evaluation of pedestrian-related autonomous driving functions. In addition to VVE-V2P integration, real-vehicle modeling and controller verification were conducted to support VVE-based autonomous driving tests.

Overall, this year's work provided a complete autonomous driving safety research framework including safety-critical controller design, DRL-based decision-making, simulation evaluation, HIL validation, VVE testing, V2P integration, and real-vehicle controller verification. The proposed methods improve the safety and robustness of autonomous driving systems in road traffic including VRU interaction scenarios.

We plan to focus on work zone safety in our year 4 project. Looking ahead, future work will focus on further improving the realism, scalability, and experimental validation of the proposed framework. High-fidelity nonlinear vehicle models and nonlinear tire models will be incorporated to better evaluate controller robustness under realistic vehicle dynamics. More advanced DRL algorithms will be investigated to improve performance in dense and highly interactive traffic scenarios. Additional sensitivity analyses and ablation studies will be conducted to better understand the contribution of each module. Finally, the VVE framework will be further extended with mixed-reality technologies, including XR-based pedestrian and driver interaction, to enhance the realism and repeatability of VRU safety experiments.

References

[1] World Health Organization, *Global status report on road safety 2015*. Geneva: World Health Organization, 2015. Accessed: Oct. 24, 2023. [Online]. Available: https://iris.who.int/handle/10665/189242

[2] P. M. Salmon, K. L. Young, and M. G. Lenné, "Investigating the Role of Roadway Environment in Driving Errors: An on Road Study," *Proc. Hum. Factors Ergon. Soc. Annu. Meet.*, vol. 55, no. 1, pp. 1879–1883, Sep. 2011, doi: 10.1177/1071181311551391.

[3] A. Medina, S. Lee, W. Wierwille, and R. Hanowski, "Relationship between Infrastructure, Driver Error, and Critical Incidents," *Proc. Hum. Factors Ergon. Soc. Annu. Meet.*, vol. 48, pp. 2075–2079, Sep. 2004, doi: 10.1177/154193120404801661.

[4] H. Chen, X. Cao, B. Aksun-Guvenc, and L. Guvenc, "Vehicle-in-Virtual-Environment (VVE) Method for Developing and Evaluating VRU Safety of Connected and Autonomous Driving with Year 2 Focus on Bicyclist Safety." Accessed: Jul. 25, 2026. [Online]. Available: https://rosap.ntl.bts.gov

[5] H. Chen, X. Cao, B. Aksun-Guvenc, and L. Guvenc, "Vehicle-in-Virtual-Environment (VVE) Method for Developing and Evaluating VRU Safety of Connected and Autonomous Driving." Accessed: Jul. 25, 2026. [Online]. Available: https://rosap.ntl.bts.gov

[6] S. Y. Gelbal, B. Aksun-Guvenc, and L. Guvenc, "Collision Avoidance of Low Speed Autonomous Shuttles with Pedestrians | International Journal of Automotive Technology | Springer Nature Link." Accessed: Jul. 28, 2026. [Online]. Available: https://link.springer.com/article/10.1007/s12239-020-0087-7

[7] S. Y. Gelbal, S. Arslan, H. Wang, B. Aksun-Guvenc, and L. Guvenc, "Elastic band based pedestrian collision avoidance using V2X communication," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2017, pp. 270–276. doi: 10.1109/IVS.2017.7995731.

[8] H. Wang, A. Tota, B. Aksun-Guvenc, and L. Guvenc, "Real time implementation of socially acceptable collision avoidance of a low speed autonomous shuttle using the elastic band method," *Mechatronics*, vol. 50, pp. 341–355, Apr. 2018, doi: 10.1016/j.mechatronics.2017.11.009.

[9] S. Y. Gelbal, S. Zhu, G. A. Anantharaman, B. A. Guvenc, and L. Guvenc, "Cooperative Collision Avoidance in a Connected Vehicle Environment," Jun. 02, 2023, *arXiv*: arXiv:2306.01889. doi: 10.48550/arXiv.2306.01889.

[10] S. Y. Gelbal, B. Aksun-Guvenc, and L. Guvenc, "Vulnerable Road User Safety Using Mobile Phones with Vehicle-to-VRU Communication," *Electronics*, vol. 13, no. 2, p. 331, Jan. 2024, doi: 10.3390/electronics13020331.

[11] X. Li, A. C. A. Doss, B. Aksun-Guvenc, and L. Guvenc, "Pre-Deployment Testing of Low Speed, Urban Road Autonomous Driving in a Simulated Environment," *SAE Int. J. Adv. Curr. Pract. Mobil.*, vol. 2, no. 6, Art. no. 2020-01–0706, Apr. 2020, doi: 10.4271/2020-01-0706.

[12] B. Wen, S. Y. Gelbal, B. Aksun-Guvenc, and L. Guvenc, "Localization and Perception for Control and Decision Making of a Low Speed Autonomous Shuttle in a Campus Pilot Deployment," SAE International, Apr. 2018. doi: 10.4271/2018-01-1182.

[13] S. Y. Gelbal, B. Aksun-Guvenc, and L. Guvenc, "SmartShuttle: a unified, scalable and replicable approach to connected and automated driving in a smart city," in *Proceedings of the 2nd International Workshop on Science of Smart City Operations and Platforms Engineering*, in SCOPE '17. New York, NY, USA: Association for Computing Machinery,

2017, pp. 57–62. doi: 10.1145/3063386.3063761.
[14] X. Li, S. Zhu, B. Aksun-Guvenc, and L. Guvenc, "Development and Evaluation of Path and Speed Profile Planning and Tracking Control for an Autonomous Shuttle Using a Realistic, Virtual Simulation Environment," *J. Intell. Robot. Syst.*, vol. 101, no. 2, p. 42, Feb. 2021, doi: 10.1007/s10846-021-01316-2.
[15] X. Li, S. Zhu, S. Y. Gelbal, M. Cantas, B. Aksun-Guvenc, and L. Guvenc, "A Unified, Scalable and Replicable Approach to Development, Implementation and HIL Evaluation of Autonomous Shuttles for Use in a Smart City", SAE International. Accessed: Jul. 28, 2026. [Online]. Available: https://www.sae.org/papers/a-unified-scalable-replicable-approach-development-implementation-hil-evaluation-autonomous-shuttles-use-a-smart-city-2019-01-0493
[16] O. Kavas-Torris, N. Lackey, and L. Guvenc, "Simulating the Effect of Autonomous Vehicles on Roadway Mobility in a Microscopic Traffic Simulator," *Int. J. Automot. Technol.*, vol. 22, no. 3, pp. 713–733, Jun. 2021, doi: 10.1007/s12239-021-0066-7.
[17] O. Kavas-Torris, S. Y. Gelbal, M. R. Cantas, B. Aksun Guvenc, and L. Guvenc, "V2X Communication between Connected and Automated Vehicles (CAVs) and Unmanned Aerial Vehicles (UAVs)," *Sensors*, vol. 22, no. 22, p. 8941, Jan. 2022, doi: 10.3390/s22228941.
[18] L. Guvenc, B. Aksun-Guvenc, S. Zhu, and S. Y. Gelbal, "Autonomous Road Vehicle Path Planning and Tracking Control | Wiley," Wiley.com. Accessed: Sep. 01, 2025. [Online]. Available: https://www.wiley.com/en-us/Autonomous+Road+Vehicle+Path+Planning+and+Tracking+Control-p-9781119747949
[19] H. Chen and B. Aksun-Guvenc, "Deep Reinforcement Learning Based Collision Avoidance of Automated Driving Agent," SAE International, Apr. 2024. doi: 10.4271/2024-01-2556.
[20] H. Chen, X. Cao, L. Guvenc, and B. Aksun-Guvenc, "Deep-Reinforcement-Learning-Based Collision Avoidance of Autonomous Driving System for Vulnerable Road User Safety," *Electronics*, vol. 13, no. 10, Art. no. 10, Jan. 2024, doi: 10.3390/electronics13101952.
[21] S. Velupillai and L. Guvenc, "Laser Scanners for Driver-Assistance Systems in Intelligent Vehicles [Applications of Control]," *IEEE Control Syst. Mag.*, vol. 29, no. 2, pp. 17–19, Apr. 2009, doi: 10.1109/MSP.2008.931716.
[22] İ. Altay, B. Aksun Güvenç, and L. Güvenç, "Lidar Data Analysis for Time to Headway Determination in the DriveSafe Project Field Tests," *Int. J. Veh. Technol.*, vol. 2013, no. 1, p. 749896, 2013, doi: 10.1155/2013/749896.
[23] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," Oct. 22, 2014, *arXiv*: arXiv:1311.2524. doi: 10.48550/arXiv.1311.2524.
[24] R. Girshick, "Fast R-CNN," Sep. 27, 2015, *arXiv*: arXiv:1504.08083. doi: 10.48550/arXiv.1504.08083.
[25] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," Jan. 06, 2016, *arXiv*: arXiv:1506.01497. doi: 10.48550/arXiv.1506.01497.
[26] P. Sun *et al.*, "Sparse R-CNN: End-to-End Object Detection with Learnable Proposals," Apr. 26, 2021, *arXiv*: arXiv:2011.12450. doi: 10.48550/arXiv.2011.12450.
[27] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," May 09, 2016, *arXiv*: arXiv:1506.02640. doi: 10.48550/arXiv.1506.02640.
[28] Q. Liu, H. Ye, S. Wang, and Z. Xu, "YOLOv8-CB: Dense Pedestrian Detection Algorithm

Based on In-Vehicle Camera," *Electronics*, vol. 13, no. 1, Art. no. 1, Jan. 2024, doi: 10.3390/electronics13010236.
[29] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "PointPillars: Fast Encoders for Object Detection from Point Clouds," May 07, 2019, *arXiv*: arXiv:1812.05784. doi: 10.48550/arXiv.1812.05784.
[30] W. Zheng, W. Tang, S. Chen, L. Jiang, and C.-W. Fu, "CIA-SSD: Confident IoU-Aware Single-Stage Object Detector From Point Cloud," Dec. 05, 2020, *arXiv*: arXiv:2012.03015. doi: 10.48550/arXiv.2012.03015.
[31] S. Muhammad and G.-W. Kim, "Visual Object Detection Based LiDAR Point Cloud Classification," in *2020 IEEE International Conference on Big Data and Smart Computing (BigComp)*, Feb. 2020, pp. 438–440. doi: 10.1109/BigComp48618.2020.00-32.
[32] R. Sahba, A. Sahba, M. Jamshidi, and P. Rad, "3D Object Detection Based on LiDAR Data," in *2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, Oct. 2019, pp. 0511–0514. doi: 10.1109/UEMCON47517.2019.8993088.
[33] L. Liu, J. He, K. Ren, Z. Xiao, and Y. Hou, "A LiDAR–Camera Fusion 3D Object Detection Algorithm," *Information*, vol. 13, no. 4, Art. no. 4, Apr. 2022, doi: 10.3390/info13040169.
[34] A. Y. Naich and J. R. Carrión, "LiDAR-Based Intensity-Aware Outdoor 3D Object Detection," *Sensors*, vol. 24, no. 9, Art. no. 9, Jan. 2024, doi: 10.3390/s24092942.
[35] G. Jocher, A. Chaurasia, and J. Qiu, *Ultralytics YOLO*. (Jan. 2023). Python. Accessed: Aug. 26, 2024. [Online]. Available: https://github.com/ultralytics/ultralytics
[36] "CARLA - v20 carla_v20," Roboflow. Accessed: Aug. 26, 2024. [Online]. Available: https://universe.roboflow.com/alec-hantson-student-howest-be/carla-izloa/dataset/20
[37] J. Zhu, Z. Lian and Z. Jiang, "A Spatio-Temporal Graph Transformer Network for Multi-Pedestrain Trajectory Prediction | IEEE Conference Publication | IEEE Xplore." Accessed: Aug. 29, 2024. [Online]. Available: https://ieeexplore-ieee-org.proxy.lib.ohio-state.edu/document/9927798
[38] G. Rains, A. Faircloth, C. Thai, R. Raper, "Evaluation of a Simple Pure Pursuit Path-Following Algorithm for an Autonomous, Articulated-Steer Vehicle." Accessed: Oct. 23, 2023. [Online]. Available: https://doi.org/10.13031/aea.30.10347
[39] Y. Chen and J. J. Zhu, "Pure Pursuit Guidance for Car-Like Ground Vehicle Trajectory Tracking," presented at the ASME 2017 Dynamic Systems and Control Conference, American Society of Mechanical Engineers Digital Collection, Nov. 2017. doi: 10.1115/DSCC2017-5376.
[40] G. M. Hoffmann, C. J. Tomlin, M. Montemerlo, and S. Thrun, "Autonomous Automobile Trajectory Tracking for Off-Road Driving: Controller Design, Experimental Validation and Racing," in *2007 American Control Conference*, Jul. 2007, pp. 2296–2301. doi: 10.1109/ACC.2007.4282788.
[41] B. Aksun-Güvenç, L. Güvenç, and S. Karaman, "Robust MIMO disturbance observer analysis and design with application to active car steering," *Int. J. Robust Nonlinear Control*, vol. 20, no. 8, pp. 873–891, 2010, doi: 10.1002/rnc.1476.
[42] L. Güvenç, B. Aksun-Güvenç, B. Demirel, and M. T. Emirler, *Control of Mechatronic Systems*. IET Digital Library, 2017. doi: 10.1049/PBCE104E.
[43] X. Chen, X. Liu, and M. Zhang, "Autonomous Vehicle Lane-Change Control Based on Model Predictive Control with Control Barrier Function," in *2024 IEEE 13th Data Driven Control and Learning Systems Conference (DDCLS)*, May 2024, pp. 1267–1272. doi:

10.1109/DDCLS61622.2024.10606562.
[44] A. Thirugnanam, J. Zeng, and K. Sreenath, “Safety-Critical Control and Planning for Obstacle Avoidance between Polytopes with Control Barrier Functions,” May 31, 2022, *arXiv*: arXiv:2109.12313. doi: 10.48550/arXiv.2109.12313.
[45] X. Cao *et al.*, “Nonlinear Model Predictive Control-Based Reverse Path-Planning and Path-Tracking Control of a Vehicle with Trailer System,” Sep. 01, 2025, *arXiv*: arXiv:2509.01820. doi: 10.48550/arXiv.2509.01820.
[46] H. Zhou, L. Guvenc, and Z. Liu, “Design and evaluation of path following controller based on MPC for autonomous vehicle,” in *Proceedings of the 36th Chinese Control Conference, CCC 2017*, IEEE Computer Society, Sep. 2017, pp. 9934–9939. doi: 10.23919/ChiCC.2017.8028942.
[47] A. D. Ames, J. W. Grizzle, and P. Tabuada, “Control barrier function based quadratic programs with application to adaptive cruise control,” in *53rd IEEE Conference on Decision and Control*, Dec. 2014, pp. 6271–6278. doi: 10.1109/CDC.2014.7040372.
[48] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, “Control Barrier Function Based Quadratic Programs for Safety Critical Systems,” *IEEE Trans. Autom. Control*, vol. 62, no. 8, pp. 3861–3876, Aug. 2017, doi: 10.1109/TAC.2016.2638961.
[49] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, “Control Barrier Functions: Theory and Applications,” in *2019 18th European Control Conference (ECC)*, Jun. 2019, pp. 3420–3431. doi: 10.23919/ECC.2019.8796030.
[50] W. Xiao and C. Belta, “High-Order Control Barrier Functions,” *IEEE Trans. Autom. Control*, vol. 67, no. 7, pp. 3655–3662, Jul. 2022, doi: 10.1109/TAC.2021.3105491.
[51] A. E. Chriat and C. Sun, “High-Order Control Lyapunov–Barrier Functions for Real-Time Optimal Control of Constrained Non-Affine Systems,” *Mathematics*, vol. 12, no. 24, Art. no. 24, Jan. 2024, doi: 10.3390/math12244015.
[52] K. Wong, M. Stölzle, W. Xiao, C. D. Santina, D. Rus, and G. Zardini, “Contact-Aware Safety in Soft Robots Using High-Order Control Barrier and Lyapunov Functions,” May 05, 2025, *arXiv*: arXiv:2505.03841. doi: 10.48550/arXiv.2505.03841.
[53] X. Cao, H. Chen, L. Guvenc, and B. Aksun-Guvenc, “Communication Disturbance Observer Based Delay-Tolerant Control for Autonomous Driving Systems,” *Sensors*, vol. 25, no. 20, p. 6381, Jan. 2025, doi: 10.3390/s25206381.
[54] H. Chen, X. Cao, L. Guvenc, and B. Aksun-Guvenc, “High Order Control Lyapunov Function - Control Barrier Function - Quadratic Programming Based Autonomous Driving Controller for Bicyclist Safety,” Dec. 14, 2025, *arXiv*: arXiv:2512.12776. doi: 10.48550/arXiv.2512.12776.
[55] L. Guvenc, B. A. Guvenc, and M. T. Emirler, “Connected and Autonomous Vehicles,” in *Internet of Things and Data Analytics Handbook*, John Wiley & Sons, Ltd, 2017, pp. 581–595. doi: 10.1002/9781119173601.ch35.
[56] P. Ghorai, A. Eskandarian, Y.-K. Kim, and G. Mehr, “State Estimation and Motion Prediction of Vehicles and Vulnerable Road Users for Cooperative Autonomous Driving: A Survey,” *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 16983–17002, Oct. 2022, doi: 10.1109/TITS.2022.3160932.
[57] P. Markiewicz, M. Długosz, and P. Skruch, “Review of tracking and object detection systems for advanced driver assistance and autonomous driving applications with focus on vulnerable road users sensing,” in *Trends in Advanced Intelligent Control, Optimization and Automation*, Springer, Cham, 2017, pp. 224–237. doi: 10.1007/978-3-319-60699-6_22.

[58] D. Yang, H. Zhang, E. Yurtsever, K. A. Redmill, and Ü. Özgüner, "Predicting Pedestrian Crossing Intention With Feature Fusion and Spatio-Temporal Attention," *IEEE Trans. Intell. Veh.*, vol. 7, no. 2, pp. 221–230, Jun. 2022, doi: 10.1109/TIV.2022.3162719.
[59] H. Zhang, D. Yang, E. Yurtsever, K. A. Redmill, and Ü. Özgüner, "Faraway-Frustum: Dealing with Lidar Sparsity for 3D Object Detection using Fusion," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, Sep. 2021, pp. 2646–2652. doi: 10.1109/ITSC48978.2021.9564990.
[60] Y. Guo, Y. Liu, B. Wang, P. Huang, H. Xu, and Z. Bai, "Trajectory planning framework for autonomous vehicles based on collision injury prediction for vulnerable road users," *Accid. Anal. Prev.*, vol. 203, p. 107610, Aug. 2024, doi: 10.1016/j.aap.2024.107610.
[61] X. Cao, H. Chen, B. Aksun-Guvenc, and L. Guvenc, "Modified Hybrid A* Collision-Free Path-Planning for Automated Reverse Parking," Dec. 12, 2025, *arXiv*: arXiv:2512.12021. doi: 10.48550/arXiv.2512.12021.
[62] X. Cao *et al.*, "Hybrid A*-Based Reverse Path-Planning of a Vehicle with Single Trailer," *Electronics*, vol. 15, no. 5, p. 1114, Jan. 2026, doi: 10.3390/electronics15051114.
[63] M. Bojarski *et al.*, "End to End Learning for Self-Driving Cars," Apr. 25, 2016, *arXiv*: arXiv:1604.07316. doi: 10.48550/arXiv.1604.07316.
[64] A. Kendall *et al.*, "Learning to Drive in a Day," Sep. 11, 2018, *arXiv*: arXiv:1807.00412. doi: 10.48550/arXiv.1807.00412.
[65] B. R. Kiran *et al.*, "Deep Reinforcement Learning for Autonomous Driving: A Survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 6, pp. 4909–4926, Jun. 2022, doi: 10.1109/TITS.2021.3054625.
[66] F. Ye, S. Zhang, P. Wang, and C.-Y. Chan, "A Survey of Deep Reinforcement Learning Algorithms for Motion Planning and Control of Autonomous Vehicles," in *2021 IEEE Intelligent Vehicles Symposium (IV)*, Jul. 2021, pp. 1073–1080. doi: 10.1109/IV48863.2021.9575880.
[67] Z. Zhu and H. Zhao, "A Survey of Deep RL and IL for Autonomous Driving Policy Learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 14043–14065, Sep. 2022, doi: 10.1109/TITS.2021.3134702.
[68] Z. Zhang, C. Peng, M. Zhu, E. Yurtsever, K. Redmill "An Uncertainty-Weighted Decision Transformer for Navigation in Dense, Complex Driving Scenarios." Accessed: Sep. 30, 2025. [Online]. Available: https://arxiv.org/abs/2509.13132
[69] Z. Cao *et al.*, "Reinforcement Learning based Control of Imitative Policies for Near-Accident Driving," Jun. 30, 2020, *arXiv*: arXiv:2007.00178. doi: 10.48550/arXiv.2007.00178.
[70] A. Aksjonov and V. Kyrki, "A Safety-Critical Decision-Making and Control Framework Combining Machine-Learning-Based and Rule-Based Algorithms," *SAE Int. J. Veh. Dyn. Stab. NVH*, vol. 7, no. 3, pp. 10-07-03–0018, Jun. 2023, doi: 10.4271/10-07-03-0018.
[71] K. Makantasis, M. Kontorinaki, I. Nikolos "Deep reinforcement-learning-based driving policy for autonomous road vehicles", IET Intelligent Transport Systems, Wiley Online Library. Accessed: Oct. 24, 2023. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/full/10.1049/iet-its.2019.0249
[72] F. Merola, F. Falchi, C. Gennaro, and M. Di Benedetto, "Reinforced Damage Minimization in Critical Events for Self-driving Vehicles", in *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, Online Streaming, --- Select a Country ---: SCITEPRESS - Science and Technology Publications, 2022, pp. 258–266. doi: 10.5220/0010908000003124.

[73] H. Chen, F. Zhang, and B. Aksun-Guvenc, “Collision Avoidance in Autonomous Vehicles Using the Control Lyapunov Function–Control Barrier Function–Quadratic Programming Approach with Deep Reinforcement Learning Decision-Making,” *Electronics*, vol. 14, no. 3, p. 557, Jan. 2025, doi: 10.3390/electronics14030557.

[74] H. Chen and B. Aksun-Guvenc, “Hierarchical Deep Reinforcement Learning-Based Path Planning with Underlying High-Order Control Lyapunov Function—Control Barrier Function—Quadratic Programming Collision Avoidance Path Tracking Control of Lane-Changing Maneuvers for Autonomous Vehicles,” *Electronics*, vol. 14, no. 14, p. 2776, Jan. 2025, doi: 10.3390/electronics14142776.

[75] Y. Ding, H. Zhong, Y. Qian, L. Wang, and Y. Xie, “Lane-Change Collision Avoidance Control for Automated Vehicles with Control Barrier Functions,” *Int. J. Automot. Technol.*, vol. 24, no. 3, pp. 739–748, Jun. 2023, doi: 10.1007/s12239-023-0061-2.

[76] I. Jang and H. J. Kim, “Safe Control for Navigation in Cluttered Space Using Multiple Lyapunov-Based Control Barrier Functions,” *IEEE Robot. Autom. Lett.*, vol. 9, no. 3, pp. 2056–2063, Mar. 2024, doi: 10.1109/LRA.2024.3349917.

[77] A. A. Nasab and M. H. Asemani, “Control of Mobile Robots Using Control Barrier Functions in Presence of Fault,” in *2023 9th International Conference on Control, Instrumentation and Automation (ICCIA)*, Dec. 2023, pp. 1–6. doi: 10.1109/ICCIA61416.2023.10506347.

[78] A. Alan, A. J. Taylor, C. R. He, A. D. Ames, and G. Orosz, “Control Barrier Functions and Input-to-State Safety With Application to Automated Vehicles,” *IEEE Trans. Control Syst. Technol.*, vol. 31, no. 6, pp. 2744–2759, Nov. 2023, doi: 10.1109/TCST.2023.3286090.

[79] S. Nageshrao, E. Tseng, and D. Filev, “Autonomous Highway Driving using Deep Reinforcement Learning,” Mar. 29, 2019, *arXiv*: arXiv:1904.00035. doi: 10.48550/arXiv.1904.00035.

[80] Z. Zhang, C. Peng, E. Yurtsever, and K. A. Redmill, “Bootstrapping Reinforcement Learning with Sub-optimal Policies for Autonomous Driving,” Sep. 04, 2025, *arXiv*: arXiv:2509.04712. doi: 10.48550/arXiv.2509.04712.

[81] T. P. Lillicrap *et al.*, “Continuous control with deep reinforcement learning,” in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2016. Accessed: Feb. 19, 2026. [Online]. Available: http://arxiv.org/abs/1509.02971

[82] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal Policy Optimization Algorithms,” Aug. 28, 2017, *arXiv*: arXiv:1707.06347. doi: 10.48550/arXiv.1707.06347.

[83] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, “Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor,” in *Proceedings of the 35th International Conference on Machine Learning*, PMLR, Jul. 2018, pp. 1861–1870. Accessed: Feb. 19, 2026. [Online]. Available: https://proceedings.mlr.press/v80/haarnoja18b.html

[84] P. Christodoulou, “Soft Actor-Critic for Discrete Action Settings,” Oct. 18, 2019, *arXiv*: arXiv:1910.07207. doi: 10.48550/arXiv.1910.07207.

[85] E. Leurent, *An Environment for Autonomous Driving Decision-Making*. (May 2018). Python. Accessed: Jun. 14, 2025. [Online]. Available: https://github.com/eleurent/highway-env

[86] M. Jaritz, R. de Charette, M. Toromanoff, E. Perot, and F. Nashashibi, “End-to-End Race Driving with Deep Reinforcement Learning,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 2070–2075. doi: 10.1109/ICRA.2018.8460934.

[87] S. H. Ashwin and R. Naveen Raj, “Deep reinforcement learning for autonomous vehicles: lane keep and overtaking scenarios with collision avoidance,” *Int. J. Inf. Technol.*, vol. 15, no. 7, pp. 3541–3553, Oct. 2023, doi: 10.1007/s41870-023-01412-6.
[88] A. J. M. Muzahid, S. F. Kamarulzaman, Md. A. Rahman, and A. H. Alenezi, “Deep Reinforcement Learning-Based Driving Strategy for Avoidance of Chain Collisions and Its Safety Efficiency Analysis in Autonomous Vehicles,” *IEEE Access*, vol. 10, pp. 43303–43319, 2022, doi: 10.1109/ACCESS.2022.3167812.
[89] Y. Chen and C. G. Cassandras, “Optimal Sequencing and Motion Control in a Roundabout With Safety and Comfort Guarantees,” *IEEE Trans. Intell. Transp. Syst.*, vol. 26, no. 11, pp. 19148–19162, Nov. 2025, doi: 10.1109/TITS.2025.3593441.
[90] X. Li, L. Guvenc, and B. Aksun-Guvenc, “Autonomous Vehicle Decision-Making with Policy Prediction for Handling a Round Intersection,” *Electronics*, vol. 12, no. 22, p. 4670, Jan. 2023, doi: 10.3390/electronics12224670.
[91] W.-D. Shen, Y.-C. Lu, H.-Y. Wei, and H. Zhang, “Enhancing Intersection Safety through ISAC-Enabled Sidelink Communication in Next Generation Vehicular Networks,” in *2025 IEEE VTS Asia Pacific Wireless Communications Symposium (APWCS)*, Aug. 2025, pp. 1–5. doi: 10.1109/APWCS67981.2025.11151905.
[92] A. Karalakou, M. Rostami-Shahrbabaki, F. Rempe, and K. Bogenberger, “A Deep Reinforcement Learning Approach for Controlling Autonomous Vehicles in Lane-Free Roundabouts,” in *2025 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2025, pp. 2348–2354. doi: 10.1109/IV64158.2025.11097805.
[93] Z. Gu, W. Zeng, and Z. Zheng, “Spatio-Temporal Cooperative Decision-Making for Mixed Traffic at Unsignalized Intersections Based on Deep Reinforcement Learning,” in *2025 9th CAA International Conference on Vehicular Control and Intelligence (CVCI)*, Oct. 2025, pp. 1–6. doi: 10.1109/CVCI66304.2025.11348173.
[94] M. Treiber, A. Hennecke, and D. Helbing, “Congested traffic states in empirical observations and microscopic simulations,” *Phys. Rev. E*, vol. 62, no. 2, pp. 1805–1824, Aug. 2000, doi: 10.1103/PhysRevE.62.1805.
[95] E. Leurent, *An Environment for Autonomous Driving Decision-Making*. (May 2018). Python. Accessed: Feb. 19, 2026. [Online]. Available: https://github.com/eleurent/highway-env
[96] X. Cao, H. Chen, S. Y. Gelbal, B. Aksun Guvenc, and L. Guvenc, “Vehicle-in-Virtual-Environment Method for ADAS and Connected and Automated Driving Function Development, Demonstration and Evaluation”, SAE International. Accessed: Feb. 19, 2026. [Online]. Available: https://www.sae.org/papers/vehicle-virtual-environment-method-adas-connected-automated-driving-function-development-demonstration-evaluation-2024-01-1967
[97] H. Chen, X. Cao, L. Guvenc, and B. Aksun Guvenc, “Developing a Vehicle-in-Virtual-Environment (VVE) Based Autonomous Driving Function Development and Evaluation Pipeline for Vulnerable Road User Safety,” SAE International, Warrendale, PA, SAE Technical Paper 2025-01–8061, Apr. 2025. doi: 10.4271/2025-01-8061.
[98] L. Pariota *et al.*, “Integrating tools for an effective testing of connected and automated vehicles technologies,” *IET Intell. Transp. Syst.*, vol. 14, no. 9, pp. 1025–1033, 2020, doi: 10.1049/iet-its.2019.0678.
[99] S. Chen, N.-N. Zheng, Y. Chen, and S. Zhang, “A Novel Integrated Simulation and Testing Platform for Self-Driving Cars With Hardware in the Loop,” *IEEE Trans. Intell. Veh.*, vol. PP, pp. 1–1, May 2019, doi: 10.1109/TIV.2019.2919470.

[100] Ş. Y. Gelbal, S. Tamilarasan, M. R. Cantaş, L. Güvenç, and B. Aksun-Güvenç, "A connected and autonomous vehicle hardware-in-the-loop simulator for developing automated driving algorithms," in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct. 2017, pp. 3397–3402. doi: 10.1109/SMC.2017.8123155.

[101] S. Oncu *et al.*, "Robust Yaw Stability Controller Design for a Light Commercial Vehicle Using a Hardware in the Loop Steering Test Rig," in *2007 IEEE Intelligent Vehicles Symposium*, Jun. 2007, pp. 852–859. doi: 10.1109/IVS.2007.4290223.

[102] S. Oncu, L. Guvenc, S. Ersolmaz, S. Ozturk, K. Serdar. N. Kilic, M. Sinal, "Steer-by-Wire Control of a Light Commercial Vehicle Using a Hardware-in-the-Loop Test Setup." Accessed: Jul. 28, 2026. [Online]. Available: https://polen.itu.edu.tr/entities/publication/e70417c6-a1dd-428b-b632-3797fc7987af

[103] M. R. Cantas, O. Kavas, S. Tamilarasan, S. Y. Gelbal, and L. Guvenc, "Use of Hardware in the Loop (HIL) Simulation for Developing Connected Autonomous Vehicle (CAV) Applications," SAE International, Apr. 2019. doi: 10.4271/2019-01-1063.

[104] X. Cao, H. Chen, S. Y. Gelbal, B. Aksun-Guvenc, and L. Guvenc, "Vehicle-in-Virtual-Environment (VVE) Method for Autonomous Driving System Development, Evaluation and Demonstration," *Sensors*, vol. 23, no. 11, p. 5088, Jan. 2023, doi: 10.3390/s23115088.

[105] S. Y. Gelbal, B. Aksun Guvenc, and L. Guvenc, "Vehicle in Virtual Environment (VVE) Method of Autonomous Driving Function Evaluation and Development ", SAE International. Accessed: Jul. 28, 2026. [Online]. Available: https://www.sae.org/papers/vehicle-virtual-environment-vve-method-autonomous-driving-function-evaluation-development-2023-01-0820

[106] L. Guvenc, and B. Aksun-Guvenc "Vehicle-in-virtual-environment (VVE) methods and systems for autonomous driving system". U.S. Patent 12,654,723 B2, filed March 8, 2022, and issued June 16, 2026.

[107] R. Rao, "A Simulation Framework for Real-Time, Pedestrian and Autonomous Vehicle Interaction Using Shared Virtual Reality," The Ohio State University, 2026. Accessed: Jul. 28, 2026. [Online]. Available: https://kb.osu.edu/handle/1811/107124

[108] X. Cao and L. Guvenc, "Path Planning and Robust Path Tracking Control of an Automated Parallel Parking Maneuver," SAE International, Warrendale, PA, SAE Technical Paper 2024-01–2558, Apr. 2024. doi: 10.4271/2024-01-2558.

[109] L. Güvenç, B. Aksun-Güvenç, B. Demirel, and M. T. Emirler, *Control of Mechatronic Systems*. IET Digital Library, 2017. doi: 10.1049/PBCE104E.

[110] F. Ma *et al.*, "Stability Design for the Homogeneous Platoon with Communication Time Delay," *Automot. Innov.*, vol. 3, no. 2, pp. 101–110, Jun. 2020, doi: 10.1007/s42154-020-00102-4.

[111] S. Necipoglu, S. A. Cebeci, Y. E. Has, L. Guvenc, and C. Basdogan, "Robust Repetitive Controller for Fast AFM Imaging," *IEEE Trans. Nanotechnol.*, vol. 10, no. 5, pp. 1074–1082, Sep. 2011, doi: 10.1109/TNANO.2011.2106797.

[112] L. Gu¨venc¸, "Stability and Performance Robustness Analysis of Repetitive Control Systems Using Structured Singular Values," *J. Dyn. Syst. Meas. Control*, vol. 118, no. 3, pp. 593–597, Sep. 1996, doi: 10.1115/1.2801185.

[113] H. Wang, S. Y. Gelbal, and L. Guvenc, "Multi-Objective Digital PID Controller Design in Parameter Space and its Application to Automated Path Following," *IEEE Access*, vol. 9, pp. 46874–46885, 2021, doi: 10.1109/ACCESS.2021.3066925.

[114] M. T. Emirler *et al.*, "Lateral stability control of fully electric vehicles," *Int. J. Automot.*

Technol., vol. 16, no. 2, pp. 317–328, Apr. 2015, doi: 10.1007/s12239-015-0034-1.

[115] H. Chen, B. Aksun-Guvenc, 2026, “Safe Hierarchical Autonomous Driving Framework: Integrating SAC-D Planning with HOCLF-HOCBF-QP Constrained Low-Level Control,” *IEEE Open Journal of Vehicular Technology*, vol. 7, pp. 1474-1489, https://doi.org/10.1109/OJVT.2026.3691614