OpenConvoy: Universal Platform for Real-World Testing of Cooperative Driving Systems

Owen Burns, Hossein Maghsoumi, Israel Charles, Yaser Fallah
College of Engineering and Computer Science
University of Central Florida
Orlando, United States
https://orcid.org/0009-0007-1826-373X

***Abstract*—Cooperative driving, enabled by communication between automated vehicle systems, promises significant benefits to fuel efficiency, road capacity, and safety over single-vehicle driver assistance systems such as adaptive cruise control (ACC). However, the responsible development and implementation of these algorithms pose substantial challenges due to the need for extensive real-world testing. We address this issue by introducing OpenConvoy, an open and extensible framework designed for the implementation and assessment of cooperative driving policies on physical connected and autonomous vehicles (CAVs). We demonstrate the capabilities of OpenConvoy through a series of experiments on a convoy of multi-scale vehicles controlled by Platooning, showcasing the stability of our system across different vehicle configurations and its ability to effectively measure convoy cohesion across driving scenarios, including varying degrees of communication loss.**

***Index Terms*—cooperative driving systems, connected autonomous vehicles, real-world testing platform, platooning.**

I. Introduction

Rapid advancements in autonomous vehicle technology and vehicle-to-vehicle communication (V2V) have opened up new avenues for enhancing transportation efficiency, safety, and sustainability. One promising application within this domain is cooperative driving, where cars communicate intentions to ensure cohesive movement. This coordination enables shorter safe inter-vehicle distances, reducing emissions by reducing drag on the cars in the convoy and increasing highway throughput [**?**].

Despite these potential benefits, developing safe cooperative driving policies remains a complex challenge. Thorough testing must be undertaken to assess the resilience of such algorithms to issues such as rough terrain, communication loss, and noisy sensor readings. The complexity of real-world driving environments further motivates the development of joint perception and decision making frameworks to improve the awareness and coordination of connected and autonomous vehicles (CAVs) to beyond the human level. Current frameworks limit the testing of solutions to these problems to simulated environments or very specific hardware implementations (see II), limiting the general applicability of results. Thus, there is an urgent need for a unified platform for assessing the performance of arbitrary cooperative driving implementations on various autonomous vehicles in a consistent manner. In this paper, we address this problem and introduce OpenConvoy to fill that gap. The OpenConvoy platform is publicly available in [1] for use by researchers and developers of cooperative driving systems and other humanitarian systems.

A. Background

Cooperative driving systems use Vehicle-to-Vehicle (V2V) and Vehicle-to-Everything (V2X) communication to enhance situational awareness among cars in close proximity by sharing information such as speed, acceleration, and braking in real time. These systems have been demonstrated to provide a number of benefits, particularly improved highway throughput and fuel efficiency [2]. This is achieved by enabling vehicles to safely travel in closer proximity than they normally would without this heightened awareness. Implementations of this concept can typically be deconstructed along the lines of spacing policy, communication policy, and controller.

In cooperative driving, the spacing policy defines the physical structure of the platoon, typically categorized as either platooning or Cooperative Adaptive Cruise Control (CACC), with the former using fixed following distances and the latter using fixed following times. The communication policy defines the structure of V2V communications, including the policy for sending messages (either time- or event-triggered [3]) and the network topology that determines who receives them. Network topologies are commonly either bidirectional or predecessor following with varying degrees of lookahead [4]. These policies are visualized in comparison to individual driving (Adaptive Cruise Control (ACC)) in Fig. 1. Based on the information received from the other vehicles and the objective of the spacing policy, the controller determines the motion to be executed at the vehicle level; these controllers can vary widely from simple linear controllers to advanced Model Predictive Control (MPC) implementations [5].

B. Motivation and Problem Domain

High reliability in diverse conditions is a critical challenge for cooperative driving systems. Testing these systems on vehicles in various conditions is usually costly and complex. As

This work was partially supported by the National Science Foundation under Grant CNS-1932037. The code and supplementary material are available at: https://github.com/owenonline/cavrel-platooning.

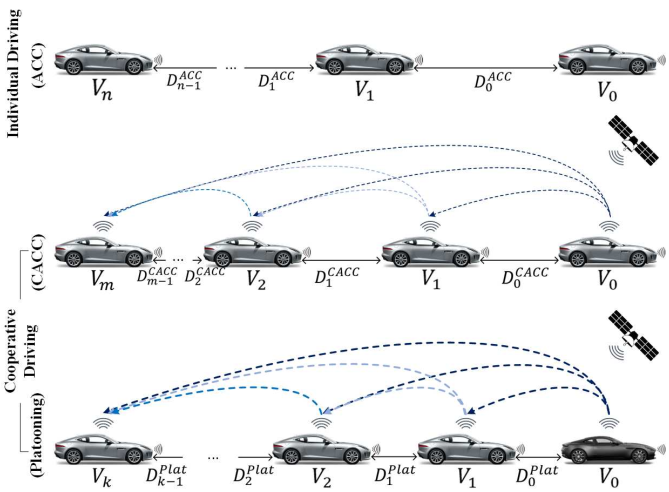

Fig. 1. Comparison of ACC, CACC, and Platooning, each using an all-predecessor IFT. The distances from each vehicle (V) to the vehicle behind it in each system are represented as D^{ACC}, D^{CACC}, and D^{Plat}. The relation $D^{\text{ACC}} > D^{\text{CACC}} > D^{\text{Plat}}$ implies that $k > m > n$.

TABLE I
COOPERATIVE DRIVING IMPLEMENTATIONS BY SPACING POLICY, COMMUNICATION POLICY, AND CONTROLLER

Cooperative Driving Implementations				
Reference	***Spacing Policy***	***Information Flow Topology (IFT)***	***Communication Trigger***	***Model-based Controller***
[6]	M-CACC	All BD	Time	Yes
[3]	CACC	All PD	Event	Yes
[4]	CACC	N-lookahead PD	Time	Yes
[7]	M-CACC	1-lookahead PD	Time	No
[5]	Platooning	All PD	Time	Yes
[8]	Platooning	All PD	Event	Yes
[9]	Platooning	Leader Following	Event	Yes
[10]	CACC	1-lookahead PD	Event	Yes

BD: Bidirectional, PD: Predecessor, M-CACC: modified CACC

shown in Table. I, various cooperative driving implementations differ in spacing policies, communication strategies, and controllers. Despite these differences, the fundamental logic for message broadcasting and hardware communication remains consistent. This highlights an opportunity to create a unified platform to reduce the challenges and costs of testing and implementing cooperative driving algorithms across diverse vehicle types and configurations.

C. Contributions

We present OpenConvoy, an open and modular platform that streamlines the implementation and testing of cooperative driving algorithms on diverse autonomous vehicles. This platform directly supports both Robot Operating System (ROS1 and ROS2) and uses the Micro Air Vehicle Link (MAVLink) protocol, enabling its application to a wide variety of vehicle types and control stacks with minimal modification. We demonstrate the stability of our system by employing it in testing platooning with time-triggered communication and an all-predecessor-following Information Flow Topology (IFT) on a convoy of multi-scale vehicles.

II. Related Work

Interest in autonomous driving has spurred the development of a wide variety of platforms to streamline the implementation and testing of autonomous vehicles. A summary of the types of systems in comparison to OpenConvoy is given in Table II.

The prohibitive cost associated with field testing has made simulation an integral part of autonomous vehicle development. At an abstract level, platforms such as SUMO [11] and Veins [12] simulate multi-agent traffic dynamics, but do not provide vehicle-level data. At a more concrete level, autonomous vehicle (AV) sims such as Carla [13], LGSVL [14], and CarSim [15] provide vehicle-level data for single-agent systems while cooperative autonomous vehicle (CAV) simulators such as OpenCDA [16], AutoCastSim [17], V2Xverse [18], and [19] do the same for multi-agent systems. Besides supporting multi-agent simulation, CAV simulators include functionality to simulate V2V communication and support custom cooperative driving algorithms.

Despite advances in simulation techniques, real world testing remains essential to capture the full breadth of possibilities which may occur in actual driving scenarios [20]. AV testing frameworks have been developed to assess the track driving capabilities of autonomous vehicles by generating information-rich scenarios to overlay on top of live sensor data [21] [22], however they do not include high-level control functionality for autonomous vehicles themselves and do not assess multi-agent scenarios involving V2V communication. Approaches supporting high-level vehicle control include commercial self-driving systems such as those from Nvidia [23] and Hexagon [24] and self-driving frameworks like Autoware [25], though these focus on single-agent scenarios. For high-level control of multiple agents options are more limited; despite the existence of CARMA [26] as a general purpose cooperative driving framework, it is not open hardware, working with a specific set of sensors exclusive to full-size vehicles which makes it inviable for most research use. The 1Tenth version of the project was cancelled before release [27], and as a result most CAV work is conducted either in-sim or on small, purpose built research platforms like MicroIV [28] which are not open hardware and don't support custom cooperative driving (CD) algorithm implementations. This leaves the gap which OpenConvoy addresses, the need for an extensible cooperative driving platform compatible with open hardware support.

III. System Architecture

The problem OpenConvoy addresses is facilitating the implementation of arbitrary cooperative driving policies on arbitrary vehicle hardware without requiring vehicle-specific code.

TABLE II
COMPARISON BETWEEN OPENCONVOY AND EXISTING AUTONOMOUS DRIVING PLATFORMS

Refs	*Type*	*Vehicle-Level*	*Multi-Agent*	*Real World*	*High-Level Vehicle Control*	*V2V Comm*	*Open Hardware*	*Custom CD Algo Support*	*Availability*
[11], [12]	Traffic SIMs	No	Yes	No	N/A	N/A	N/A	N/A	OS
[13], [14], [15]	AV SIMs	Yes	No	No	Yes	No	N/A	N/A	OS
[16], [17], [19] [18], [29], [30]	CAV SIMs	Yes	Yes	No	Yes	Yes	N/A	Yes	OS
[21], [22]	AV Testing	Yes	No	Yes	No	No	Yes	No	OS
[25]	Autoware	Yes	No	Yes	Yes	No	Yes	No	OS
[23], [24], [31]	Pre-built AVs	Yes	No	Yes	Yes	No	No	No	COM
[28]	Research Systems	Yes	Yes	Yes	Yes	Yes	No	No	OS
[26]	CARMA	Yes	Yes	Yes	Yes	Yes	No	Yes	OS
[27]	CARMA-1tenth	Yes	Yes	Yes	Yes	Yes	Yes	Yes	NO
[1]	OpenConvoy	Yes	Yes	Yes	Yes	Yes	Yes	Yes	OS

OS: Open Source, COM: Commercial, NO: Not released, AV: Autonomous Vehicle, CAV: Connected and Autonomous Vehicle, CD: Cooperative driving

It addresses this by treating the spacing policy, communication policy, and controllers as standalone components and handling the rest of the vehicle control and communication internally in a hardware agnostic manner. In the rest of the section we formulate this problem concretely and outline the core components of the OpenConvoy architecture.

A. Problem formulation

Let $i \in 0, 1, ...n$ denote the vehicles of a convoy where vehicle 0 is the leader. For each vehicle i, let $\mathcal{R}^i$ be a data structure containing the most recent states of the vehicles in the convoy it receives messages from according to the IFT, including itself and its own current state. Concretely,

$$\mathcal{R}_j{}^i = \{s_j{}^{n-k}, s_j{}^{n-k+1}, ..., s_j{}^n\} \tag{1}$$

When vehicle i receives a message from vehicle j, the IFT of the system determines whether the message from j should be saved; this Rx gate is formulated as:

$$\rho : (i, j) \to \{0, 1\} \tag{2}$$

Similarly, every $\mathcal{B}$ seconds vehicle i determines whether it should broadcast its current state based on $\mathcal{R}^i$; this Tx gate is formulated as:

$$\tau : (\mathcal{R}^i) \to \{0, 1\} \tag{3}$$

At each time step t, vehicle i calculates its target speed v_t and heading θ_t based on the spacing policy σ as follows:

$$v_t, \theta_t = \arg\min_{v,\theta}[\sigma(v, \theta, \mathcal{R}^i)] \tag{4}$$

Finally, the controllers χ_v and χ_θ each transform the target speed and heading into the applied speed and heading, which is then actuated on the vehicle.

$$v_a = \chi_v(v_t, \mathcal{R}^i), v_t = \chi_\theta(\theta_t, \mathcal{R}^i) \tag{5}$$

This definition allows arbitrary definition of τ, ρ, σ, χ_v, and χ_θ to cover the breadth of cooperative driving implementations

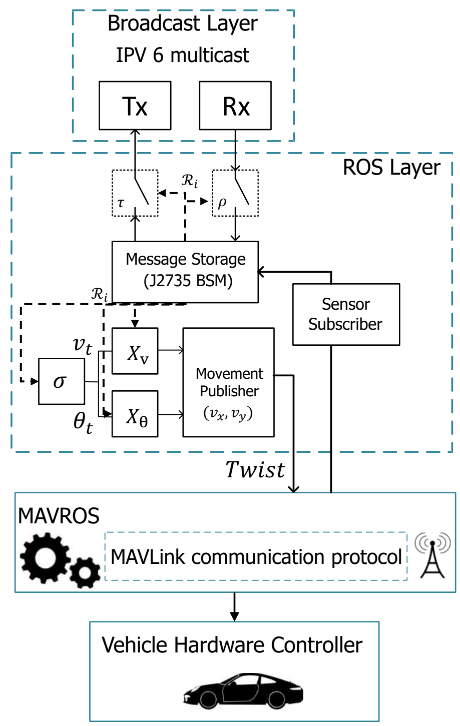

Fig. 2. Block diagram of OpenConvoy platform

without any ROS- or hardware-specific logic. The remaining components are explained below, with Fig. 2 being a block diagram of the overall system.

B. Mission Controller

The core of the OpenConvoy architecture is the Mission Controller, a ROS node implemented for ROS 1 and 2 that controls the vehicle motors, routes information between the cooperative driving policy components, handles V2V communication, handles conversions between coordinate frames of different vehicles in the convoy, and logs position information. An overview of how the Mission Controller operates when receiving and transmitting messages and when executing vehicle motion is given in Fig. 3. The Mission Controller also ensures that if any vehicle encounters an issue, all vehicles in the convoy stop to prevent crashes at short following distances, and logs time-synced position data for all vehicles to allow easy analysis post-testing.

C. Hardware Interface

To support the widest breadth of vehicle configurations, OpenConvoy uses MAVLink ROS (MavRos) [32] to communicate with a flight controller running either PX4 Autopilot [33] or ArduPilot Autopilot [34], which in turn controls the vehicle's motors. Because either firmware can run on a multitude of different flight controllers, and most flight controllers, Electronic Speed Controllers (ESCs), and Global Positioning System (GPS) receivers are interoperable, OpenConvoy maintains near-universal compatibility with lab-scale autonomous vehicles.

D. V2V Communication

To ensure that our platform can seamlessly interact with other cooperative driving systems adhering to the same standard, and that the vehicle states stored in $\mathcal{R}^i$ are in a widely accepted format, OpenConvoy's messages follow the SAE J2735 Basic Safety Messages (BSM) [35] standard. BSMs encapsulate essential vehicle state information such as position, velocity, and acceleration, enabling real-time data exchange crucial for cooperative maneuvers. Internet Protocol version 6 (IPv6) multicast communication is used to transmit BSMs between vehicles, ensuring simultaneous delivery and avoiding network configuration associated with IPv4 broadcasts.

E. Usage Details

Cooperative driving policies can be implemented by creating a subclass of the Mission Controller and overriding the methods corresponding to the communication policy, spacing policy, and controller to implement their desired policy. All vehicles running Mission Controllers will move according to their cooperative driving policy automatically. Specific instructions can be found in the project GitHub [1].

IV. Experiments

We conducted multiple experiments on a string of three small-scale vehicles, using OpenConvoy to implement platooning with an all-predecessor following IFT and time-triggered communication with a non-model based controller. We tested the performance of the convoy with varying degrees of communication loss in order to simulate the environment of a crowded network or imperfect transmission equipment.

A. Hardware Setup

For our experiments, we assembled three small-scale vehicles: two at a 1/10$^{\text{th}}$ scale and one at a 1/6$^{\text{th}}$ scale, each with different motors, servos, and speed controllers. Fig. 4 illustrates the hardware components of two of these vehicles,

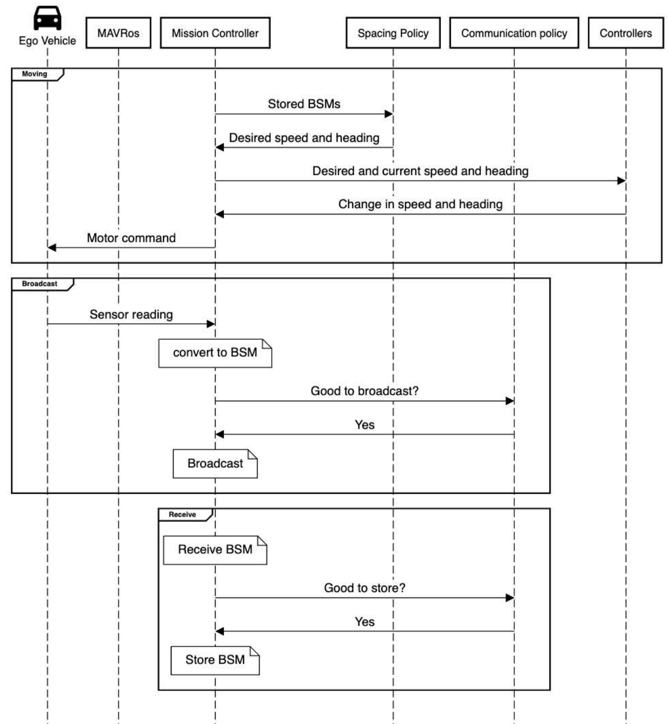

Fig. 3. Sequence diagram of OpenConvoy platform.

highlighting their design and setup. For each of the vehicles, we used a PixHawk 6C flight controller and M9N GPS receiver, with the PixHawk directly connecting to the ESC and steering servo in each vehicle. We used PX4 in all cases, though ArduPilot could be used interchangably due to its support for Mavros. One of the 1/10$^{\text{th}}$ scale vehicles used ROS 1 and Python 2 on an Nvidia Jetson TX2 while both other vehicles used an Nvidia Jetson AGX Xavier with ROS 2 and Python 3.

B. Implementation Details

Using all-predecessor following, we let τ always return 1 and ρ return 1 if the broadcaster is a predecessor in the platoon. We define σ to be the sum of costs of the distance (in the plane arrived at by performing a local East-North-Up (ENU) transformation on all relevent GPS coordinates) between the point representing the correct following distance and the projected point one time step forwards for the ego vehicle for each predecessor of the ego vehicle in the platoon. For χ_v and χ_θ we use a Proportional-Derivative (PD) and Stanley controller respectively.

The experiments were conducted with these three vehicles under a controlled setting. The leader vehicle's trajectory was designed to mimic a realistic highway driving scenario, with a time-varying speed starting at 1 m/s and jumping to 2 m/s before returning to 1 m/s. This setup ensured that the following vehicles had to adapt to sudden speed changes, providing valuable data on the efficacy of our control algorithms in maintaining desired inter-vehicle distances and overall platooning stability. Higher speeds were ruled out due to the

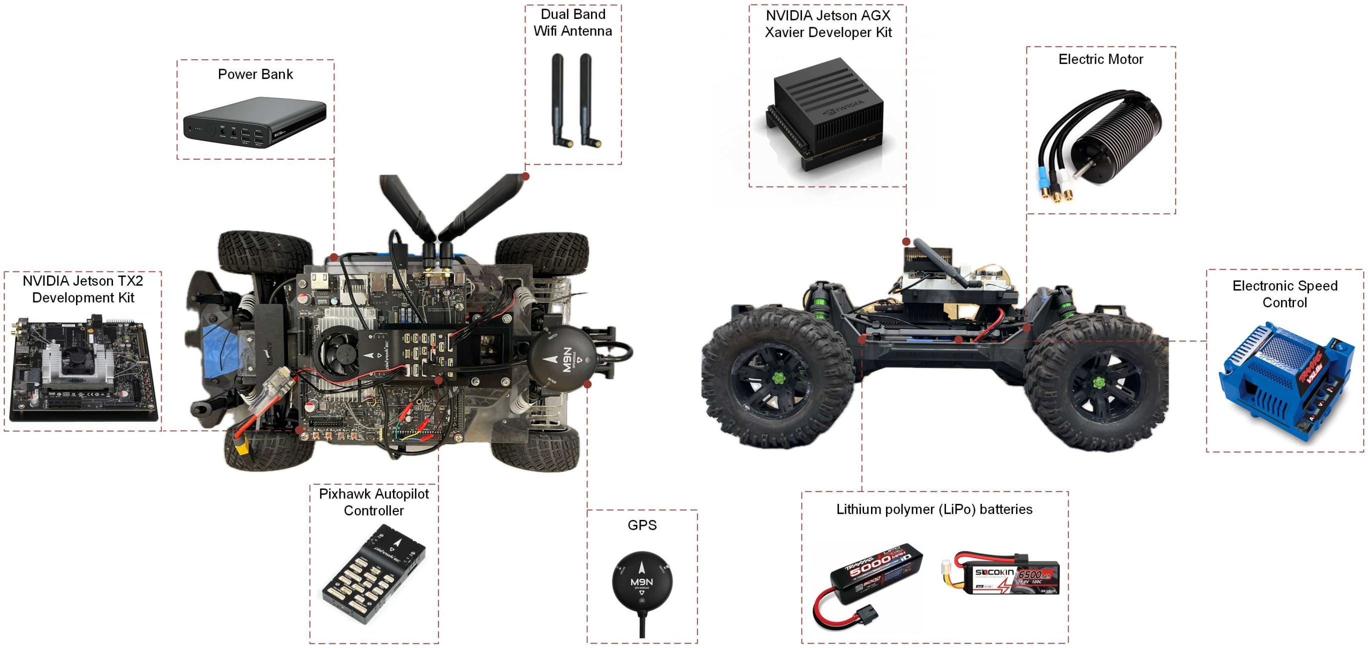

Fig. 4. Hardware Components for 1/10th (left) and 1/6th (right) Vehicles

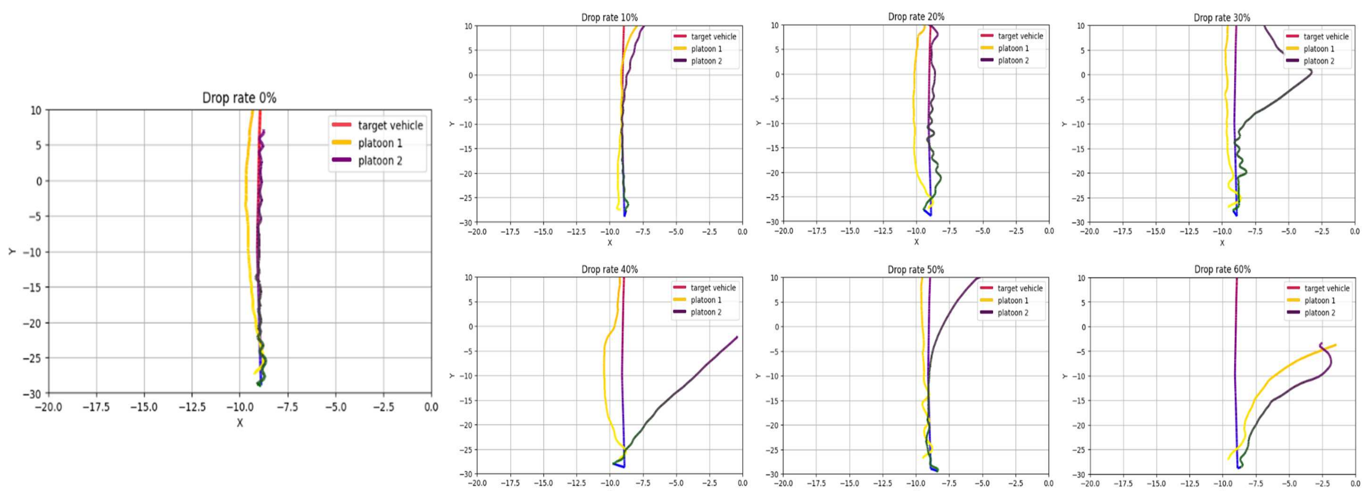

Fig. 5. Vehicle Trajectories

low top speed of our lowest powered following vehicle, which struggled to cope with the additional weight of the power bank and companion computer.

C. Evaluation Metrics

In evaluating the performance of our platooning experiments, we utilized two key metrics: Platooning Error [36] and Speed Difference [37]. These metrics provided a comprehensive assessment of our control algorithms' effectiveness in maintaining desired distances and ensuring traffic flow stability.

1) Platooning Error: As introduced in [36], Platooning Error is defined as the absolute value of the difference between the actual distance gap and the desired distance gap (15 meters) in meters. To capture a statistical sense of worst-case behavior and account for the characteristics of wireless networks, we chose the 95th percentile of the error's absolute value. This metric allows us to evaluate how well the vehicles maintain the desired inter-vehicle spacing, even under varying conditions.

2) Speed Difference: The variation between the highest and lowest speeds among all vehicles in the string at each time step serves as an effective indicator of traffic flow. As outlined in [37], this metric, referred to as Speed Difference, is utilized to assess the performance of the platooning system. By evaluating the speed difference, we can determine how consistently the vehicles in the string are moving in relation to each other, which is crucial for maintaining smooth and

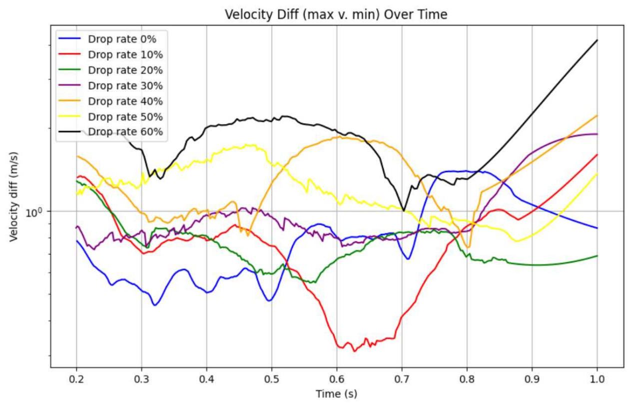

Fig. 6. Max inter-vehicle velocity difference over time

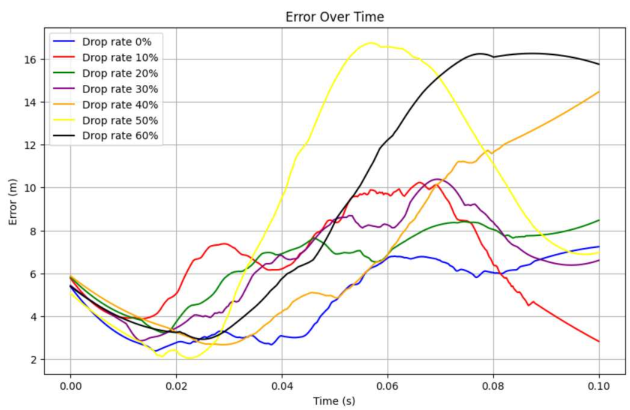

Fig. 7. Max inter-vehicle platooning error over time

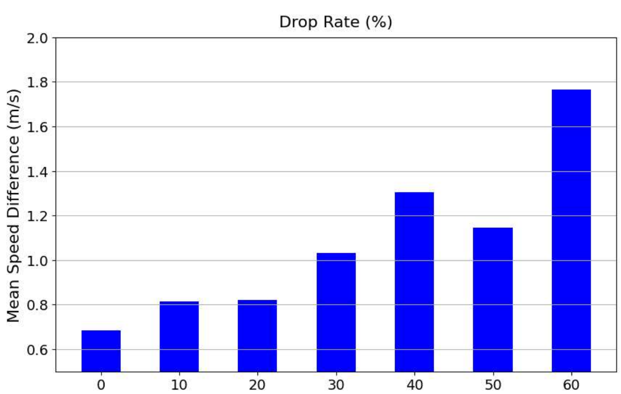

Fig. 8. Mean velocity error over drop rates

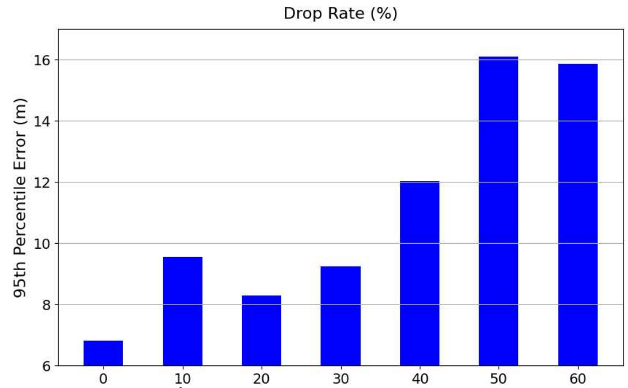

Fig. 9. 95^{th} percentile platooning error over drop rates

efficient traffic flow.

These metrics together provide a detailed evaluation of the platooning system's performance, highlighting both the accuracy in maintaining desired distances and the stability of vehicle speeds within the platoon.

D. Results

Fig. 5 shows the vehicle trajectory for the lead vehicle and follower vehicles in scenarios with variable Packet Error Rates (PER) ranging from 0% to 60%. As shown in this figure, with a drop rate between 0% and 20%, follower vehicles 1 and 2 follow the lead vehicle with good accuracy. In the drop rate range of 30% to 50%, although the first follower vehicle responds well and follows the lead vehicle with minor errors, the second follower vehicle experiences an increased error rate at times. At a drop rate of 60%, the first follower vehicle loses the path of the lead vehicle, which leads to a path error for the second follower vehicle as well.

Fig. 6 shows the difference between the maximum speed value and the minimum speed value at different time intervals for variable PER scenarios. It can be seen that this difference increases with the rise in drop rate in most time intervals.

Fig. 7 illustrates the platooning error over time intervals for various PER scenarios. Although, in most time intervals, the error for drop rate values between 0% and 30% is generally compact and close to each other, the error rate in some time intervals for drop rate values of 40%, 50%, and 60% is significantly higher than the error values for drop rates between 0% and 30%.

For a better comparison of the maximum and minimum speed differences in platooning, the average speed difference over variable PER is shown in Fig. 8. This figure clearly shows an increasing trend in the difference as the drop rate increases.

Additionally, the 95^{th} percentile error over different values of PER is shown in Fig. 9, highlighting the impact of increased PER on platooning performance. These results are directly consistent with the experiments carried out in simulator in [5], demonstrating the ability of OpenConvoy to provide an effective platform to bridge the sim2real gap for cooperative driving.

V. Conclusion

We present OpenConvoy to address the lack of a platform for easily implementing cooperative driving algorithms on platoons of real vehicles, reducing the work required to perform the kind of rigorous real-world testing which is required for the safe development and deployment of cooperative driving systems. We demonstrate the compatibility of OpenConvoy across multi-scale autonomous vehicles and its ability to rigorously assess the performance of cooperative driving implementations across varying communication landscapes, successfully replicating the results of previous simulator-based works. Future lines of work will include expanding testing to larger numbers of vehicles and implementing the methods of more papers, in particular creating an easily extensible basis for implementing MPC.

REFERENCES

[1] Owen Burns & Hossein Maghsoumi, "Openconvoy platform," https://github.com/owenonline/cavrel-platooning, 2024, Accessed: 2024-05-29.

[2] Xiao-Yun Lu Hao Liu, Steven E. Shladover and Xingan (David) Kan, "Freeway vehicle fuel efficiency improvement via cooperative adaptive cruise control," *Journal of Intelligent Transportation Systems*, vol. 25, no. 6, pp. 574–586, 2021.

[3] Mahdi Razzaghpour, Rodolfo Valiente, Mahdi Zaman, and Yaser P. Fallah, "Predictive model-based and control-aware communication strategies for cooperative adaptive cruise control," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 4, pp. 232–243, 2023.

[4] Mahdi Razzaghpour, Sahand Mosharafian, Arash Raftari, Javad Mohammadpour Velni, and Yaser P. Fallah, "Impact of information flow topology on safety of tightly-coupled connected and automated vehicle platoons utilizing stochastic control," in *2022 European Control Conference (ECC)*, 2022, pp. 27–33.

[5] Mahdi Razzaghpour, Shahriar Shahram, Rodolfo Valiente, and Yaser P. Fallah, "Impact of communication loss on mpc based cooperative adaptive cruise control and platooning," in *2021 IEEE 94th Vehicular Technology Conference (VTC2021-Fall)*, 2021, pp. 01–07.

[6] Valerio Turri, Bart Besselink, and Karl H. Johansson, "Cooperative look-ahead control for fuel-efficient and safe heavy-duty vehicle platooning," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 1, pp. 12–28, 2017.

[7] Paul Wijnbergen, Mark Jeeninga, and Bart Besselink, "Nonlinear spacing policies for vehicle platoons: A geometric approach to decentralized control," *Systems & Control Letters*, vol. 153, pp. 104954, 2021.

[8] Guangyu Wu, Guang Chen, Hao Zhang, and Chao Huang, "Fully distributed event-triggered vehicular platooning with actuator uncertainties," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 7, pp. 6601–6612, 2021.

[9] Yue Wei, Wang Liyuan, and Guo Ge, "Event-triggered platoon control of vehicles with time-varying delay and probabilistic faults," *Mechanical Systems and Signal Processing*, vol. 87, pp. 96–117, 2017, Signal Processing and Control challenges for Smart Vehicles.

[10] Shixi Wen, Ge Guo, Bo Chen, and Xiue Gao, "Cooperative adaptive cruise control of vehicles using a resource-efficient communication mechanism," *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 1, pp. 127–140, 2019.

[11] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner, "Microscopic traffic simulation using sumo," in *The 21st IEEE International Conference on Intelligent Transportation Systems*. 2018, IEEE.

[12] Christoph Sommer, David Eckhoff, Alexander Brummer, Dominik S. Buse, Florian Hagenauer, Stefan Joerer, and Michele Segata, *Veins: The Open Source Vehicular Network Simulation Framework*, pp. 215–252, Springer International Publishing, Cham, 2019.

[13] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun, "Carla: An open urban driving simulator," 2017.

[14] Guodong Rong, Byung Hyun Shin, Hadi Tabatabaee, Qiang Lu, Steve Lemke, Mārtiņš Možeiko, Eric Boise, Geehoon Uhm, Mark Gerow, Shalin Mehta, Eugene Agafonov, Tae Hyung Kim, Eric Sterner, Keunhae Ushiroda, Michael Reyes, Dmitry Zelenkovsky, and Seonman Kim, "Lgsvl simulator: A high fidelity simulator for autonomous driving," 2020.

[15] Richard Johansson, David Williams, Anders Berglund, and Pierre Nugues, "Carsim: a system to visualize written road accident reports as animated 3d scenes," in *Proceedings of the 2nd Workshop on Text Meaning and Interpretation*, USA, 2004, TextMean '04, p. 57–64, Association for Computational Linguistics.

[16] Runsheng Xu, Yi Guo, Xu Han, Xin Xia, Hao Xiang, and Jiaqi Ma, "Opencda:an open cooperative driving automation framework integrated with co-simulation," 2021.

[17] Jiaxun Cui, Hang Qiu, Dian Chen, Peter Stone, and Yuke Zhu, "Coopernaut: End-to-end driving with cooperative perception for networked vehicles," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, IEEE.

[18] Genjia Liu, Yue Hu, Chenxin Xu, Weibo Mao, Junhao Ge, Zhengxiang Huang, Yifan Lu, Yinda Xu, Junkai Xia, Yafei Wang, and Siheng Chen, "Towards collaborative autonomous driving: Simulation platform and end-to-end system," 2024.

[19] Dongyao Jia, Jie Sun, Anshuman Sharma, Zuduo Zheng, and Bingyi Liu, "Integrated simulation platform for conventional, connected and automated driving: A design from cyber–physical systems perspective," *Transportation Research Part C: Emerging Technologies*, vol. 124, pp. 102984, 2021.

[20] Philip Koopman and Michael Wagner, "Challenges in autonomous vehicle testing and validation," *SAE International Journal of Transportation Safety*, vol. 4, no. 1, pp. 15–24, 2016.

[21] Shuo Feng, Yiheng Feng, Xintao Yan, Shengyin Shen, Shaobing Xu, and Henry X. Liu, "Safety assessment of highly automated driving systems in test tracks: A new framework," *Accident Analysis Prevention*, vol. 144, pp. 105664, 2020.

[22] Shuo Feng, Xintao Yan, Haowei Sun, Yiheng Feng, and Henry X. Liu, "Intelligent driving intelligence test for autonomous vehicles with naturalistic and adversarial environment," *Nature Communications*, vol. 12, no. 1, pp. 748, Feb. 2021.

[23] NVIDIA, "Self-Driving Cars," https://www.nvidia.com/en-us/self-driving-cars/, 2024, Accessed: 2024-08-02.

[24] Hexagon, "Autonomous Vehicle Research Platforms," https://hexagon.com/solutions/autonomous-vehicle-research-platforms, 2024, Accessed: 2024-08-02.

[25] Shinpei Kato, Shota Tokunaga, Yuya Maruyama, Seiya Maeda, Manato Hirabayashi, Yuki Kitsukawa, Abraham Monrroy, Tomohito Ando, Yusuke Fujii, and Takuya Azumi, "Autoware on board: Enabling autonomous vehicles with embedded systems," in *Proceedings of the 9th ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*, 2018, pp. 287–296.

[26] Federal Highway Administration, "FHWA Cooperative Automation Research: CARMA Proof-of-Concept TSMO Use Case Testing: CARMA Cooperative Perception Low-Level Concept of Operations," Tech. Rep., Federal Highway Administration, Washington, DC, 2022.

[27] U.S. Department of Transportation Federal Highway Administration, "CARMA 1 Tenth," https://github.com/usdot-fhwa-stol/carma-1-tenth, 2024, Accessed: 2024-08-02.

[28] Wei Tang, Ming Yang, Zheming Lv, Qiyang Qian, Ti Su, Bing Wang, and Chunxiang Wang, "Microiv: A cooperative driving hardware simulation platform for cooperative-its," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 10, pp. 9173–9182, 2018.

[29] Daniel Bischoff, Florian Alexander Schiegg, Tobias Meuser, Dieter Schuller, Nils Dycke, and Ralf Steinmetz, "What cooperation costs: Quality of communication and cooperation costs for cooperative vehicular maneuvering in large-scale scenarios," in *International Conference on Vehicle Technology and Intelligent Transport Systems*, 2020.

[30] Viktor Lizenberg, Mhd Redwan Alkurdi, Ulrich Eberle, and Frank Köster, "Intelligent co-simulation framework for cooperative driving functions," in *2021 IEEE 17th International Conference on Intelligent Computer Communication and Processing (ICCP)*, 2021, pp. 109–115.

[31] U.S. Department of Transportation, "Connected Vehicle Research," https://www.its.dot.gov/research_archives/connected_vehicle/dot_cvbrochure.htm, 2024, Accessed: 2024-08-02.

[32] MAVLink Dev Team, "Mavros - mavlink extendable communication node for ros with proxy for ground control station," 2024, Accessed: 2024-05-27.

[33] Lorenz Meier, Dominik Honegger, and Marc Pollefeys, "Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 6235–6240.

[34] ArduPilot Dev Team, "Ardupilot," 2024, Accessed: 2024-05-27.

[35] Farid Ahmed and Zaid, "Basic safety messages (bsm)," 2019, Accessed: 2024-05-27.

[36] Steven E Shladover, Christopher Nowakowski, Xiao-Yun Lu, and Robert Ferlis, "Cooperative adaptive cruise control: Definitions and operating concepts," *Transportation Research Record*, vol. 2489, no. 1, pp. 145–152, 2015.

[37] Mahdi Razzaghpour, Shahriar Shahram, Rodolfo Valiente, and Yaser P Fallah, "Impact of communication loss on mpc based cooperative adaptive cruise control and platooning," in *2021 IEEE 94th Vehicular Technology Conference (VTC2021-Fall)*. IEEE, 2021, pp. 01–07.