OpenCDA-∞: A Closed-loop Benchmarking Platform for End-to-end Evaluation of Cooperative Perception (Supplementary Material)

Chia-Ju Chen[1], **Runsheng Xu**[1], **Wei Shao**[2], **Junshan Zhang**[2], **Zhengzhong Tu**[3]
[1]UCLA, [2]UC Davis, [3]Texas A&M University

Contents

38th Conference on Neural Information Processing Systems (NeurIPS 2024) Track on Datasets and Benchmarks.

A Appendix

A.1 Detailed Specifications of OpenCDA-∞

In this section, we will detail the specifications of our proposed OpenCDA-∞ simulation and benchmarking platform for the planning-oriented, safety-critical evaluation of cooperative perception models in a systematic approach.

A.1.1 The OpenCDA Simulation Platform

OpenCDA [1] is an open-source co-simulation-based research and engineering framework tailored for prototyping, developing, and testing cooperative driving automation (CDA) systems. Integrating automated driving simulation tools such as CARLA [2] and SUMO [3], OpenCDA provides a versatile environment to evaluate various CDA applications. OpenCDA supports a range of functionalities, including perception, localization, planning, control, and V2X communication. As illustrated in Fig. 1, OpenCDA organically combines multiple core components: simulation tools, a Python-based Cooperative Driving Automation (CDA) system, and a comprehensive scenario manager, enabling researchers to simulate complex cooperative driving scenarios with high flexibility.

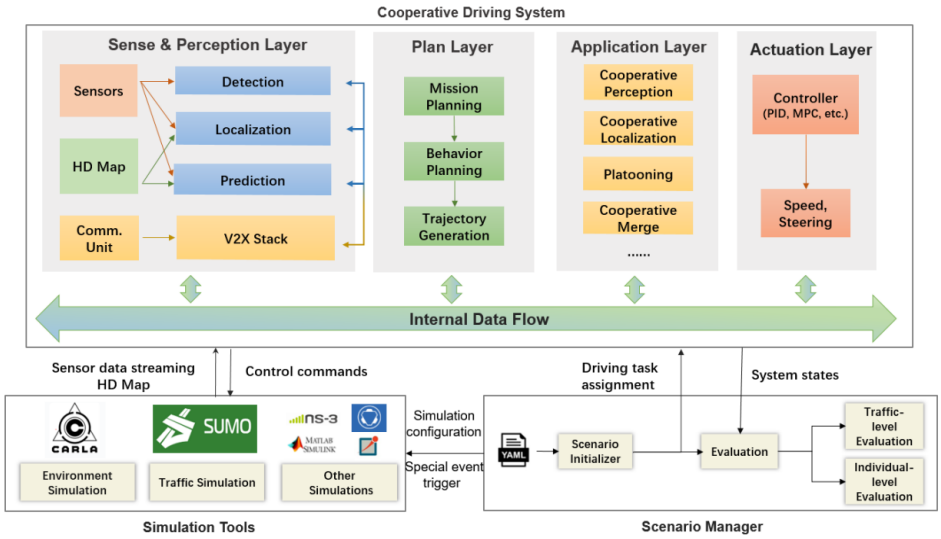


Figure 1: **The overall system design of OpenCDA.** The full-stack software of the designed open-source cooperative driving automation (OpenCDA) system interacts with simulation tools to simulate and test the system-level performance in pre-defined scenarios in Carla Towns and Culver City.

Simulation Tools Integration. OpenCDA employs CARLA [2], an open-source driving simulator powered by Unreal Engine, known for its high-quality rendering capabilities and versatile server-client architecture. While CARLA excels in detailed driving simulations, its limitations in managing large-scale traffic are further mitigated by our further integration of SUMO [3], a comprehensive traffic simulator adept at modeling realistic traffic behavior and scenarios. This dual-tool approach allows researchers to choose between detailed driving simulations with CARLA, broad traffic simulations with SUMO, or a co-simulation approach, to achieve complex, on-demand scenario and traffic simulation for each specific requirement.

Scenario Management. The scenario manager in OpenCDA is a sophisticated module segmented into four primary components: the configuration file, initializer, event trigger, and evaluation functions. It blends static elements (e.g., road structures and infrastructure defined by CARLA's assets) with dynamic elements (e.g., traffic flow and weather conditions controlled via YAML files). The configuration loader parses these files to guide the simulation, then designating tasks for the respective Connected and Autonomous Vehicles (CAVs). This system allows for the simulation of complex scenarios, including rare events, and enables performance evaluations at both the individual vehicle level in CARLA and at the broader traffic level in SUMO.

Cooperative Driving Automation System: OpenCDA's cooperative driving system, structured in modular layers, leverages CARLA and SUMO through a streamlined API to perform cooperative driving tasks. Sensors mounted on CAVs collect raw environmental data, which is processed through sequential layers: sensing, perception, planning, and actuation. This architecture supports simulations of both cooperative systems and single vehicle intelligence, facilitating mixed-traffic scenarios. The cooperation is primarily activated in the application layer, where CAVs exchange critical data such as positions and intents through V2X communication, aligning on cooperative strategies like platooning. This layer's protocols dynamically adapt based on cooperative needs, such as enhancing object detection through shared sensor data fusion.

Modularity and Customization. A distinctive feature of OpenCDA is its modular framework, where each layer comes with default protocols and algorithms that can be easily replaced with advanced ones. This modularity not only allows comprehensive evaluation of the entire CDA system but also facilitates the comparison of individual algorithms within a consistent framework. The built-in algorithms, including those for cooperative platooning, serve as reproducible baselines to provide a valuable reference for emerging research.

A.1.2 The OpenCOOD Cooperative Detection Toolkit

The aforementioned OpenCDA platform does not support running cooperative detection models on the fly. Instead, it directly retrieves bounding boxes from the server, bypassing the need to run any sophisticated detection models actively within the simulation environment. This architectural choice, while efficient, presented limitations, especially for researchers and developers keen on investigating the immediate impacts of cooperative detection algorithms on autonomous driving behaviors.

The OpenCOOD [4] Cooperative Detection Toolkit offers a comprehensive suite of tools and algorithms optimized for collaborative perception. Fig. 2 visualizes the visualization of the bounding box overlay on top of the LiDAR point clouds and camera images. It mainly supports offline cooperative detection model training and testing, based on the pre-exported simulation data generated by simulation platforms like OpenCDA. It also supports training and evaluating real-world datasets like DAIR-V2X [5] if configured properly.

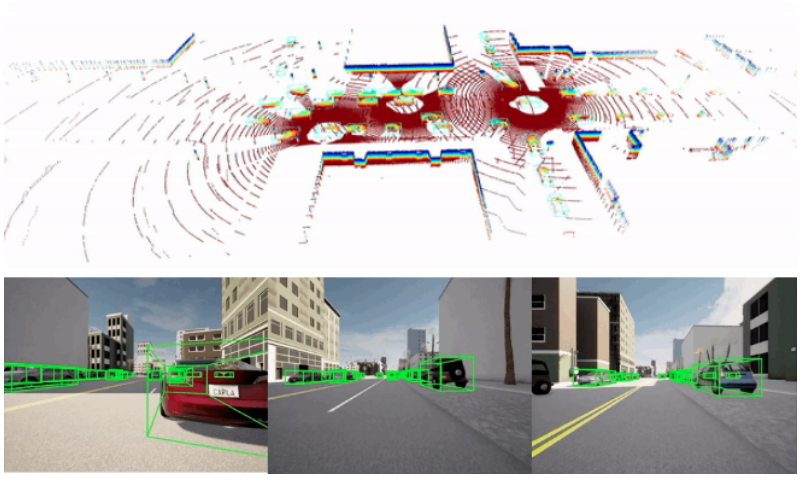


Figure 2: **Demo of OpenCOOD Cooperative Detection Toolkit.** It offers a comprehensive benchmarking platform for evaluating cooperative detection models.

The OpenCOOD framework adopts a system design that considers the V2X/V2V perception problem a multi-agent perception system, where a number of agents of different types (e.g., smart infrastructure or connected and automated vehicles (CAV)) communicate, perceive, and share information/features with each other. To simulate the real-world scenarios, OpenCOOD assumes that all the agents have imperfect locations (e.g., subject to location error) and there exists an uncertain time delay during communication (by simulating a feature transmission delay). The ultimate design goal is to develop a performant and robust V2X/V2V feature fusion module that capably generates accurate bonding boxes under the above-mentioned noise. As shown in Fig. 3, the OpenCOOD framework

consists of five major components: 1) V2X metadata sharing, 2) LiDAR feature extraction, 3) feature compression & sharing, 4) V2X fusion network, 5) a detection head.

❶ **V2X metadata sharing.** In the initial collaboration stage, each agent within the communication range shares metadata such as poses, extrinsics, and agent type. One agent is selected as the ego vehicle to construct a V2X graph, where nodes represent CAVs or infrastructure and edges represent directional V2X communication channels. OpenCOOD assumes metadata transmission is well-synchronized. Upon receiving the ego vehicle's pose, connected agents project their own LiDAR point clouds to the ego vehicle's coordinate frame before feature extraction.

❷ **Feature extraction**: OpenCOOD implemented various LiDAR feature extractors: PointPillar [6], Pixor [7], VoxelNet [8], and SECOND [9]. By default, the anchor-based PointPillar method is recommended to extract visual features from point clouds due to its low inference latency and optimized memory usage. Raw point clouds are converted into a stacked pillar tensor, scattered to a 2D pseudo-image, and fed into the PointPillar backbone. The backbone extracts feature maps $F_i^t \in \mathbb{R}^{H \times W \times C}$, representing agent i's features at time t_i with height H, width W, and channels C.

❸ **Compression and sharing**: To reduce transmission bandwidth, a series of 1×1 convolutions progressively compresses the feature maps along the channel dimension. The compressed features, with size (H, W, C') (where $C' \ll C$), are transmitted to the ego vehicle and projected back to (H, W, C) using a coupled 1×1 convolution decoder. Due to inevitable time gaps between data capture by connected agents and feature reception by the ego vehicle, features are often temporally misaligned. To correct this delay-induced spatial misalignment, OpenCOOD uses a spatial-temporal correction module (STCM), which employs a differential transformation and sampling operator Γ_ξ to spatially warp the feature maps. An ROI mask prevents the network from focusing on padded zeros caused by spatial warp.

❹ **V2X/V2V fusion network**: The intermediate features $H_i = \Gamma_\xi(F_i^t) \in \mathbb{R}^{H \times W \times C}$ aggregated from connected agents are fed into the V2X/V2V fusion network, the core component of OpenCOOD, for inter-agent and intra-agent feature fusion. High-resolution feature maps are maintained throughout the network, as the absence of high-definition features significantly impairs object detection performance.

❺ **Detection head**: After receiving the final fused feature maps, two 1×1 convolution layers are applied for box regression and classification. The regression output is $(x, y, z, w, l, h, \theta)$, denoting the position, size, and yaw angle of predefined anchor boxes. The classification output is the confidence score of being an object or background for each anchor box. Smooth L_1 loss is used for regression and focal loss for classification.

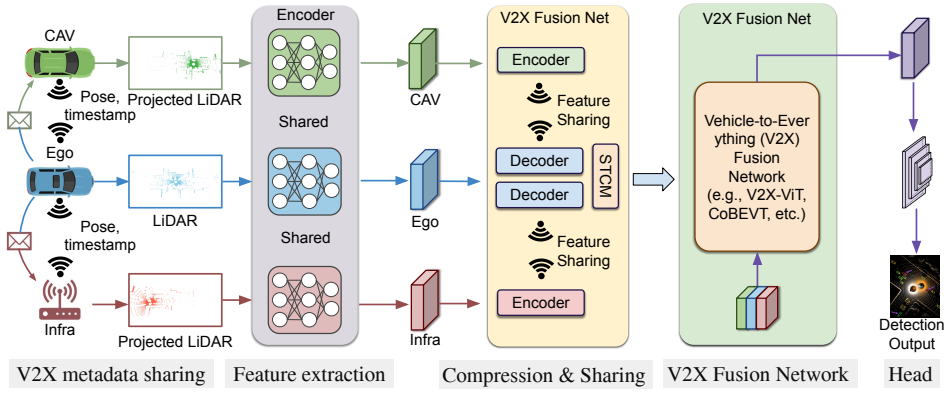


Figure 3: **The system design of OpenCOOD detection architecture.**

This framework effectively addresses the challenges of V2X perception, enabling robust and efficient perception in autonomous driving systems through advanced cooperative fusion strategies. We have included several baseline and state-of-the-art models to conduct the experiments on OpenCDA-∞:

1. **No Fusion**: only uses the ego vehicle's own LiDAR point cloud without collaboration.
2. **Early Fusion**: directly aggregates the raw LiDAR point clouds and learns fusion on top of the raw data, requiring the highest transmission bandwidth as the raw data is large in size.

3. **Late Fusion**: receives all the final detection results and applies non-maximum impression to produce the final ego results. This method requires the least transmission bandwidth as it only requires transmitting the bounding box axis.
4. **OPV2V [4] (intermediate fusion)**: transmits intermediate neural features (e.g., PointPillar features) to ego car where ego applies an attention model to fuse them.
5. **V2X-ViT [10] (intermedia fusion)**: transmits intermediate neural features where the ego applies a V2X-ViT transformer to fuse them.

Implementation details. All the models were trained on the OPV2V training set [4], where the set splits are 6,764/1,981/2,719 frames. The number of vehicles is between 2 and 7 for the entire dataset. During training, a random CAV was selected as the ego vehicle; during the test phase, a fixed ego was defined for a fair comparison. The communication range for each agent was set as 70 meters, ignoring CAVs beyond this radius range. The voxel resolution for the PointPillar backbone is 0.4 meters in both height and width. By default, the feature transmission compression rate is 32 for all the compared intermediate fusion methods. We follow the original papers to set up the hyperparameters for each specific model. The Adam optimizer [11] was employed for training, with an initial learning rate of 0.001, steadily decaying every 10 epochs by a factor of 0.1. We trained all the models using the Tesla V100, and all the model training took about a week to finish.

Integration into OpenCDA-∞. We have made further efforts to bridge the gap by amalgamation of cooperative detection models implemented in OpenCOOD into OpenCDA. Specifically, by seamlessly compiling OpenCOOD [4] as an additional `MLManager` component, OpenCDA can now not only run a diverse array of cooperative detection models on the fly but also allow the outputs of these models to directly steer the planning and decision-making processes of its autonomous agents. This enriched feature makes it possible to investigate the influence of state-of-the-art cooperative perception models at a system level, which can more faithfully simulate real-world scenarios. With this new module, the OpenCDA-∞ would require at least 8GB GPU memory to run the simulation to capably load the OpenCOOD models for online detection.

A.1.3 OpenSCENARIO Add-ons

By default, OpenCDA utilizes the built-in CARLA traffic manager to simulate vehicle dynamics. During scenario initialization, routes are computed based on initial spawn and destination positions. Additionally, auxiliary vehicles are spawned randomly within predefined ranges and probability distributions, resulting in non-deterministic behavior in each run. However, this setup offers limited granular control over individual actor behavior, making specific scenario generation challenging.

OpenSCENARIO [12] is a standardized XML-based language designed for defining driving scenarios, providing a structured approach to create reproducible, configurable, and complex simulations. It enables scripting scenarios ranging from simple straight-road driving to complex urban environments with multiple dynamic actors. The framework supports encoding high-level traffic rules and participant behaviors. ScenarioRunner is an extension to CARLA that facilitates the execution and evaluation of scenarios described using OpenSCENARIO or a Python interface. It translates high-level OpenSCENARIO XML-based definitions into sequences of actions and events within the CARLA simulation, effectively bridging the gap between text-defined scenarios and the CARLA simulation environment.

Our study presents twelve reproducible and challenging scenarios developed with OpenSCENARIO, wherein cooperative perception proves more beneficial than no collaboration but still struggles due to the complexity of the scenarios. Furthermore, we introduce a programming paradigm demonstrating how OpenCDA can consume scenarios defined by OpenSCENARIO. The below listing illustrates an exemplary XML configuration file defining an OpenSCENARIO scene specification.

```
<?xml version="1.0"?>
<scenarios>
    <scenario name="Scenario_4" type="Scenario_4" town="Town03"
    enable_cav="True">  # Define the scene.
        <ego_vehicle x="23.8" y="3.7" z="0.3" yaw="0" model="vehicle.
    nissan.patrol" />  # Ego vehicle
        <other_actor x="50.8" y="3.7" z="-500" yaw="0" model="vehicle.
    dodge.charger_2020" />  # Other actors
```

```
        <other_actor x="58.8" y="7.4" z="-500" yaw="0" model="vehicle.
    dodge.charger_2020" />  # Other actors
        <other_actor x="58.8" y="3.7" z="-500" yaw="0" model="vehicle.
    dodge.charger_2020" rolename="cav_1" color="(0,0,255)" />  # CAV
    vehicle
        <weather cloudiness="0" precipitation="0"
    precipitation_deposits="0" wind_intensity="0" sun_azimuth_angle="0
    " sun_altitude_angle="75" />  # Define the scene specs.
     </scenario>
</scenarios>
```

Listing 1: An exemplary XML config file to define an OpenSCENARIO.

A.1.4 Trajectory Prediction

In its current architecture, OpenCDA lacks inherent support for trajectory prediction. This limitation significantly constrains our ability to understand how the predictable future trajectories of other vehicles influence subsequent planning activities, which are critical in automated driving systems. To better emulate real-world traffic scenarios and simulate driver behaviors, this study introduces a trajectory prediction module supporting various commonly used behavior models. This addition aims to enhance the realism and complexity of the simulation environment.

We implement five prediction models, each designed to capture specific driving behaviors, thereby facilitating nuanced simulations. These models include:

❶ **Constant Velocity**: This model is suitable for scenarios involving consistent traffic flow on highways or steady-state vehicle motion in urban environments with light traffic, where vehicle speed remains relatively constant. The corresponding formula for position over time is expressed as $x = vt$, where x denotes displacement, v is constant velocity, and t is time.

❷ **Constant Acceleration**: Ideal for scenarios where vehicles are continuously accelerating or decelerating, such as entering or exiting highways. The motion is mathematically described by $v = u + at$, where v is the final velocity, u is the initial velocity, a is constant acceleration, and t is time.

❸ **Constant Speed and Yaw Rate**: This model applies to situations where vehicles are moving at a constant speed and direction, such as when a vehicle is steadily cornering or changing lanes at a fixed rate. It can be represented as $\theta = \omega t$, where θ is the yaw angle, ω is the constant yaw rate, and t is time.

❹ **Constant Acceleration and Yaw Rate**: Used in more complex scenarios where a vehicle is accelerating or decelerating while simultaneously changing direction at a constant rate, such as taking an exit ramp at varying speeds. This scenario combines the equations $x = vt$ and $\theta = \omega t$ to capture both linear and angular movements.

❺ **Physics Oracle Model**: This highly precise model is employed in the most complex scenarios requiring predictions of various driving behaviors, including sudden stops, emergency maneuvers, and high-speed chases. It is an ensemble of various physical laws and principles, encompassing both linear and rotational dynamics to capture the full range of possible vehicle behaviors in response to dynamic environments.

These models collectively advance the realism and fidelity of the simulation environment, providing a more comprehensive tool for understanding and predicting vehicle behaviors in automated driving systems.

A.1.5 Robust Planning with Prediction

A.1.6 Robust Behavior Planning with Trajectory Prediction

OpenCDA designs planning behavior using a rule-based finite-state machine system, conditional on specific traffic dynamics. However, without the presence of the prediction module, we have observed that the default planning behavior falls short in handling numerous challenging scenarios. These include intersections with orthogonal traffic, complex lane merging, and vehicles abruptly emerging

Algorithm 1 Robust Behavior Planning Algorithm.

```
Initialize the Carla world with YAML and XML configs
Get the current vehicle state and sensor data
Calculate the global route
while vehicle is running and far from destination do
    Update vehicle state and sensor data
    Run online cooperative perception models
    Do collision check for observed vehicles
    Run trajectory prediction for actors
    Do collision check for predicted trajectories
    if red light detected ahead then
        Brake immediately
    end if
    if in intersection then
        Decelerate
    end if
    if slow vehicle detected in front then
        Do collision check for overtaking
        if overtake allowed and safe to overtake then
            Perform overtaking
        else
            follow the front vehicle
        end if
    else if safe to push then
        Acceralte to temporary waypoints
    else
        Continue current action or follow the trajectory
    end if
    Update control commands to actuators
end while
```

from blind spots. These findings highlight the limitations of the current planning algorithm. Thus, we are implementing new planning components to accommodate the existence of predicted trajectories.

Algo. 1 depicts the detailed robust behavior planning algorithm in our proposed simulation platform, OpenCDA-∞. The system consists of several major states: calculating a global route, performing lane changes, formulating and executing temporary routes, overtaking, following the front vehicle, and decelerating when the ego vehicle is proximate to other obstacles. Upon integrating the proposed prediction model, future trajectories of visible actors become predictable. Consequently, we designed a mechanism to determine if the ego vehicle will collide with other obstacle vehicles within the next k seconds. Given the uncertainty of velocity and traffic conditions, we propose configurable parameters that delineate the range of possible future positions, as detailed in Fig. 2 of main paper, to account for variations in locations.

Specifically, suppose the planned waypoints of the ego car are $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_K\}$, and the predicted trajectory of a threat car is $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_K\}$. We run a collision check at a uniform time series $T = \{t_1, t_2, \ldots, t_K\}$ and check the minimum possible distance between the two vehicles at future locations within a given scanning window that accounts for prediction error:

$$L = \min_{t \in T} \min_{r \in [t-\tau, t+\tau]} \|\mathbf{x}_r - \mathbf{y}_r\|_2 . \tag{1}$$

If L is less than some predefined distance, then there is a potential collision, and the planning algorithm will adjust accordingly.

This enhancement in trajectory prediction and behavior planning bridges the gap between the simulation environment and real-world driving behavior. The simulation platform can now handle more complex scenarios, such as those defined in our proposed OPV2V-Safety benchmarks.

A.2 Detailed Specifications of OPV2V-Safety

In this section, we will detail the proposed OPV2V-Safety dataset, a supplementary set of challenging scenarios for evaluating the safety of cooperative perception models in our proposed closed-loop OpenCDA-∞ simulation benchmark.

A.2.1 Design Details

Simulation Platform. We utilize our proposed OpenCDA-∞, the enhanced co-simulation platform for V2X/V2V cooperative automation research based on CARLA, as our simulation toolkit. Integrated with the OpenSCENARIO add-on, we customized the behaviors of all the other actors within the simulation and used the XML config file to control the behavior of the ego vehicle and CAVs. All the detection results from the compiled OpenCOOD models (e.g., OPV2V, V2X-ViT) are loaded prior to simulation and inference at per-frame to generate bounding boxes on the fly. These detection results are employed as input for the downstream trajectory prediction and robust planning. After the simulation has concluded, we record the evaluation metrics and generate our comparison report.

Scenario Setting. We generate the scenarios based on the eight default towns, which are directly available in CARLA, for easier reproduction. In each scenario, we follow the safety-critical pre-crash traffic from NHTSA to set up the driving route of the ego vehicle and the vehicle prone to colliding with it. We also spawn multiple large trucks as obstacles to block the sensors (LiDAR and cameras) in the ego car, making the scenarios even more challenging. To tailor them in the context of V2V co-perception, we initiate another CAV in collaboration with ego to capably 'see' the collision vehicle ahead of the ego vehicle. Our dataset has an average of two intelligent vehicles in each scenario.

Sensor Configuration. Similar to previous works [4, 13], we configured each CAV to be equipped with 4 RGB cameras facing front, back, left, and right respectively, so that they collectively cover the whole 360° panoramic view. Each camera has 110° field-of-view (FOV) and reads out 800×600 RGB frames. A 64-channel LiDAR with 1.3M points per second, 120-meter capturing range is mounted on the middle of the vehicle roof. We record LiDAR point clouds and camera frames at 10 Hz and save the corresponding positional data and timestamp as additional metadata.

A.3 Related works

There exist two highly relevant end-to-end benchmarks and datasets: ReasonNet [14] and Coopernaut [15]. Both ReasonNet and Coopernaut are significant contributions to the field, each with unique approaches to addressing challenges in cooperative perception and autonomous driving. Here we discuss the striking difference of our proposed framework as compared to theirs. ReasonNet focuses on global and temporal reasoning for handling occlusions and predicting the future behavior of objects, which is crucial for navigating complex urban environments. However, we'd like to highlight that our work fundamentally differs from ReasonNet: ReasonNet leverages global and temporal reasoning to solve occlusion problems, while our OpenCDA-∞ offers a unique contribution to employ multi-agent cooperative/V2V perception, where multiple vehicles are connected to share view sights within a collaborative network, for obstacle handling. The things in common are both to handle severe occlusion, but our methodologies and use cases are strikingly different. Additionally, our OPV2V-Safety dataset specifically targets pre-crash situations, adding another layer of stress testing under extreme conditions that complement the scenarios addressed by ReasonNet. We have designed the scenarios in such a way that without connected vehicles, the crash will likely happen, but with connected vehicles, the ego car will be able to observe the occluded regions, thus yielding smoother and better planning results. Coopernaut introduces an end-to-end learning model using cross-vehicle perception via V2V communications, with a focus on vision-based cooperative driving. It employs the AUTOCASTSIM framework to evaluate the model in challenging, accident-prone scenarios. Coopernaut focuses on imitation learning-based planning algorithms for end-to-end driving. In contrast, our OpenCDA-∞ supports both modular components (that are widely deployed by the industry), as well as end-to-end approaches. Empowered by OpenCDA, our frameworks feature more advanced mobility features like platooning. Lastly, OpenCDA-∞ provides a comprehensive platform to evaluate the system-wide impacts of these perception models on autonomous driving safety and performance.

A.4 Data Analysis and Visualization

Inspired by the guidelines provided by the National Highway Traffic Safety Administration (NHTSA) [16], we carefully crafted 12 pre-crash scenarios, representing challenging driving conditions. To push the envelope further, we introduced additional obstacles in each scenario, simulating situations where a single vehicle's visibility is critically hindered, making it vulnerable to accidents. This serves as a foundation to explore the potential of multi-agent cooperative perception in ameliorating these visibility issues.

Figure Fig. 4 presents the pivotal moments just before potential crashes for all 12 scenarios, encapsulating a diverse range of daily hazardous driving situations. In particular, our designs include: 1) Left turn obstacles A/B/C, 2) Right turn obstacle A, 3) Straight obstacle A, 4) Merging obstacle A/B, 5) Unprotected left turn A, 6) Highway merging A, 7) Lane-crossing turn A, 8) Zigzagging A, and 9) Sudden stopping A. A comprehensive breakdown of these scenarios can be found in Tab. 1. The scenarios span various road configurations, including 4-way intersections, T-intersections, straight segments, midblocks, entrance ramps, highways, rural roads, and more. In each setting, the ego vehicle embarks on a pre-set path, relying on planning algorithms implemented in OpenCDA to react to unforeseen road events. Adhering to traffic norms and proactively avoiding potential hazards is paramount for the ego vehicle.

A.4.1 Specifications of Scenarios

The detailed descriptions and specifications of our carefully curated scenarios are presented in Tab. 1. It should be noted that we have pre-designed a larger space of scenarios, but have conducted pre-screening to filter out non-interesting scenarios. The remaining ones in OPV2V-Safety can indeed challenge existing cooperative perception systems, as shown in Tab. 1 of the main paper. We should note that we don't use any personal or sensitive information as our simulation are purely based on Carla simulation using OpenCDA-∞. We also did not include any pedestrians in our benchmark.

A.4.2 Evaluation Metrics

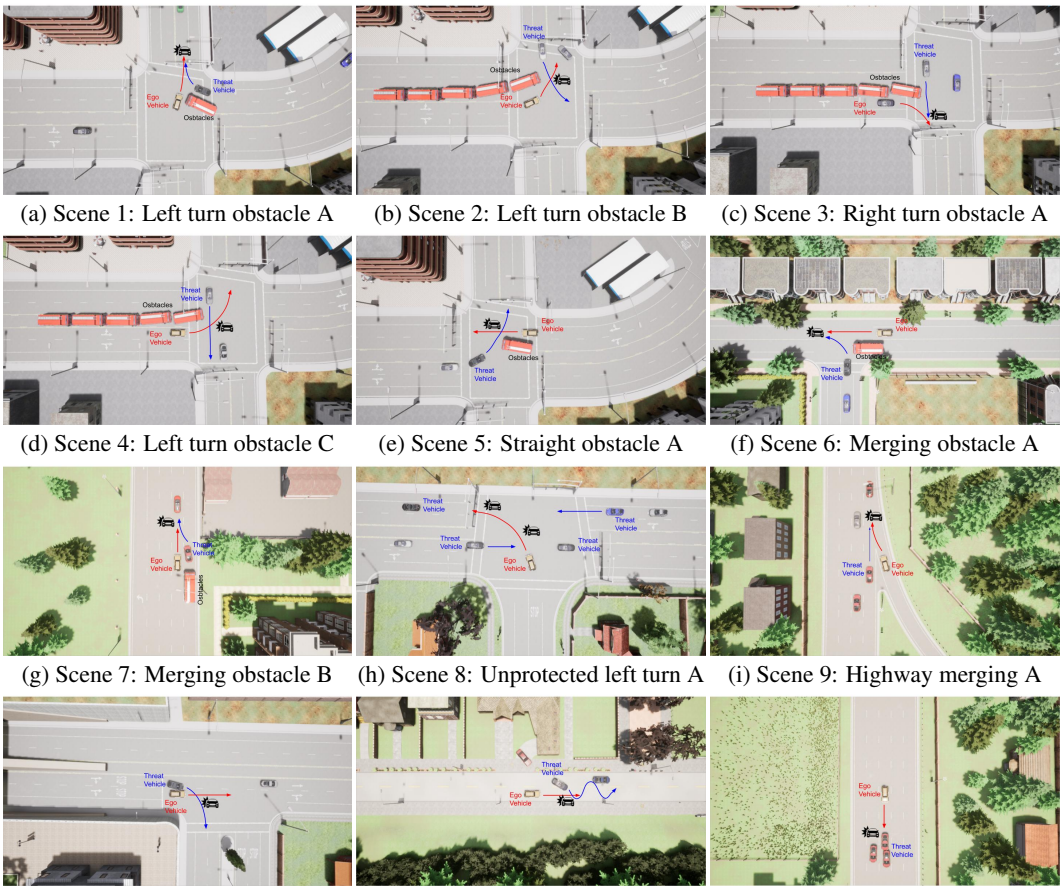
Here we detail the evaluation metrics as used in our OPV2V-Safety benchmark, as supplementary to our main paper. Specifically, we evaluate the performance on four levels: **Model level**, **Safety level**, **Efficiency level**, and **Stability level**. Finally, we propose a new **System level** overall score method to summarize the overall performance as a weighted sum of all the evaluation metrics.

❶ **Model Level**: The de facto evaluation metrics to compare 3D detection models are Average Precision (AP) at different Intersection-over-Union (IoU) thresholds. These metrics merely assess the accuracy and robustness of the model in detecting objects/vehicles in 3D space. Following [4], detection performance is evaluated in the range of $x \in [-140, 140]m$, $y \in [-40, 40]m$ near the ego vehicle. The broadcast range is 70 meters, and messages beyond this communication range will be discarded by ego. We report AP@0.3, @0.5, and @0.7 for each individual scenario respectively.

❷ **Safety Level**: Ensuring safety remains critical for any automated driving system. To effectively gauge safety, we consider two key metrics:

- Collision Rate (CR): This metric represents the frequency of collisions encountered by the vehicle, typically expressed as a ratio of collisions to total driving time or distance. A lower CR indicates safer driving behavior.
- Time-to-Collision (TTC): TTC quantifies the time it would take for two vehicles to collide if they continue at their current speed and trajectory. A smaller TTC suggests a higher risk situation. Monitoring this metric helps in determining how often an autonomous system comes close to a potential collision.
- Off-Road (OR): Measures instances when the vehicle unintentionally departs from the designated roadway, indicating potential control or perception failures. Any non-zero OR is generally considered undesirable in automated driving.

❸ **Efficiency Level**: Metrics in this category gauge the operational efficiency of the autonomous system. We include metrics like 1) Time-to-Destination (TTD), which evaluates the time taken for an autonomous vehicle to reach its destination, 2) Average Speed (AS) which records the average speed magnitude when completing the route, and 3) Average Route Distance (ARD) that calculates



(a) Scene 1: Left turn obstacle A (b) Scene 2: Left turn obstacle B (c) Scene 3: Right turn obstacle A

(d) Scene 4: Left turn obstacle C (e) Scene 5: Straight obstacle A (f) Scene 6: Merging obstacle A

(g) Scene 7: Merging obstacle B (h) Scene 8: Unprotected left turn A (i) Scene 9: Highway merging A

(j) Scene 10: Lane-crossing turn A (k) Scene 11: Zigzagging A (l) Scene 12: Studden stopping A

Figure 4: The visualization of potential pre-crash moments in the OPV2V-Safety Benchmark.

the overall route distance in meters. These metrics are effective tools to compare the efficiency of the autonomous driving algorithms during simulation.

❹ **Stability Level**: The stability level metrics are an indicator of the driving skills of AD models. Ensuring that an autonomous vehicle operates smoothly and predictably not only heightens passenger comfort but also augments the trust that other road users place in autonomous systems. We follow existing works and consider two metrics accordingly: average acceleration (ACC), and average yaw rate (AYR). The average acceleration gauges the consistency in speed adjustments, effectively preventing abrupt stops or rapid accelerations that can be jarring for occupants. On the other hand, the average yaw rate is a testament to the vehicle's steadiness during turns, ensuring that lane changes and cornering are executed with precision and grace. Collectively, these metrics provide a holistic assessment of the vehicle's ability to maneuver safely, confidently, and comfortably in diverse driving conditions.

❺ **System Level**: This level amalgamates metrics from the preceding tiers to produce a comprehensive performance indicator. By employing a weighted summation of these metrics, we gain an integrated perspective of the autonomous system's efficacy across diverse conditions. The weighting coefficients can be tailored to align with the unique criteria and emphases of any given evaluation. Specifically, we calculate the overall score as:

$$\text{OS} = \sum_{i=1}^{n} w_i \times M_i, \tag{2}$$

where M_i is the normalized metric score for $i-th$ evaluation metric:

$$M_i = \begin{cases} m_i/m_i^{max} & \text{if } m_i \text{ is the higher the better} \\ 1-(m_i/m_i^{max}) & \text{if } m_i \text{ is the lower the better} \end{cases} \tag{3}$$

Table 1: Detailed descriptions of carefully designed scenarios in the OPV2V-Safety dataset.

SCENARIO	Len (s)	Description
1: Left turn obstacle A	17.5	The ego vehicle attempts a left turn at an intersection, with its line of sight to opposite lanes obstructed by a truck, masking potentially hazardous oncoming vehicles from the right.
2: Left turn obstacle B	17.2	The ego vehicle makes a left turn at an intersection, whose view to the left is blocked by obstacle trucks, while a threat vehicle initiates a left turn from its perpendicular left.
3: Right turn obstacle A	13.6	The ego car begins a right turn at an intersection with view blocked by trucks, hindering the view of a potentially hazardous vehicle approaching from a perpendicular direction.
4: Left turn obstacle C	23.8	The ego vehicle initiates a right turn at an intersection, with its vision hampered by trucks, while a threatening vehicle approaches straight on from a perpendicular path.
5: Straight obstacle A	15.1	The ego vehicle advances straight through an intersection, but its forward view is blocked by obstacle trucks as a vehicle from the opposing direction attempts a left turn.
6: Merging obstacle A	21.3	Driving straight on a T-intersection, the ego vehicle faces a challenging situation as a vehicle from a perpendicular direction attempts a left merge, all while trucks block its view.
7: Merging obstacle B	16.9	As the ego vehicle cruises on a straight segment, an abrupt merge attempt by a vehicle from an adjacent lane takes place, with the ego's situational awareness hampered by trucks.
8: Unprotected left turn A	15.1	The ego vehicle, aiming for a left merge on a T-intersection, has to navigate through a chaotic mix of bidirectional traffic.
9: Highway merging A	18.8	Approaching the main lanes of a highway via an entrance ramp, the ego vehicle confronts fast-paced traffic densely packed with vehicles.
10: Lane-crossing turn A	20.3	As the ego vehicle moves straight ahead on a T-intersection, an erratic vehicle from the left lane suddenly cuts across, aiming to access a perpendicular lane.
11: Zigzagging A	15.9	The ego vehicle maintains a straight trajectory when a neighboring vehicle engages in a perilous double lane-change, switching lanes back and forth, typically without signaling.
12: Sudden stopping A	19.0	While the ego vehicle travels on a straight path, the leading vehicle unexpectedly halts, posing an immediate rear-end collision risk.

Table 2: Constants and weights used to compute the overall score.

	Safety Level			**Efficiency Level**			**Stability Level**	
Symbol	CR↓	TTC↑	OR↓	TTD↓	AS↑	ARD↓	ACC↓	AYR↓
$m_i^{\max}$	1	8	0.1	20	25	85	0.5	0.15
w_i	0.495	0.099	0.099	0.05	0.05	0.05	0.02	0.02

where m_i^{max} is a constant suggesting the maximum allowed value of this metric. We follow prior works [17] and set the constants and weights used to calculate the overall metrics as shown in Tab. 2.

A.4.3 Data License

The dataset will be released under the MIT License (MIT):

A.5 Additional Experiments

A.5.1 Impact of Communication Latency

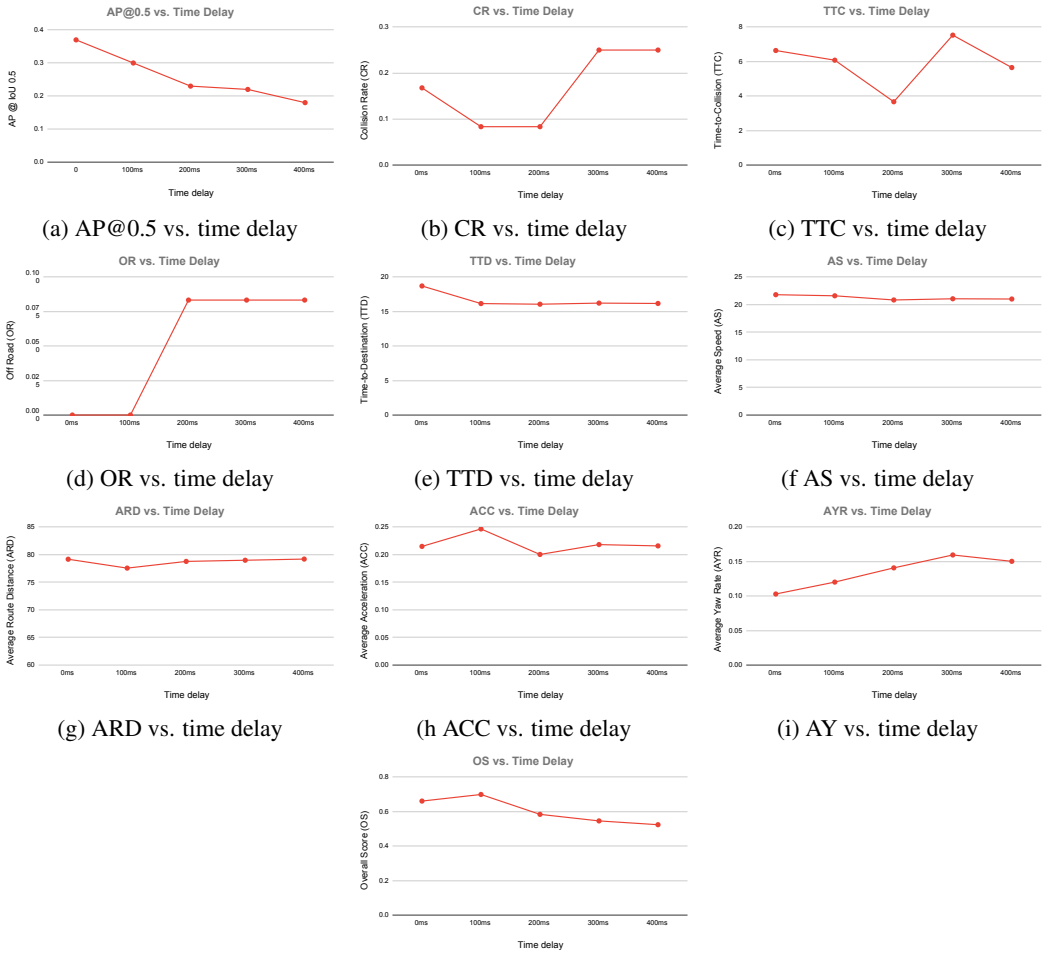
An essential aspect of autonomous driving systems is their real-time responsiveness, especially when evaluating detection accuracy, safety, and other critical metrics. Communication latency can introduce inaccurate coordinate transformations and delayed sensing information. Failure to properly handle these challenges can make the system vulnerable [10]. To address this, we investigated the impact of time delays in communication, focusing on a range of delays from 0 to 400 milliseconds. For our case study, we utilized the balanced model OPV2V, which achieves top-tier performance regardless of V2V communication status. As illustrated in Fig. 5(a), detection accuracy (measured as AP0.5) significantly declines with increased time delay, showing a 51% reduction in performance with a 400 ms delay compared to no delay.

In terms of safety metrics, the reduced detection accuracy does not correlate straightforwardly with degraded collision risk (CR) and time-to-collision (TTC). Interestingly, CR initially decreases and then increases at 100 ms and 300 ms delays, respectively. TTC fluctuates without a clear pattern, showing no consistent relationship with time delay. The occurrence rate (OR) spikes at 200 ms delay but remains steady at higher latencies. Efficiency metrics, such as time to detection (TTD), average speed (AS), and average route deviation (ARD), remain relatively stable and are minimally impacted by time delay. Regarding stability, acceleration (ACC) shows random effects, while average yaw rate (ARY) increases with delays up to 300 ms before dropping at 400 ms.

Overall, the system-level operational safety (OS) metric exhibits a general declining trend as time delay increases from 0 to 400 ms. However, the OS score surprisingly improves at 100 ms latency. This suggests that while minor delays in V2V communication can significantly reduce the detection accuracy of a cooperative model, they may not detrimentally affect overall planning performance when evaluated from a system perspective. This finding underscores the significance of our novel planning-oriented benchmarking pipeline for cooperative perception, which offers a more holistic view of testing and evaluation. It provides critical insights into the safety and operational performance of V2V algorithms, emphasizing the importance of comprehensive system-level assessments.

A.6 Limitations

Although our proposed OPV2V-Safety benchmark includes a variety of challenging scenarios, it may still lack coverage of some critical edge cases and rare events that could significantly impact safety. Real-world driving involves a vast array of unpredictable situations that are challenging to



(a) AP@0.5 vs. time delay (b) CR vs. time delay (c) TTC vs. time delay

(d) OR vs. time delay (e) TTD vs. time delay (f AS vs. time delay

(g) ARD vs. time delay (h ACC vs. time delay (i) AY vs. time delay

(j) OS vs. time delay

Figure 5: **Investigating the impact of time delay on the proposed evaluation metrics using the OPV2V model as a case study.** We observe that (a) AP@0.5 is decreasing gradually as the time delay increases from 0 to 400 ms, while the collision rate (CR) and other safety-related metrics like TTC do not necessarily deteriorate. Suprisingly, the overall score (OS) increases at 100 ms time delay but drops largely when latency surpasses 200 ms.

comprehensively model and test in a simulation environment. While simulation is an effective tool for synthesizing data, there is always a gap between simulation results and real-world performance. The benchmark's results might not fully generalize to real-world deployments due to differences in sensor configurations, vehicle dynamics, and environmental interactions that are difficult to model accurately in a simulation. Addressing these limitations through continuous refinement and incorporating real-world data and scenarios will be crucial for the reliability and safety of autonomous driving systems. We call for rigorous on-road testing to ensure that models trained on simulation datasets would improve the safety of real-world complex scenarios.

A.7 Potential Negative Societal Impacts

We carefully designed twelve safety-critical scenarios to challenge the current cooperative perception systems. Releasing this source data would provide malicious hackers with additional resources to design adversarial or backdoor attacks in real-world scenarios. Moreover, our V2V cooperative systems would require to collect vast amounts of data for each vehicle, including detailed information about vehicle locations, passenger identities, and surrounding environments. This can raise significant privacy concerns regarding who has access to this data and how it is used. Failure to properly

handle this sensitive data can lead to misuse or unauthorized surveillance. The integration of V2V or V2X communication systems introduces new cybersecurity vulnerabilities. Malicious actors could potentially hack into these systems from a weak node in the network, causing entire graph malfunctions or data breaches. This networked risk can pose serious threats to public safety and security. Despite showing impressive performance and potential, future researchers should explore such techniques more responsibly.

References

[1] Runsheng Xu, Yi Guo, Xu Han, Xin Xia, Hao Xiang, and Jiaqi Ma. Opencda: an open cooperative driving automation framework integrated with co-simulation. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 1155–1162. IEEE, 2021. 2

[2] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017. 2

[3] Cristina Olaverri-Monreal, Javier Errea-Moreno, Alberto Díaz-Álvarez, Carlos Biurrun-Quel, Luis Serrano-Arriezu, and Markus Kuba. Connection of the sumo microscopic traffic simulator and the unity 3d game engine to evaluate v2x communication-based systems. *Sensors*, 18(12):4399, 2018. 2

[4] Runsheng Xu, Hao Xiang, Xin Xia, Xu Han, Jinlong Li, and Jiaqi Ma. Opv2v: An open benchmark dataset and fusion pipeline for perception with vehicle-to-vehicle communication. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2583–2589. IEEE, 2022. 3, 5, 8, 9

[5] Haibao Yu, Yizhen Luo, Mao Shu, Yiyi Huo, Zebang Yang, Yifeng Shi, Zhenglong Guo, Hanyu Li, Xing Hu, Jirui Yuan, et al. Dair-v2x: A large-scale dataset for vehicle-infrastructure cooperative 3d object detection. In *CVPR*, pages 21361–21370, 2022. 3

[6] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *CVPR*, pages 12697–12705, 2019. 4

[7] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7652–7660, 2018. 4

[8] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4490–4499, 2018. 4

[9] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 4

[10] Runsheng Xu, Hao Xiang, Zhengzhong Tu, Xin Xia, Ming-Hsuan Yang, and Jiaqi Ma. V2x-vit: Vehicle-to-everything cooperative perception with vision transformer. In *ECCV*, pages 107–124. Springer, 2022. 5, 12

[11] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[12] CARLA authors. Openscenario support. https://carla-scenariorunner.readthedocs.io/en/latest/openscenario_support/, 2023. 5

[13] Yiming Li, Ziyan An, Zixun Wang, Yiqi Zhong, Siheng Chen, and Chen Feng. V2x-sim: A virtual collaborative perception dataset for autonomous driving. *arXiv preprint arXiv:2202.08449*, 2022. 8

[14] Hao Shao, Letian Wang, Ruobing Chen, Steven L Waslander, Hongsheng Li, and Yu Liu. Reasonnet: End-to-end driving with temporal and global reasoning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13723–13733, 2023. 8

[15] Jiaxun Cui, Hang Qiu, Dian Chen, Peter Stone, and Yuke Zhu. Coopernaut: end-to-end driving with cooperative perception for networked vehicles. In *CVPR*, pages 17252–17262, 2022. 8

[16] Wassim G Najm, Raja Ranganathan, Gowrishankar Srinivasan, John D Smith, Samuel Toma, Elizabeth Swanson, August Burgett, et al. Description of light-vehicle pre-crash scenarios for safety applications based on vehicle-to-vehicle communications. Technical report, United States. National Highway Traffic Safety Administration, 2013. 9

[17] Chejian Xu, Wenhao Ding, Weijie Lyu, Zuxin Liu, Shuai Wang, Yihan He, Hanjiang Hu, Ding Zhao, and Bo Li. Safebench: A benchmarking platform for safety evaluation of autonomous vehicles. *Advances in Neural Information Processing Systems*, 35:25667–25682, 2022. 11