CARMEN+: Center for Automated Vehicle Research with Multimodal AssurrEd Navigation

A USDOT Tier-1 University Transportation Center

Final Report: Misbehavior Detection in Uncertainty Aware Object Level Cooperative Perception for Motion Planning

P.I.	Project Info:
Keith A. Redmill, Ph.D.	Grant No. 69A3552348327
https://orcid.org/0000-0003-1332-1332	DUNS: 832127323
Vignesh Srinivasan	EIN #: 31-6025986
https://orcid.org/0009-0007-7579-3913	Project Effective: October 30, 2023
The Ohio State University	Project End: August 30, 2025
Department of Electrical and Computer Engineering	Submission: October 24, 2025

Technical Report Documentation Page

1. Report No. Pending assignment.	**2. Government Accession No** n/a	**3. Recipient's Catalog No.** n/a
4. Title and Subtitle Misbehavior Detection in Uncertainty Aware Object Level Cooperative Perception for Motion Planning		**5. Report Date** September 30, 2025 **6. Performing Organization Code** n/a
7. Author(s) Keith A. Redmill, Ph.D. https://orcid.org/0000-0003-1332-1332 Vignesh Srinivasan https://orcid.org/0009-0007-7579-3913		**8. Performing Organization Report No.** n/a
9. Performing Organization Name and Address The Ohio State University Center for Automotive Research 930 Kinnear Road Columbus, OH 43212		**10. Work Unit No. (TRAIS)** n/a **11. Contract or Grant No.** 69A3552348327
12. Sponsoring Agency Name and Address CARMEN+ University Transportation Center The Ohio State University 930 Kinnear Road Columbus, OH 43212		**13. Type of Report and Period Covered** Final Report. October 2023 to August 2025 **14. Sponsoring Agency Code** USDOT

15. Supplementary Notes

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, by a grant 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.

16. Abstract

Cooperative perception broadens the sensing range of an automated vehicle, especially in urban settings where other road users and infrastructure create occlusions. However, it raises safety concerns because received information can be incorrect due to faults or malicious attacks, which can lead to unsafe planning. We study object-level fusion strategies and related attack models in which an attacker communicates spoofed objects or alters reported object locations. Our object-level fusion merges detections from connected vehicles using inverse variance weighted box fusion, removes highly uncertain detections, and forwards fused detections and variances to a tracker. We validate on the V2V4Real dataset and observe that the method outperforms standard late fusion and surpasses several state-of-the-art intermediate fusion schemes. We address data trust with a misbehavior detection scheme that uses overlapping fields of view to test the contextual validity of received data and to update dynamic, source-specific trust scores based on consistency. Fusion is performed only after these scores are evaluated. The scheme is tested in two scenarios, a T junction with a spoofed vehicle and a case with a communicating vehicle that has erroneous pose, and it identifies misbehaving vehicles and excludes them from fusion. We consider uncertainty-aware trajectory planning by formulating an optimal control problem solved with model predictive control with probabilistic constraints. We validate the stack in four scenarios, a left turn at a T junction with an adversarial non-communicating vehicle, a collaborative perception case with an occluded non-communicating vehicle at a T-junction, a cooperative perception case with an occluded non-communicating stopped vehicle that requires collision avoidance during a right turn at a four-way intersection, and a similar case with the addition of a spoofed vehicle.

17. Key Words Cooperative Perception, Uncertainty Quantification, Multi-Object Tracking, Trust, Misbehavior Detection, Object Spoofing, CARLA Simulation, Model Predictive Control, Chance Constraints, Urban Traffic, Cooperative Driving		**18. Distribution Statement** No restrictions.	
19. Security Classif.(of this report) Unclassified	**20. Security Classif.(of this page)** Unclassified	**21. No Pages** 79 pages	**22. Price** n/a

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

Contents

List of Figures

List of Tables

Abstract

Cooperative perception extends each automated vehicle's field of view through perception data sharing among nearby connected vehicles, thereby improving AV situational awareness, especially in complex urban traffic scenarios where other road users and infrastructure components can create occlusions that limit the view of an individual AV. However, cooperative perception raises safety concerns because received information can be incorrect, false, or misleading, whether due to faulty sensors, noise, processing errors or even deliberate malicious attacks. These errors can lead to incorrect or even unsafe planning and decision making by nearby connected agents.

Successful cooperative perception requires defining what information to share and the corresponding fusion strategy. While significant research is being conducted on improving perception modules and communication systems to share object or feature level data, the quantification and use of perception uncertainty, and its effects on downstream tasks such as cooperative object tracking, system reliability and security, and vehicle motion planning and control, remain largely unexplored.

In this work, we focus on late fusion and corresponding attack models, where an attacker can spoof vehicles or alter reported object locations. We adopt object level collaboration for bandwidth efficiency and ease of deployment, in contrast to early fusion's high bandwidth demands and intermediate fusion's need for knowledge of each vehicle's sensors and sensor processing model architecture.

We first develop an object-level fusion strategy that fuses detections from nearby connected vehicles using inverse variance weighted box fusion, suppresses high uncertainty detections via each detection's uncertainty ellipse, and passes the fused detections and variances to the downstream object tracker. We also calibrate predicted uncertainties. The method, evaluated on the V2V4Real dataset, significantly outperforms standard late fusion and surpasses several state of the art intermediate fusion schemes.

We next address data trust in cooperative perception by developing a misbehavior detection approach that uses overlapping fields of view to verify the contextual validity of communicated object data and to update dynamic trust scores for each source based on consistency. Data fusion is performed only after these scores are evaluated. We test our approach using the OpenCDA simulation framework built on CARLA on two scenarios, a T-junction with a spoofed vehicle and a case involving a communicating vehicle with erroneous pose information. The results demonstrate that the algorithm identifies misbehaving vehicles and excludes them from the fusion process.

Finally, we approach uncertainty-aware trajectory planning by constructing an optimal control problem, solved using model predictive control, with probabilistic constraints to balance risk and progress as well as performance, safety, and road boundaries, thereby enabling non-conservative motion planning. We validate the resulting AV stack in four scenarios: a single non-communicating vehicle adversarially perturbs its trajectory as the AV attempts a left turn at a T-junction, a multi-vehicle collaborative perception scenario with an occluded non-communicating vehicle approaching a T-intersection through which the AV must pass, a multi-vehicle cooperative perception scenario involving an occluded non-communicating stopped obstacle vehicle that requires the AV to plan and execute a collision avoidance maneuver while completing a right turn at a four-way intersection, and the same 4-way intersection scenario with the addition of a spoofed obstacle vehicle. The results demonstrate that the ego vehicle successfully avoids both dynamic and static obstacles and also identifies the source of the malicious spoofed vehicle.

Note: The results presented in this report are also being submitted for potential publication in preferred academic conferences and/or journals.

Chapter 1

Uncertainty-Aware Object Level Cooperative Perception and Tracking

Automated vehicles face significant challenges in scenarios involving potentially occluded objects. To tackle this issue, recent research investigates the use of vehicle-to-vehicle (V2V) or vehicle-to-everything (V2X) communication to exchange perception data among connected automated vehicles, thereby extending their collective field of view. However, much of the existing work primarily centers on sharing and fusing object-level or feature-level data, while the uncertainties tied to these perceptions remain largely underexplored in cooperative perception. Moreover, how these uncertainties affect downstream tasks, such as cooperative object tracking, behavior planning, and control, continues to be an active research area. In this chapter, we propose a method that merges object-level perception data from nearby connected vehicles using an inverse variance-based weighted box-fusion scheme. Our approach filters out highly uncertain detections by taking into account the uncertainty ellipses surrounding each detection and integrates the fused variances into our downstream object tracking module. The proposed method is evaluated on the V2V4Real dataset. Quantitative experiments demonstrate that our approach outperforms the standard late fusion scheme by 27.4% in AP@0.5 and 48.6% in AP@0.7. Additionally, it surpasses several other state-of-the-art intermediate fusion schemes by a considerable margin.

1.1 Introduction

Object detection and localization is a fundamental requirement for the safe operation of automated vehicles (AV). With recent advances in deep learning, Deep Neural Networks (DNN) have emerged as premier tools in perception tasks due to their effectiveness in extracting key features from the surrounding environment. However, deep learning-based perception has issues with trustworthiness due to the inherent tendency of modern neural networks to be overconfident in their predictions [3]. A reliable system should be accurate when it is confident and indicate high uncertainty when it is less accurate. In a typical AV pipeline, the detection results are propagated to downstream tasks such as object tracking and behavior planners. However, this implies that any errors present in the outputs of the perception module will subsequently affect the decisions of the AV. Thus, it is important that the object detector not only provide information about its surroundings but also the uncertainties associated with each predicted object.

Deep Ensembles [4] and Monte-Carlo Dropout [5] are two of the most common approaches for estimating uncertainties. However, both methods are computationally expensive, with the former requiring an ensemble of DNNs and the latter necessitating several forward passes during the inference stage. Deploying Dropout-based uncertainty estimation in real-time scenarios is challenging due to its time-consuming nature. To address both issues, Pitropov et al. proposed LiDAR-MIMO [6], an extension of the Multi-Input Multi-Output (MIMO) [7] uncertainty estimation method for 3D object detection in LiDAR point clouds. The key contribution of the original paper is that subnetworks within the neural network can be trained to make independent predictions for multiple inputs. The key difference between these two methods is the input representation: LiDAR-MIMO performs multi-input fusion on pseudo bird's-eye-view (BEV) maps generated by passing the raw point cloud through a pillar feature network [8], whereas MIMO operates directly on raw images.

Automated vehicles not only suffer from perception uncertainties but also environmental occlusions. Current AV systems rely solely on individual perception. These systems are vulnerable to perception and tracking related issues due to the presence of occluded objects, thereby affecting the safety of all participants. They also experience limitations due to limited sensing ranges. To overcome these issues, especially in urban traffic scenarios, early works in the connected automated vehicles (CAVs) setting proposed the transmission of object-level data along with vehicle state information (e.g. position, speed) [9]. Communicating object-level information enables CAVs to navigate safely in areas where road users and infrastructure cause severe occlusions. However, object-level information is not the only way to share perception. In general, there are three fusion schemes for CAVs: **Early Fusion**, where connected vehicles communicate raw point-cloud or video data; **Late Fusion**, where CAVs share fully processed perception outputs; and **Intermediate Fusion**, where CAVs transmit feature-level information to nearby participants.

1.1.1 Early Fusion

Early fusion integrates raw sensory data at the earliest stage of the perception pipeline. Due to the irregular and sparse characteristics of LiDAR point clouds, early fusion commonly employs direct aggregation strategies. Arnold et al. [10] proposed a centralized fusion architecture that uses multiple sensors in the infrastructure. They validated their approach on datasets generated with the CARLA simulation platform, focusing on complex urban scenarios such as T-junctions and roundabouts. While early fusion generally provides superior accuracy, it requires significant bandwidth and computational resources to transmit and process raw sensor data, thus relying heavily on high-performance edge computing infrastructure. There also can be significant issues related to sensor calibration and alignment errors and potential communication delays.

1.1.2 Late Fusion

Late fusion operates at the object level, combining independently detected bounding boxes from distributed perception nodes through spatial-temporal alignment and fusion. Andreas Rauch et al. [9] proposed one of the earliest late fusion methodologies, termed Car2X-based perception, employing a Constant Turn Rate and Acceleration (CTRA) model integrated with an Unscented Kalman Filter (UKF) to robustly handle noisy measurements and alignment errors. While late fusion significantly reduces bandwidth requirements compared to early fusion, it introduces cumulative errors arising from sensor uncertainties and localization inaccuracies.

1.1.3 Intermediate Fusion

Intermediate fusion represents a compromise between early and late fusion strategies, communicating intermediate-level representations such as pseudo-bird's eye view (BEV) features derived from perception modules. Qi Chen et al. [11] introduced F-Cooper, one of the pioneering intermediate fusion methods. Intermediate fusion provides a balanced trade-off between the accuracy benefits of early fusion and the resource efficiency of late fusion. However, it requires compatibility among perception architectures, potentially mandating disclosure of proprietary details by original equipment manufacturers (OEMs).

1.1.4 Approach and Contributions

For this study, we will only consider the late fusion setting, as it is bandwidth-efficient and model-agnostic. Additionally, we assume that each vehicle communicates the uncertainty associated with each detection. CAVs with overlapping fields of view may observe the same object. When they exchange this information, it is necessary to fuse similar detections to avoid duplicates in the object list. In our late-fusion setting, we fuse detections using their predicted uncertainties [12] [13]. For object tracking, we modify AB3DMOT [14], the conventional Kalman filter-based tracking method, to incorporate the merged predicted uncertainties. Additionally, we adapt the life cycle management of trajectory tracks to a detection score-based method, replacing the usual max-age parameter [15]. The major contributions of this chapter are as follows:

- Utilizing an inverse-variance Weighted-Box Fusion (WBF) scheme [13] to fuse perception outputs from nearby connected vehicles.
- Filtering out uncertain detections by analyzing the uncertainty ellipses [16].
- Adapting AB3DMOT to incorporate predicted uncertainties and track objects using score-based life cycle management [15].

1.2 Related Works

Cooperative perception has recently gained significant traction, as it extends each CAV's field of view, enhancing both the safety and reliability of the AV pipeline. Research in cooperative perception is ongoing. Arnold et al. [10] introduced some of the earliest results on cooperative perception, where they synthesized a new multi-view dataset using KITTI [17] and analyzed both early and late fusion schemes. F-Cooper is among the first works exploring intermediate fusion [11]. V2X-VIT [18] and CoBEVT [19] employ transformer-based architectures for feature-level cooperative perception. When2Comm [20] proposes a unified framework that learns both how to construct communication groups and when to communicate.

OPV2V [21] is a large-scale simulated dataset for V2V perception, while V2V4Real [2] is the first open-source, real-world, multi-modal, and multi-task dataset for V2V perception. Additionally, V2V4Real offers a 3D cooperative tracking benchmark, where detection outputs are fed into a Kalman filter, similar to AB3DMOT [14].

We focus on evaluating our fusion scheme within a Tracking-By-Detection (TBD) setting using the V2V4Real benchmark. Our proposed late fusion scheme outperforms the benchmark's best-performing intermediate fusion method, CoBEVT, in both detection and tracking.

1.3 Method - Cooperative Perception and Tracking Pipeline

We present the full perception and tracking pipeline in Figure 1.1, where $\mathbf{d}$ denotes detections represented by bounding boxes, c the confidence score of each detection, and $\mathbf{R}$ the variances

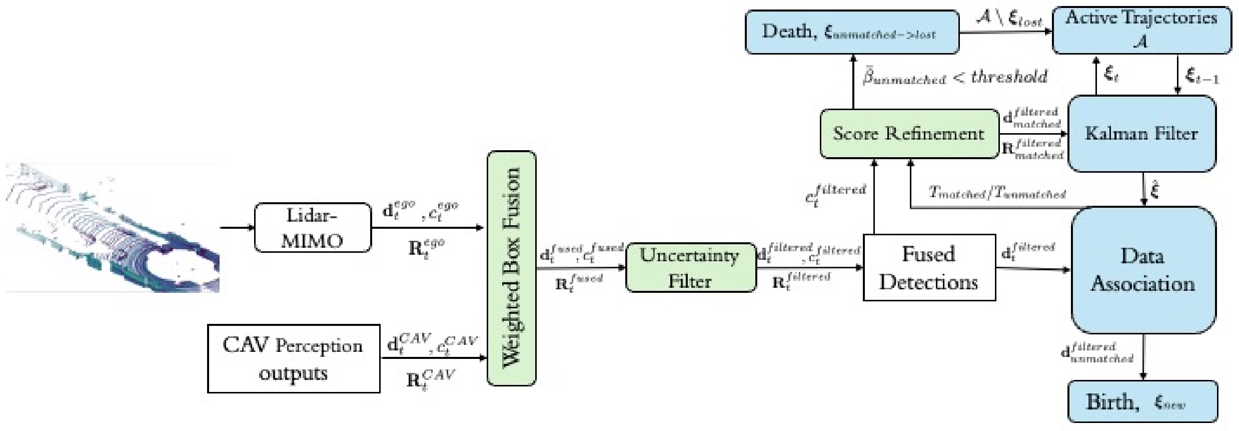

Death, $\xi_{unmatched->lost}$
$\mathcal{A} \setminus \xi_{lost}$
Active Trajectories $\mathcal{A}$
$\tilde{\beta}_{unmatched} < threshold$
ξ_t
ξ_{t-1}
Score Refinement
$\mathbf{d}^{filtered}_{matched}$
$\mathbf{R}^{filtered}_{matched}$
Kalman Filter
$c_t^{filtered}$
$T_{matched}/T_{unmatched}$
$\hat{\xi}$
Lidar-MIMO
$\mathbf{d}_t^{ego}, c_t^{ego}$
$\mathbf{R}_t^{ego}$
Weighted Box Fusion
$\mathbf{d}_t^{fused}, c_t^{fused}$
$\mathbf{R}_t^{fused}$
Uncertainty Filter
$\mathbf{d}_t^{filtered}, c_t^{filtered}$
$\mathbf{R}_t^{filtered}$
Fused Detections
$\mathbf{d}_t^{filtered}$
Data Association
CAV Perception outputs
$\mathbf{d}_t^{CAV}, c_t^{CAV}$
$\mathbf{R}_t^{CAV}$
$\mathbf{d}^{filtered}_{unmatched}$
Birth, ξ_{new}

Figure 1.1: Cooperative Perception and Tracking Pipeline

associated with each bounding-box. The bounding boxes are defined by the component vector $[x, y, z, h, w, l, \theta]$, where x, y, z denote the bounding box centroid coordinates, h, w, l denote the height, width, and length and θ denotes the yaw angle of the bounding box.

The pipeline receives information from nearby connected vehicles and then performs variance-weighted box fusion and uncertainty-based detection filtering. The fused and filtered detections are passed to the object tracker, which uses a score-based life-cycle management method.

1.3.1 Object Detection

We employ PointPillar [8], shown in Figure 1.2, as the base model for our LiDAR-MIMO architecture. PointPillars employs a Pillar Feature Network to efficiently process point cloud data. The input point cloud is first discretized into a sparse set of vertical pillars. Each pillar is encoded using a PointNet-based network, and the resulting features are scattered back to their corresponding pillar locations to construct a pseudo-BEV. This BEV feature map is then processed by a convolutional backbone to extract high-level features, which are fed into detection heads to predict 3D bounding boxes. The entire pipeline is highly efficient for GPU computation, as all major operations, including feature encoding and backbone processing, are implemented as 2D convolutions.

We selected two anchor heads, which is the maximum capacity of PointPillar as currently implemented [7].

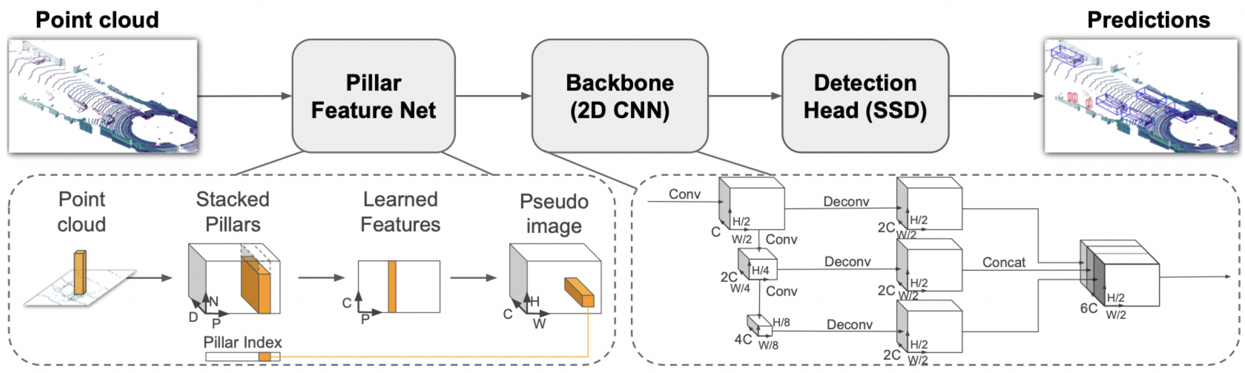


Figure 1.2: Point Pillar Architecture

1.3.2 Uncertainty Estimation

As proposed in LiDAR-MIMO [6], we stack the generated intermediate BEV maps of multiple point clouds during the training stage, rather than stacking raw point clouds. These BEV maps serve as the multiple inputs. Depending on the number of multiple outputs we want to predict, we use multiple anchor heads, each with independent classification and regression convolution layers. Additionally, we included an extra convolutional layer in each anchor head to predict uncertainties.

In deep neural networks (DNNs), two types of uncertainties typically arise: Epistemic Uncertainty (EU) and Aleatoric Uncertainty (AU). EU represents the model's uncertainty, which occurs due to limited training data. AU captures data uncertainty, caused by factors such as measurement noise or weather conditions, and is further classified into homoscedastic and heteroscedastic AU [22]. Homoscedastic AU has constant output noise for all inputs, while heteroscedastic AU features varying observation noise.

Here, AU is estimated using log-variance, $\lambda_* = \log \sigma_*^2$, to prevent numerical stability issues. The log-variance is regressed using the following equation [23] [24]:

$$L_{reg-var} = \frac{1}{2} \exp\left(-\boldsymbol{\lambda}^T\right) (\mathbf{y}_{gt} - \mathbf{y}) \odot (\mathbf{y}_{gt} - \mathbf{y}) + \frac{1}{2} \boldsymbol{\lambda}^T \mathbf{1}, \tag{1.1}$$

where, $\mathbf{y}_{gt}$ and $\mathbf{y}$ represent the ground truth regression targets and regression outputs, namely the bounding boxes, $\odot$ is the Hadamard product between two matrices, and $\boldsymbol{\lambda}$ is the vector of predicted variances corresponding to the regression outputs. The bounding boxes representing each detection are assumed to be drawn from a multivariate Gaussian distribution with uncorrelated coordinates. Accordingly, each coordinate has an associated mean and variance. The variance reflects confidence in the detection. Higher confidence corresponds to lower variances for the bounding box coordinates, indicating greater certainty about the object's location and size. Therefore, the loss function is derived from the Negative Log-Likelihood (NLL) of the multivariate Gaussian distribution. For the uncertainty in yaw, however, we use the NLL of the von Mises probability density function (PDF) [25]

$$\begin{aligned} L_{yaw-var} = \log I_0(\exp(-s_{yaw})) - \exp(-s_{yaw}) \cos(\theta - \theta_t) \\ + \lambda_V ELU(s - s_0) \end{aligned} \tag{1.2}$$

where I_0 is the 0-order Bessel function, $s_{yaw} \approx \log(\sigma_{yaw}^2)$, ELU represents the regularizing activation function, λ_V is the regularization coefficient, s_0 is the offset for ELU, θ is the predicted orientation, and θ_t is the ground truth orientation [25].

1.3.3 Loss Function

We train the model using a multi-loss function. For 3D bounding box regression and classification, we employ smooth-l_1 [26] and focal loss [27], respectively. We denote these losses by $L_{\ell_1\text{-reg}}$ (regression) and $L_{\text{focal-cls}}$ (classification). Additionally, we incorporate the uncertainty estimation loss functions shown in Equation 1.1 and 1.2. The total loss function is formulated as follows:

$$\begin{aligned} L_{total} &= \alpha_{reg} L_{l_1-reg} + \alpha_{cls} L_{focal-cls} \\ &+ \alpha_{reg-var} L_{reg-var} + \alpha_{yaw-var} L_{yaw} \end{aligned} \tag{1.3}$$

where the coefficients α_{reg}, α_{cls}, $\alpha_{\text{reg-var}}$, and $\alpha_{\text{yaw-var}}$ are scalar weights that control the contribution of each term to the total loss.

1.3.4 Late Fusion Scheme

The traditional late fusion scheme utilizes Non-Max Suppression (NMS) to output the final bounding boxes. Detections from multiple CAVs are sorted by their detection confidences, and the detection box with the highest confidence is selected. Co-visible detections from multiple CAVs that have significant overlap with this box are filtered out. However, this approach requires a predetermined threshold to discard redundant boxes, which can cause issues when the highest confidence box does not accurately represent the actual location of the object. Boxes with lower confidence but better representations of the object's location may be discarded if they overlap significantly with the highest confidence box. To address this, we propose using Weighted Box Fusion (WBF) [13] to merge bounding boxes from multiple CAVs. Using Intersection-over-Union (IoU) as the association metric, we form clusters of matched boxes across multiple vehicles, using θ_{wbf} as the matching threshold. Within these clusters, we perform weighted averaging to obtain the final bounding box representing the object.

While the original WBF paper used confidence as the weighting factor, we instead use inverse variance weighting based on the predicted variances. Comparative results between confidence and variance-based weighting are presented in the next section. For example, if we have 'K' matched boxes in the cluster, then the weight corresponding to i^{th} x-coordinate is as follows:

$$w_{\sigma^2_{x_i}} = \frac{\frac{1}{\sigma^2_{x_i}}}{\sum_{j=1}^{K} \frac{1}{\sigma^2_{x_j}}} \tag{1.4}$$

where σ^2_x denotes the variance of the x-position. Throughout the remainder of this chapter, we use $\sigma^2_x, \sigma^2_y, \sigma^2_z, \sigma^2_h, \sigma^2_w, \sigma^2_l, \sigma^2_\theta$ to denote the variances corresponding to x, y, z, h, w, l, θ, respectively.

While coordinate-wise inverse-variance averaging works for positions and physical dimensions, it cannot be applied naively to orientation. Due to the front-back longitudinal symmetries present in many vehicles, LiDAR-only perception often exhibits 0/180°$ flips in predicted orientation. Two CAVs may observe the same object but assign opposite headings. If both estimates have low variances, a direct average can yield an infeasible orientation (e.g., 90°). To avoid this, we select the orientation of the highest-confidence bounding box as the fused orientation. To determine the overall confidence of the merged box, we compute the overall variance of the bounding box and use inverse variance weighted averaging to obtain the weighted average prediction [12]. For example, if we have K matched boxes in the cluster, the fused confidence score is given as

$$p_k \;:=\; \frac{1}{\sigma^2_{k,x}} + \frac{1}{\sigma^2_{k,y}} + \frac{1}{\sigma^2_{k,z}} + \frac{1}{\sigma^2_{k,h}} + \frac{1}{\sigma^2_{k,w}} + \frac{1}{\sigma^2_{k,l}} + \frac{1}{\sigma^2_{k,\theta}}, \qquad k = 1, \ldots, K, \tag{1.5}$$

$$s_{\text{fused}} = \frac{\sum_{k=1}^{K} p_k \, s_k}{\sum_{k=1}^{K} p_k}. \tag{1.6}$$

1.3.5 Uncertainty Ellipse Filtering

After merging the perception outputs from nearby CAVs, we filter out highly uncertain detections from our detection list. Each bounding box is associated with an uncertainty ellipse, derived from the bounding box's dimensions and fused variances [16]. We then discard highly uncertain detections using the method described in Algorithm 1.

Algorithm 1 Uncertainty-Based Detection Filtering

Input: Detection list $\mathcal{D} = \{(B_i, \Sigma_i)\}_{i=1}^{n}$
where B_i is the bounding box, and Σ_i is the associated uncertainty (covariance matrix).

```
1: for each detection $i \in \mathcal{D}$ do
2:     $E_i \leftarrow \textsc{CreateUncertaintyEllipse}(B_i, \Sigma_i)$      ▷ Generate uncertainty ellipse
3:     $UI \leftarrow \frac{|E_i \setminus B_i|}{|E_i|}$      ▷ Uncertainty Index
4:     if $UI > \theta_{UI}$ then
5:         $\mathcal{D} \leftarrow \mathcal{D} \setminus \{(B_i, \Sigma_i)\}$      ▷ Remove detection
6:     end if
7: end for
```

The basic idea of Algorithm 1 is to geometrically assess uncertainty in detections. We first construct uncertainty ellipses by treating predicted uncertainties as additive quantities as described below (the detailed derivation was not provided in [16]). Following this, we remove uncertain detections by computing the Uncertainty Index $UI := \frac{|E_i \setminus B_i|}{|E_i|}$. This measures the fraction of the ellipse area due to added uncertainty, that is, the part outside the bounding box B_i. If UI exceeds a threshold θ_{UI}, we discard the detection.

Uncertainty Ellipse Construction.

Consider a rectangular oriented bounding box of length l (longitudinal), width w (lateral), and orientation (yaw) θ about the z-axis. Let (x, y) denote the longitudinal and lateral axes, respectively. The total uncertainty in the lateral and longitudinal directions is obtained by adding σ_w^2 and σ_l^2 to the σ_x^2- and σ_y^2, respectively, as in [16]. However, these are vector operations. Using the triangle law for vector addition for the standard deviations in the (x, y) coordinates and (l, w) dimensions, the total lateral and longitudinal variances are:

$$\sigma_{\text{lat}} = \sqrt{\sigma_x^2 + \sigma_w^2}, \qquad \sigma_{\text{lon}} = \sqrt{\sigma_y^2 + \sigma_l^2}. \tag{1.7}$$

To incorporate orientation uncertainty as an additive quantity, define axis-aligned half-extents $a(\theta)$ and $b(\theta)$ as the sums of the absolute projection components along lateral and longitudinal directions:

$$a(\theta) = \tfrac{1}{2}(l\,|\cos\theta| + w\,|\sin\theta|), \qquad b(\theta) = \tfrac{1}{2}(w\,|\cos\theta| + l\,|\sin\theta|). \tag{1.8}$$

We consider small $\theta \in [-\sigma_\theta, \sigma_\theta]$ centered at 0 (with $\sigma_\theta > 0$). As we are only working with additive variance components, without loss of generality, we restrict our analysis to $[0, \sigma_\theta]$:

$$a(\theta) = \tfrac{1}{2}(l \cos\theta + w \sin\theta)\,, \qquad b(\theta) = \tfrac{1}{2}(w \cos\theta + l \sin\theta)\,. \tag{1.9}$$

The worst-case added noise due to orientation uncertainty is obtained by maximizing the additive quantities relative to the nominal half-extents after axis alignment:

$$\Delta_a = \max_{\theta\in[0,\sigma_\theta]} a(\theta) - \frac{l}{2}, \qquad \Delta_b = \max_{\theta\in[0,\sigma_\theta]} b(\theta) - \frac{w}{2}. \tag{1.10}$$

Differentiating $a'(\theta) = \frac{1}{2}(-l \sin\theta + w\cos\theta)$ gives the stationary point $\tan\theta^\star = \frac{w}{l}$. For $\sigma_\theta < \arctan(w/l)$, the maximum occurs at $\theta = \sigma_\theta$ as the tangent function is increasing on the given interval, yielding:

$$\Delta_a = \tfrac{1}{2}(l\cos\sigma_\theta + w\sin\sigma_\theta - l)\,, \qquad \Delta_b = \tfrac{1}{2}(w\cos\sigma_\theta + l\sin\sigma_\theta - w)\,. \tag{1.11}$$

Let $\tau \coloneqq \tan\sigma_\theta$. Using $\sin\sigma_\theta = \dfrac{\tau}{\sqrt{1+\tau^2}}$ and $\cos\sigma_\theta = \dfrac{1}{\sqrt{1+\tau^2}}$, we obtain:

$$\Delta_a = \frac{1}{2}\left(\frac{l + w\tau}{\sqrt{1+\tau^2}} - l\right) = \frac{l}{2}\left(\frac{1+\frac{w}{l}\tau}{\sqrt{1+\tau^2}} - 1\right), \qquad \Delta_b = \frac{1}{2}\left(\frac{w + l\tau}{\sqrt{1+\tau^2}} - w\right) = \frac{w}{2}\left(\frac{1+\frac{l}{w}\tau}{\sqrt{1+\tau^2}} - 1\right). \tag{1.12}$$

After rationalizing and rearranging terms, we obtain:

$$\Delta_a = \frac{l}{2}\left(1 - \frac{w}{\sqrt{w^2 + l^2\tan^2\sigma_\theta}}\sqrt{1+\tan^2\sigma_\theta}\right), \qquad \Delta_b = \frac{w}{l}\,\Delta_a. \tag{1.13}$$

Thus, the ellipse semi-axes are given by

$$L_a = \frac{l}{2} + \sigma_{\text{lon}} + \Delta_a, \qquad L_b = \frac{w}{2} + \sigma_{\text{lat}} + \Delta_b. \tag{1.14}$$

1.3.6 Tracking

Tracking-by-Detection (TBD) is performed using a method based on AB3DMOT, modified to incorporate the merged predicted uncertainties. The entire cooperative perception and tracking pipeline is shown in Figure 1.1. The state of a tracklet $\boldsymbol{\xi}$ at time t is given as follows:

$$\boldsymbol{\xi_t} = \left(x_t^{\boldsymbol{\xi}}, y_t^{\boldsymbol{\xi}}, z_t^{\boldsymbol{\xi}}, \theta_t^{\boldsymbol{\xi}}, h_t^{\boldsymbol{\xi}}, w_t^{\boldsymbol{\xi}}, l_t^{\boldsymbol{\xi}}, dx_t^{\boldsymbol{\xi}}, dy_t^{\boldsymbol{\xi}}, dz_t^{\boldsymbol{\xi}}\right)^\top \tag{1.15}$$

where $(x_t^{\boldsymbol{\xi}}, y_t^{\boldsymbol{\xi}}, z_t^{\boldsymbol{\xi}})$ denotes the centroid of the tracklet, $\theta_t^{\boldsymbol{\xi}}$ is the yaw angle, and $(h_t^{\boldsymbol{\xi}}, w_t^{\boldsymbol{\xi}}, l_t^{\boldsymbol{\xi}})$ are the height, width, and length of the tracklet, respectively. Velocity is given by $(dx_t^{\boldsymbol{\xi}}, dy_t^{\boldsymbol{\xi}}, dz_t^{\boldsymbol{\xi}})$.

We use the constant velocity model as our motion model. The Kalman Filter Prediction step is as follows:

$$\hat{\boldsymbol{\xi}}_t = \mathbf{F}\boldsymbol{\xi}_{t-1} \tag{1.16}$$

$$\mathbf{\hat{P}_t} = \mathbf{F}\mathbf{P_{t-1}}\mathbf{F}^T + \mathbf{Q} \tag{1.17}$$

where $\hat{\boldsymbol{\xi}}_t \in \mathbb{R}^{10}$ is the predicted state of the tracklet at time t and $\boldsymbol{\xi}_{t-1} \in \mathbb{R}^{10}$ is the estimated state of the tracklet at time $t-1$. $\mathbf{F} \in \mathbb{R}^{10\times10}$ is the state transition matrix for the constant velocity model. $\hat{\mathbf{P}}_\mathbf{t} \in \mathbb{R}^{10\times10}$ is the predicted state covariance at time t, and $\mathbf{P_{t-1}} \in \mathbb{R}^{10\times10}$ is the estimated state covariance at time $t-1$. The matrix $\mathbf{Q} \in \mathbb{R}^{10\times10}$ is the process model noise covariance. Each detection is represented by $d_t = \left(x_t^d, y_t^d, z_t^d, \theta_t^d, l_t^d, w_t^d, h_t^d\right)$ and corresponding variances $\mathbf{R}_t = \text{diag}\left(\sigma^2_{x_t^d}, \sigma^2_{y_t^d}, \sigma^2_{z_t^d}, \sigma^2_{\theta_t^d}, \sigma^2_{l_t^d}, \sigma^2_{w_t^d}, \sigma^2_{h_t^d}\right)$.

We use the Hungarian Algorithm for data association [28] with Generalized IoU (GIoU) as the similarity metric. A detection is matched with a motion-model–propagated track if their GIoU exceeds the threshold θ_{gIoU3D}. For each matched pair, we perform the Kalman Filter update step as follows:

$$\mathbf{S_t} = \mathbf{H}\hat{\mathbf{P}}_\mathbf{t}\mathbf{H}^T + \mathbf{R}_t \tag{1.18}$$

$$\mathbf{K}_t = \hat{\mathbf{P}}_\mathbf{t}\mathbf{H}^T\mathbf{S}_\mathbf{t}^{-1} \tag{1.19}$$

$$\boldsymbol{\xi}_t = \hat{\boldsymbol{\xi}}_t + \mathbf{K}_t(d_t - \mathbf{H}\hat{\boldsymbol{\xi}}_t) \tag{1.20}$$

$$\mathbf{P_t} = (\mathbf{I} - \mathbf{K}_t\mathbf{H})\hat{\mathbf{P}}_\mathbf{t} \tag{1.21}$$

where $\mathbf{S_t} \in \mathbb{R}^{7\times7}$ is the innovation matrix, $\mathbf{H} = \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix} \in \mathbb{R}^{7\times10}$ is the observation matrix, and $\mathbf{R}_t \in \mathbb{R}^{7\times7}$ is the observation noise covariance matrix for the detection. We use the predicted variances from the fusion scheme as the observed noise.

During object tracking, objects might leave the scene or new objects might enter the scene, thereby requiring a life-cycle management scheme that manages the initialization and removal of tracks, where a track is defined as the ongoing representation of an object in the AVs world model. Age-based schemes birth new tracks when an object first enters the field of view and remove existing tracks if the tracked object has not been updated after a sufficiently long period of time and the latest detections indicate that the object no longer persists within the scene. This allows the tracker to reduce false positives. A max-age parameter describes the number of frames considered before removing a track from the current object list. Various other life-cycle management approaches exist in the literature, such as the confidence-based schemes presented in [15]. To improve robustness in case of missed detections, we move to a score-based management approach instead of a max-age scheme.

Score Refinement

During the Kalman filter prediction step, the confidence of all tracks is decayed by a factor of γ, similar to [29]. This is done to avoid overconfident tracklets. When a detection is matched to an active track, the confidence in the track is increased using the update function [15]:

$$\beta_t = 1 - (1 - \beta_{t-1})(1 - c_t)$$

where β_t is the trajectory confidence at time t, and c_t is the detection confidence.

We consider the running mean of tracklet confidence to determine whether to retain or remove the tracklet from the trajectory list. If the running mean, after score refinement, falls below a threshold η_{traj}, the tracklet is removed from the tracklist, the list of active trajectories [29].

1.4 Experimental Results

In this section, we evaluate our late fusion scheme using inverse variance weighting. We demonstrate that our late fusion scheme, combined with uncertain detection filtering, significantly outperforms state-of-the-art cooperative perception algorithms on the V2V4Real benchmark.

1.4.1 Datasets

We train our model and evaluate cooperative perception and tracking using the V2V4Real dataset [2]. The V2V4Real dataset is a real-world collection of data gathered by two connected vehicles, each equipped with two cameras, one LiDAR, a GPS receiver, and an IMU. The data was collected in Columbus, Ohio. The distance between the two vehicles maintained at 150m to ensure overlap in their fields of view. The dataset is divided into training, validation, and testing sets, containing 14,210, 2,000, and 3,986 frames, respectively.

We selected two anchor heads, which is the maximum capacity of PointPillar as currently implemented [7]. The model was trained for 140 epochs with a batch size of 3, a learning rate of 0.002, and a learning rate decay using cosine annealing.

1.4.2 Evaluation Metrics

To maintain consistency, we adopt the same evaluation metrics used in V2V4Real [2]. We report the object detection results using Average Precision (AP) as the evaluation metric. The model is evaluated under two conditions: 1) the synchronous setting, where data transmission between CAVs is assumed to be instantaneous, and 2) the asynchronous setting, where communication delays are introduced, simulating a delay by using the point clouds of non-ego vehicles from the previous timestamp.

For object tracking, in addition to the CLEAR metrics [30], the tracking algorithm is assessed using two key metrics: Average Multi-Object Tracking Accuracy (AMOTA) and Average Multi-Object Tracking Precision (AMOTP) [14] [2]. These metrics take into account confidence scores, False Positives (FP), False Negatives (FN), True Positives (TP), and identity switches across various recall levels. Other metrics, such as scaled-AMOTA (sAMOTA), Multi-Object Tracking Accuracy (MOTA), Mostly Tracked Trajectories (MT), and Mostly Lost Trajectories (ML), are also evaluated. A default threshold of 0.25 IoU, as established in [14], is used for evaluation.

1.4.3 Implementation Details

During model training, we adopt the same hyperparameters for $\alpha_{reg} = 2.0$ and $\alpha_{cls} = 1.0$ as specified in [8]. The default $\lambda_V = 1.0$ from [25] is used for ELU regularization. We set $\alpha_{reg-var} = \alpha_{yaw-var} = 0.025$. In the post-processing stage, the threshold for our weighted box fusion scheme is set to $\theta_{wbf} = 0.15$, and for detection filtering, a threshold of $\theta_{UI} = 0.4$ is applied. Finally, during the object tracking phase, we employ a similarity threshold of $\theta_{gIoU3D} = -0.2$ for data association. Additionally, we set $\gamma = 0.5$ for score refinement and $\eta_{traj} = 0.1$ for life cycle management, respectively.

1.4.4 Quantitative Results - Detection Performance

The performance of our proposed WBF with uncertain detection filtering scheme is shown in Table 1.1. All values are given as percentages. AP@IoU = 0.5/0.7 indicates the IoU thresholds used for evaluation. Our late fusion approach demonstrates a substantial improvement over the standard late fusion method, achieving a 27.4% increase in AP@0.5 and a 48.6% increase in AP@0.7. Furthermore, our method outperforms other state-of-the-art intermediate fusion cooperative perception algorithms. To identify the components contributing to these performance gains, we also report results for WBF alone and WBF in conjunction with the uncertain detection filter. The WBF-based approach shows the greatest improvement compared to the standard NMS, while the best detection performance is achieved when both WBF and the uncertain detection filter are employed together. However, from Table 1.1, we also observe that AP@0.7 at 0–30 m in the no-fusion case outperforms both NMS-based and WBF-based post-processing methods. Similar results are reported in [2]. A likely reason is co-visible observations: when objects are close to the vehicle, it can localize and detect them well unless there are occlusions. When cooperative perception is introduced and nearby connected vehicles share perception information, co-visible objects may have their localization influenced by other vehicles' perceptions, which can either improve or degrade overall detection accuracy.

We note in Table 1.1 that results for asynchronous perception do not vary much across different WBF implementations, since we are still operating on object-level perception outputs. However, the uncertainty-filtered results remain better than the standard WBF method.

Table 1.1: Comparison of Fusion Methods Results: All the baseline methods are from the V2V4Real Paper [2]. All values are percentages. Best results are typeset in bold font

Method	**Sync (AP@IoU=0.5/0.7)**			
	Overall	**0-30m**	**30-50m**	**50-100m**
No Fusion	39.8/22.0	69.2/42.6	29.3/14.4	4.8/1.6
Late Fusion	55.0/26.7	73.5/36.8	43.7/22.2	36.2/15.1
Early Fusion	59.7/32.1	76.1/46.3	42.5/20.8	47.6/22.5
F-Cooper [11]	60.7/31.8	80.8/46.9	45.6/23.6	32.8/17.1
V2VNet [31]	64.5/34.3	80.6/51.4	52.6/26.6	42.6/17.1
AttFuse [21]	64.7/33.6	79.8/44.1	53.1/29.3	43.6/19.9
V2X-ViT [18]	64.9/36.9	82.0/**55.3**	51.7/26.6	43.2/21.1
CoBEVT [19]	66.5/36.0	82.3/51.1	52.1/28.2	49.1/17.5
No Fusion	48.2/32.2	73.6/51.7	35.9/23.4	8.3/4.6
No Fusion - Ellipse Filtered	48.6/32.7	73.8/52.2	36.6/23.9	8.1/4.7
WBF (Using Confidence)	69.0/38.5	81.9/45.8	54.5/32.3	55.9/31.4
WBF	69.8/39.0	**82.4**/46.4	57.1/32.6	57.5/33.4
WBF-Ellipse Filtered	**70.2/39.7**	**82.4**/46.6	**57.2/33.7**	**57.8/34.3**

Method	**Async (AP@IoU=0.5/0.7)**			
	Overall	**0-30m**	**30-50m**	**50-100m**
No Fusion	39.8/22.0	69.2/42.6	29.3/14.4	4.8/1.6
Late Fusion	50.2/22.4	70.7/34.2	41.0/19.8	26.1/7.8
Early Fusion	52.1/25.8	74.6/43.6	34.5/16.3	30.2/9.5
F-Cooper [11]	53.6/26.7	79.0/44.1	38.7/19.5	18.1/6.0
V2VNet [31]	56.4/28.5	78.6/48.0	44.2/21.5	25.6/6.9
AttFuse [21]	57.7/27.5	78.6/41.4	45.5/23.8	27.2/9.0
V2X-ViT [18]	55.9/29.3	79.7/50.4	43.3/21.1	24.9/7.0
CoBEVT [19]	58.6/29.7	**80.3/48.3**	44.7/22.8	30.5/8.7
No Fusion	48.2/32.2	73.6/51.7	35.9/23.4	8.3/4.6
No Fusion - Ellipse Filtered	48.6/32.7	73.8/52.2	36.6/23.9	8.1/4.7
WBF (Using Confidence)	63.0/32.6	78.9/43.2	**52.5**/28.7	38.9/15.7
WBF	63.1/32.4	79.0/42.9	**52.5**/28.6	**39.0**/15.8
WBF-Ellipse Filtered	**63.3/33.9**	79.2/43.1	**52.5/29.3**	38.9/**16.3**

1.4.5 Quantitative Results - Tracking Performance

The performance of our score-based tracking method is presented in Table 1.2. All values are given as percentages. We observe a 41.7% improvement in AMOTA, which can be attributed to both enhanced detection performance and the use of score-based life cycle management. Notably, a significant portion of this improvement 34.3% is due to improved detection performance, even when using the same max-age life cycle management as outlined in [2].

Table 1.2: Comparison of Cooperative Tracking Methods Results: All the baseline methods are from the V2V4Real Paper [2]. All values are percentages. Best results are typeset in bold font.

Method	AMOTA ↑	AMOTP ↑	sAMOTA ↑	MOTA ↑	MT ↑	ML ↓
No Fusion	16.08	41.60	53.84	43.46	29.41	60.18
Late Fusion	29.28	51.08	71.05	59.89	45.25	31.22
Early Fusion	26.19	48.15	67.34	60.87	40.95	32.13
F-Cooper [11]	23.29	43.11	65.63	58.34	35.75	38.91
AttFuse [21]	28.64	50.48	73.21	63.03	46.38	28.05
V2VNet [31]	30.48	54.28	75.53	64.85	48.19	27.83
V2X-ViT [18]	30.85	54.32	74.01	64.82	45.93	26.47
CoBEVT [19]	32.12	55.61	77.65	63.75	47.29	30.32
Standard (Max-Age parameter)	39.32	57.15	86.58	86.63	64.99	13.19
Score-Based	**41.49**	**58.70**	**89.34**	**87.80**	**66.43**	**12.71**

1.4.6 Qualitative Results

We present qualitative results from a highway scenario in the dataset. Figure 1.3 shows bounding-box visualizations using Open3D [32]. The top image shows the no-fusion case, whereas the bottom image shows late fusion using our variance-weighted post-processing scheme. Note the numbered vehicle labels in each image. We observe that in the no-fusion scenario for CAV-1 there are no predictions for non-connected vehicles 1, 3, 4, 5, 6, and 7 due to occlusions and objects lying outside CAV-1's LiDAR range. In the fusion scenario, while vehicle 1 remains undetected, CAV-1 perceives vehicles 3, 4, and 7 using CAV-2's perception information. Vehicles 5 and 6 are still not perceived because both are occluded from CAV-2's perspective. We also note a slight degradation in the localization of vehicle 2, likely because it is observed by both CAV-1 and CAV-2 and suboptimal variance-weighting of observations from different viewpoints shifts the bounding box. This effect stems from the trained detector rather than the post-processing scheme. Future work could include optimized weighting strategies that assign priority to vehicles based on co-visible observations.

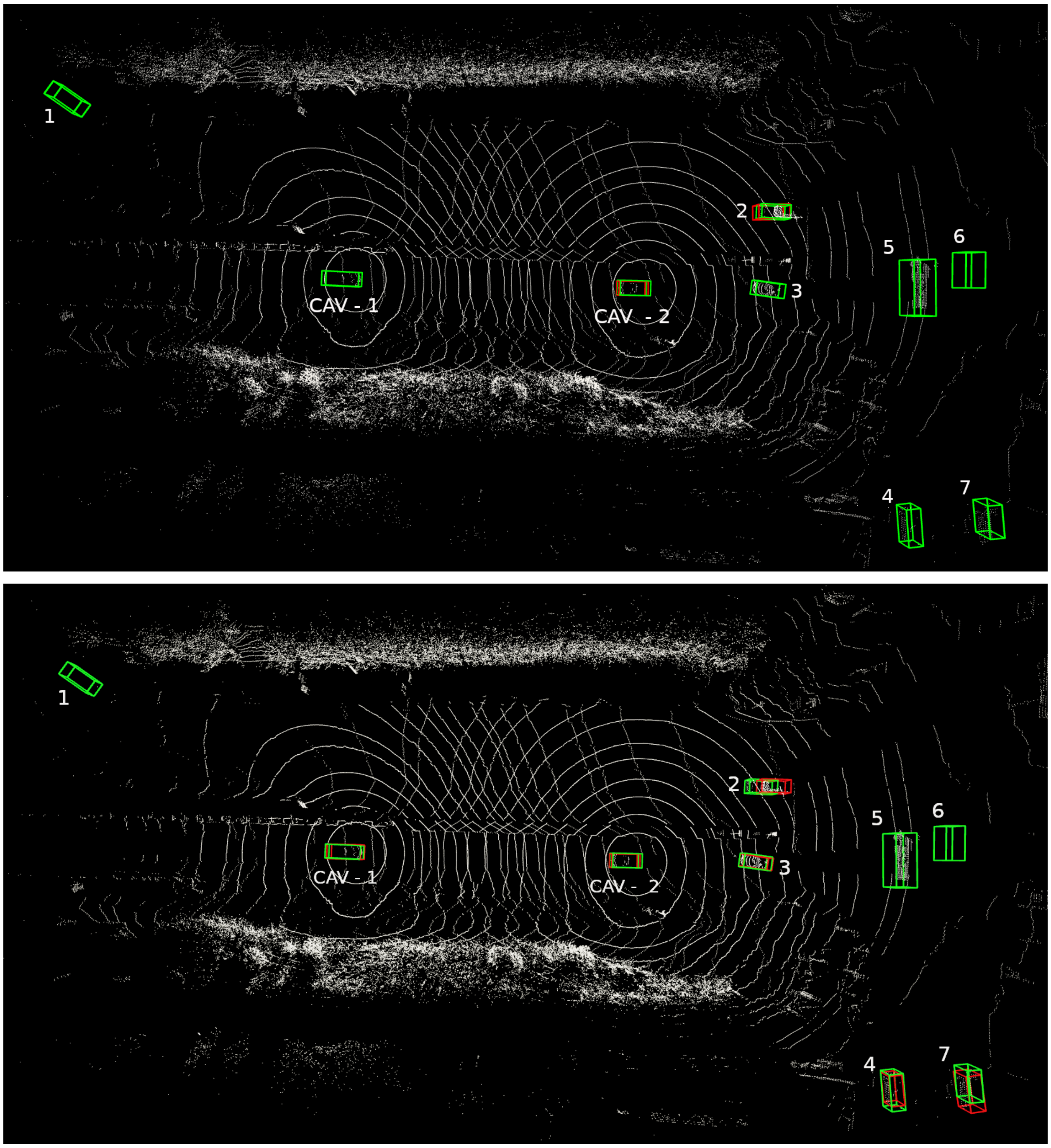


Figure 1.3: Qualitative results of cooperative 3D object detection. Green and red 3D bounding boxes represent the ground truth and predictions, respectively. The top image shows results with no fusion, while the bottom image shows results with fusion.

1.5 Uncertainty Calibration

In safety-critical applications such as automated driving, object detectors must predict not only the locations and sizes of obstacles but also quantify their associated uncertainties. Downstream control tasks, such as behavior planning, depend on these uncertainty estimates to execute efficient and non-conservative maneuvers. However, if the uncertainty of a detector is miscalibrated, such as underestimating the true error with high confidence, the control algorithm may fail to assign the appropriate priorities to these observations, potentially resulting in unsafe behavior. Küppers et al. observed that modern object detectors frequently overestimate spatial uncertainty compared to the observed errors [33]. To address this, they introduced several post-hoc uncertainty calibration methods for probabilistic regression, which are briefly summarized in Table 1.3.

Table 1.3: Summary of Uncertainty Calibration Methods

Method	**Applicability**	**Cross-Dim.**	**Output Distribution**
Isotonic Regression	Univariate, nonparametric	No	Arbitrary (per-dim)
Variance Scaling	Univariate, Gaussian	No	Gaussian
GP-Beta	Univariate, Beta family	No	Beta
GP-Normal	Uni/multivariate, Gaussian process	Yes	Gaussian
GP-Cauchy	Univariate, Cauchy	No	Cauchy (per-dim)

Uncertainty calibration is often characterized by several complementary metrics, each capturing a different facet of calibration quality. Negative Log-Likelihood (NLL) quantifies the similarity between the predicted probability distribution and the observed outcomes, providing a global measure of the degree to which the predicted uncertainties match the actual distribution of errors. NLL is appropriate when the goal is to assess the overall quality of probabilistic predictions, regardless of the uncertainty parameterization, and is particularly relevant for comparing models with different underlying distributional assumptions.

In contrast, Uncertainty Calibration Error (UCE) and Expected Normalized Calibration Error (ENCE) specifically quantify the quality of variance (or standard deviation) predictions in regression settings. UCE measures the absolute difference between the predicted variance and the observed mean squared error within bins of similar predicted variance, making it a direct indicator of how well the model's uncertainty matches the empirical error.

$$\mathrm{UCE} = \sum_{m=1}^{M} \frac{N_m}{N} \left| \mathrm{MSE}(m) - \mathrm{MV}(m) \right| \tag{1.22}$$

where $\mathrm{MSE}(m)$ and $\mathrm{MV}(m)$ are the mean squared error and the mean variance in bin m, respectively, and N_m is the number of samples in bin m.

ENCE normalizes this difference by the root mean variance in each bin, expressing the miscalibration as a relative error and making it robust to changes in scale across output spaces.

$$\mathrm{ENCE} = \frac{1}{M} \sum_{m=1}^{M} \frac{\left| \mathrm{RMSE}(m) - \mathrm{RMV}(m) \right|}{\mathrm{RMV}(m)} \tag{1.23}$$

where $\mathrm{RMSE}(m)$ and $\mathrm{RMV}(m)$ are the root mean squared error and the root mean variance within bin m, respectively.

In summary, NLL is best used to evaluate the global fit of the predicted distribution (including tail behavior and overall uncertainty shape), while UCE and ENCE are more suited for assessing whether predicted variances are statistically consistent with empirical errors.

However, there is a fundamental issue when applying traditional uncertainty calibration techniques to orientation estimation. Conventional post-hoc calibration methods, such as isotonic regression and GP-Normal, are fundamentally limited when applied to orientation (θ) calibration in 3D object detection. Orientation errors are inherently circular and often exhibit a bimodal distribution: prediction errors cluster around 0 and $\pm\pi$, representing a 180° flip in vehicle orientation, as many models easily confuse the front and back of a vehicle, generating vehicle orientations of both θ and $\theta + \pi$. Isotonic regression is built for monotonic, scalar-valued targets and assumes a strictly increasing empirical CDF, making it incompatible with angular data where errors wrap around at $\pm\pi$ and the empirical error CDF contains discontinuities. This method cannot represent the two-mode structure of orientation errors, leading to smoothing across modes and miscalibrated uncertainty estimates. GP-Normal calibration, which learns a mapping from predicted outputs to a Gaussian distribution, imposes a unimodal, symmetric error model that cannot represent the mixture of near-zero and near-π errors. When fit to bimodal data, it compensates by inflating the predicted variance to cover both peaks, resulting in overly wide confidence intervals and high UCE, ENCE, and NLL values for θ, even after calibration. This failure to accurately represent uncertainty in orientation propagates to downstream tasks, where either overconfident or underconfident predictions can degrade behavior planning and safety in autonomous systems.

To address the limitations of traditional uncertainty calibration for orientation, Kato & Kato propose a *bimodal uncertainty model* for orientation errors in 3D detection [34]. Rather than modeling the orientation error $\Delta\alpha = \theta_{\text{pred}} - \theta_{\text{true}}$ with a single unimodal distribution, they use a mixture of two Von Mises distributions: one centered at 0 and one at π. The calibrated error density is:

$$p_{\text{ori}}(\Delta\alpha \mid 0, \kappa, \lambda) = \lambda\, p_{\text{Mises}}(\Delta\alpha \mid \mu = 0, \kappa) + (1 - \lambda)\, p_{\text{Mises}}(\Delta\alpha \mid \mu = \pi, \kappa), \tag{1.24}$$

where $p_{\text{Mises}}(\cdot \mid \mu, \kappa)$ is the Von Mises ("circular normal") distribution with mean μ and concentration κ. The mixing weight λ represents the probability that the prediction is close to the true orientation. This two-component model explicitly captures both correct and flipped predictions, effectively handling bimodal and periodic error structures. Unlike unimodal models, this approach resolves the ambiguity of 180° flips and is crucial for a proper calibration of the orientation uncertainty.

1.5.1 Uncertainty Calibration Results

We reserved a small subset of the validation set from the dataset for calibration and split it into two parts, namely *calibration-train* and *calibration-test*. On *calibration-train*, we ran a no-fusion inference to collect predicted bounding boxes and their uncertainty parameters, matched each prediction to a ground-truth box using a high IoU threshold (IoU $\geq$ 0.5), and discarded unmatched detections. We then trained the uncertainty calibrator on the resulting matched pairs. We used the NetCal–framework for GP-Normal training [35], while bimodal orientation calibration was implemented with our own code. Finally, in *calibration-test*, we repeated the inference and matching and validated the calibrator. The results below present the calibrator's performance on the V2V4Real test set.

As shown in Table 1.4, the uncertainty metrics improve across most dimensions when comparing Calibrated to Uncalibrated results. The position and physical size dimensions (x, y, z, h, w, l) were calibrated using the GP–Normal method, while the orientation θ was calibrated using the bimodal theta approach. For ENCE and UCE, Calibrated mostly outperforms Uncalibrated in every dimension. NLL also generally improves under Calibrated (e.g., x: 1.88$\rightarrow$0.37, l: 1.42$\rightarrow$0.76). As expected, θ has a large NLL value despite calibration, indicating that orientation is still the hardest

Table 1.4: Calibration Metrics per Bounding Box Parameter (Calibrated vs. Uncalibrated)

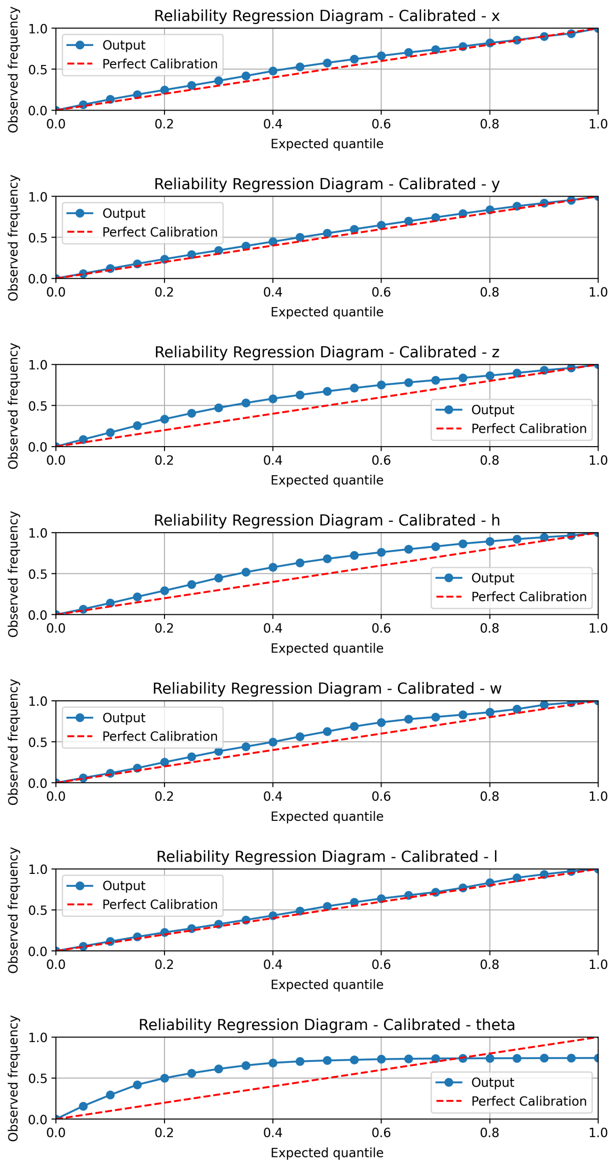
Metric	Mode	x	y	z	h	w	l	θ
ENCE	Cal	0.3370	0.3421	0.5838	0.6005	0.4013	0.2583	5.7187
	Uncal	0.4710	0.3452	0.5630	0.5502	0.4134	0.5292	12.3077
UCE	Cal	0.03590	0.01084	0.05622	0.02245	0.03840	0.12989	2.54775
	Uncal	0.08601	0.01524	0.06090	0.00373	0.03317	0.18633	2.59190
NLL	Cal	0.36804	−0.35605	0.15131	−0.24416	−0.02955	0.76232	25.47431
	Uncal	1.88299	−0.13210	1.75568	−0.17219	0.55146	1.41978	334.95805

component to model and calibrate accurately. The large NLL values are due to the limitations of using *LiDAR-only* detections for orientation, which rely on the learned values.

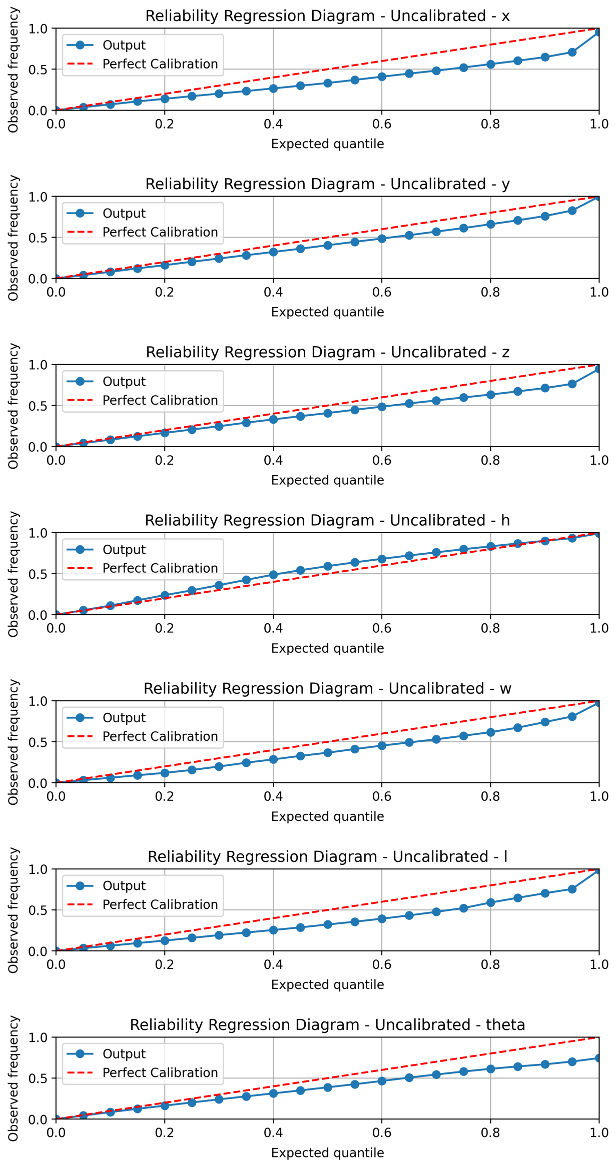
As shown in Figure 1.4, the Calibrated reliability curves mostly lie closer to perfect-calibration than the Uncalibrated curves, indicating improved uncertainty calibration. This trend is consistent with the quantitative results reported in Table 1.4. However, while there is a significant reduction in UCE, ENCE, and NLL for θ, the calibrated uncertainties appear to be underconfident. This arises from orientation errors as not all headings are corrected before calibration, so a small number of outliers skew the scaling even though their count has decreased with orientation prediction and correction. We will explore improved calibration methods that explicitly handle orientation correction prior to calibration in future work.

1.6 Conclusions

We propose an inverse-variance weighted box fusion scheme for cooperative perception. The fused variances from the merged boxes are used to construct uncertainty ellipses, which are then employed to filter out detections with high uncertainty. These fused detections and variances are integrated into our score-based object tracking algorithm. We also performed uncertainty calibration and reported the corresponding results. For future work, we plan to explore the calibration of predicted uncertainties and their application in cooperative perception and tracking.



(a) Calibrated



(b) Uncalibrated

Figure 1.4: Calibrated vs Uncalibrated Reliability Regression Diagrams for Bounding Box Parameters

Chapter 2

Misbehavior Detection for Collaborative Perception

Intervehicle communications, such as Collective Perception Messages (CPMs), provide enhanced information about a vehicle's surroundings, supplementing the data provided by Basic Safety Messages (BSMs) and thereby boosting environmental awareness for connected and automated vehicles (CAVs) particularly in scenarios involving occluded vehicles. However, accepting perception data from nearby connected vehicles, especially when it cannot be locally verified, can pose significant risks due to the potential consequences of erroneous or malicious information in the network. This work focuses on the challenge of data trust in collaborative perception by proposing a misbehavior detection scheme that utilizes the overlapping fields of views (FoVs) of multiple vehicles. These shared observations allow for verifying the contextual validity of vehicle-to-vehicle (V2V) data, thereby enhancing the overall trustworthiness of V2V systems. The proposed detection scheme is tested using the OpenCDA simulation framework, built on CARLA, in a T-junction scenario, where an attacker injects spoofed vehicles targeting the ego vehicle amidst other CAVs. Simulation results demonstrate the effectiveness of the scheme in identifying misbehavior and mitigating its impact on the ego vehicle's decision-making. Additionally, we evaluate an erroneous-pose attack in which the attacker transmits incorrect or corrupted pose information to the ego vehicle. As pose is required to transform remote detections into the ego vehicle frame, this experiment tests the robustness of the geometric-consistency framework under position and navigation errors.

2.1 Introduction

The perception module of an automated vehicle (AV) is critical for safety, as its performance directly influences driving behavior. However, AV perception is inherently constrained by the limited capabilities of onboard sensors such as LiDARs and cameras. LiDAR and camera sensors cannot see through occlusions and provide low-resolution data for distant objects, and their views of the world are often restricted by their mounting position and FoV, making it difficult to capture the entire driving environment. In addition, cameras are sensitive to lighting conditions and lack accurate depth perception, further contributing to imperfect detection. LiDAR-based collaborative perception algorithms have recently gained traction, enabling nearby connected vehicles to exchange perception data, such as raw sensor inputs, internal feature maps, or processed bounding boxes, which are then fused onboard to perform object detection [10,21]. This approach has demonstrated significant improvements in detection accuracy compared to traditional CAV methods that rely solely on sharing BSMs.

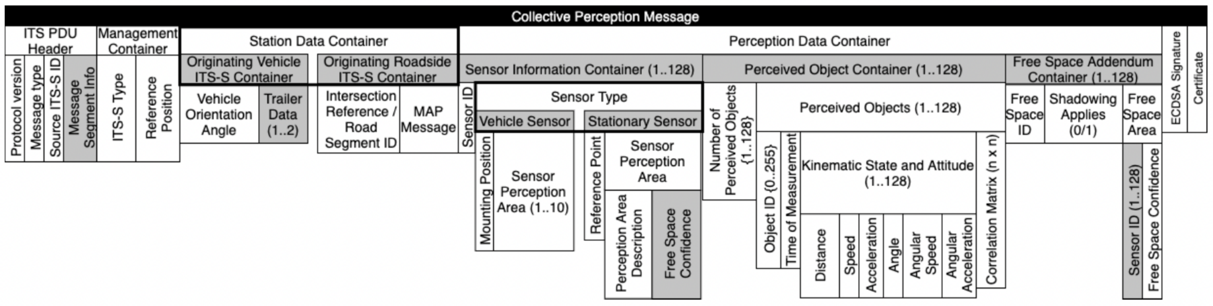


Figure 2.1: ETSI Cooperative Perception Message Structure [1]

However, cooperative perception introduces significant vulnerabilities to vehicle safety, as the perception module begins to rely on information from nearby vehicles that may be compromised or malicious. An attacker can inject spoofed data—such as fake objects or altered perception results—into the V2V communication network, misleading the perception systems of surrounding vehicles. This false information can result in incorrect decision-making by victim vehicles, potentially causing accidents, unsafe maneuvers, or disruptions to overall traffic flow.

In this work, we focus on late-fusion-based attack models, where an attacker can easily generate spoofed vehicles or alter the reported locations of objects. We adopt object-level collaboration due to its bandwidth efficiency and ease of deployment compared to early fusion, which demands high bandwidth, and intermediate fusion, which requires knowledge of each participating vehicle's sensing and sensor processing model architecture within the network. To mitigate such attacks, we propose a free-space consistency analysis that identifies inconsistencies in the shared information from nearby connected vehicles. In this approach, each vehicle is required to transmit its perceived free-space area along with detected objects and standard BSMs, per the Collective Perception Service specifications defined in the ETSI TS-103-324 Technical Report [36]. Figure 2.1 shows the CPM format prescribed by ETSI. As indicated, in addition to communicating perceived-object information in the *Perceived Object Container*, CAVs must also send the *Free Space Area* (Free Space Addendum Container) [1]. Only the grey fields are optional but white fields are mandatory.

We assign dynamic trust scores to each communicating node (i.e., connected vehicle), updating these scores based on the consistency of their shared information relative to that of other vehicles in the network [37]. Data fusion is performed only after evaluating these trust scores to ensure robustness. We evaluate the proposed misbehavior detection algorithm in a simulated T-junction scenario, featuring an attacker who injects a spoofed vehicle, two benign connected vehicles, and an ego vehicle. The results demonstrate that the algorithm effectively identifies misbehaving vehicles and excludes them from the fusion process, thus mitigating their impact on the AV's decision-making. The effectiveness of attack mitigation is evaluated using the trust score, in addition to standard classification metrics (e.g., true positives, false positives, precision, and recall) over the duration of the entire scenario. We also evaluate an erroneous pose attack in the same T-junction setup, where the attacker transmits incorrect or corrupted pose instead of spoofed detections. Because pose is used to transform local detections into the ego vehicle frame, large pose errors cause footprint misplacement and misalignment with real objects. We test the geometric-consistency framework under varying translation and yaw perturbations and quantify its ability to identify pose errors.

2.2 Related Works

2.2.1 CAVs and Cooperative Perception

Research in connected vehicles spans several decades, driven by the promise of improved safety, roadway efficiency, and reduced fuel consumption. Early influential work under California's PATH program included the development of sophisticated systems such as Adaptive Cruise Control (ACC) and Cooperative Adaptive Cruise Control (CACC), particularly tailored for heavy-duty trucks (HDTs). Similarly, initiatives like the Grand Cooperative Driving Challenge (GCDC), held in the Netherlands in 2011 and 2016, accelerated research in cooperative driving by emphasizing cooperative vehicle-following strategies to improve roadway throughput and efficiency. Furthermore, SAE International established essential standards, including SAE J2735 and SAE J2945, providing foundational guidelines for Dedicated Short-Range Communication (DSRC) and Cellular V2X (C-V2X), which underpin vehicle-to-vehicle (V2V) interactions critical to cooperative driving systems.

Modern AVs are often equipped with cameras and LiDAR sensors to enable enhanced scene understanding. With advancements in neural networks that support 3D object detection using multi-sensor data fusion, robust automated driving has become increasingly feasible, allowing AVs to accurately perceive their surroundings and navigate safely [8, 38–40]. However, AVs still face limitations in their FoV due to sensor range constraints and environmental occlusions. These limitations can be addressed by enabling vehicles to share perceptual information, in addition to BSMs, with nearby connected vehicles to achieve a more comprehensive and holistic view of the environment [9, 41].

There has been significant progress in cooperative perception, supported by the release of several open-source datasets such as OPV2V, V2V4Real, and V2X-Sim, which span both simulated and real-world settings to facilitate benchmarking and evaluation [2, 21, 42]. However, ongoing research is focused on overcoming key challenges such as communication bandwidth constraints, latency and localization errors, ensuring privacy-preserving data exchange, and safeguarding against potential cyberattacks [43–46].

2.2.2 Attacks on CAVs

CAVs face diverse cyberattack threats, broadly categorized into in-vehicle network, vehicle-to-everything (V2X) network, and other attack types [47, 48]. In-vehicle attacks include remote sensor manipulation, GPS spoofing, ECU software flashing, and impersonation, all of which can disrupt vehicle operations, compromise privacy, or hijack vehicle control [49–51]. V2X attacks, such as message replay, various manipulations of network routing, and data falsification, exploit communication vulnerabilities, generate misleading safety responses, or compromise confidentiality through eavesdropping or key cracking [52]. Other threats include infrastructure disruptions, slight deviations that cause hazardous conditions, and targeted attacks on machine learning systems integral to AV technologies [53]. These attacks collectively represent significant cyber risks, highlighting cybersecurity's complexity and dynamic nature in CAV ecosystems. For our study, we will examine threats involving V2V communication, a subset of V2X networks. Even authenticated CAVs within vehicular networks are vulnerable to spoofed or bogus information disseminated by certified vehicles. Several prominent attack types include [54]:

1. **Jamming Attack:** A Denial of Service (DoS) attack that floods the transmission channel with dummy messages, preventing legitimate communication by occupying the medium.

2. **False Location Information:** Involves sending out altered or incorrect BSMs to mislead other vehicles, often resulting in the appearance of ghost vehicles.

3. **Bogus Object Attack:** The attacker transmits false CPMs, falsely indicating the presence of an object, which can manipulate vehicle behavior or gain unauthorized access to infrastructure.

4. **False Alert Information:** Disseminating abnormal traffic-related messages such as fake congestion, emergency braking, or accident alerts to disrupt normal vehicle operations and induce unnecessary maneuvers.

5. **Message Forwarding Attack:** The attacker drops or delays critical messages before forwarding them, disrupting reliable and timely safety communication.

Our specific focus will be on the **Bogus Object Attack**, where vehicles maliciously broadcast false CPMs to influence network behavior and safety-critical vehicle decisions.

2.2.3 Mitigation Strategies

Ambrosin et al. proposed a layered Misbehavior Detection System (MDS) for V2X applications that verifies shared perception data before fusion with local perception and before forwarding the information to control and planning, while allowing algorithm-agnostic integration of diverse detection techniques [55]. The MDS categorizes four types of anomalies:

1. **Message-level Anomaly:** Identified through direct inspection of message content, such as out-of-range data or duplicate reports.

2. **Model-level Anomaly:** Detected when the reported object state significantly deviates from predictions based on a motion model. This requires temporal tracking of each object.

3. **System-level Anomaly:** Inferred from inconsistent spatial relationships among objects reported by different senders, such as distinct objects occupying overlapping positions.

4. **Perception-level Anomaly:** Arises when shared object data conflict with the ego vehicle's local perception of the environment.

This multi-tiered verification enhances the trustworthiness and robustness of cooperative perception in connected autonomous systems. Building on the generalized misbehavior detection framework, which identifies inconsistencies at the message, model, system, and perception levels, Liu et al. proposed MISO-V, a practical implementation of these principles using occupancy grid consistency [37]. MISO-V verifies the CPMs by evaluating the spatial alignment of the reported objects with the fused occupancy grid and ego vehicle's FoV. Based on this comparison, the sender's trust score is dynamically updated: consistent data increases trust, while spatial inconsistencies trigger penalization. This mechanism supports real-time filtering of unreliable information, ensuring that only trustworthy data influences perception fusion and trajectory planning. Tsukada, Manabu, et al. proposed a Kalman filter-based approach that improves the performance of BSM tampering detectors by utilizing observation data shared through CPM [56]. However, the approach does not consider the possibility of falsification within the CPM data itself. Ali, M. Shabbir and Pierre Merdrignac proposed a misbehavior detection algorithm (LCP-MBDA) to identify false stop information attacks in V2X communications [57]. These attacks involve the dissemination of deceptive data about sudden vehicle stops via CPMs, potentially disrupting traffic flow and compromising road safety through unnecessary hard braking, stopping, or obstacle dodge maneuvers. Their approach uses a vehicle perception model to compare locally detected objects with previously reported vehicles in CPMs. If a reported vehicle cannot be confirmed through local perception, the sender is flagged for misbehavior.

2.3 Method

In our study, we propose a misbehavior detection algorithm that integrates key principles from the generalized MDS architecture, MISO-V, and LCP-MBDA, employing geometric consistency-based verification. The generalized MDS architecture offers an algorithm-agnostic and modular foundation for diverse detection approaches but does not specify particular consistency-checking mechanisms. LCP-MBDA focuses exclusively on false stop scenarios, using simplistic proximity-based queries to validate stationary vehicle positions from CPMs without clearly detailing the map querying method employed in the algorithm. MISO-V, most closely related to our proposed algorithm, utilizes occupancy-grid fusion, marking cells occupied or free based on unanimous vehicle agreement, with conflicting cell classifications indicate potential misbehavior. However, this occupancy-grid approach may generate increased false positives due to inherent discretization errors, localization inaccuracies, and bounding box approximation issues, and lacks explicit methods for querying bounding boxes in the presence of inconsistent grid information. To overcome these shortcomings, our proposed method leverages polygonal representations, allowing arbitrary shape modeling, eliminating discretization errors, and enabling sophisticated geometric operations for precise consistency checking [58].

2.3.1 Proposed Scheme

This section describes the geometric consistency-based misbehavior detection algorithm designed to detect persistent attacks in cooperative perception systems. The consistency verification is performed in two stages. The first stage evaluates pairwise consistency between all communicating CAVs. The second stage performs consistency checks with respect to the ego vehicle's FoV. The ego FoV check is performed after the CAV-level consistency check because the ego vehicle's perception is assumed to be fully trustworthy.

2.3.2 System Model

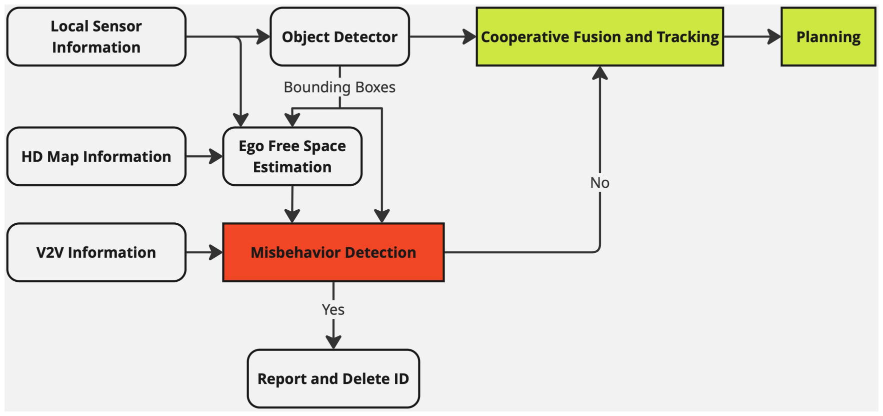


Figure 2.2: Misbehavior Detection System Overview

As shown in Figure 2.2, the block diagram presents a cooperative perception and decision-making architecture that integrates LiDAR-based object detection with V2V-enabled information sharing. Local LiDAR data is processed using a PointPillars-based object detector to generate 3D bounding boxes and their associated uncertainty estimates and free space around the ego vehicle is estimated using the LiDAR point cloud and HD map information through ground segmentation and removal of occupied regions [8]. Perception information, including bounding boxes with associated uncertainty and free space estimates, is received from surrounding connected vehicles via V2V communication.

The misbehavior detection module performs consistency checks across all shared information by evaluating whether object detections reported by one vehicle contradict the free space regions published by another. In addition, consistency is checked against the ego vehicle's own perception data, which is always assumed to be trustworthy. Contradictions may occur, for example, if an object reported by a neighboring vehicle lies within the ego vehicle's perceived free space or if multiple vehicles publish conflicting spatial information. Occlusion-aware reasoning and geometric checks are used to detect such inconsistencies. Vehicles identified as misbehaving are flagged and permanently excluded from subsequent cooperative fusion and tracking stages.

Valid data is fused using the variance-weighted box fusion method, as detailed in Chapter 1. The resulting object estimates are tracked using an Extended Kalman Filter (EKF) based on a Constant Turn Rate and Velocity (CTRV) motion model [59, 60]. Short-term motion prediction is performed using the same CTRV model, which will also be utilized in the collision avoidance and trajectory planning module described in the next chapter.

Assumptions

For our analysis, we assume that the information received from all nearby connected vehicles is time-synchronized. This allows for consistent temporal alignment during cooperative perception and fusion. In future work, it will be necessary to implement time-forward propagation of detected objects and perceived free space to account for communication delays and processing latency.

2.3.3 Free Space Estimation

The free space estimation algorithm processes LiDAR data to identify drivable regions around the ego vehicle by first applying RANSAC to eliminate ground points and then utilizing DBSCAN to cluster non-ground points into obstacles [61, 62]. While our perception algorithm focuses on identifying vehicles, DBSCAN captures all object clusters, whether or not they are vehicles, that could cast LiDAR shadows and obstruct free space. After filtering out ground and object points, valid LiDAR points are radially projected to define the free space boundaries. Because our primary interest lies in spoofed vehicles that could disrupt the ego vehicle's behavior, we restrict our analysis to the drivable region by applying an in-lane mask that excludes objects and areas outside lane boundaries, ensuring the algorithm concentrates on potential threats within the ego vehicle's operational space. Finally, the resulting free space region is represented as a polygon using the Shapely library [63], enabling geometric operations that are crucial for consistency-based checks (see Algorithm 2).

2.3.4 CAV Free Space Consistency

The CAV free space consistency procedure flow chart shown in Figure 2.3 outlines a structured approach for evaluating the consistency of free space information reported by CAVs. Initially, the algorithm performs pairwise intersection checks of the reported free spaces $F_i \cap F_j$ for all vehicles $i \neq j$. If an intersection is identified, it calculates the convex hull and the corresponding bounding box to localize conflicting regions. Vehicles whose reported bounding boxes intersect these conflicting regions have their trust scores decreased. Conversely, if no intersection between free spaces is found, the algorithm further evaluates consistency by checking if bounding boxes reported by one vehicle intersect the free space of another vehicle, indicating potential misbehavior.

Algorithm 2 Free Space Estimation

```
1: procedure FreeSpaceEstimation(lidarData)
2:     pcd ← VoxelDownsample(lidarData)
3:     Filtering Step:
4:         Remove points outside LiDAR Range
5:         Remove points within the ego-vehicle bounding box
6:     inLaneMask ← GenerateLaneMask(pcd, HDMap)
7:     planeModel, groundIndices ← RANSAC(pcd)
8:     pointHeights ← ComputeValidHeights(pcd, planeModel)
9:     objectClusters ← DBSCAN(pcd, non-ground points)    ▷ Separate object clusters using
       information from groundIndices
10:    objectClusters ← ValidObjects(pcd, objectClusters, size, shape, range)    ▷ Filter clusters
       based size, shape and range
11:    objectMask ← BuildObjectMask(pcd, objectClusters)
12:    Compute Free Space:
13:        Exclude pcd points outside lane boundaries using inLaneMask
14:        Exclude pcd points labeled as objects using objectMask
15:        Compute boundary points along radial directions from the sensor
16:        Mark the closest valid point per direction as a free space boundary
17:    freeArea ← BuildPolygon(free space boundary points)
18:    return freeArea
19: end procedure
```

Again, trust scores for inconsistent vehicles are penalized. Vehicles whose trust scores fall below a predetermined threshold are immediately removed; otherwise, they are subjected to an additional *ego FoV* consistency check for final verification.

2.3.5 Ego FoV Consistency

The ego FoV consistency procedure flowchart depicted in Figure 2.4 details a structured verification procedure for bounding boxes reported by CAVs relative to the ego vehicle's perception. This step is applied after the CAV Free Space Consistency check and serves as a secondary validation layer. Initially, the intersection over area (IoA) between each CAV's bounding box and the ego vehicle's free space (F_{ego}) is computed. If the IoA exceeds a predefined threshold τ, indicating substantial overlap with the ego's FoV, the algorithm evaluates spatial consistency by comparing the center coordinates of the overlapping bounding boxes. Specifically, it computes the norm distance between each CAV-reported bounding box and all ego-detected bounding boxes. If a corresponding ego detection is found within a distance threshold $d_{\text{threshold}}$, the report is deemed consistent—allowing for minor localization inaccuracies—and the trust score of the CAV is increased. Otherwise, the CAV is marked inconsistent and its trust score is penalized. Vehicles with trust scores falling below the removal threshold are excluded from further participation, while consistent vehicles are retained for downstream cooperative fusion. Bounding boxes that either lack sufficient IoA or remain spatially consistent result in trust score increases.

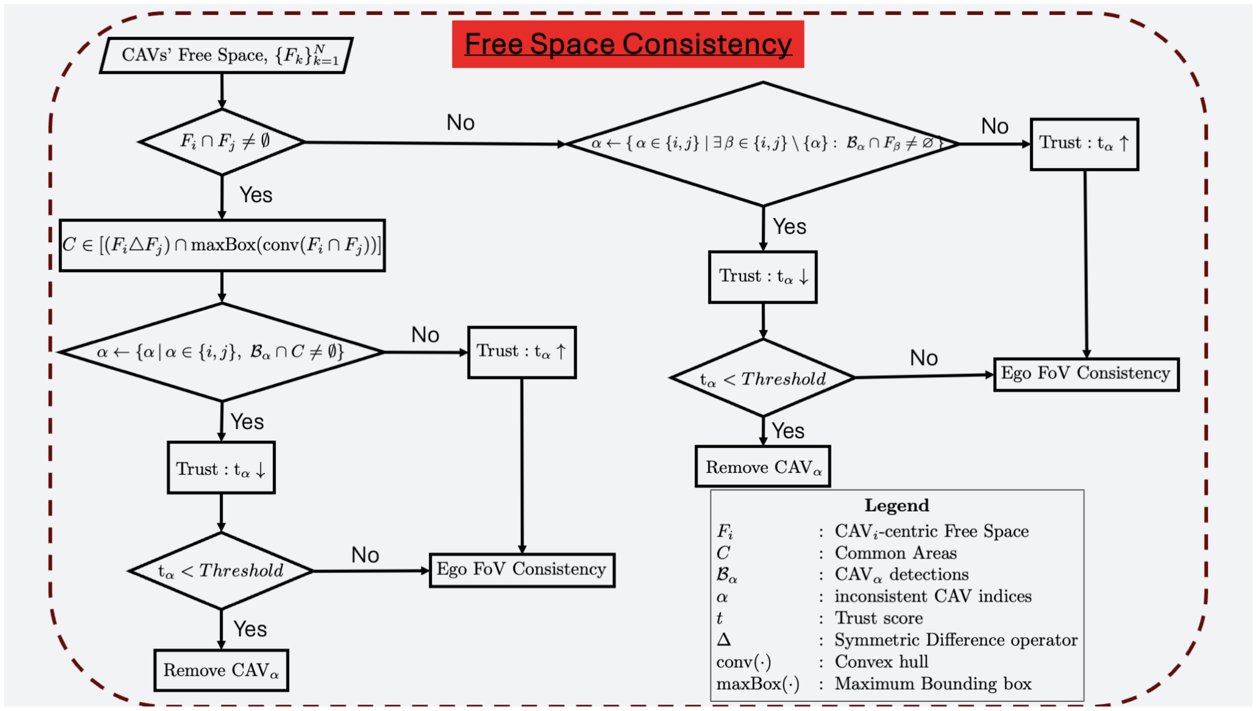

Free Space Consistency
CAVs' Free Space, $\{F_k\}_{k=1}^N$
$F_i \cap F_j \neq \emptyset$
No
Yes
$C \in [(F_i \triangle F_j) \cap \text{maxBox}(\text{conv}(F_i \cap F_j))]$
$\alpha \leftarrow \{\alpha \mid \alpha \in \{i,j\},\ \mathcal{B}_\alpha \cap C \neq \emptyset\}$
Trust : $t_\alpha \uparrow$
Trust : $t_\alpha \downarrow$
$t_\alpha < Threshold$
Ego FoV Consistency
Remove CAV$_\alpha$
$\alpha \leftarrow \{\alpha \in \{i,j\} \mid \exists \beta \in \{i,j\} \setminus \{\alpha\} : \mathcal{B}_\alpha \cap F_\beta \neq \emptyset\}$
Legend
F_i : CAV$_i$-centric Free Space
C : Common Areas
$\mathcal{B}_\alpha$: CAV$_\alpha$ detections
α : inconsistent CAV indices
t : Trust score
Δ : Symmetric Difference operator
conv(·) : Convex hull
maxBox(·) : Maximum Bounding box

Figure 2.3: Approach for Free Space Consistency Check: Left branch handles overlapping free space whereas the right branch handles disjoint free space.

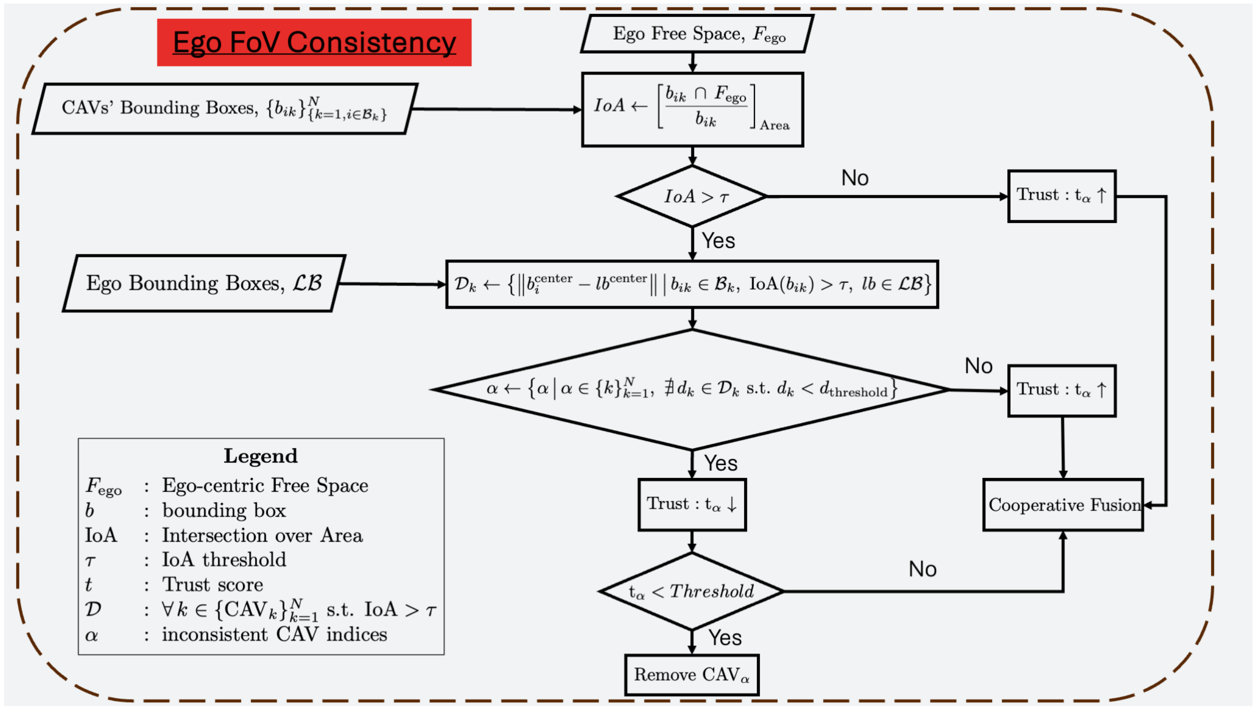

Ego FoV Consistency
Ego Free Space, F_{ego}
CAVs' Bounding Boxes, $\{b_{ik}\}_{\{k=1, i \in \mathcal{B}_k\}}^N$
$IoA \leftarrow \left[\frac{b_{ik} \cap F_{\text{ego}}}{b_{ik}}\right]_{\text{Area}}$
$IoA > \tau$
No
Yes
Trust : $t_\alpha \uparrow$
Ego Bounding Boxes, $\mathcal{LB}$
$\mathcal{D}_k \leftarrow \{\|b_i^{\text{center}} - lb^{\text{center}}\| \mid b_{ik} \in \mathcal{B}_k,\ \text{IoA}(b_{ik}) > \tau,\ lb \in \mathcal{LB}\}$
$\alpha \leftarrow \{\alpha \mid \alpha \in \{k\}_{k=1}^N,\ \nexists d_k \in \mathcal{D}_k \text{ s.t. } d_k < d_{\text{threshold}}\}$
Trust : $t_\alpha \downarrow$
Cooperative Fusion
$t_\alpha < Threshold$
Remove CAV$_\alpha$
Legend
F_{ego} : Ego-centric Free Space
b : bounding box
IoA : Intersection over Area
τ : IoA threshold
t : Trust score
$\mathcal{D}$: $\forall k \in \{\text{CAV}_k\}_{k=1}^N$ s.t. IoA $> \tau$
α : inconsistent CAV indices

Figure 2.4: Approach for Ego Field of View Consistency Check

2.4 Implementation

The overall detection process is outlined in Algorithm 3. The trust score of each CAV is updated on the basis of whether it is inconsistent. If a CAV is inconsistent, its current trust score is penalized by multiplying it with a decay factor $p < 1$. If consistent, the score is increased by scaling it with $\frac{1}{p}$, capped at a maximum of 0.9. An exponential moving average with weight α is used to smooth the update, balancing past and current behavior [37].

Algorithm 3 Misbehavior Detection

```
1: procedure DETECTION(EgoInformation, CAVInformation)
2:     for each CAV_Content ∈ CAVInformation do
3:         CAV_Content.CAV_ID.inconsistent ← false
4:     end for
5:     $\mathcal{B}$ ← EgoInformation.BoundingBoxes
6:     $\mathcal{F}_{\text{ego}}$ ← EgoInformation.FreeSpace
7:     FREESPACECONSISTENCY(CAVInformation)
8:     CAVInformation ← REMOVELOWTRUSTSCORES(CAVInformation,
               trustThreshold)
9:     EGOFOVCONSISTENCY($\mathcal{F}_{\text{ego}}$, $\mathcal{B}$,
               CAVInformation, overlapThreshold, distanceThreshold)
10:    updatedCAVInformation ← REMOVELOWTRUSTSCORES(CAVInformation,
               trustThreshold)
11:    return updatedCAVInformation                ▷ Filtered data for perception fusion
12: end procedure
```

All CAVs are initialized with a trust score of 0.7. The ego vehicle is always assigned a trust score of 1.0 and is excluded from trust updates. The trust score update procedure is described in Algorithm 4.

Algorithm 4 Update Trust Score

```
1: procedure UPDATETRUSTSCORE(CAV_Content)
2:     $t$ ← CAV_Content.CAV_ID.trustScore
3:     if CAV_Content.CAV_ID.inconsistent then
4:         newTrust ← $p \cdot t$
5:         CAV_Content.CAV_ID.trustScore ← $\alpha \cdot t + (1 - \alpha)\cdot$ newTrust
6:     else
7:         newTrust ← $\min\left(\frac{1}{p} \cdot t,\ 0.9\right)\Bigg($
8:         CAV_Content.CAV_ID.trustScore ← $\alpha \cdot t + (1 - \alpha)\cdot$ newTrust
9:     end if
10: end procedure
```

2.4.1 CAVs Free Space Consistency

The Free Space Consistency check, as illustrated in Figure 2.3, evaluates the mutual consistency of free space information shared among CAVs. The overall structure and integration of these steps are detailed in Algorithm 5. It begins by performing pairwise comparisons of the free space regions

F_i, F_j reported by each pair of connected vehicles. If there is no overlap between these free spaces, that is $F_i \cap F_j = \emptyset$, the procedure HandleDisjointFreeSpace shown in Algorithm 6 is invoked. This procedure examines bounding boxes reported by each vehicle against the other vehicle's free space to detect spatial inconsistencies.

If the free spaces do overlap, that is $F_i \cap F_j \neq \emptyset$, the procedure HandleOverlappingFreeSpace shown in Algorithm 7 is employed. It computes the convex hull of the intersection and creates a bounding box $\text{maxBox}(\text{conv}(F_i \cap F_j))$ to constrain the analysis to local regions of agreement. The intersection of this bounding box with the symmetric difference $F_i \triangle F_j$ isolates conflicting areas where free space estimates differ, removing distant and irrelevant geometries. Object bounding boxes from each vehicle are then checked against these conflicting regions to determine inconsistencies. Vehicles identified as inconsistent have their trust scores decreased, and vehicles whose trust scores fall below a specified threshold are subsequently removed.

Algorithm 5 Free Space Consistency Check

```
 1: procedure FreeSpaceConsistency(CAVInformation)
 2:     $\mathcal{E} \leftarrow$ CAVInformation.Boxes
 3:     for each pair ($\text{CAV\_Content}_i$, $\text{CAV\_Content}_j$) in CAVInformation do
 4:         $F_i, F_j \leftarrow \text{CAV\_Content}_i$.FreeSpace, $\text{CAV\_Content}_j$.FreeSpace
 5:         $\mathcal{B}_i, \mathcal{B}_j \leftarrow \text{CAV\_Content}_i$.BoundingBoxes, $\text{CAV\_Content}_j$.BoundingBoxes
 6:         $I \leftarrow F_i \cap F_j$
 7:         if $I = \emptyset$ then
 8:             HandleDisjointFreeSpace($\mathcal{B}_\alpha, F_\alpha, \mathcal{E}, I, \text{CAV\_Content}_\alpha$)      ▷ $\alpha \in \{i, j\}$
 9:         else
10:             HandleOverlappingFreeSpace($\mathcal{B}_\alpha, F_\alpha, \mathcal{E}, I, \text{CAV\_Content}_\alpha$)   ▷ $\alpha \in \{i, j\}$
11:         end if
12:     end for
13: end procedure
```

After completing these pairwise consistency checks, the algorithm proceeds with ego FoV consistency verification. Only vehicles passing this subsequent verification step have their trust scores increased.

Algorithm 6 Handle Disjoint Free Space ($I = \emptyset$)

```
Require: $\mathcal{B}_\alpha$: Bounding Boxes (Detections) of $\text{CAV}_\alpha$
Require: $F_\alpha$: $\text{CAV}_\alpha$-centric free space
Require: $\mathcal{E}$: footprints of all CAVs
Require: $I$: intersection free-space area of $\text{CAV}_i \cap \text{CAV}_j$
Require: $\text{CAV\_Content}_\alpha$: CPM contents of $\text{CAV}_\alpha$
Require: $\tilde{\mathcal{B}}_\alpha$: potentially false-positive detections
Require: $\mathcal{R}_\alpha$: suspected-spoofed/false-positive detections
 1: procedure HandleDisjointFreeSpace($\mathcal{B}_\alpha, F_\alpha, \mathcal{E}, I, \text{CAV\_Content}_\alpha$)    ▷ $\alpha \in \{i, j\}$
 2:     $\mathcal{L}_\alpha \leftarrow \{b \in \mathcal{B}_\alpha \mid b \cap F_{\bar{\alpha}} \neq \emptyset\}$,    $\text{Flag}_\alpha \leftarrow (\mathcal{L}_\alpha \neq \emptyset)$,    $\forall \alpha \in \{i, j\}$
 3:     if $\text{Flag}_i$ and not $\text{Flag}_j$ then
 4:         $\text{CAV\_Content}_i$.CAV_ID.inconsistent ← true
 5:         Update trust scores of $\text{CAV\_Content}_i$
 6:     else if $\text{Flag}_j$ and not $\text{Flag}_i$ then
 7:         $\text{CAV\_Content}_j$.CAV_ID.inconsistent ← true
 8:         Update trust scores of $\text{CAV\_Content}_j$
 9:     else if $\text{Flag}_i$ and $\text{Flag}_j$ then
10:         $\text{MatchedPairs} \leftarrow \{(b_i, b_j) \in \mathcal{L}_i \times \mathcal{L}_j \mid \text{IoU}(b_i, b_j) \geq \delta\}$
11:         $\text{Unmatched}_\alpha \leftarrow \mathcal{B}_\alpha \setminus \{b_\alpha \mid \exists(\cdot, b_\alpha) \in \text{MatchedPairs}\}$,    $\forall \alpha \in \{i, j\}$
12:         $\tilde{\mathcal{B}}_\alpha \leftarrow \{b \in \text{Unmatched}_\alpha \mid \forall \ell \in \mathcal{E},\ b \cap \ell = \emptyset\}$,    $\forall \alpha \in \{i, j\}$
13:         $\mathcal{R}_\alpha \leftarrow \{b \in \tilde{\mathcal{B}}_\alpha \mid \text{IoA}(b, F_{\bar{\alpha}}) \geq \tau\}$,    $\forall \alpha \in \{i, j\}$
14:         if $\mathcal{R}_\alpha \neq \emptyset$ then
15:             $\text{CAV\_Content}_\alpha$.CAV_ID.inconsistent ← true
16:             Update trust scores of $\text{CAV\_Content}_\alpha$
17:         end if
18:     else
19:         continue
20:     end if
21: end procedure
```

Algorithm 7 Handle Overlapping Free Space ($I \neq \emptyset$)

```
Require: $\mathcal{B}_\alpha$: Bounding Boxes (Detections) of CAV$_\alpha$
Require: $F_\alpha$: CAV$_\alpha$-centric free space
Require: $\mathcal{E}$: footprints of all CAVs
Require: $I$: intersection free-space area of CAV$_i \cap$ CAV$_j$
Require: CAV_Content$_\alpha$: CPM contents of CAV$_\alpha$
Require: $C$: common area
Require: $\tilde{\mathcal{B}}_\alpha$: potentially false-positive detections
Require: $\mathcal{R}_\alpha$: suspected-spoofed/false-positive detections
 1: procedure HandleOverlappingFreeSpace($\mathcal{B}_\alpha, F_\alpha, \mathcal{E}, I$, CAV_Content$_\alpha$)        ▷ $\alpha \in \{i, j\}$
 2:     CommonAreas ← $\{C \mid C \in [(F_i \triangle F_j) \cap \text{maxBox}(\text{conv}(I))]\,,\ \text{Area}(C) \geq \varepsilon\}$
 3:     for each $C \in$ CommonAreas do
 4:         $\mathcal{L}_\alpha \leftarrow \{b \in \mathcal{B}_\alpha \mid b \cap C \neq \emptyset\}$,    Flag$_\alpha \leftarrow (\mathcal{L}_\alpha \neq \emptyset)$,    $\forall \alpha \in \{i, j\}$
 5:         if Flag$_i$ and not Flag$_j$ then
 6:             CAV_Content$_i$.CAV_ID.inconsistent ← true
 7:             Update trust scores of CAV_Content$_i$
 8:             break
 9:         else if Flag$_j$ and not Flag$_i$ then
10:             CAV_Content$_j$.CAV_ID.inconsistent ← true
11:             Update trust scores of CAV_Content$_j$
12:             break
13:         else if Flag$_i$ and Flag$_j$ then
14:             MatchedPairs ← $\{(b_i, b_j) \in \mathcal{L}_i \times \mathcal{L}_j \mid \text{IoU}(b_i, b_j) \geq \delta\}$
15:             Unmatched$_\alpha \leftarrow \mathcal{B}_\alpha \setminus \{b_\alpha \mid \exists(\cdot, b_\alpha) \in$ MatchedPairs$\}$,    $\forall \alpha$
16:             $\tilde{\mathcal{B}}_\alpha \leftarrow \{b \in \text{Unmatched}_\alpha \mid \forall \ell \in \mathcal{E},\ b \cap \ell = \emptyset\}$,    $\forall \alpha \in \{i, j\}$
17:             $\mathcal{R}_\alpha \leftarrow \{b \in \tilde{\mathcal{B}}_\alpha \mid \text{IoA}(b, C) \geq \tau\}$,    $\forall \alpha \in \{i, j\}$
18:             if $\mathcal{R}_\alpha \neq \emptyset$ then
19:                 CAV_Content$_\alpha$.CAV_ID.inconsistent ← true
20:                 Update trust scores of CAV_Content$_\alpha$
21:                 break
22:             end if
23:         else
24:             continue
25:         end if
26:     end for
27: end procedure
```

2.4.2 Ego FoV Consistency

As shown in Figure 2.4, the Ego FoV Consistency module verifies whether object detections shared by CAVs are spatially consistent with the ego vehicle's free space and its own detections. For each bounding box b_i reported by a CAV, the intersection over area (IoA) with respect to the ego's free space F_{ego} is computed. If the IoA exceeds a threshold τ (overlapThreshold), indicating significant overlap with the ego's free space, the detection is flagged for further consistency checking.

To determine alignment with ego vehicle detections, the algorithm compares the bounding box centers of the CAV and ego detections. If a corresponding ego vehicle detection is found within a specified distance threshold, the detection is considered consistent, accounting for minor localization errors. If no such match is found, the CAV is marked inconsistent and its trust score is penalized. CAVs with trust scores falling below the threshold are removed, while consistent ones are passed on for cooperative fusion. The complete procedure is outlined in Algorithm 8.

Algorithm 8 Ego FoV Consistency Check

```
Require: $\mathcal{F}_{\text{ego}}$: Ego Free Space
Require: $\mathcal{LB}$: Local (Ego) Bounding Boxes (Detections)
 1: procedure EgoFoVConsistency($\mathcal{F}_{\text{ego}}$, $\mathcal{LB}$,
    CAVInformation, overlapThreshold, distanceThreshold)
 2:     for each CAV_Content ∈ CAVInformation do
 3:         $\mathcal{B} \leftarrow$ CAV_Content.BoundingBoxes
 4:         for each $b_i \in \mathcal{B}$ do
                                                    ▷ $\mathcal{F}_{\text{ego}}$: Ego Free Space
 5:             $\text{IoA} \leftarrow \left[\frac{b_i \cap \mathcal{F}_{\text{ego}}}{b_i}\right]\Big(_{\text{Area}}$
 6:             if IoA > overlapThreshold then
                                                    ▷ $\mathcal{LB}$: Local Bounding Boxes
 7:                 if $\nexists\, lb \in \mathcal{LB}$ s.t. $\|b_i.\text{center} - lb.\text{center}\| <$ distanceThreshold then
 8:                     CAV_Content.CAV_ID.inconsistent ← true
 9:                     UpdateTrustScore(CAV_Content)
10:                     break
11:                 else
12:                     continue
13:                 end if
14:             end if
15:         end for
16:     end for
17: end procedure
```

2.5 Experimental Results

2.5.1 Datasets

For the object detection task, we adopted the logic and architecture from Chapter 1, which established a late fusion scheme for detection and tracking and compared it with existing baselines. Our aim was to integrate that module into the pipeline for misbehavior detection and control. To support this application, we retrained the model on a dataset tailored to our scenarios of interest. We utilize a subset of the OPV2V dataset, focusing specifically on data collected from CARLA's Town03 and Town05 environments. A total of 1,609 frames were used for training and 467 frames for validation. These data samples were selected from the OPV2V dataset to match our scenario-specific requirements. Additionally, we collected custom data using the OpenCDA framework in Town03 and Town05. This dataset includes 2,785 frames for training, 365 frames for validation, and 420 frames for testing. In the collected dataset, we ensure that the number of CAVs per frame ranges between 2 and 7, consistent with the configuration used in the OPV2V dataset. In total, we use 4,394 frames for trainig, 832 frames for validation, and 420 frames testing.

2.5.2 Simulation Settings

To simulate cooperative perception scenarios, we utilize the OpenCDA framework, which integrates cooperative driving pipelines with standard automated driving system (ADS) components [64]. OpenCDA combines three core modules: automated driving simulation using CARLA, traffic simulation using SUMO, and a co-simulation engine that synchronizes both environments [65, 66]. The framework builds upon standard ADS platforms and supports various classes of data exchange between vehicles, infrastructure, and other road users. We integrated the OpenCOOD model library for cooperative perception [67]. For multi-object tracking, we modify AB3DMOT and replace the constant-velocity/linear-KF motion model with a Constant-Turn-Rate-Velocity (CTRV) model and an Extended Kalman Filter (EKF) [14].

2.5.3 CPM Spoof Attack Scenario

Scenario Description

As seen in Figure 2.5, the ego vehicle approaches the intersection intending to make a left turn. It observes one connected vehicle directly ahead within its field of view. A second connected vehicle, not visible to the ego vehicle, is approaching the intersection from the ego's left; its presence and location are communicated via V2V messages. Additionally, other connected vehicles report a non-connected green vehicle entering the intersection from the ego's right, which is also not visible to the ego vehicle. The attacker is stationary and is fully occluded from all vehicles.

All vehicles are equipped with LiDAR sensors having a maximum range of 120 meters, with intensity degrading as distance increases. V2V communication is restricted to a broadcast range of 70 meters to be consistent with the OPV2V dataset configuration.

Attacker Model

As show in Figure 2.6, the attacker broadcasts a spoofed vehicle positioned behind the green vehicle. The transmitted bounding box mimics a plausible dynamic obstacle, as the attacker positions it within the drivable regions of the HD map. To enhance the effectiveness of the attack, the bounding box is placed in areas that are likely occluded from the ego vehicle's FoV, minimizing the chance of inconsistencies in detection. As the ego vehicle approaches the intersection, the

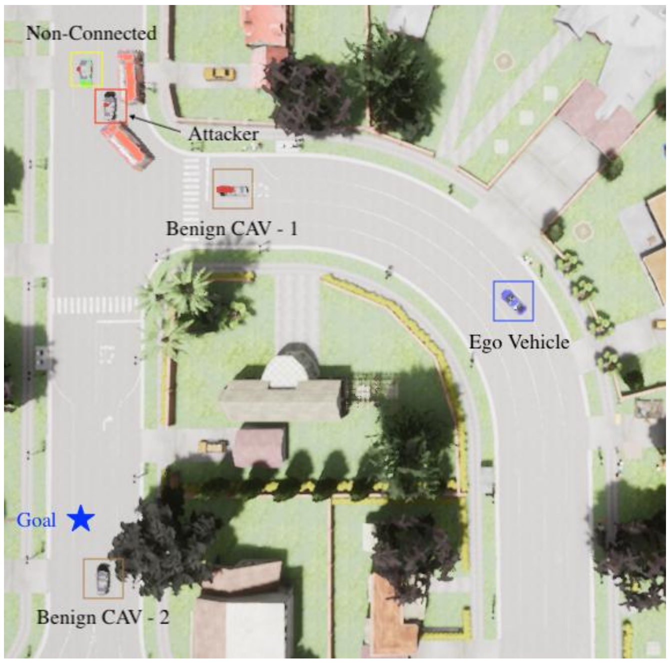


Figure 2.5: Overhead View of Scenario

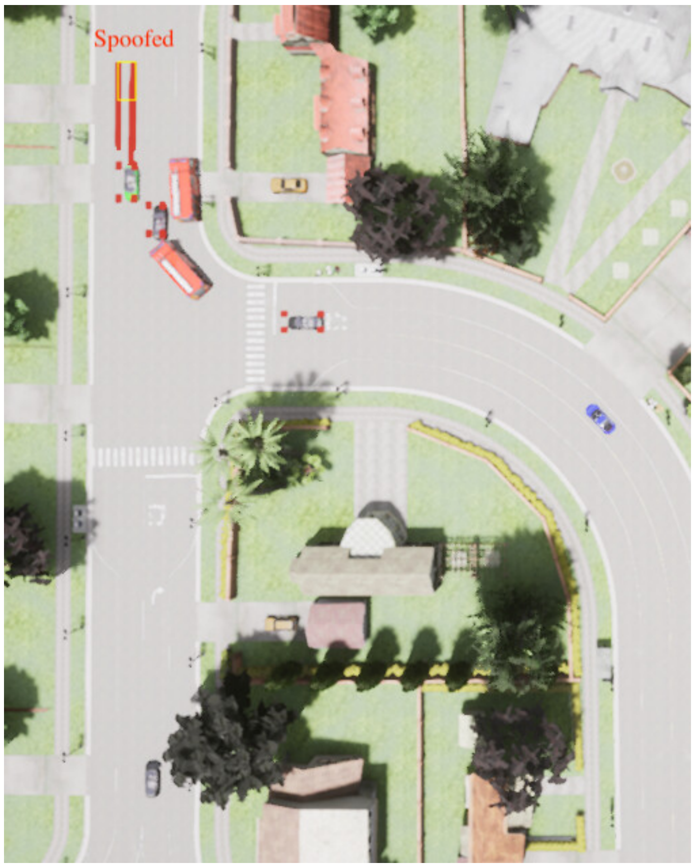


Figure 2.6: Spoofed Vehicle Insertion

attacker transitions the spoofed object's motion model from a Constant Velocity (CV) model to a CTRV model, directing it along a new path that mimics a left-turning vehicle. This maneuver should induces the ego vehicle to brake as it approaches the intersection, despite the absence of any real vehicle. By adapting the spoofed motion to the context and visibility constraints, the attacker ensures a near 100% success rate in deceiving the ego vehicle.

Assumptions

As the attacker must communicate free space information to the ego vehicle, it maintains the consistency between the spoofed vehicle's presence and the perceived free space by updating the free space polygon to exclude the area occupied by the spoofed bounding box. A ray-based approach is used to model occlusion: rays are cast from the attacker's position to the outer corners of the spoofed bounding box and extended outward to define an occlusion region that simulates the shadow cast by the object. This occlusion polygon is also subtracted from the free space. It is assumed that the attack is persistent, with the attacker continuously transmitting the spoofed vehicle's bounding box.

2.5.4 CPM Spoof Attack Quantitative Results

Trust Score Evaluation

The CAVs provide perception information once they enter the ego vehicle's communication range. Figure 2.7 illustrates the inconsistency in the overlapping regions between benign CAV-1 and an attacker, leading to a decrease in the trust score. Since this study focuses exclusively on spoofed vehicles rather than object removal attacks, a consistency check with just one benign vehicle suffices to identify malicious entities within the network. As inconsistencies persist, the trust score continues to decline and the vehicle is classified as potentially misbehaving. Once the trust score falls below a predefined threshold, the vehicle is definitively marked as an attacker and further information from this sender is disregarded. Figure 2.8 shows the trend of the trust score throughout the duration of the scenario, continuing until the ego vehicle reaches its destination. The trust threshold is set to 0.5.

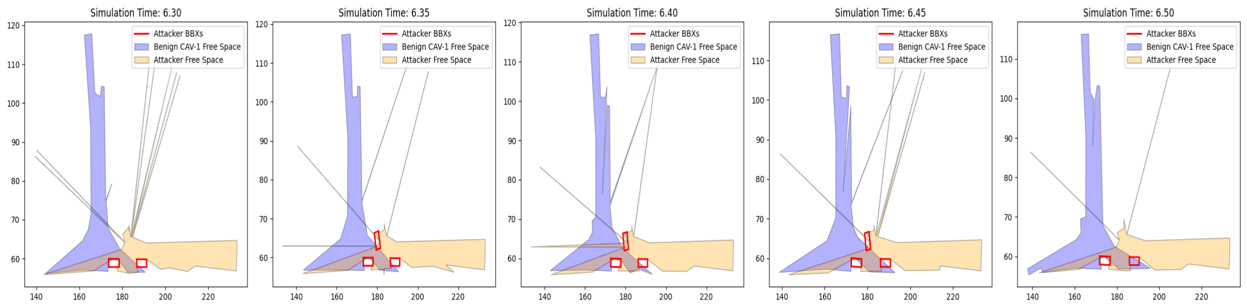


Figure 2.7: Illustration of Free Space Inconsistency

Potential misbehavior is identified based on trust score trends: an increasing trust score indicates potentially benign behavior, while a decreasing score suggests potential misbehavior. Vehicles with trust scores dropping below the threshold are permanently classified as misbehaving.

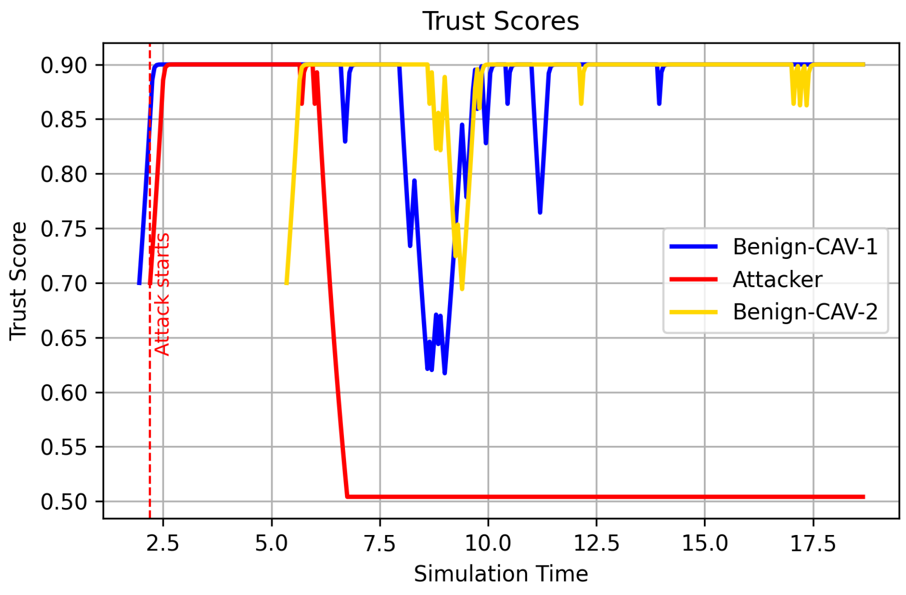


Figure 2.8: Trust Scores Evolution over Time - Threshold: 0.5

Performance Metrics

We analyze the evolution of the F_1 score, False Positive Rate (FPR), and True Positive Rate (TPR) over time. These metrics are calculated based on classification outcomes defined as follows:

- **True Positive (TP)**: Attacker correctly classified as potentially misbehaving.
- **False Positive (FP)**: Benign vehicle incorrectly classified as potentially misbehaving.
- **True Negative (TN)**: Benign vehicle correctly classified as benign.
- **False Negative (FN)**: Attacker incorrectly classified as benign.

The formulas used for the metrics are given by:

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.1}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{2.2}$$

$$\text{F}_1 \text{ score} = \frac{2 \cdot \text{TP}}{2 \cdot \text{TP} + \text{FP} + \text{FN}} \tag{2.3}$$

For offline classification analysis, we record TP, FP, TN, and FN for the entire scenario duration across varying thresholds.

Performance Evaluation

As expected, reducing the detection threshold leads to a decrease in TPR due to an increase in FNs. For instance, Figure 2.9 illustrates the trust score dynamics at a threshold of 0.3.

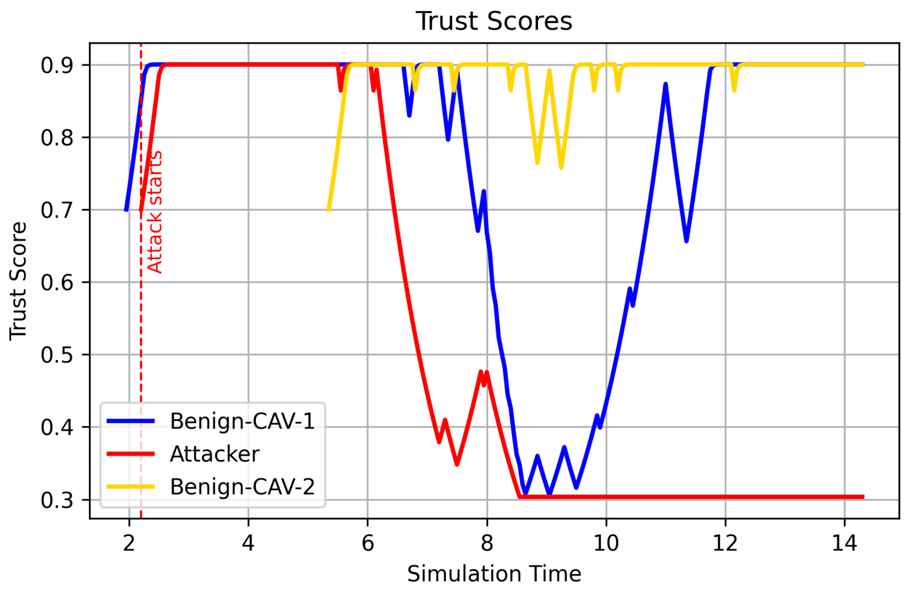


Figure 2.9: Trust Scores Evolution over Time - Threshold: 0.3

At this lower threshold, benign CAV-1 temporarily loses overlapping visibility with the attacker due to the occlusion caused by the non-connected green vehicle. Consequently, the attacker's trust score briefly increases, as no inconsistencies are detected during this interval. The detection of the attacker resumes only after benign CAV-2 enters the intersection, restoring overlapping visibility and thereby revealing inconsistencies again.

Threshold values within the range $[0.5, 0.6]$ yield relatively similar performance in terms of TPR and FPR as shown in Figure 2.10–2.11. Thus, a threshold of 0.5 is selected for subsequent time-series analysis as shown in Figure 2.12. The ego vehicle first identifies a potentially misbehaving vehicle around $6.3\,s$, at which point inconsistencies between the free space reported by benign CAV-1 and the attacker—stemming from the presence of a spoofed object—are initially observed. Given the persistent nature of the attack, the attacker's trust score continues to decline as inconsistencies increase when benign CAV-1 progresses further into the intersection. At approximately $7\,s$, the attacker's trust score falls below the threshold, resulting in the vehicle being blacklisted and subsequent communications from it being ignored. Non-negligible false positives are noted when benign CAV-1 and benign CAV-2 share overlapping observations, likely attributable to erroneous object detections by the perception module under increased traffic density and complexity at the intersection. Such false positives can be mitigated by employing more robust object detection algorithms. Overall, this approach is considered effective for misbehavior detection, particularly against persistent attacks in realistic traffic scenarios involving partial and full occlusions.

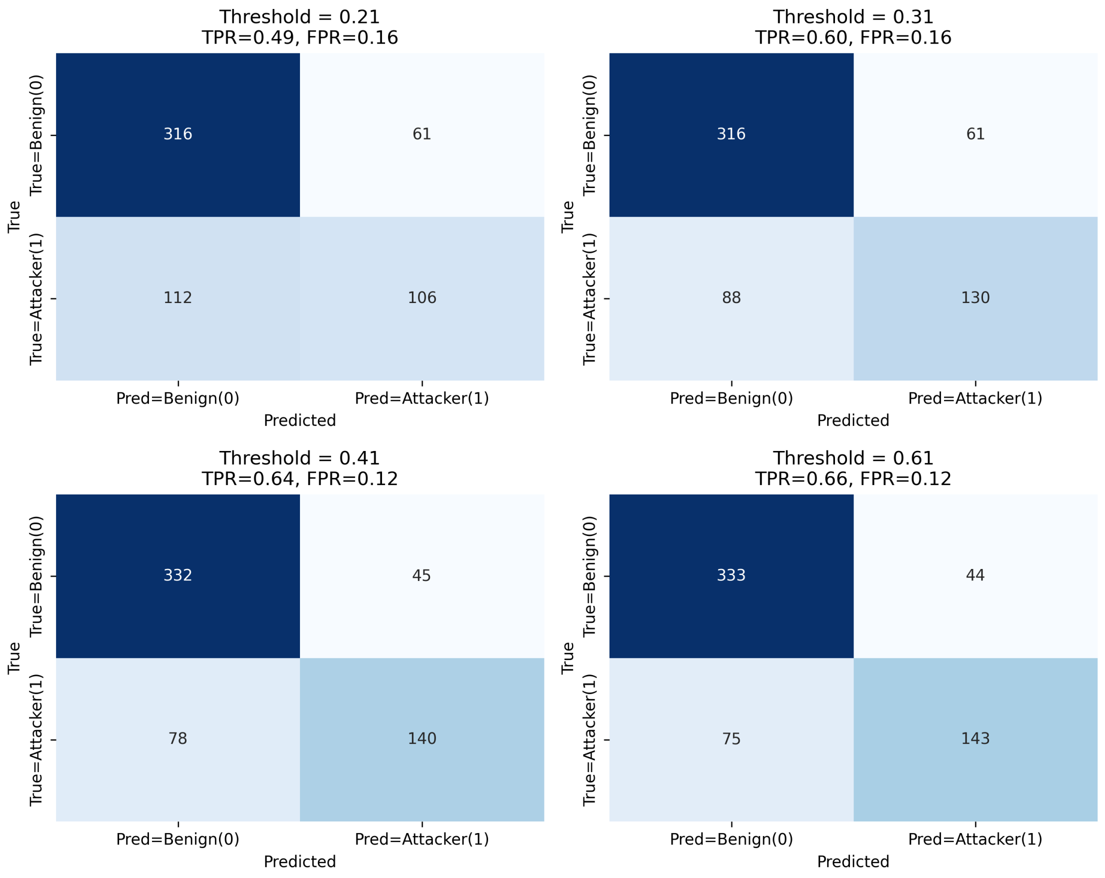


Figure 2.10: Classification Results for Varying Trust Thresholds

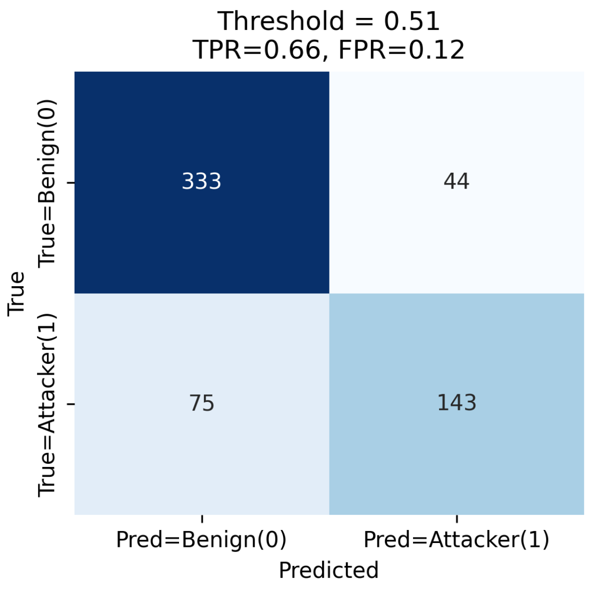
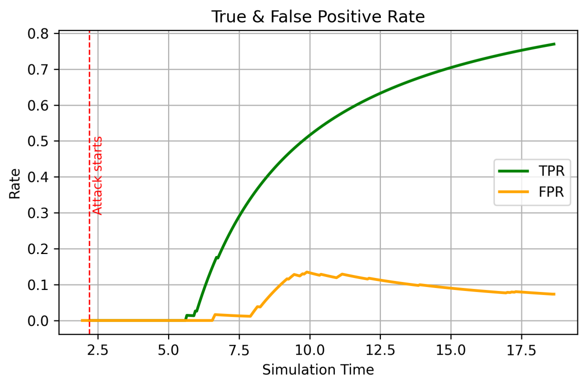
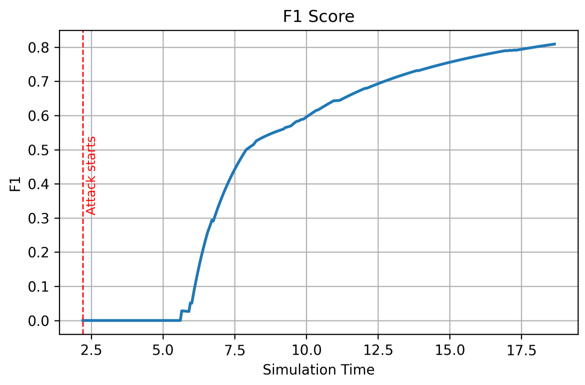


Figure 2.11: Classification Results for Trust Threshold = 0.5



(a) TPR vs FPR over Time



(b) F1 Score over Time

Figure 2.12: TPR vs FPR and F1 Score for Trust Threshold = 0.5

2.5.5 Erroneous Pose Information Scenario

Consider a malicious vehicle that transmits corrupted pose information to the ego vehicle. Because pose parameters define the $SE(2)$ transform used to express detections in the ego frame, such errors propagate through the spatial transformation and bias the reported positions, orientations and headings of all objects contributed by the malicious vehicle, degrading fusion and downstream tracking.

Let the true pose of the malicious vehicle in global coordinates be

$$\mathbf{p}_M^G \triangleq \begin{bmatrix} x_M^G & y_M^G & \theta_M^G \end{bmatrix}^\top ,$$

and let the malicious vehicle broadcast a corrupted pose

$$\hat{\mathbf{p}}_M^G = \mathbf{p}_M^G + \Delta\mathbf{p}_M^G, \qquad \Delta\mathbf{p}_M^G \triangleq \begin{bmatrix} \Delta x_M & \Delta y_M & \Delta\theta_M \end{bmatrix}^\top .$$

We model the perturbations as bounded, independent, uniformly distributed random variables [68]:

$$\Delta x_M \sim \mathcal{U}(x_{\min}, x_{\max}), \quad \Delta y_M \sim \mathcal{U}(y_{\min}, y_{\max}), \quad \Delta\theta_M \sim \mathcal{U}(\theta_{\min}, \theta_{\max}).$$

Let the true pose of the ego vehicle in global coordinates be

$$\mathbf{p}_E^G \triangleq \begin{bmatrix} x_E^G & y_E^G & \theta_E^G \end{bmatrix}^\top ,$$

The rotation matrix for an arbitrary orientation is given by

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

Define the global homogeneous transforms

$${}^{G}T_E = \begin{bmatrix} R(\theta_E^G) & \begin{bmatrix} x_E^G \\ y_E^G \end{bmatrix} \\ 0 & 1 \end{bmatrix} \qquad {}^{G}T_M = \begin{bmatrix} R(\theta_M^G) & \begin{bmatrix} x_M^G \\ y_M^G \end{bmatrix} \\ 0 & 1 \end{bmatrix} \qquad {}^{G}T_{\hat{M}} = \begin{bmatrix} R(\theta_M^G + \Delta\theta_M) & \begin{bmatrix} x_M^G + \Delta x_M \\ y_M^G + \Delta y_M \end{bmatrix} \\ 0 & 1 \end{bmatrix}.$$

The transformation of the malicious vehicle's true pose to the ego vehicle frame is obtained by performing the following operation:

$$ {}^{E}T_{M} = \left({}^{G}T_{E}\right)^{-1}\,{}^{G}T_{M} = \begin{bmatrix} R(\theta_E^G)^\top R(\theta_M^G) & R(\theta_E^G)^\top \left(\begin{bmatrix} x_M^G \\ y_M^G \end{bmatrix} - \begin{bmatrix} x_E^G \\ y_E^G \end{bmatrix} \right) \\ 0 & 1 \end{bmatrix}. $$

Similarly, the transformation matrix for the corrupted pose is given by

$$ {}^{E}T_{\hat{M}} = \left({}^{G}T_{E}\right)^{-1}\,{}^{G}T_{\hat{M}} = \begin{bmatrix} R(\theta_E^G)^\top R(\theta_M^G + \Delta\theta_M) & R(\theta_E^G)^\top \left(\begin{bmatrix} x_M^G + \Delta x_M \\ y_M^G + \Delta y_M \end{bmatrix} - \begin{bmatrix} x_E^G \\ y_E^G \end{bmatrix} \right) \\ 0 & 1 \end{bmatrix}. $$

Any object with homogeneous coordinates $\mathbf{p}^M = [X^M,\, Y^M,\, 1]^\top$ in the malicious vehicle frame M is reported in the ego vehicle frame as

$$ \mathbf{p}^E_{true} = {}^{E}T_{M}\,\mathbf{p}^M, \qquad \mathbf{p}^E_{corr} = {}^{E}T_{\hat{M}}\,\mathbf{p}^M $$

where $\mathbf{p}^E_{true}$ and $\mathbf{p}^E_{corr}$ are the uncorrupted and corrupted points in the ego vehicle frame.

Define the error due to corrupted pose as

$$ \delta\mathbf{p}^E \triangleq \mathbf{p}^E_{\text{corr}} - \mathbf{p}^E_{\text{true}} = \begin{bmatrix} \delta\mathbf{r}_E \\ 0 \end{bmatrix} $$

where $\delta\mathbf{r}_E$ is given by

$$ \delta\mathbf{r}_E = R(\theta_E^G)^\top \left(\begin{bmatrix} \Delta x_M \\ \Delta y_M \end{bmatrix} + R(\theta_M^G)\big(R(\Delta\theta_M) - I\big) \begin{bmatrix} X^M \\ Y^M \end{bmatrix} \right). $$

It can be seen that the error decomposes into a constant bias term originating from the global translation error $[\Delta x_M, \Delta y_M]^\top$ when rotated into the ego vehicle frame and a distance-dependent yaw error.

Assumptions

- The malicious vehicle is the only agent transmitting corrupted pose information as defined above.
- A single offset $\Delta\mathbf{p}_M = \begin{bmatrix} \Delta x_M & \Delta y_M & \Delta\theta_M \end{bmatrix}^\top$ is sampled once at the start of the simulation from the specified uniform bounds and is held *constant* for the entire duration.
- All other cooperative vehicles are assumed to transmit their *true* pose information.

2.5.6 Erroneous Pose Information Evaluation

Detection is based on geometric consistency within overlapping fields of view (FoVs) of the ego vehicle and other CAVs.

- *Small errors.* If $\max|\Delta x_M| < 0.4$ m, $\max|\Delta y_M| < 0.4$ m, and $\max|\Delta\theta_M| < 10^\circ$ (≈ 0.1745 rad), the induced shifts in the projected boxes are minor, data association remains stable, and the detector typically does not flag the malicious vehicle.
- *Moderate to large errors.* For translation deviations > 0.4 m in either axis or larger yaw offsets, discrepancies are observable when the ego vehicle and malicious vehicle FoVs overlap. Malicious vehicle observations, when expressed in the ego vehicle frame under biased extrinsics (pose-corrupted $SE(2)$ transform), exhibit geometric inconsistency with ego vehicle and benign vehicle projections of the same objects.

- *Free-space consistency.* This test is less discriminative because neither source is fully trusted. A benign vehicle and a malicious vehicle may sense the same object, but the malicious vehicle pose-biased projection can intersect the estimated free space of the benign vehicle, creating occupancy conflicts that are down weighted rather than conclusively identified.
- *Matching threshold.* Cross-view association uses a generalized IoU threshold of GIoU ≥ 0.0, tolerating slightly overlapping boxes and covering both spoofed vehicle and erroneous case.
- *Scaling with network size.* Larger error magnitudes improve detectability, but robust identification of pose corruption benefits from greater viewpoint diversity given by more cooperating benign vehicles, especially when the malicious vehicle's reported free space remains internally consistent with its own detections.

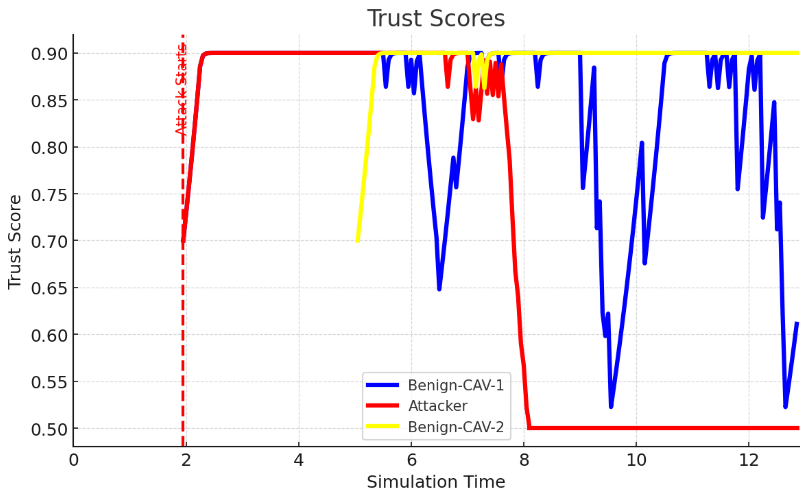


Figure 2.13: Trust Scores Evolution over Time - Erroneous Pose

Using the same three-CAV configuration with an additional non-connected vehicle as in the spoof case, Figure 2.13 shows that FreeSpaceConsistency does not flag the malicious vehicle early, consistent with the limitations described above. In this experiment, the translation offsets are drawn uniformly with random sign, $\Delta x, \Delta y \sim \mathcal{U}\big([-0.45, -0.4] \cup [0.4, 0.45]\big)$ m, and the yaw offset is symmetric, $\Delta\theta \sim \mathcal{U}(-10^\circ, 10^\circ)$. In this setting, the ego vehicle flags the malicious vehicle as misbehaving only via ego FoV consistency check.

Discussion. To assess the magnitude of pose errors that produce noticeable transformation error in the ego vehicle frame, we ran a parameter sweep. We began with large bounds on translation error, $|\Delta x_M|, |\Delta y_M| \approx 1$ m, and progressively reduced them. A similar sweep was performed for yaw, starting at 20° (approximately 0.3491 rad). We observed that orientation error alone is not sufficient to identify potential malicious vehicles because geometric consistency is evaluated through overlaps. Non-negligible overlap can still occur with the actual footprint observed from a different perspective by another CAV. Such cases can be resolved during the fusion stage using postprocessing functions and therefore are not reliable indicators of misbehavior.

We also varied the number of cooperating CAVs to assess the effect of viewpoint on detectability and to estimate the minimum number of benign vehicles needed to identify potential pose corruption. When only one benign vehicle and one malicious vehicle were present, results were often inconclusive for the free space consistency test as neither source could be assumed correct a priori.

2.6 Conclusion

A geometric consistency-based misbehavior detection algorithm was developed to identify bogus vehicle attacks, specifically addressing scenarios involving only the addition of fake vehicles rather than their removal. The attack is assumed to be persistent, and successful detection requires at least one benign vehicle with a field-of-view (FoV) overlapping that of the attacker. If vehicle removal attacks were to be considered, at least two benign vehicles with mutually overlapping FoVs would be necessary for reliable detection.

In the presence of incorrect pose information, the algorithm performs well only up to a point, its effectiveness being limited by (i) the need for more benign vehicles to provide viewpoint diversity or (ii) sufficient overlap of the ego vehicle - malicious vehicle FoVs. Future work should leverage motion propagation of bounding boxes and time-propagated free-space representations, and analyze patterns in cross-view overlap offsets to discriminate sensor noise from pose errors.

While geometric operations implemented using Shapely are optimized, the pairwise consistency checks currently exhibit a slower computational complexity of $o(N^2)$, where N is the number of CAVs within communication range. This highlights the potential for further optimization through spatial tree structures such as R-trees, enabling more efficient geometric querying. Another avenue for reducing computation involves adopting a priority-based analysis, where only vehicles offering the most relevant information for the ego vehicle's maneuver are considered instead of processing every vehicle within communication range.

Future work should extend the approach to scenarios involving non-persistent attacks and relax the assumption of synchronized observations, necessitating forward-time propagation of free-space representations.

Chapter 3

Uncertainty-Aware Model Predictive Control

3.1 Introduction

Safety is a fundamental objective in the development of automated vehicles (AVs). Perception modules within AV systems employ deep learning (DL) techniques for feature extraction and environmental state estimation, using annotated datasets for training. The outputs of these perception modules are propagated to downstream modules responsible for decision making and motion planning. As a result, any errors or uncertainties present in DL-based perception propagate through the system and can adversely affect overall safety [69].

To mitigate this, it is essential not only to quantify the uncertainty associated with perceived objects but also to incorporate these uncertainties into the control algorithm, particularly for collision avoidance. Among the various methods explored in the literature, Model Predictive Control (MPC) is especially well suited for trajectory planning in AVs. MPC solves a constrained Optimal Control Problem (OCP) at each iteration, producing a sequence of control actions that minimize a specified cost function while satisfying system and safety constraints. Only the first action of the optimal control solution is applied to the system, after which the prediction horizon is advanced, and the optimization problem is re-solved using updated state and environment information. In AV motion planning, the cost function is typically designed to penalize deviations from the reference trajectory or desired behavior, while constraints maintain safety and dynamic feasibility [70].

In MPC-based motion planning, a standard approach is to predict the trajectories of dynamic objects using a motion model with additive process noise, capturing the stochasticity in their behavior. Rather than initializing tracks with large state uncertainties that would require several measurements to correct, explicitly predicting uncertainties could improve track initialization, track association, and enable better physics-based propagation over the planning horizon, even though such propagation schemes often degrade over longer horizons. Object states and associated uncertainties are maintained via Kalman filtering.

Our pipeline comprises three modules: perception, tracking, and control. The perception module follows the late-fusion cooperative scheme described in Chapter 1. For tracking, we modify AB3DMOT to perform online tracking and replace the constant-velocity/linear-KF motion model with a Constant-Turn-Rate-Velocity (CTRV) model and an Extended Kalman Filter (EKF) [14]. The predicted uncertainties are incorporated within the tracker to yield improved state estimates. The resulting tracked objects and their covariance estimates are passed to the control module for probabilistic collision avoidance and safe trajectory planning.

Remark: Assumptions are stated in every relevant section; unless explicitly stated otherwise, they apply to the overall problem.

3.2 Related Works

Interest in uncertainty-aware motion planning is growing in recent literature, with a focus not only on estimating uncertainties in object detection and corresponding intentions but also on incorporating this information into downstream tasks such as motion planning and control. Estimating uncertainties for both detection and intention helps avoid overly conservative solutions.

One such work describes a framework that predicts multiple trajectories for nearby dynamic obstacles (DOs) and employs an Interactive Multiple Model (IMM) Kalman filter to maintain their intention estimates [70]. This information is passed to a Stochastic Model Predictive Control (SMPC) algorithm, which provides the control sequence. With the help of an IMM filter, the model can increase or decrease the likelihood of a DO's intention estimates as new measurements are received, thereby reducing control conservatism. However, the paper assumes that all objects are visible and that the corresponding noise matrices are fixed. Additionally, the intention possibilities are predetermined for each obstacle. Another related work addresses multi-modal predictions at intersections [71]. It uses the MultiPath framework [72], which outputs a Gaussian Mixture Model (GMM) over future trajectories with corresponding per time-step uncertainties across the prediction horizon. This information is then passed to an SMPC algorithm that optimizes over a class of feedback policies to produce control actions. However, this approach requires a top-down (BEV) image of the scene that captures all nearby agents, essentially assuming complete visibility (i.e., no occlusions).

Both approaches described above assume full visibility. In practice, modern AVs perceive the environment via onboard sensors and map priors rather than through a perfect oracle. Thus, perceived object uncertainty should be attributed to sensor noise and environmental factors. Perception-uncertainty–aware planning was explored in [69], which proposes Perception-Uncertainty-aware Rapidly-exploring Random Trees (PU-RRT). They use an uncertainty-aware 3D object detector to produce bounding boxes with associated uncertainties, then transform these boxes into enlarged ellipses that account for both spatial and localization uncertainty. The resulting obstacles are passed to an RRT planner to generate feasible, collision-free trajectories. However, although this introduces a useful mechanism for incorporating perception uncertainty via ellipses around detections, the constraint formulation appears incomplete. Additionally, replanning is performed only with respect to the current observations. There is no trajectory-prediction model that accounts for the future intentions and motions of nearby detected obstacles, issues that can increase the risk of collision. Similarly, other works on chance-constrained motion planning [73–78] propose various methods for propagating obstacle uncertainty and incorporating it into the MPC framework.

The works above consider only single-vehicle perception. In complex scenarios such as urban intersections and T-junctions, planning often becomes conservative due to occlusions. This can be mitigated via cooperative perception, where nearby CAVs communicate perception information, improving FoV and enabling safe execution of complex maneuvers (e.g., left turns) [79]. Zhang [80] presents one such motion-planning framework with integrated perception under varying weather conditions. It leverages both semantic and geometric information, using a CNN for weather classification and a LiDAR-based perception module for object detection. The system performs object-level fusion and passes the results to a deterministic MPC for lane-change operations in an intersection scenario and a highway scenario. However, it does not model the uncertainties associated with detected objects and does not address more complex maneuvers such as left turns.

Similarly, Zhao [81] presents a deterministic MPC approach in which CAVs transmit perception information to a base station at the intersection for coordinated, safe driving aimed at improving comfort, efficiency, and safety, and reducing fuel consumption. However, the paper does not specify the perception modality and does not account for uncertainty in detected objects.

In our work, we consider object-level cooperative perception wherein nearby CAVs communicate detected objects along with their associated uncertainties. We use a centralized scheme at the ego vehicle, which performs object fusion and maintains a tracker for the fused objects. The tracked objects are then provided to an MPC algorithm for reference trajectory tracking. We assume a single intention for each detected obstacle.

3.3 Method

This section details the problem formulation: the vehicle model, perception information integration, chance constraint formulation and the optimal control problem.

3.3.1 Ego Vehicle Model

We model the ego vehicle with a simplified kinematic bicycle model since we are operating at low speeds. In the kinematic bicycle model, the vehicle's position in the global reference frame is denoted by the coordinates X and Y. The rate of change in these coordinates depends on the longitudinal velocity of the vehicle V and the heading angle ψ, as well as the slip angle $\beta(u_2)$, which is determined by the input of the steering u_2. These variables are illustrated in Figure 3.1. The vehicle's motion in the global coordinate system is described by the following equations:

$$\dot{X} = V\cos(\psi + \beta(u_2)) \tag{3.1}$$

$$\dot{Y} = V\sin(\psi + \beta(u_2)) \tag{3.2}$$

$$\dot{V} = u_1 \tag{3.3}$$

$$\dot{\psi} = \frac{V}{l_r}\sin(\beta(u_2)) \tag{3.4}$$

where

- X, Y is the vehicle's position in the global frame,
- V is the velocity of the vehicle,
- u_1 represents the acceleration command,
- ψ represents the heading angle,
- $\beta(u_2)$ is the angle of slip at the center of gravity, which represents the lateral displacement due to steering.

The slip angle at the center of gravity is given by:

$$\beta(u_2) = \arctan\left(\tan(u_2)\frac{l_r}{l_f + l_r}\right)\Bigg(\tag{3.5}$$

These equations describe the vehicle's motion in the global coordinate frame, where the position (X, Y) evolves based on the velocity and heading dynamics.

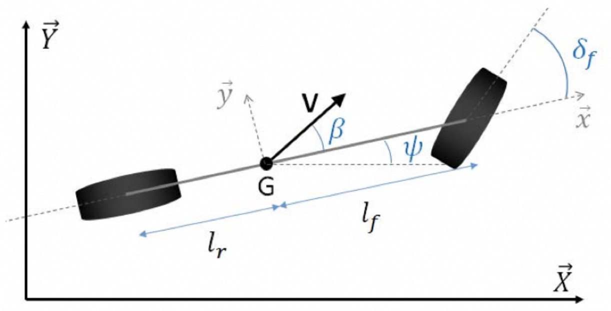


Figure 3.1: Kinematic Bicycle Model

3.3.2 Dynamic Obstacle Tracking and Prediction

Object detection is performed using a PointPillars-based architecture as detailed in Chapter 1. The predicted 3D bounding boxes and their associated variances are used to initialize tracks in a CTRV-EKF tracker. Because velocity and yaw rate are not observable state variables, high prior variances are assigned to these states. As new measurements arrive, track-to-measurement association is performed via geometric overlap using the Generalized Intersection over Union (GIoU) metric.

Assumptions

- In cooperative perception, the ego vehicle may appear in a connected automated vehicle's (CAV) field of view (FoV), resulting in fused detections that correspond to the ego vehicle itself. Prior to track association, such instances are removed using a GIoU-based comparison with the ego bounding box.
- In our current framework, we assume that only a single intention and the corresponding trajectory is predicted for each perceived object, based on past observations.

Generalized Intersection over Union (GIoU) Given two bounding boxes A and B, and their smallest enclosing box C, the IoU and Generalized IoU are defined as:

$$\mathrm{IoU}(A, B) = \frac{\mathrm{area}(A \cap B)}{\mathrm{area}(A \cup B)} \tag{3.6}$$

$$\mathrm{GIoU}(A, B) = \mathrm{IoU}(A, B) - \frac{\mathrm{area}(C \setminus (A \cup B))}{\mathrm{area}(C)} \tag{3.7}$$

where area$(\cdot)$ denotes the area (or volume in 3D), and C is the minimal enclosing box containing both A and B.

Instead of employing a fixed noise covariance matrix R, the predicted per-detection variances are used (i) to set the initial state covariance P_0 for each new track, and (ii) to construct a measurement-specific noise matrix R_k. After each EKF update, the filtered bounding boxes and their associated covariances are provided to the control module for chance-constrained MPC.

State and Measurement Vectors The CTRV model tracks a nine-dimensional state:

$$\mathbf{x} = \begin{bmatrix} x & y & z & \theta & l & w & h & v & \omega \end{bmatrix}^\top \tag{3.8}$$

with measurements

$$\mathbf{z} = \begin{bmatrix} x & y & z & \theta & l & w & h \end{bmatrix}^\top \tag{3.9}$$

where

- (x, y, z) is the 3D center of the object in Unreal[1] coordinates [m],
- θ is the yaw (heading) about the vertical axis [rad],
- (l, w, h) are the bounding-box dimensions in Unreal coordinates [m],
- v is the velocity [m/s],
- ω is the yaw rate [rad/s].

Discrete-Time State Transition (CTRV Model) For a discrete time step Δt, the nonlinear state transition is given by:

$$x_{k+1} = \begin{cases} x_k + \dfrac{v_k}{\omega_k}\left(\sin(\theta_k + \omega_k \Delta t) - \sin(\theta_k)\right), & \omega_k \neq 0 \\ x_k + v_k \Delta t \cos(\theta_k), & \omega_k \approx 0 \end{cases} \tag{3.10}$$

$$y_{k+1} = \begin{cases} y_k + \dfrac{v_k}{\omega_k}\left(-\cos(\theta_k + \omega_k \Delta t) + \cos(\theta_k)\right), & \omega_k \neq 0 \\ y_k + v_k \Delta t \sin(\theta_k), & \omega_k \approx 0 \end{cases} \tag{3.11}$$

$$\theta_{k+1} = \theta_k + \omega_k \Delta t. \tag{3.12}$$

$$\tag{3.13}$$

Extended Kalman Filter The Jacobian A_k of the nonlinear transition function $f(\cdot)$ with respect to the state $\mathbf{x}_k$ is computed at each time step for the EKF update. For $\omega_k \neq 0$, the relevant nonzero entries are:

$$A_k = \frac{\partial f}{\partial \mathbf{x}}_{\hat{\mathbf{x}}_{k-1|k-1}} \tag{3.14}$$

where

$$\frac{\partial x_{k+1}}{\partial \theta_k} = \frac{v_k}{\omega_k}\left(-\cos(\theta_k) + \cos(\theta_k + \omega_k \Delta t)\right) \tag{3.15}$$

$$\frac{\partial x_{k+1}}{\partial v_k} = \frac{\sin(\theta_k + \omega_k \Delta t) - \sin(\theta_k)}{\omega_k} \tag{3.16}$$

$$\frac{\partial x_{k+1}}{\partial \omega_k} = \frac{v_k}{\omega_k^2}\left(\sin(\theta_k) - \sin(\theta_k + \omega_k \Delta t)\right) + \frac{v_k \Delta t}{\omega_k}\cos(\theta_k + \omega_k \Delta t) \tag{3.17}$$

$$\frac{\partial y_{k+1}}{\partial \theta_k} = \frac{v_k}{\omega_k}\left(\sin(\theta_k + \omega_k \Delta t) - \sin(\theta_k)\right) \tag{3.18}$$

$$\frac{\partial y_{k+1}}{\partial v_k} = \frac{-\cos(\theta_k + \omega_k \Delta t) + \cos(\theta_k)}{\omega_k} \tag{3.19}$$

$$\frac{\partial y_{k+1}}{\partial \omega_k} = \frac{v_k}{\omega_k^2}\left(-\cos(\theta_k) + \cos(\theta_k + \omega_k \Delta t)\right) + \frac{v_k \Delta t}{\omega_k}\sin(\theta_k + \omega_k \Delta t) \tag{3.20}$$

$$\frac{\partial \theta_{k+1}}{\partial \omega_k} = \Delta t \tag{3.21}$$

[1]Unreal uses a left-handed coordinate system (X–Forward, Y–Right, Z–Up)

All other entries correspond to the identity matrix, as the remaining state variables are assumed constant during propagation.

The measurement function extracts the first seven elements of the state:

$$\mathbf{z}_k = H\mathbf{x}_k \tag{3.22}$$

where the measurement matrix H is

$$H = \begin{bmatrix} I_{7\times7} & 0_{7\times2} \end{bmatrix} \tag{3.23}$$

The EKF predict and update equations are given as follows:

Prediction:

$$\hat{\mathbf{x}}_{k|k-1} = f(\hat{\mathbf{x}}_{k-1|k-1}) \tag{3.24}$$

$$P_{k|k-1} = A_k P_{k-1|k-1} A_k^\top + Q_k \tag{3.25}$$

Update:

$$\mathbf{y}_k = \mathbf{z}_k - h(\hat{\mathbf{x}}_{k|k-1}) \tag{3.26}$$

$$S_k = H_k P_{k|k-1} H_k^\top + R_k \tag{3.27}$$

$$K_k = P_{k|k-1} H_k^\top S_k^{-1} \tag{3.28}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + K_k \mathbf{y}_k \tag{3.29}$$

$$P_{k|k} = (I - K_k H_k) P_{k|k-1} \tag{3.30}$$

where $A_k = \frac{\partial f}{\partial \mathbf{x}}\Big|_{\hat{\mathbf{x}}_{k-1|k-1}}$ is the Jacobian state transition, Q_k is the process noise covariance, $H_k = \frac{\partial h}{\partial \mathbf{x}}\Big|_{\hat{\mathbf{x}}_{k|k-1}}$ is the Jacobian measurement and R_k is the noise covariance measurement.

Trajectory Prediction For physics-based trajectory prediction of perceived objects, we utilize the CTRV model to propagate the object state. During trajectory prediction over the planning horizon, the state covariances are recursively propagated using the EKF prediction equations. Due to ambiguity in LiDAR-based perception, the measured yaw can occasionally flip, differing from the tracked orientation by approximately π, which is impossible for objects in consecutive frames. To address this, an orientation correction is applied prior to filtering using

$$\text{If} \quad (\hat{\theta}_k - \theta_k^{\text{track}}) > \frac{\pi}{2} \quad \Longrightarrow \quad \hat{\theta}_k \leftarrow (\hat{\theta}_k + \pi). \tag{3.31}$$

To avoid erratic yaw updates and compounding errors from perception, a first-order low-pass filter is applied to the predicted yaw:

$$\hat{y}_k = \text{wrap}\left(\tilde{y}_{k-1} + \omega_k \Delta t\right) \tag{3.32}$$

$$\tilde{y}_k = (1-\alpha)\tilde{y}_{k-1} + \alpha \hat{y}_k \tag{3.33}$$

where

- $\tilde{y}_k$ is the smoothed yaw at step k,
- $\hat{y}_k$ is the predicted yaw,

- ω_k is the predicted yaw rate,
- $\alpha \in (0,1)$ is the smoothing factor,
- wrap$(\cdot)$ ensures angles remain in $(-\pi, \pi]$.

This filtering avoids unrealistic trajectory predictions, such as loops or heading reversals.

3.3.3 Chance Constraints

Here we present the methodology utilized to reformulate the Gaussian Chance Constraints to deterministic ones to obtain a tractable optimization problem.

Assumptions

- Independent Gaussian variances: Variances along each coordinate axis are treated as independent; cross-covariances are neglected.
- Single-intention prediction: Each obstacle provides only one predicted trajectory (no multi-modal intentions).
- Perfect ego localization: The ego vehicle's pose is assumed perfectly known, so its uncertainty is not considered in the constraint.

General Linear Chance Constraint Reformulation

We describe how linear chance constraints are transformed to deterministic constraints. We first briefly explain how scalar constraints are transformed to deterministic inequalities. Following this, we extend the analysis to vectors using hyperplane definitions.

Scalar case Consider $X \sim \mathcal{N}(\mu, \sigma^2)$ with a risk tolerance of $\delta \in (0, \frac{1}{2})$. With a δ risk tolerance, we bound the probability of violation as follows:

$$\Pr(X < 0) \leq \delta \iff \mu \geq c, \tag{3.34}$$

with

$$c = \sqrt{2}\,\sigma \operatorname{erf}^{-1}(1 - 2\delta), \tag{3.35}$$

and

$$\operatorname{erf}(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2}\, dt. \tag{3.36}$$

Linear–Gaussian multivariate case Let the state vector satisfy

$$\mathbf{x} \sim \mathcal{N}(\hat{\mathbf{x}}, \boldsymbol{\Sigma}), \tag{3.37}$$

with mean $\hat{\mathbf{x}} \in \mathbb{R}^n$ and covariance $\boldsymbol{\Sigma} \succ 0$. Constraint satisfaction is assessed using the signed margin (unnormalized perpendicular distance) to the hyperplane $\mathbf{a}^\top \mathbf{x} = b$:

$$V := \mathbf{a}^\top \mathbf{x} - b. \tag{3.38}$$

The constraint is satisfied iff $V > 0$. Impose the risk tolerance

$$\Pr(V \leq 0) \leq \delta. \tag{3.39}$$

Since $V \sim \mathcal{N}(\mathbf{a}^\top \hat{\mathbf{x}} - b,\ \mathbf{a}^\top \boldsymbol{\Sigma}\, \mathbf{a})$, we have the closed form solution

$$\Pr(V \le 0) \ = \ \tfrac{1}{2} + \tfrac{1}{2}\ \mathrm{erf}\!\left(\frac{b - \mathbf{a}^\top \hat{\mathbf{x}}}{\sqrt{2\,\mathbf{a}^\top \boldsymbol{\Sigma}\,\mathbf{a}}}\right)\Bigg(\tag{3.40}$$

and enforcing $\Pr(V \le 0) \le \delta$ yields the deterministic inequality

$$\mathbf{a}^\top \hat{\mathbf{x}} - b \ \ge \ c, \tag{3.41}$$

with threshold

$$c \ = \ \mathrm{erf}^{-1}(1 - 2\delta)\,\sqrt{2\,\mathbf{a}^\top \boldsymbol{\Sigma}\,\mathbf{a}} \ = \ z_{1-\delta}\,\sqrt{\mathbf{a}^\top \boldsymbol{\Sigma}\,\mathbf{a}}, \qquad z_{1-\delta} := \Phi^{-1}(1-\delta). \tag{3.42}$$

Multiple linear constraints and nonconvex sets For a polyhedral set $A\mathbf{x} \le \mathbf{b}$, apply (3.41) row-wise to each $(\mathbf{a}_i^\top, b_i)$. For nonconvex geometry, we may use a disjunctive (union–bound) relaxation over a finite collection of supporting half-spaces [82].

Chance-Constraint Reformulation for a Single Dynamic Obstacle

Here, we describe how we transform ellipsoidal chance constraints to deterministic constraints for a single dynamic obstacle.

At prediction step k the obstacle's center in the global frame is $\mathbf{o}_k = [x_k^{\mathrm{o}},\ y_k^{\mathrm{o}}]^\top$ with heading ψ_k^{o}. Let the ego vehicle's position be $\mathbf{p}_k = [x_k,\ y_k]^\top$ and define the relative vector between the two as

$$\mathbf{r}_k \ = \ \mathbf{p}_k - \mathbf{o}_k. \tag{3.43}$$

Ellipse footprint We model each rectangular obstacle of width w and length l by an ellipse with half-axes (a, b) defined in the obstacle's local frame. The formulation for incorporating spatial uncertainties follows [69]. However, where [69] uses the inscribed ellipse ($a = \frac{l}{2}$, $b = \frac{w}{2}$), we instead adopt the minimal oriented circumscribed ellipse that encloses the rectangle:

$$a_{\mathrm{base}} = \frac{\sqrt{2}}{2}\,l, \qquad b_{\mathrm{base}} = \frac{\sqrt{2}}{2}\,w. \tag{3.44}$$

This change avoids under–approximating the footprint especially in cases of highly certain detections and yields a slightly conservative outer bound that contains the entire rectangle.

Inflated safety ellipse Let $\Sigma_k \in \mathbb{R}^{2\times 2}$ be the obstacle's planar position covariance. The inflated ellipse used for collision checking is obtained as the Minkowski sum of the ego and obstacle ellipses. In the obstacle–aligned frame this reduces to adding half–axes:

$$a_{\mathrm{tot}} \ = \ a_{\mathrm{ego}} + a_{\mathrm{obs}}, \qquad b_{\mathrm{tot}} \ = \ b_{\mathrm{ego}} + b_{\mathrm{obs}}. \tag{3.45}$$

Define the rotation into the obstacle frame $R(\psi_k)$ and the scaling operator

$$\Theta_k \ = \ R(\psi_k)^\top \, \mathrm{diag}\!\big(1/a_{\mathrm{tot}},\ 1/b_{\mathrm{tot}}\big)\Bigg(\qquad R(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix}, \tag{3.46}$$

so the normalized relative position is $\boldsymbol{\xi}_k = \Theta_k\,\mathbf{r}_k$.

Collision set and chance constraint (single obstacle o_k) With $\mathbf{r}_k = \mathbf{p}_k - \mathbf{o}_k$ and $\boldsymbol{\xi}_k = \Theta_k\, \mathbf{r}_k$, define the ellipsoidal collision region

$$C^k_{\text{ego},o} \;:=\; \{\mathbf{r_k} \in \mathbb{R}^2 \quad \|\mathbf{r_k}\|_{\Omega_k} \le 1 \;\;, \qquad \Omega_k := \Theta_k^\top \Theta_k, \tag{3.47}$$

where $\|\mathbf{v}\|_\Omega := \sqrt{\mathbf{v}^\top \Omega\, \mathbf{v}}$. The collision–probability requirement for a single obstacle is

$$\Pr\bigl(\mathbf{r}_k \in C^k_{\text{ego},o}\bigr) \le\; \varepsilon_k \quad\iff\quad \Pr\bigl(\|\Theta_k \mathbf{r}_k\|_2 \le 1\bigr) \le\; \varepsilon_k. \tag{3.48}$$

Gaussian model and spherical transform [73] Assume a Gaussian model for the relative position

$$\mathbf{r}_k \sim \mathcal{N}\bigl(\hat{\mathbf{r}}_k,\, \Sigma_{r,k}\bigr), \qquad \hat{\mathbf{r}}_k := \hat{\mathbf{p}}_k - \hat{\mathbf{o}}_k,\; \Sigma_{r,k} := \mathrm{Cov}(\mathbf{p}_k - \mathbf{o}_k) \;\; \text{(i.e., } \Sigma_{p,k} + \Sigma_{o,k} \text{ if independent)}, \tag{3.49}$$

and transform to the scaled ("spherical") region

$$\boldsymbol{\xi}_k = \Theta_k \mathbf{r}_k \;\sim\; \mathcal{N}\bigl(\hat{\boldsymbol{\xi}}_k,\, \Sigma_{\xi,k}\bigr), \qquad \hat{\boldsymbol{\xi}}_k := \Theta_k \hat{\mathbf{r}}_k,\; \Sigma_{\xi,k} := \Theta_k \Sigma_{r,k} \Theta_k^\top. \tag{3.50}$$

In these coordinates the collision set is the unit ball $\tilde{C}^k := \{\boldsymbol{\xi} \mid \|\boldsymbol{\xi}\|_2 \le 1\}$, and

$$\Pr\bigl(\mathbf{r}_k \in C^k_{\text{ego},o}\bigr) \;=\; \Pr\bigl(\boldsymbol{\xi}_k \in \tilde{C}^k\bigr) = \int_{\|\boldsymbol{\xi}\| \le 1} \mathcal{N}\bigl(\boldsymbol{\xi} \;\; \hat{\boldsymbol{\xi}}_k, \Sigma_{\xi,k}\bigr)\, d\boldsymbol{\xi}. \tag{3.51}$$

Half–space relaxation Following [73], we perform a conservative over-approximation of the unit ball by its supporting half–space at $\hat{\boldsymbol{\xi}}_k$:

$$\tilde{C}^k \;\subset\; \hat{C}^k := \{\boldsymbol{\xi} \;\; \mathbf{a}_k^\top \boldsymbol{\xi} \le b_k \;\;, \qquad \mathbf{a}_k := \frac{\hat{\boldsymbol{\xi}}_k}{\|\hat{\boldsymbol{\xi}}_k\|_2}, \;\; b_k := 1. \tag{3.52}$$

Because $\tilde{C}^k \subset \hat{C}^k$, we have $\Pr(\boldsymbol{\xi}_k \in \tilde{C}^k) \le \Pr(\boldsymbol{\xi}_k \in \hat{C}^k)$. Thus, we have a sufficient condition for (3.48) if the following constraint is satisfied

$$\Pr\bigl(\mathbf{a}_k^\top \boldsymbol{\xi}_k \le b_k\bigr) \le\; \varepsilon_k. \tag{3.53}$$

Deterministic reformulation Since $\mathbf{a}_k^\top \boldsymbol{\xi}_k \sim \mathcal{N}\bigl(\mathbf{a}_k^\top \hat{\boldsymbol{\xi}}_k,\; \mathbf{a}_k^\top \Sigma_{\xi,k} \mathbf{a}_k\bigr)$, (3.53) is equivalent to the deterministic inequality using the above mentioned deterministic reformulation of linear chance constraints

$$\mathbf{a}_k^\top \hat{\boldsymbol{\xi}}_k - b_k \;\ge\; z_{1-\varepsilon_k} \sqrt{\mathbf{a}_k^\top \Sigma_{\xi,k} \mathbf{a}_k}, \qquad z_{1-\varepsilon_k} = \Phi^{-1}(1-\varepsilon_k) \;=\; \sqrt{2}\, \mathrm{erf}^{-1}(1-2\varepsilon_k), \tag{3.54}$$

which, with (3.52), simplifies to

$$\|\hat{\boldsymbol{\xi}}_k\|_2 - 1 \;\ge\; z_{1-\varepsilon_k} \sqrt{\frac{\hat{\boldsymbol{\xi}}_k^\top \Sigma_{\xi,k} \hat{\boldsymbol{\xi}}_k}{\|\hat{\boldsymbol{\xi}}_k\|_2^2}}. \tag{3.55}$$

Equivalently, defining the risk margin $m_k := z_{1-\varepsilon_k} \sqrt{\hat{\boldsymbol{\xi}}_k^\top \Sigma_{\xi,k} \hat{\boldsymbol{\xi}}_k} / \|\hat{\boldsymbol{\xi}}_k\|_2$, the constraint is essentially given by $\|\hat{\boldsymbol{\xi}}_k\|_2 - 1 \ge m_k$.

Non-uniform distance-based ε assignment In this work, we utilize the predicted covariance matrices for the entire trajectory within the above chance-constraint framework. Instead of uniformly assigning the collision-risk budget throughout the prediction horizon, we focus on the obstacle trajectory segments that interact with the ego vehicle trajectory.

Let $\mathbf{p_e}[l], \mathbf{p_k}[l] \in \mathbb{R}^2$ be the predicted ego and obstacle-k positions at step l and define the Euclidean separation as:

$$d_{l,k} \triangleq \quad \mathbf{p_e}[l] - \mathbf{p_k}[l] \quad _2. \tag{3.56}$$

With ego vehicle length l_{ego}, we set the interaction length

$$L_{\text{win}} \;=\; 3\, l_{\text{ego}}, \tag{3.57}$$

and declare obstacle k interacting if at any step it comes within this distance:

$$\mathcal{K}_{\text{int}} \;\triangleq\; \Big\{ k \in \{1, \dots, N_o\} \;:\; \exists\, l \in \{0, \dots, N-1\} \text{ s.t. } d_{l,k} \le L_{\text{win}} \Big\}.\Big(\tag{3.58}$$

For each interacting obstacle $k \in \mathcal{K}_{\text{int}}$, the interaction window is

$$\mathcal{I}_k \;\triangleq\; \{\, l\Big(\in \{0, \dots, N-1\} \;:\; d_{l,k} \le L_{\text{win}} \quad . \tag{3.59}$$

The risk is distributed only over $\mathcal{I}_k$. For $l \notin \mathcal{I}_k$ (or $k \notin \mathcal{K}_{\text{int}}$) we revert to the deterministic collision avoidance check that does not use predicted uncertainties. Because the risk margin increases as ε decreases, applying it outside the interaction window is unnecessary and would cause the ego vehicle to deviate from its reference trajectory earlier than required to satisfy Equation 3.55, thereby increasing overall lateral deviation. Accordingly, we apply the probability risk margin only within the prediction interaction window. For segments outside the interaction window, we set $\varepsilon = 0.5$, thereby nullifying the impact of predicted uncertainties. The above uses a point-wise Euclidean threshold. It can be tightened by detecting intersections between successive line segments of the ego and obstacle paths (poly-line segment intersection).

Let N_o be the number of perceived obstacles, and let $\varepsilon_{\text{total}} > 0$ denote the total risk budget on the prediction horizon of length N. Let $N_{rel} \le N_o$ denote the number of relevant objects, where relevance is determined by the trajectory interaction logic described above. We then divide the budget per relevant obstacle and then per time step [83]. For simplicity,

$$\varepsilon_{\text{obs},k} \;\triangleq\; \frac{\varepsilon_{\text{total}}}{N_r el}, \qquad k = 1, \dots, N_o, \tag{3.60}$$

and we assign stepwise risks $\{\varepsilon_{l,k}\}_{l=0}^{N-1}$ satisfying

$$\sum_{l=0}^{N-1}\Big(\varepsilon_{l,k} \;\le\; \varepsilon_{\text{obs},k} \quad (\forall k), \qquad \sum_{k=1}^{N_o}\sum_{l=0}^{N-1}\Big(\varepsilon_{l,k} \;\le\; \varepsilon_{\text{total}}. \tag{3.61}$$

For obstacle k, define the closest index $t_c(k) \in \arg\min_{l \in \{0,\dots,N-1\}} d_{l,k}$, where $d_{l,k}$ is the predicted ego–obstacle distance at step l. We introduce an interaction window $\mathcal{I}_k \subseteq \{0, \dots, N-1\}$ centered at $t_c(k)$ and assign nonnegative, normalized weights $\{w_{l|k}\}_{l \in \mathcal{I}_k}$ that decrease with temporal distance from $t_c(k)$, following [83]:

$$w_{l|k} \ge 0, \qquad \sum_{l \in \mathcal{I}_k}\Big(w_{l|k} = 1, \qquad w_{l|k} = g\big(|l - t_c(k)|\big),\Big(\; g(\cdot) \text{ monotone decreasing.} \tag{3.62}$$

The distance–based allocation on the window is

$$\tilde{\varepsilon}_{l,k} \;=\; \varepsilon_{\mathrm{obs},k}\, w_{l|k}, \qquad l \in \mathcal{I}_k. \tag{3.63}$$

For steps outside the interaction window ($l \notin \mathcal{I}_k$), we revert to the deterministic collision–avoidance formulation that does not use the predicted uncertainties for collision checking; i.e., no uncertainty–driven risk allocation is applied outside $\mathcal{I}_k$.

Very small risks can over-inflate the safety ellipse, so we impose a per-step floor $\varepsilon_{\min} > 0$ on the window and then preserve the per-obstacle budget:

$$\varepsilon_{l,k} \;\leftarrow\; \max\{\tilde{\varepsilon}_{l,k},\, \varepsilon_{\min}\}, \quad l \in \mathcal{I}_k, \qquad \sum_{l \in \mathcal{I}_k} \varepsilon_{l,k} \;=\; \varepsilon_{\mathrm{obs},k}. \tag{3.64}$$

If the floor causes $\sum_{l \in \mathcal{I}_k} \varepsilon_{l,k} > \varepsilon_{\mathrm{obs},k}$, we subtract the excess uniformly from indices with $\varepsilon_{l,k} > \varepsilon_{\min}$ until the inequality holds. This avoids a vanishing $\varepsilon_{l,k}$.

3.3.4 Lane Boundary Constraint

Here, we present admissible lane boundary conditions that ensure lane-keeping and allow lane changes to avoid obstacles.

Let the ego vehicle's position at time step k be $\mathbf{p}_k = [x_k \; y_k]^\top \in \mathbb{R}^2$. At each time step, we define two parallel lines that bound a convex corridor and require the next position $\mathbf{p}_{k+1}$ to lie within that convex corridor. To allow for potential lane changes and obstacle avoidance along the current path, we construct a wide yet valid road corridor by selecting all outermost lanes that travel in the same direction as the reference trajectory. That is, using the HD map and the reference sample at step k:

- From the reference point at k, query the neighboring drivable lanes on both sides.
- Retain only those lanes whose forward vectors satisfy a same-direction test, e.g., $\mathbf{f}_{\mathrm{k}}^\top \mathbf{f}_{\mathrm{neighbor}} \geq 0$.

Let $\mathbf{p}_L[k], \mathbf{p}_R[k] \in \mathbb{R}^2$ denote the outer world-frame points of the left and right edges of the road at the k^{th} time step, obtained by shrinking each edge inward by a small safety margin $m_{\mathrm{safety}} > 0$ to avoid driving directly on lane extremes.

Define the across-road normal and the corresponding signed normal projections of the edge points

$$\mathbf{a}_k \;=\; \mathbf{p}_L[k] - \mathbf{p}_R[k], \qquad v_L \;=\; \mathbf{a}_k^\top \mathbf{p}_L[k], \qquad v_R \;=\; \mathbf{a}_k^\top \mathbf{p}_R[k]. \tag{3.65}$$

and the corresponding bounds

$$g_{\min}[k] \;=\; \min\{v_L,\, v_R\}, \qquad g_{\max}[k] \;=\; \max\{v_L,\, v_R\}. \tag{3.66}$$

To ensure that the entire vehicle remains inside the lane, shrink the feasible region on both sides as follows:

$$m_{\mathrm{tot}} \;=\; \frac{w_{\mathrm{veh}}}{2}, \tag{3.67}$$

where w_{veh} is the vehicle width. Note that the base safety margin from lane extremes has already been applied when constructing $\mathbf{p}_L[k]$ and $\mathbf{p}_R[k]$. Moving each boundary line inward by m_{tot} perpendicularly is equivalent to shifting the right-hand side of the linear constraints by

$$\delta_k \;=\; m_{\mathrm{tot}}\, \|\mathbf{a}_k\|_2. \tag{3.68}$$

This follows from the signed distance formula to a line $\mathbf{a}_k^\top \mathbf{p} = g$ given by

$$\operatorname{dist}(\mathbf{p},\ \{\mathbf{a}_k^\top \mathbf{x} = g\}) \ = \ \frac{\mathbf{a}_k^\top \mathbf{p} - g}{\|\mathbf{a}_k\|_2}, \tag{3.69}$$

and shows that requiring at least m_{tot} meters of clearance from each boundary is exactly the same as replacing g by $g \pm m_{\text{tot}} \|\mathbf{a}_k\|_2$ [84].

Apply the symmetric inset to both sides:

$$g_{\text{lo}}[k] \ = \ g_{\min}[k] + \delta_k, \qquad g_{\text{hi}}[k] \ = \ g_{\max}[k] - \delta_k. \tag{3.70}$$

Finally, define the lane-boundary set at step k as

$$\mathcal{LB}_k \ := \ \Big\{ \mathbf{p} \in \mathbb{R}^2 \quad g_{\text{lo}}[k] \ \le \ \mathbf{a}_k^\top \mathbf{p} \ \le \ g_{\text{hi}}[k] \Big\}.\Bigg(\tag{3.71}$$

3.3.5 Trajectory Planning OCP

We construct a quadratic cost function that penalizes (i) state-tracking error, (ii) control effort, and (iii) control variation over the prediction horizon N. Let $\mathbf{x}_k \in \mathbb{R}^{n_x}$, $\mathbf{u}_k = [\, a_k,\ \delta_k \,]^\top \in \mathbb{R}^2$, and let $\mathbf{p}_k \in \mathbb{R}^2$ denote the planar position extracted from $\mathbf{x}_k$. Feasibility is enforced through box constraints, velocity, acceleration, jerk, and steering rate limits, lane–boundary set membership $\mathbf{p}_k \in \mathcal{LB}_k$, and the collision avoidance chance constraints introduced above. The notation for the obstacle chance constraint and the set lane-boundary is consistent with the previous formulation (same $\mathcal{LB}_k$, $\Theta_k^{(j)}$ and $\mathbf{p}_{o,k}^{(j)}$).

$$\begin{aligned}
\min_{\{\mathbf{x}_k, \mathbf{u}_k\}} \quad & \sum_{k=0}^{N-1} \big(\mathbf{x}_k - \mathbf{x}_k^{\text{ref}}\big)^\top W \big(\mathbf{x}_k - \mathbf{x}_k^{\text{ref}}\big) + (\mathbf{x}_N - \mathbf{x}_N^{\text{ref}})^\top W_N (\mathbf{x}_N - \mathbf{x}_N^{\text{ref}}) \\
& \quad + \sum_{k=0}^{N-1} \mathbf{u}_k^\top R \mathbf{u}_k + \sum_{k=1}^{N-1} (\mathbf{u}_k - \mathbf{u}_{k-1})^\top S (\mathbf{u}_k - \mathbf{u}_{k-1}) \\
\text{s.t.} \quad & \mathbf{x}_{k+1} \ = \ \Phi_{T_s}\big(\mathbf{x}_k, \mathbf{u}_k\big), \qquad k = 0, \dots, N-1 \\
& v_{\min} \le v_k \le v_{\max}, \qquad k = 0, \dots, N \\
& a_{\min} \le a_k \le a_{\max}, \quad \delta_{\min} \le \delta_k \le \delta_{\max}, \qquad k = 0, \dots, N-1 \\
& |a_k - a_{k-1}| \le j_a T_s, \qquad |\delta_k - \delta_{k-1}| \le \alpha_\delta T_s, \qquad k = 1, \dots, N-1 \\
& \mathbf{p}_k \in \mathcal{LB}_k, \qquad k = 0, \dots, N \\
& \Pr\Big(\ \Theta_k^{(j)}\big(\mathbf{p}_k - \mathbf{p}_{o,k}^{(j)}\big) \ _2 \ \ge \ 1\Big) \ \ge \ 1 - \varepsilon_k^{(j)}, \qquad k = 0, \dots, N, \ \ j = 1, \dots, N_o \\
& \mathbf{x}_0 \ = \ \mathbf{x}^{\text{meas}}
\end{aligned} \tag{3.72}$$

$$N \ = \ \frac{T_h}{T_s}, \qquad T_h > 0 \text{ (horizon duration)}, \ \ T_s > 0 \text{ (sampling time)}, \ \ N \in \mathbb{N},$$

with Φ_{T_s} being the RK4 one–step map of $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$. Here $W, W_N \succ 0$ penalizes state tracking error, $R \succ 0$ penalizes input effort, and $S \succ 0$ penalizes input variation.

Solution We implement the OCP in `CasADi` and solve it with an interior-point method (IPOPT) using the following policies:

- *Receding horizon:* At each time step, if the previous solution is feasible, we warm-start the optimization problem by advancing it using the dynamics model.
- *Lane corridor:* $\mathcal{LB}_k$ is precomputed from the HD map.
- *Chance constraints:* $\Theta_k^{(j)}$ and $\mathbf{p}_{o,k}^{(j)}$ are updated online from tracker means and covariance matrices.

Fallback policy If the optimization is infeasible, the ego vehicle is required to decelerate along the current path until feasibility is restored. This can result in the vehicle coming to a complete stop.

3.4 Experimental Results

In this section, we present three experiments that demonstrate the effectiveness of our uncertainty aware MPC framework, along with a fourth that integrates the misbehavior-detection framework presented in Chapter 2. The first is a single-vehicle perception case with adversarial agents that can perturb their own paths, where the ego vehicle attempts a left turn at a T-junction. The second is a cooperative-perception scenario at a T-junction, where the ego vehicle again attempts a left turn in the presence of oncoming, occluded agent. The third and fourth involve a four-way intersection where the ego vehicle executes a right turn in the presence of occluded obstacles, and then the same maneuver with both occluded and spoofed obstacles.

3.4.1 Datasets

We utilize the same dataset generated for the object detection task in Chapter 2.

3.4.2 Scenario Description - Single Vehicle Perception in a T-Junction

The scenario consists of an ego vehicle executing a left turn at a T-Junction. As the ego vehicle enters the intersection, it detects a non-communicating oncoming vehicle, denoted NC-AV-2, approaching from its left on the adjacent lane on the opposite side. This oncoming vehicle is observed to be in a right turn only lane. Under normal behavior, it is expected to yield to the ego vehicle, which has already initiated the left turn, and to complete its right turn without encroaching on the ego vehicle's path. However, in this scenario, the oncoming vehicle exhibits adversarial behavior: instead of yielding, it deviates from its nominal path and executes an aggressive maneuver that encroaches into the ego vehicle's path. This adversarial action results in a potential collision. The ego vehicle must continuously predict the oncoming vehicle's future trajectory and take appropriate action to avoid collision while adhering to track boundaries.

Figure 3.2 corresponds to the single-vehicle perception scenario without adversarial movement from NC-AV-2. The green path denotes the expected trajectory of the non-connected vehicle.

Figure 3.3 depicts a single-vehicle perception scenario in which NC-AV-2 executes an adversarial maneuver that encroaches on the ego vehicle's lane. Within the intersection region $\mathcal{R}$, we randomly select four waypoints $\{s_i\}_{i=1}^4$ along the nominal path $p^{\text{nom}}(s)$. At each selected waypoint, we randomly draw an independent lateral deviation $d_i \sim \mathcal{U}([0.4, 0.6])$ m and apply it using

$$q_i \;=\; p^{\text{nom}}(s_i) \;+\; \sigma_i \, d_i \, e_\perp(s_i), \qquad \sigma_i \in \{-1, +1\}, \tag{3.73}$$

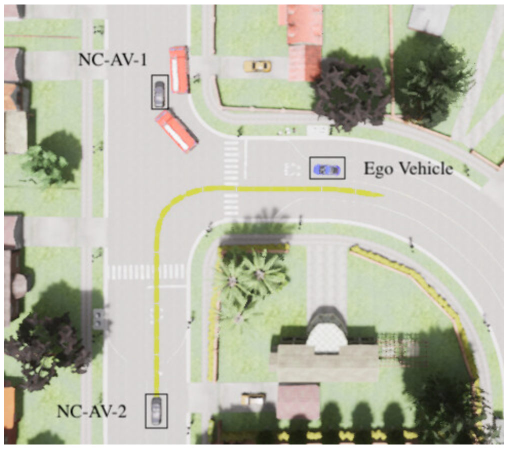


Figure 3.2: Single-vehicle perception: Without adversarial behavior from NC-AV-2

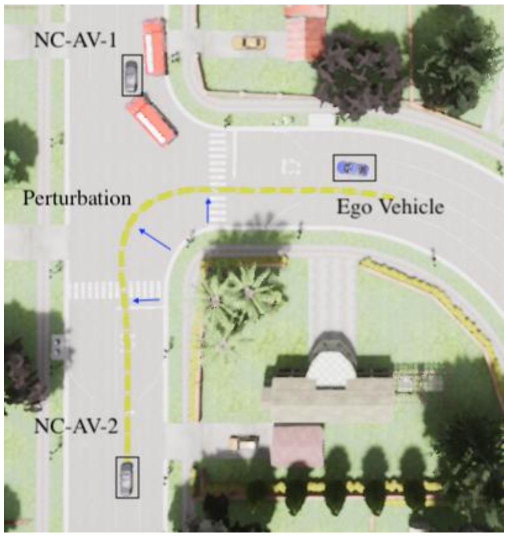


Figure 3.3: Single Vehicle Perception: With adversarial behavior from NC-AV-2

where $e_{\perp}(s)$ is any unit lateral direction orthogonal to the path heading at s and σ_i is chosen to bias the deviation toward the ego vehicle's lane.
Note. When the adversarial lateral deviation d_i exceeds 0.6 m, we observe increased infeasibilities due to lane–boundary constraints. Accordingly, we cap d_i when we sample from the uniform distribution $\mathcal{U}([0.4, 0.6])$ m.

3.4.3 Scenario Description - Cooperative Perception in a T-Junction

This scenario is set at a T-junction involving three connected automated vehicles (CAVs), including the ego vehicle. The ego vehicle is preparing to enter the main intersection, intending to execute a left turn. A non-connected, occluded vehicle, denoted NC-AV, is approaching from the right (relative to the ego vehicle's heading) and also intends to make a left turn through the intersection. Due to the presence of occlusions, the non-connected vehicle has no visibility of the ego vehicle and thus incorrectly assumes it has the right of way as it enters the intersection. The ego vehicle leverages cooperative perception by receiving shared perception data from CAV-1 and

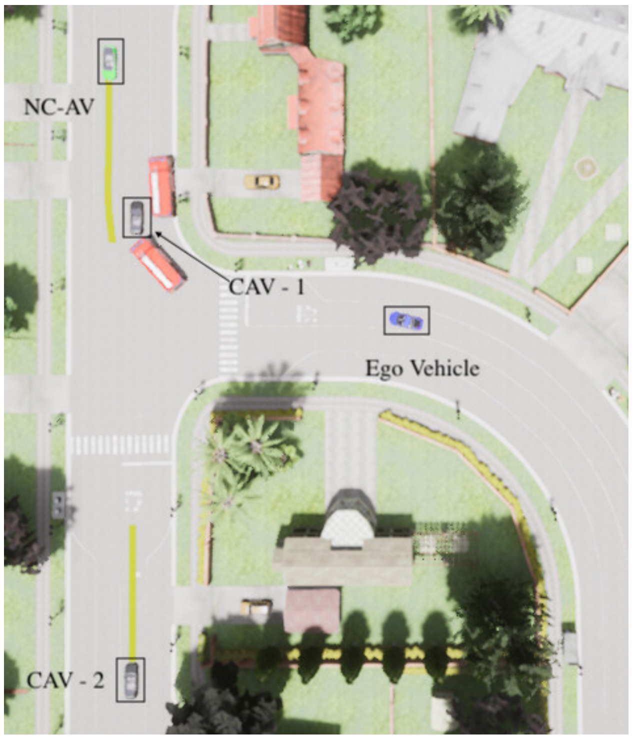


Figure 3.4: Cooperative Perception at a T-junction: Connected vehicles CAV-1 and CAV-2 share their observations of the non-connected vehicle NC-AV with the ego vehicle.

CAV-2. Using this information, the ego vehicle is able to infer the state and predicted intent of the occluded vehicle. As the first vehicle to enter the intersection, it is necessary for the ego vehicle to execute a non-conservative maneuver, rather than waiting, in order to proceed safely and efficiently. Figure 3.4 shows the cooperative perception setup in which CAV-1 and CAV-2 relay their observations of the NC-AV to the ego vehicle.

3.4.4 Scenario Description - Cooperative Perception in an Intersection

This scenario involves 4 CAVs, including the ego vehicle, and a non-connected stationary obstacle. As the ego vehicle approaches the intersection intending to make a right turn, the stationary obstacle remains occluded and is not directly observable by the ego vehicle. To safely execute the turn, the ego vehicle must perform a lane change maneuver to avoid the obstacle and subsequently return to its original lane. Figure 3.5 shows the lane–change case in which the ego vehicle attempts a right turn but, upon receiving information from CAV-1 about a stationary vehicle, adapts its plan accordingly.

3.4.5 Scenario Description - Misbehavior Detection in an Intersection

This scenario involves four CAVs, including the ego vehicle, and a non-connected stationary obstacle. The goal matches the previously described intersection case where the ego vehicle approaches the intersection intending to turn right, while a stationary obstacle lies in its path and is not directly observable to the ego vehicle. Additionally, one of the connected vehicle acts as an attacker, sending spoofed perception to the ego vehicle. The spoofed box is placed ahead of the stationary obstacle, forcing the ego vehicle to take additional steps to avoid it.

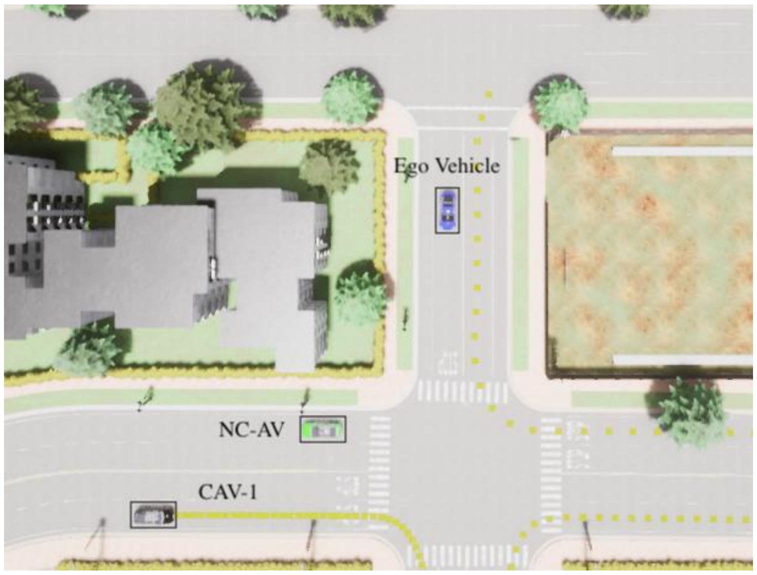
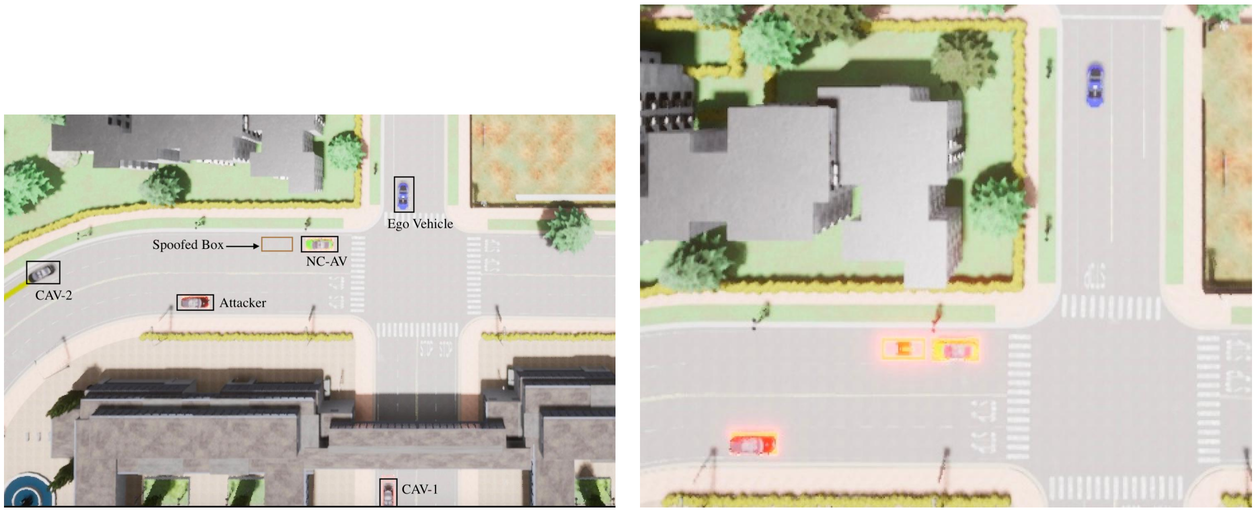


Figure 3.5: Intersection Scenario with Lane Change Based Obstacle Avoidance

Figure 3.6a and 3.6b show the scenario. Figure 3.6b shows the perception–module output from the simulation.



(a) Intersection Scenario for Misbehavior Detection (b) Perception Capture

Figure 3.6: Misbehavior Detection Scenario

3.4.6 Simulation Parameters

Vehicle parameters are listed in Table 3.1(a) together with the physics–based bounds on acceleration and steering. Scenario–specific controller settings for the T–junction are reported in Table 3.1(b). For the Intersection scenario, we increase the horizon duration to 3 s and set $W = \text{diag}(2, 2.5, 1, 1)$ and $W_N = \text{diag}(2, 3, 1, 1)$ instead of the T–junction scenario values. We

Table 3.1: Vehicle and Simulation Parameters

(a) Vehicle Parameters

Parameter	**Value**	**Parameter**	**Value**
$l_{\text{wheelbase}}$	2.875 m	l_{ego}	4.72 m
w_{ego}	2.089 m	δ	$[-1.22,\ 1.22]$ rad
a	$[-6,\ 4.5]$ m/s^2	v	$[0,\ 10]$ m/s

(b) Simulation Parameters (T–junction)

Parameter	**Value**	**Parameter**	**Value**
T_s	0.05 s	N	40
W	$\text{diag}(2, 4, 1, 1)$	W_N	$\text{diag}(2, 7, 1, 1)$
R	$\text{diag}(0.1, 0.1)$	S	$\text{diag}(0.01, 0.01)$
α_δ	0.3 rad/s	j_a	1.0 m/s^3
$\varepsilon_{\text{total}}$	0.1	$\varepsilon_{\min}$	0.001

use a constant process–noise covariance for motion prediction and uncertainty propagation:

$$Q_{(x,y,z)} = \text{diag}(10^{-4},\ 10^{-4},\ 10^{-4})\ \text{m}^2 \quad (\text{std. } 1\ \text{cm}), \qquad Q_\psi = 0.01\ \text{rad}^2,$$

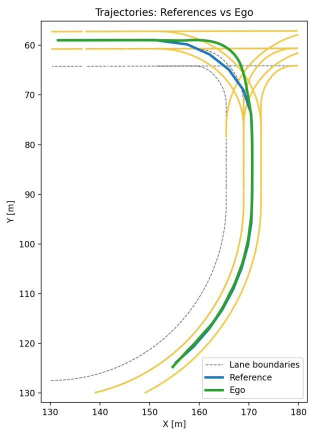
$$Q_{(h,w,l)} = \text{diag}(0.01,\ 0.01,\ 0.01)\ \text{m}^2, \qquad Q_\omega = 1\ (\text{rad/s})^2.$$

Here $Q_{(x,y)}$ is used for obstacle motion prediction and uncertainty propagation in the collision avoidance module.

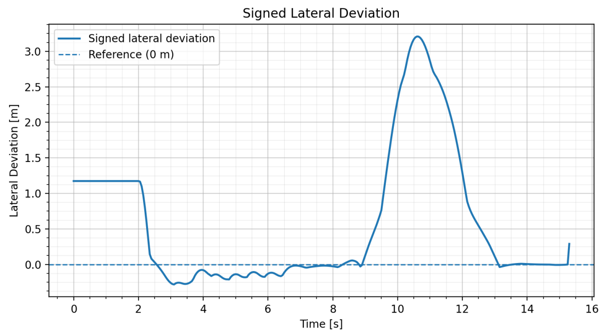
3.4.7 Quantitative Results - Single Vehicle Perception in a T-Junction

Without Adversarial Behavior

Figure 3.7a and 3.7b show, respectively, the ego vehicle's path versus the reference and the corresponding signed lateral deviation from the reference trajectory. Figure 3.8 shows the ego vehicle's speed over time.



(a) Trajectories in the CARLA Frame



(b) Signed Lateral Deviation

Figure 3.7: Ego vehicle path versus reference (left) and corresponding lateral deviation (right).

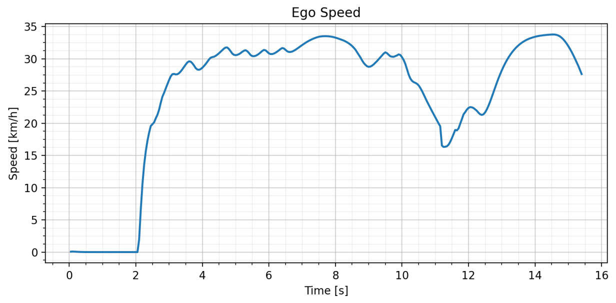


Figure 3.8: Ego Vehicle Speed in km/h

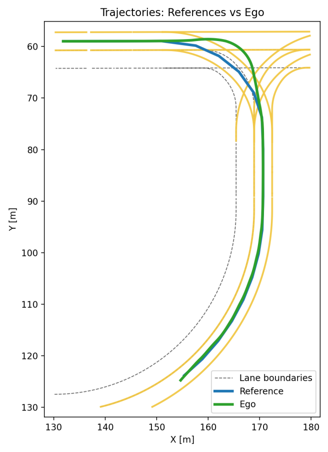
With Adversarial Behavior

Figure 3.9a and 3.9b show, respectively, the ego vehicle's path versus the reference and the corresponding signed lateral deviation with an adversarial oncoming vehicle. As expected, the signed lateral deviation is higher than in the no-perturbation case. Figure 3.10 shows the ego vehicle's speed over time for the same adversarial case.

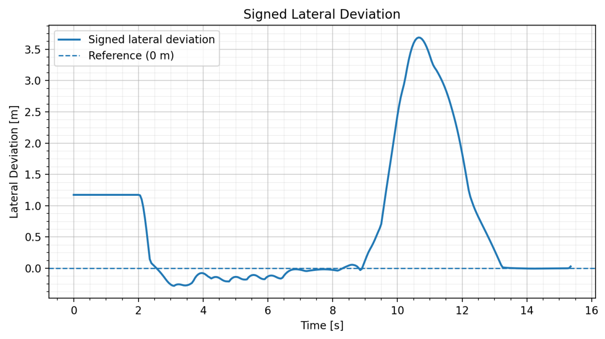
Table 3.2 summarizes the signed lateral deviation metrics and feasibility for both scenarios. We assess feasibility by counting the control steps at which the MPC solver fails to produce a solution, causing the controller to switch to the fallback policy. We also report rates.

Table 3.2: Comparison of signed lateral deviation metrics (nominal versus adversarial).

Scenario	**Mean [m]**	**RMS [m]**	**Max Absolute [m]**	**Feasibility (%)**
Without adversarial	0.66	1.11	3.28	98.40
With adversarial	0.74	1.27	3.69	98.25



(a) Trajectories in the CARLA Frame



(b) Signed Lateral Deviation

Figure 3.9: Ego vehicle path versus reference (left) and corresponding lateral deviation (right) with an adversarial oncoming vehicle.

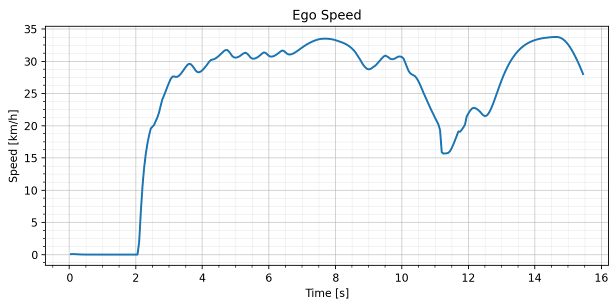


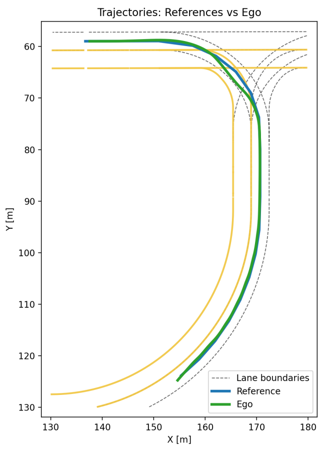
Figure 3.10: Ego Vehicle Speed in km/h: Adversarial Case

3.4.8 Quantitative Results - Cooperative Perception in a T-Junction

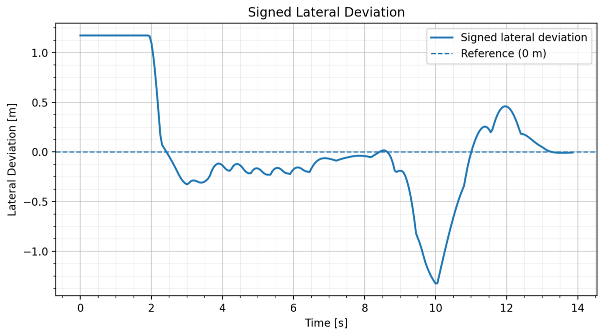
We evaluate a left-turn maneuver in which an oncoming non-connected vehicle NC-AV is initially occluded from the ego vehicle's FoV. Two connected vehicles, CAV-1 and CAV-2, transmit their detections of the NC-AV which the ego vehicle utilizes to execute a collision-free left turn. Figure 3.11a and 3.11b show, respectively, the ego vehicle path versus the reference and the corresponding signed lateral deviation. Figure 3.12 shows the ego vehicle's speed over time. Table 3.3 summarizes the signed lateral deviation metrics and feasibility.

Table 3.3: Signed lateral deviation metrics and feasibility for the cooperative-perception case.

Scenario	**Mean [m]**	**RMS [m]**	**Max Absolute [m]**	**Feasible rate [%]**
Cooperative perception	0.394	0.578079	1.324385	100.0



(a) Trajectories in the CARLA Frame



(b) Signed Lateral Deviation

Figure 3.11: Ego vehicle path versus reference (left) and corresponding lateral deviation (right) under cooperative perception.

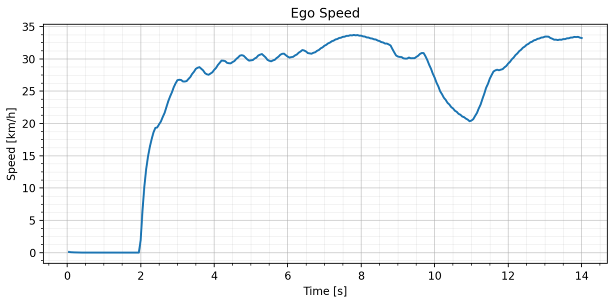


Figure 3.12: Ego Vehicle Speed in km/h

3.4.9 Quantitative Results - Cooperative Perception in an Intersection

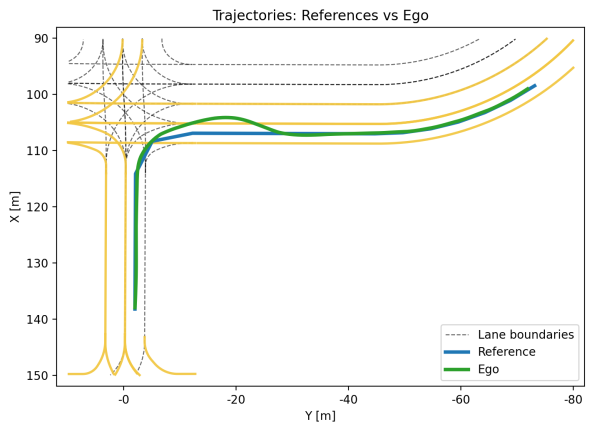
An ego vehicle approaches a four-way intersection intending to turn right. A nearby connected vehicle CAV-1 reports a stationary obstacle in the ego vehicle's lane. The ego vehicle must execute a lane-change overtake around the obstacle and then merge back into the original lane.

We evaluate two placements of the non-cooperative automated vehicle NC-AV:

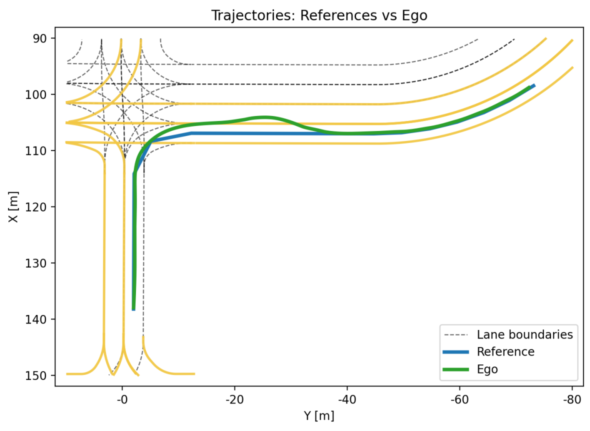
- near the intersection, and
- far, approximately $\approx 2\,\ell_{\text{ego}}$, from the intersection.

This comparison tests whether the ego vehicle can complete the lane change and merge irrespective of the obstacle's location along the path. Figure 3.13 shows the results for both intersection scenarios. Regardless of the occluded vehicle's position, the ego vehicle avoids a collision by successfully executing an avoidance-and-merge-back maneuver using perception information from CAV-1. Table 3.4 presents the lateral deviation from the reference path for both cases.

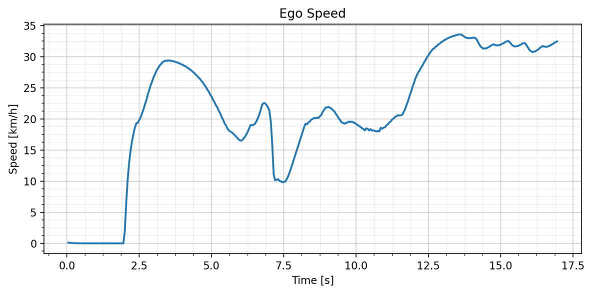
Both cases were solved with 100% feasibility.



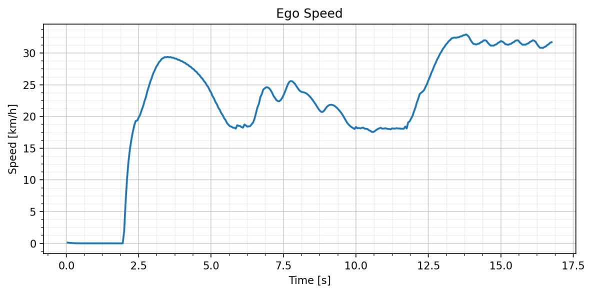
(a) Trajectories: NC-AV near Intersection



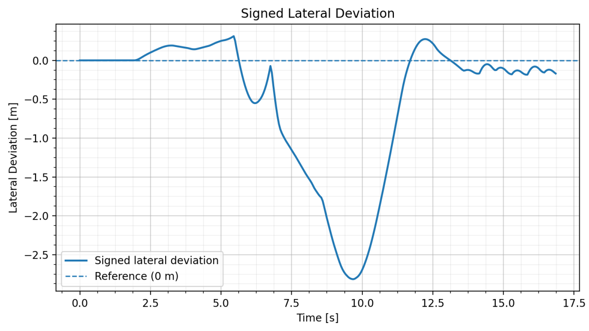
(b) Trajectories: NC-AV far from Intersection



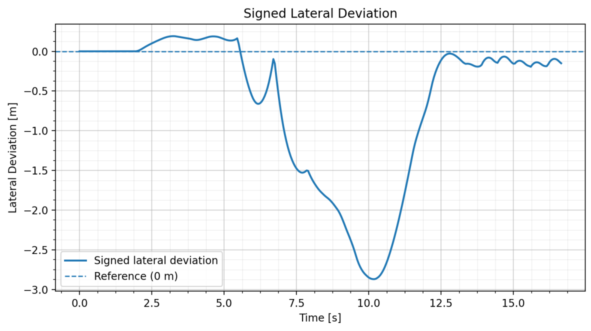
(c) Ego Speed: NC-AV near Intersection



(d) Ego Speed: NC-AV far from Intersection



(e) Signed Lateral Deviation: Near Case



(f) Signed Lateral Deviation: Far Case

Figure 3.13: Top: reference versus ego vehicle trajectories; Middle: ego vehicle speed; Bottom: signed lateral deviation

Table 3.4: Lateral Deviation Statistics [m]

Case	**Mean [m]**	**RMS [m]**	**Max Absolute [m]**
NC-AV near intersection	0.5844	1.006592	2.815058
NC-AV far from intersection	0.694405	1.127395	2.869673

3.4.10 Quantitative Results - Misbehavior Detection in an Intersection

As shown in Figure 3.14, the attacker's trust initially rises because it is the only CAV within the ego vehicle's communication range. After CAV-2 enters range, geometric inconsistencies cause the attacker's trust to decrease. The attacker is flagged before the ego vehicle reaches the intersection. As such, it does not influence the ego vehicle's behavior, and the maneuver proceeds as in the "NC-AV Near Intersection" scenario.

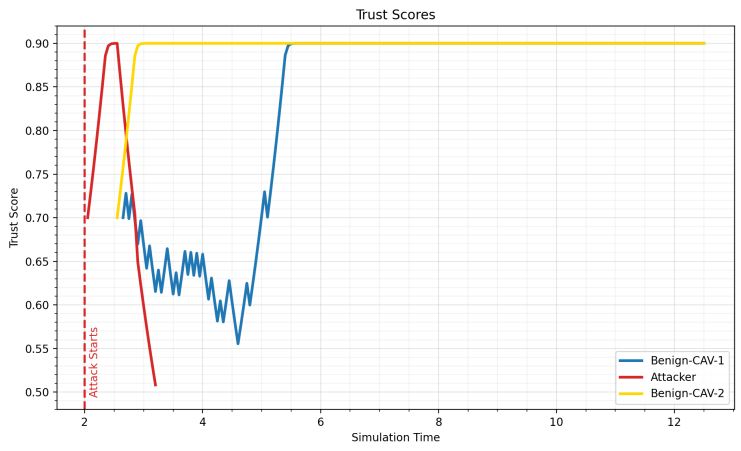


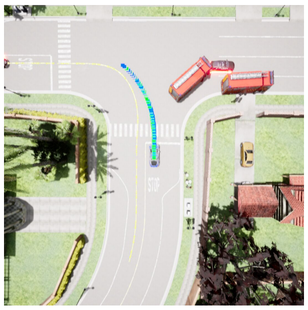
Figure 3.14: Trust Scores Evolution over Time - 4 Way Intersection Scenario

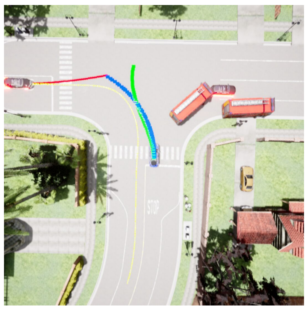
3.4.11 Qualitative Results

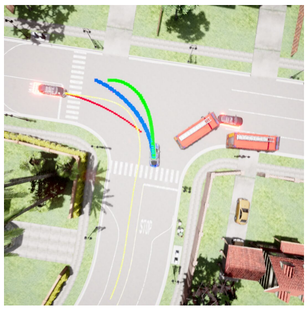
The blue path denotes the reference trajectory from the ego vehicle's current pose which is the path the ego vehicle is expected to follow while the green path is the ego vehicle's current optimal trajectory over the prediction horizon. The ego vehicle also predicts trajectories for the DOs in its tracked-object list, shown as red paths. In the figures, some predicted DO trajectories may appear to form loops at certain time steps, typically when DOs execute high-curvature turns. This effect arises from large yaw-rate uncertainty and the use of the constant-turn-rate-and-velocity (CTRV) model, whose accuracy degrades over longer horizons. This is a known limitation of physics-based kinematic models. For more accurate, multi-modal forecasts, supervised learning–based motion-prediction methods can be used.

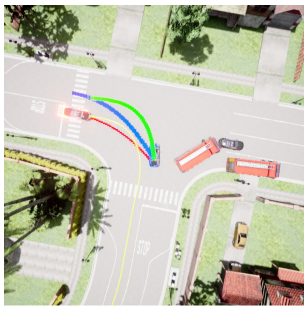
T-Junction Scenario: Single Vehicle Perception

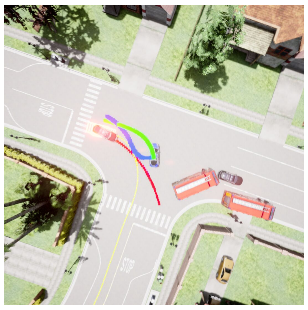
Qualitative results for single-vehicle perception in the T-junction scenario without adversarial behavior by the oncoming vehicle are shown in Figure 3.15. Under adversarial behavior, the ego vehicle follows a similar yet wider path, as evidenced by trajectory differences and increased lateral deviation. See Figure 3.7 for the nominal case and Figure 3.9 for the adversarial case.

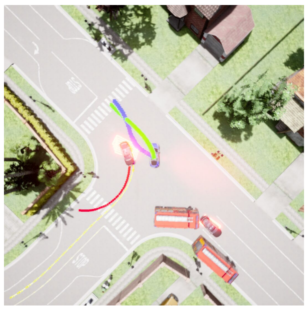

(a) $T = 8.0$ s

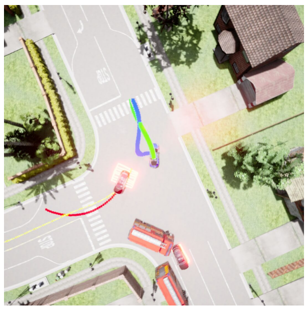

(b) $T = 8.5$ s


(c) $T = 9.0$ s


(d) $T = 9.5$ s


(e) $T = 10.0$ s


(f) $T = 10.5$ s


(g) $T = 11.0$ s

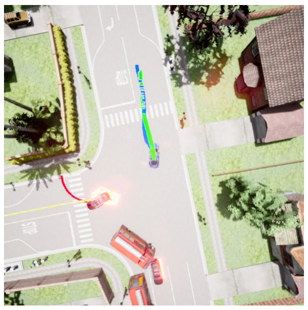

(h) $T = 11.5$ s

Figure 3.15: T-Junction Scenario: Single Vehicle Perception

T-Junction Scenario: Cooperative Perception

Qualitative results for cooperative perception in the T-junction scenario are shown in Figure 3.16. As shown in Figure 3.11, the ego vehicle deviates slightly from its reference path after CAV-1 and CAV-2 provide information about an oncoming vehicle from the right as seen from the ego vehicle's perspective.

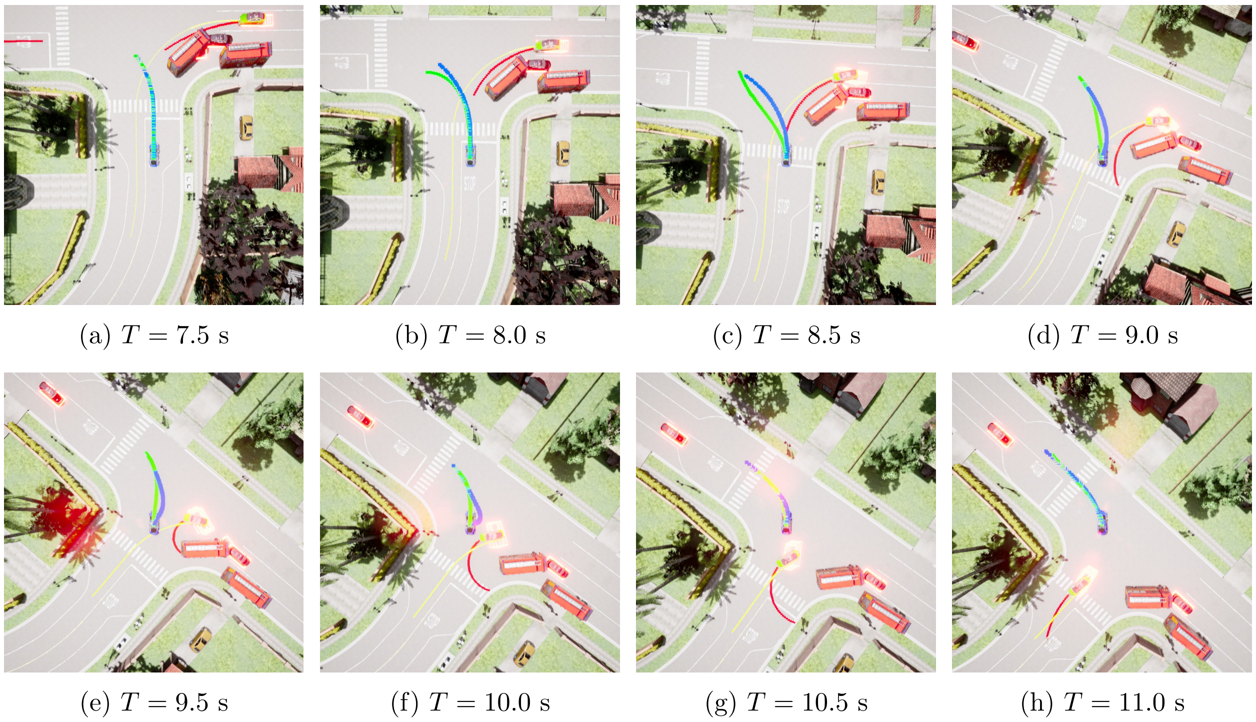
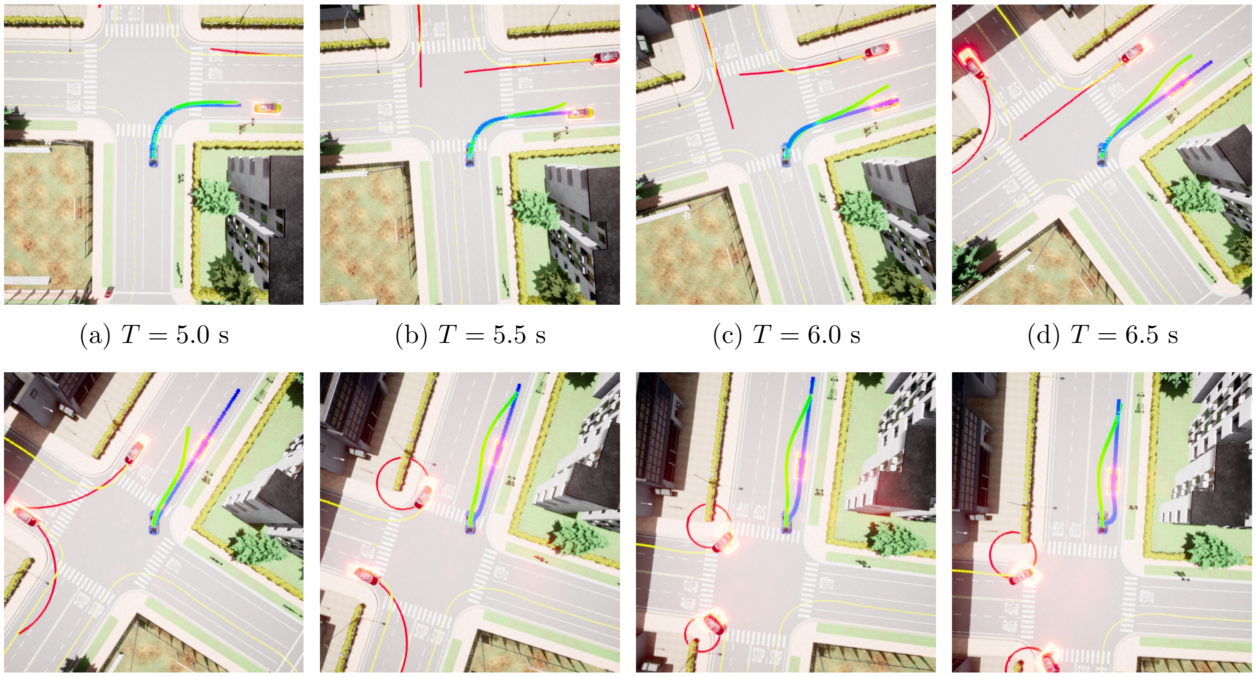

(a) $T = 7.5$ s (b) $T = 8.0$ s (c) $T = 8.5$ s (d) $T = 9.0$ s
(e) $T = 9.5$ s (f) $T = 10.0$ s (g) $T = 10.5$ s (h) $T = 11.0$ s

Figure 3.16: T-Junction Scenario: Cooperative Perception

Intersection Scenario: Cooperative Perception

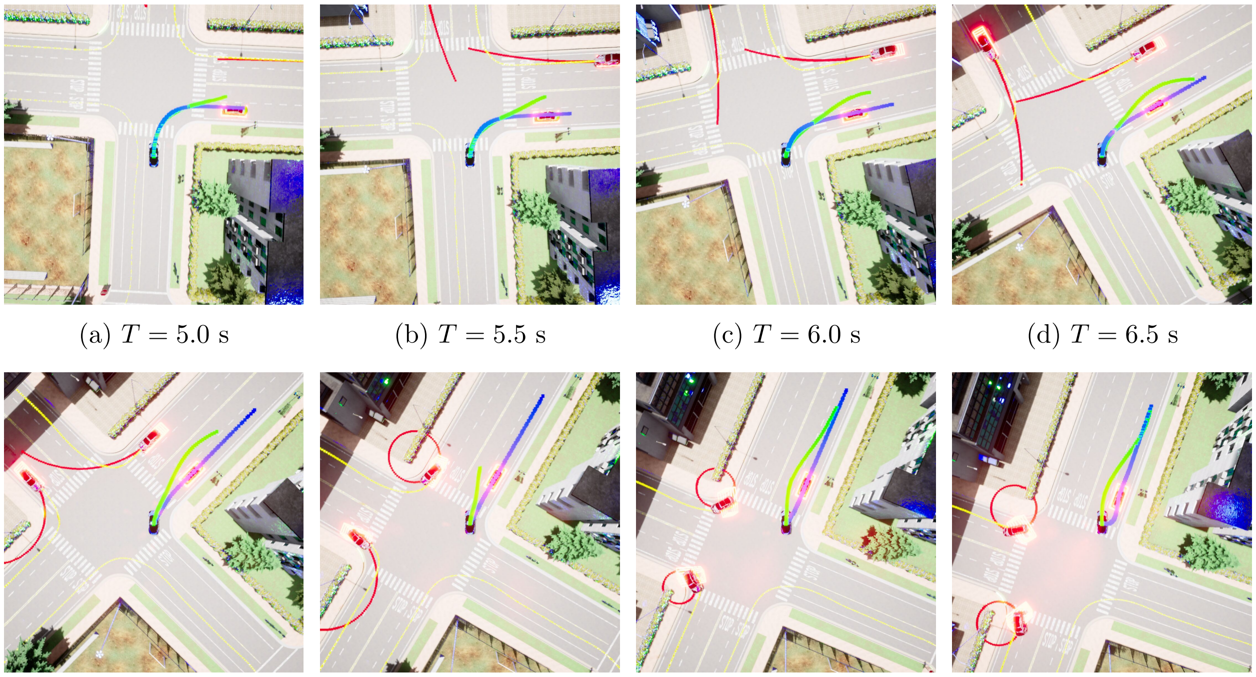
Figure 3.17 and 3.18 illustrate scenarios in which the static obstacle NC-AV is positioned far from and near the intersection, respectively. In both cases, the ego vehicle successfully avoids the stationary obstacle and completes the lane change, irrespective of the NC-AV positioning.



(a) $T = 5.0$ s (b) $T = 5.5$ s (c) $T = 6.0$ s (d) $T = 6.5$ s

(e) $T = 7.0$ s (f) $T = 7.5$ s (g) $T = 8.0$ s (h) $T = 8.5$ s

Figure 3.17: Intersection Scenario: NC-AV Away from Intersection



(a) $T = 5.0$ s (b) $T = 5.5$ s (c) $T = 6.0$ s (d) $T = 6.5$ s

(e) $T = 7.0$ s (f) $T = 7.5$ s (g) $T = 8.0$ s (h) $T = 8.5$ s

Figure 3.18: Intersection Scenario: NC-AV Near to Intersection

3.5 Discussion: Challenges Faced

Cooperative-perception pipeline design The OpenCDA framework, while robust for platooning operations and single-vehicle perception, lacks object-level cooperative perception functionality. Our primary issue was specifying and implementing the missing modules of the cooperative perception pipeline. The following modules were implemented to ensure successful perception information communication: (i) an interface for OpenCOOD models, (ii) coordinate transforms between the ego and CAV frames, (iii) object association and fusion across CAVs, and (iv) multi-object tracking.

Data Collection Although OpenCDA supports multivehicle logging, reproducible scenario design proved difficult. In high-density traffic, PID tuning was required as gridlock was observed with default controller gains. During such jams, the simulator continued recording identical frames across several consecutive timesteps, polluting the dataset with duplicate frames and inducing class imbalance. Scenario-specific PID gains, fixed vehicle classes, and spawn regions were some of the steps taken to mitigate the above mentioned issue.

Lane-keeping and Lane Change Unlike the Frenet case, where the centerline and nominal track widths keep the ego vehicle within the drivable region, our Cartesian formulation required an explicit drivable region to ensure lane-keeping while also permitting potential lane changes in case of obstacle avoidance. We adopt an MPCC-style corridor in Cartesian coordinates [84] and augment to the extreme same-direction lane boundaries (from the HD map) so the optimizer can switch lanes. This keeps a single coordinate frame for collision checking and cooperative detections, but the resulting constraints are non-convex and therefore less straightforward. In our experiments, we did not observe infeasibility during lane changes. However, at higher traffic densities the feasible set may shrink and solutions can become infeasible. If that occurs, we may need to modify the fallback policy or move to an augmented Frenet-Cartesian state with the corresponding constraints [85].

3.6 Conclusion

We designed a perception-uncertainty-aware MPC pipeline in a cooperative-perception setting. Object-level cooperative perception is implemented via the uncertainty-weighted box-level fusion approach described in Chapter 1. Fused object information is maintained on the ego vehicle's tracked-object list using an modified AB3DMOT-based tracker. The tracked objects, along with their associated uncertainties, are then fed to the MPC algorithm for collision avoidance. We investigated performance in four scenarios: a left turn at a T-junction with both single-vehicle and connected-vehicle cases) and a right turn at a four-way intersection with both real and spoofed obstacle vehicle cases. As the results indicate, the ego vehicle successfully avoids both dynamic and static obstacles. We also observed 100% feasibility in the intersection scenario, regardless of the static obstacle's position. We also evaluated a 4-way intersection in which one CAV transmitted a spoofed object to the ego vehicle and verified the effectiveness of the algorithm described in Chapter 2.

For future work, we will consider multiple intentions for dynamic obstacles to produce safer and less conservative trajectories. We may also consider learning-based motion prediction methods instead of relying on physics-based models for higher accuracy. In addition, we will explore alternative risk allocation schemes across the prediction horizon. Another potential direction is prioritizing received perception information, since not all perceived data are equally useful for trajectory planning.

Bibliography

[1] M. R. Ansari, J.-P. Monteuuis, J. Petit, and C. Chen, "V2x misbehavior and collective perception service: Considerations for standardization," in *2021 IEEE Conference on Standards for Communications and Networking (CSCN)*, pp. 124–129, 2021.

[2] R. Xu, X. Xia, J. Li, H. Li, S. Zhang, Z. Tu, Z. Meng, H. Xiang, X. Dong, R. Song, H. Yu, B. Zhou, and J. Ma, " V2V4Real: A Real-World Large-Scale Dataset for Vehicle-to-Vehicle Cooperative Perception ," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (Los Alamitos, CA, USA), pp. 13712–13722, IEEE Computer Society, June 2023.

[3] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, p. 1321–1330, JMLR.org, 2017.

[4] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, (Red Hook, NY, USA), p. 6405–6416, Curran Associates Inc., 2017.

[5] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of The 33rd International Conference on Machine Learning* (M. F. Balcan and K. Q. Weinberger, eds.), vol. 48 of *Proceedings of Machine Learning Research*, (New York, New York, USA), pp. 1050–1059, PMLR, 20–22 Jun 2016.

[6] M. Pitropov, C. Huang, V. Abdelzad, K. Czarnecki, and S. Waslander, "Lidar-mimo: Efficient uncertainty estimation for lidar-based 3d object detection," in *2022 IEEE Intelligent Vehicles Symposium (IV)*, pp. 813–820, 2022.

[7] M. Havasi, R. Jenatton, S. Fort, J. Z. Liu, J. Snoek, B. Lakshminarayanan, A. M. Dai, and D. Tran, "Training independent subnetworks for robust prediction," in *International Conference on Learning Representations*, 2021.

[8] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12689–12697, 2019.

[9] A. Rauch, F. Klanner, R. Rasshofer, and K. Dietmayer, "Car2x-based perception in a high-level fusion architecture for cooperative perception systems," in *2012 IEEE Intelligent Vehicles Symposium*, pp. 270–275, 2012.

[10] E. Arnold, M. Dianati, R. de Temple, and S. Fallah, “Cooperative perception for 3d object detection in driving scenarios using infrastructure sensors,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 3, pp. 1852–1864, 2022.

[11] Q. Chen, X. Ma, S. Tang, J. Guo, Q. Yang, and S. Fu, “F-cooper: feature based cooperative perception for autonomous vehicle edge computing system using 3d point clouds,” in *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*, SEC ’19, (New York, NY, USA), p. 88–100, Association for Computing Machinery, 2019.

[12] F. S. Roza, M. Henne, K. Roscher, and S. Günnemann, “Assessing box merging strategies and uncertainty estimation methods in multimodel object detection,” in *Proceedings of the Computer Vision – ECCV 2020 Workshops, Part VI-16* (A. Bartoli and A. Fusiello, eds.), (Cham), pp. 3–10, Springer International Publishing, 2020.

[13] R. Solovyev, W. Wang, and T. Gabruseva, “Weighted boxes fusion: Ensembling boxes from different object detection models,” *Image and Vision Computing*, vol. 107, no. 104117, 2021.

[14] X. Weng, J. Wang, D. Held, and K. Kitani, “3d multi-object tracking: A baseline and new evaluation metrics,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10359–10366, 2020.

[15] N. Benbarka, J. Schröder, and A. Zell, “Score refinement for confidence-based 3d multi-object tracking,” in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8083–8090, 2021.

[16] D. Li, B. Liu, Z. Huang, Q. Hao, D. Zhao, and B. Tian, “Safe motion planning for autonomous vehicles by quantifying uncertainties of deep learning-enabled environment perception,” *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, pp. 2318–2332, 2024.

[17] Y. Liao, J. Xie, and A. Geiger, “Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 3, pp. 3292–3310, 2023.

[18] R. Xu, H. Xiang, Z. Tu, X. Xia, M.-H. Yang, and J. Ma, “V2x-vit: Vehicle-to-everything cooperative perception with vision transformer,” in *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIX*, (Berlin, Heidelberg), p. 107–124, Springer-Verlag, 2022.

[19] R. Xu, Z. Tu, H. Xiang, W. Shao, B. Zhou, and J. Ma, “CoBEVT: Cooperative bird’s eye view semantic segmentation with sparse transformers,” in *6th Annual Conference on Robot Learning*, 2022.

[20] Y.-C. Liu, J. Tian, N. Glaser, and Z. Kira, “When2com: Multi-agent perception via communication graph grouping,” in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4105–4114, 2020.

[21] R. Xu, H. Xiang, X. Xia, X. Han, J. Li, and J. 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)*, pp. 2583–2589, 2022.

[22] E. Hüllermeier and W. Waegeman, “Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods,” *Machine Learning*, vol. 110, 03 2021.

[23] D. Feng, L. Rosenbaum, and K. Dietmayer, “Towards safe autonomous driving: Capture uncertainty in the deep neural network for lidar 3d vehicle detection,” in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 3266–3273, 2018.

[24] D. Feng, Y. Cao, L. Rosenbaum, F. Timm, and K. Dietmayer, “Leveraging uncertainties for deep multi-modal object detection in autonomous driving,” in *2020 IEEE Intelligent Vehicles Symposium (IV)*, pp. 877–884, 2020.

[25] Y. Zhong, M. Zhu, and H. Peng, “Uncertainty-aware voxel based 3d object detection and tracking with von-mises loss,” *ArXiv*, vol. abs/2011.02553, 2020.

[26] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587, 2014.

[27] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 318–327, 2020.

[28] H. W. Kuhn, “The hungarian method for the assignment problem,” *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[29] X. Li, D. Liu, Y. Wu, X. Wu, L. Zhao, and J. Gao, “Fast-poly: A fast polyhedral algorithm for 3d multi-object tracking,” *IEEE Robotics and Automation Letters*, vol. PP, pp. 1–8, 11 2024.

[30] K. Bernardin and R. Stiefelhagen, “Evaluating multiple object tracking performance: the clear mot metrics,” *EURASIP Journal on Image and Video Processing*, vol. 2008, no. 1, p. 246309, 2008.

[31] T.-H. Wang, S. Manivasagam, M. Liang, B. Yang, W. Zeng, and R. Urtasun, “V2vnet: Vehicle-to-vehicle communication for joint perception and prediction,” in *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II*, (Berlin, Heidelberg), p. 605–621, Springer-Verlag, 2020.

[32] Q.-Y. Zhou, J. Park, and V. Koltun, “Open3D: A modern library for 3D data processing,” *arXiv:1801.09847*, 2018.

[33] F. Küppers, J. Schneider, and A. Haselhoff, “Parametric and multivariate uncertainty calibration for regression and object detection,” in *European Conference on Computer Vision (ECCV) Workshops*, Springer, October 2022.

[34] Y. Kato and S. Kato, “A conditional confidence calibration method for 3d point cloud object detection,” in *2022 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1835–1844, 2022.

[35] F. Küppers, J. Kronenberger, A. Shantia, and A. Haselhoff, “Multivariate confidence calibration for object detection,” in *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

[36] E. T. S. I. (ETSI), *TS 103 324: Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Specification of the Collective Perception Service.* Sophia Antipolis Cedex, France: ETSI, June 2023.

[37] X. Liu, L. Yang, I. Alvarez, K. Sivanesan, A. Merwaday, F. Oboril, C. Buerkle, M. Sastry, and L. G. Baltar, "Miso- v: Misbehavior detection for collective perception services in vehicular communications," in *2021 IEEE Intelligent Vehicles Symposium (IV)*, pp. 369–376, 2021.

[38] L. Caltagirone, M. Bellone, L. Svensson, and M. Wahde, "Lidar–camera fusion for road detection using fully convolutional neural networks," *Robotics and Autonomous Systems*, vol. 111, pp. 125–131, 2019.

[39] Y. Li, A. W. Yu, T. Meng, B. Caine, J. Ngiam, D. Peng, J. Shen, Y. Lu, D. Zhou, Q. V. Le, A. Yuille, and M. Tan, "Deepfusion: Lidar-camera deep fusion for multi-modal 3d object detection," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17161–17170, 2022.

[40] Y. Yan, Y. Mao, and B. Li, "Second: Sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, 2018.

[41] H. Ngo, H. Fang, and H. Wang, "Cooperative perception with v2v communication for autonomous vehicles," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 9, pp. 11122–11131, 2023.

[42] Y. Li, D. Ma, Z. An, Z. Wang, Y. Zhong, S. Chen, and C. Feng, "V2x-sim: Multi-agent collaborative perception dataset and benchmark for autonomous driving," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10914–10921, 2022.

[43] J. Li, R. Xu, X. Liu, J. Ma, Z. Chi, J. Ma, and H. Yu, "Learning for vehicle-to-vehicle cooperative perception under lossy communication," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 4, pp. 2650–2660, 2023.

[44] Z. Song, F. Wen, H. Zhang, and J. Li, "A cooperative perception system robust to localization errors," in *2023 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1–6, 2023.

[45] S. Savazzi, M. Nicoli, M. Bennis, S. Kianoush, and L. Barbieri, "Opportunities of federated learning in connected, cooperative, and automated industrial systems," *IEEE Communications Magazine*, vol. 59, no. 2, pp. 16–21, 2021.

[46] R. Sedar, C. Kalalas, F. Vázquez-Gallego, L. Alonso, and J. Alonso-Zarate, "A comprehensive survey of v2x cybersecurity mechanisms and future research paths," *IEEE Open Journal of the Communications Society*, vol. 4, pp. 325–391, 2023.

[47] X. Sun, F. R. Yu, and P. Zhang, "A survey on cyber-security of connected and autonomous vehicles (cavs)," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6240–6259, 2022.

[48] Y. Takefuji, "Connected vehicle security vulnerabilities [commentary]," *IEEE Technology and Society Magazine*, vol. 37, no. 1, pp. 15–18, 2018.

[49] S. Bittl, A. A. Gonzalez, M. Myrtus, H. Beckmann, S. Sailer, and B. Eissfeller, "Emerging attacks on vanet security based on gps time spoofing," in *2015 IEEE Conference on Communications and Network Security (CNS)*, pp. 344–352, 2015.

[50] C. Sanders and Y. Wang, "Localizing spoofing attacks on vehicular gps using vehicle-to-vehicle communications," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 15656–15667, 2020.

[51] J. Sun, Y. Cao, Q. A. Chen, and Z. M. Mao, "Towards robust LiDAR-based perception in autonomous driving: General black-box adversarial sensor attack and countermeasures," in *29th USENIX Security Symposium (USENIX Security 20)*, pp. 877–894, USENIX Association, Aug. 2020.

[52] Z. El-Rewini, K. Sadatsharan, D. F. Selvaraj, S. J. Plathottam, and P. Ranganathan, "Cybersecurity challenges in vehicular communications," *Vehicular Communications*, vol. 23, p. 100214, 2020.

[53] M. Islam, M. Chowdhury, H. Li, and H. Hu, "Cybersecurity attacks in vehicle-to-infrastructure applications and their prevention," *Transportation Research Record*, vol. 2672, no. 19, pp. 66–78, 2018.

[54] M. Hadded, O. Shagdar, and P. Merdrignac, "Augmented perception by v2x cooperation (pac-v2x): Security issues and misbehavior detection solutions," in *2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)*, pp. 907–912, 2019.

[55] M. Ambrosin, L. L. Yang, X. Liu, M. R. Sastry, and I. J. Alvarez, "Design of a misbehavior detection system for objects based shared perception v2x applications," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 1165–1172, 2019.

[56] M. Tsukada, S. Arii, H. Ochiai, and H. Esaki, "Misbehavior detection using collective perception under privacy considerations," in *2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC)*, pp. 808–814, 2022.

[57] M. S. Ali and P. Merdrignac, "Distributed misbehavior detection based on vehicle perception model and cpm data collection," in *2023 IEEE 98th Vehicular Technology Conference (VTC2023-Fall)*, pp. 1–5, 2023.

[58] C. Meerpohl, M. Rick, and C. Büskens, "Free-space polygon creation based on occupancy grid maps for trajectory optimization methods," *IFAC-PapersOnLine*, vol. 52, no. 8, pp. 368–374, 2019. 10th IFAC Symposium on Intelligent Autonomous Vehicles IAV 2019.

[59] P. S. Maybeck, *Stochastic models, estimation and control.* Mathematics in science and engineering ; v. 141, New York: Academic Press, 1982.

[60] R. Schubert, E. Richter, and G. Wanielik, "Comparison and evaluation of advanced motion models for vehicle tracking," in *2008 11th International Conference on Information Fusion*, pp. 1–6, 2008.

[61] B. Wang, J. Lan, and J. Gao, "Lidar filtering in 3d object detection based on improved ransac," *Remote Sensing*, vol. 14, no. 9, 2022.

[62] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, "Dbscan revisited, revisited: Why and how you should (still) use dbscan," *ACM Trans. Database Systems (TODS)*, vol. 42, July 2017.

[63] S. Gillies, C. van der Wel, J. Van den Bossche, M. W. Taves, J. Arnott, B. C. Ward, *et al.*, "Shapely." `https://doi.org/10.5281/zenodo.7263102`, 2022.

[64] R. Xu, Y. Guo, X. Han, X. Xia, H. Xiang, and J. Ma, "Opencda: An open cooperative driving automation framework integrated with co-simulation," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pp. 1155–1162, 2021.

[65] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning* (S. Levine, V. Vanhoucke, and K. Goldberg, eds.), vol. 78 of *Proceedings of Machine Learning Research*, pp. 1–16, PMLR, 13–15 Nov 2017.

[66] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "Sumo – simulation of urban mobility: An overview," in *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation* (S. . U. o. O. Aida Omerovic, R. I. R. T. P. Diglio A. Simoni, and R. I. R. T. P. Georgiy Bobashev, eds.), ThinkMind, October 2011.

[67] D. Xu, "Opencood." `https://github.com/DerrickXuNu/OpenCOOD`, 2022.

[68] J. Zhang, I. B. Jemaa, and F. Nashashibi, "Simulation framework of misbehavior detection and mitigation for collective perception services," in *2024 IEEE Intelligent Vehicles Symposium (IV)*, pp. 2437–2442, 2024.

[69] D. Li, B. Liu, Z. Huang, Q. Hao, D. Zhao, and B. Tian, "Safe motion planning for autonomous vehicles by quantifying uncertainties of deep learning-enabled environment perception," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, pp. 2318–2332, 2024.

[70] T. Benciolini, D. Wollherr, and M. Leibold, "Non-conservative trajectory planning for automated vehicles by estimating intentions of dynamic obstacles," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 3, pp. 2463–2481, 2023.

[71] S. H. Nair, V. Govindarajan, T. Lin, C. Meissen, H. E. Tseng, and F. Borrelli, "Stochastic mpc with multi-modal predictions for traffic intersections," in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 635–640, 2022.

[72] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction," in *Proceedings of the Conference on Robot Learning* (L. P. Kaelbling, D. Kragic, and K. Sugiura, eds.), vol. 100 of *Proceedings of Machine Learning Research*, pp. 86–99, PMLR, 2020.

[73] H. Zhu and J. Alonso-Mora, "Chance-constrained collision avoidance for mavs in dynamic environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 776–783, 2019.

[74] M. Castillo-Lopez, P. Ludivig, S. A. Sajadi-Alamdari, J. L. Sanchez-Lopez, M. A. Olivares-Mendez, and H. Voos, "A real-time approach for chance-constrained motion planning with dynamic obstacles," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3620–3625, 2020.

[75] H. Lu, Q. Zong, S. Lai, B. Tian, and L. Xie, "Real-time perception-limited motion planning using sampling-based MPC," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 12, pp. 13182–13191, 2022.

[76] T. P. do Nascimento, G. F. Basso, C. E. T. Dorea, and L. M. G. Gonçalves, "Perception-driven motion control based on stochastic nonlinear model predictive controllers," *IEEE/ASME Transactions on Mechatronics*, vol. 24, no. 4, pp. 1751–1762, 2019.

[77] X. Zhang, J. Ma, Z. Cheng, S. Huang, S. S. Ge, and T. H. Lee, "Trajectory generation by chance-constrained nonlinear mpc with probabilistic prediction," *IEEE Transactions on Cybernetics*, vol. 51, no. 7, pp. 3616–3629, 2021.

[78] A. D. Bonzanini, A. Mesbah, and S. Di Cairano, "Perception-aware chance-constrained model predictive control for uncertain environments," in *2021 American Control Conference (ACC)*, pp. 2082–2087, 2021.

[79] X. Gao, C. Liu, W. Zhou, M. Xu, S. Wang, and J. Wang, "A survey of collaborative perception in intelligent vehicles at intersections," *IEEE Transactions on Intelligent Vehicles*, pp. 1–20, 2024.

[80] S. Zhang, S. Wang, S. Yu, J. J. Q. Yu, and M. Wen, "Collision avoidance predictive motion planning based on integrated perception and v2v communication," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 9640–9653, 2022.

[81] X. Zhao, J. Wang, G. Yin, and K. Zhang, "Cooperative driving for connected and automated vehicles at non-signalized intersection based on model predictive control," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 2121–2126, 2019.

[82] L. Blackmore, M. Ono, and B. C. Williams, "Chance-constrained optimal path planning with obstacles," *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1080–1094, 2011.

[83] H. Bao, Q. Kang, X. Shi, M. Zhou, H. Li, J. An, and K. Sedraoui, "Moment-based model predictive control of autonomous systems," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 4, pp. 2939–2953, 2023.

[84] A. Liniger, A. Domahidi, and M. Morari, "Optimization-based autonomous racing of 1:43 scale rc cars," *Optimal Control Applications and Methods*, vol. 36, no. 5, pp. 628–647, 2015.

[85] R. Reiter, A. Nurkanović, J. Frey, and M. Diehl, "Frenet-cartesian model representations for automotive obstacle avoidance within nonlinear mpc," *European Journal of Control*, vol. 74, p. 100847, 2023. 2023 European Control Conference Special Issue.