Spoofing in Cooperative Perception: A Structured Evaluation Using the TUMTraf-V2X Dataset

Carlos Eduardo de Sousa *†
Universidade de Brasilia†

Email: sousa.carlos@redes.unb.br*

***Abstract*—Cooperative perception enhances situational awareness in connected and automated vehicles by integrating information from heterogeneous sensing sources. However, once the cooperative perception layer is assumed trustworthy, even lightweight adversarial manipulations can distort the final shared representation of the environment. This work investigates the vulnerability of cooperative perception annotations by applying three classes of spoofing attacks—phantom object injection, random removal, and Region-of-Interest (ROI) removal—directly to the cooperative BEV annotations of the TUMTraf-V2X Mini dataset. No sensor-level tampering or fusion module is used; instead, the study focuses on how annotation-layer corruption alone affects the integrity of the perceived scene. Across 400 frames per attack, quantitative metrics reveal distinct adversarial signatures. Phantom injection consistently increases clutter and introduces plausible false obstacles. Random removal produces the highest global divergence, degrading scene completeness unpredictably. ROI-based removal generates the strongest safety-relevant impact by selectively suppressing detections within critical spatial zones despite minimal global changes. Collectively, the findings demonstrate that cooperative perception systems remain vulnerable to additive and subtractive spoofing even when raw sensor data is uncompromised. The results highlight the need for integrity mechanisms, redundancy checks, and anomaly detection strategies tailored to the semantics of cooperative perception.**

***Keywords*—cooperative perception, V2X security, spoofing attacks, cooperative BEV, dataset-based evaluation**

I. Introduction

Autonomous vehicles require accurate and robust perception to navigate complex environments. Although onboard sensors such as cameras, radar, and LiDAR have evolved significantly, they remain limited by occlusion, restricted fields of view, and environmental interference. Cooperative perception addresses these limitations by enabling vehicles and infrastructure to share sensory and semantic information through V2X communication. Recent studies show that cooperative perception increases detection range, reduces blind spots, and supports safer decision making in multi-agent settings [1]Infrastructure supported perception further strengthens these capabilities by adding static sensing units and roadside systems to the perception pipeline, improving coverage and reliability [2]

The same communication and sensing capabilities that enhance perception also expand the attack surface of autonomous vehicles. These systems are vulnerable to a wide range of cyber physical attacks, including signal manipulation, message falsification, and spoofing of sensors or V2X communication streams [3]Spoofing attacks are particularly concerning because they allow an adversary to inject artificial objects, hide real obstacles, alter spatial information, or distort the shared perception map. In cooperative settings, these effects may propagate across multiple vehicles, because agents often trust information received from others. Threat modeling research emphasizes that attackers can combine physical and cyber vectors to exploit weaknesses across sensing, communication, and control modules [4] Reviews of autonomous vehicle cybersecurity reinforce that spoofing remains one of the most critical threats due to its potential to compromise both local and cooperative perception [2]

Although several studies analyze spoofing attacks targeting individual sensors or communication channels, the literature lacks a unified method to understand how spoofed data affects each stage of the cooperative perception pipeline and how deceptive information propagates across interconnected agents. Existing works typically address sensor vulnerabilities, communication threats, or cooperative sensing architectures in isolation, without combining these dimensions into a single framework. Evaluating spoofing attacks often

requires specialized hardware and complex pipelines. Although this work includes a dataset-driven evaluation, the conceptual framework remains essential to contextualize where annotation-layer attacks fit within broader cooperative perception architectures.

The contributions of this paper are threefold. First, we introduce a unified conceptual model of cooperative perception, describing the sensing, representation, and communication processes that shape how information is produced and shared across agents. Second, we develop a structured taxonomy of spoofing attack surfaces aligned with this pipeline, clarifying how adversaries may target different stages to manipulate the perceived environment. Third, we operationalize these concepts through a dataset-driven experimental evaluation using the TUMTraf-V2X Mini benchmark, in which three spoofing attacks—phantom injection, random removal, and ROI-based concealment—are applied directly to cooperative perception annotations. This combination of conceptual analysis and empirical validation provides a practical foundation for understanding how lightweight manipulations affect the integrity of shared BEV representations.

The remainder of this paper is organized as follows. Section II reviews background concepts and related work on cooperative perception and spoofing. Section III introduces the unified conceptual model used to characterize perception stages and information flow. Section IV presents a structured taxonomy of spoofing attack surfaces aligned with this model. Section V describes the experimental setup, dataset configuration, and threat model adopted for the proof-of-concept evaluation. Section VI details the Python implementation used to generate spoofed cooperative annotations. Section VII reports the experimental results across 400 frames for three attack types. Section VIII provides a discussion of the implications of these findings, and Section IX concludes the paper and outlines directions for future work.

II. Background and Related Work

A. Cooperative perception fundamentals

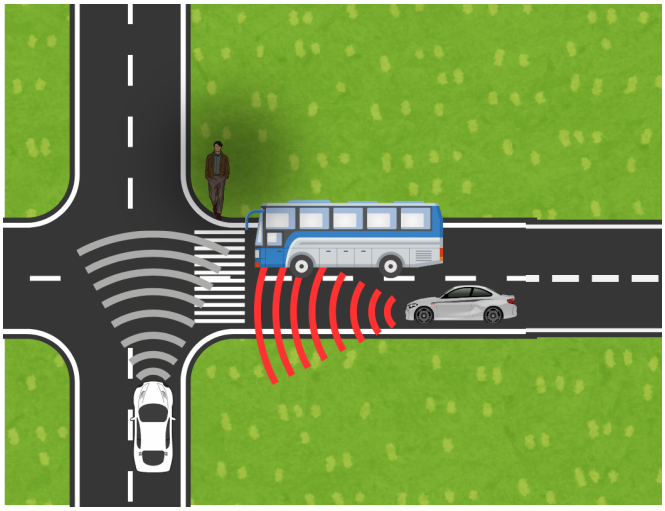
Cooperative perception emerges as an extension of the sensing capabilities of connected and automated vehicles. While onboard sensors such as lidar, radar, cameras and GNSS support local perception, they are limited by occlusions, range, field of view constraints, weather effects and computational load, as described in [3] When vehicles share information through V2X communication, they can overcome many of these limitations. Studies like [5] emphasize that V2X enables the exchange of raw or processed perception data, status information and intent, extending the awareness horizon of each vehicle. Cooperative perception supports driving tasks such as blind intersection navigation, detection of occluded pedestrians, left turn assist and do not pass warning. These tasks depend on fast, reliable and authenticated message exchange. The literature also stresses that cooperative perception depends on communication quality, fusion strategies and the integrity of shared data. Although not all works explicitly address the perception pipeline, they frequently describe the flow of information among vehicles, roadside units and cloud services, showing how perception outputs become collective decisions. Works such as Radio Jamming in V2X and B5GCyberTestV2X highlight that cooperative sensing and situational awareness are especially sensitive to communication interruptions and data manipulation.

B. Information flow in V2X systems

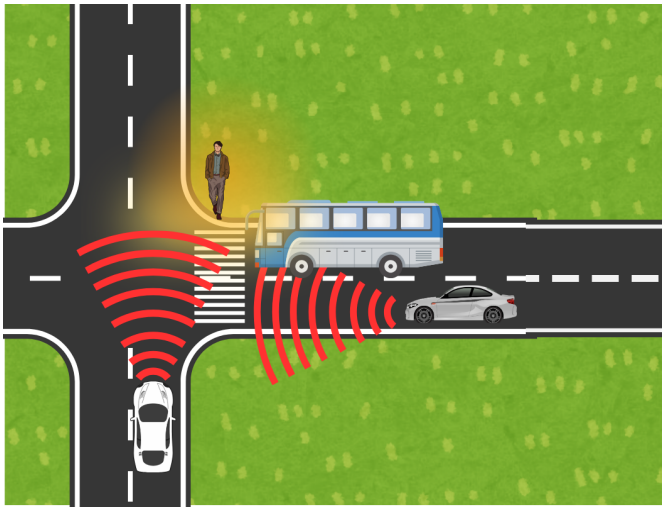
The information flow in V2X systems follows a structured chain involving sensing, pre-processing, packaging, communication and reception. According to [5] , V2X communication encompasses several modalities including vehicle to vehicle, vehicle to infrastructure, vehicle to network, vehicle to pedestrian and vehicle to grid. Each modality enables different forms of information exchange needed for coordinated driving. The flow of Basic Safety Messages, as described in [6] , provides a clear example. Vehicles periodically broadcast packets containing position, speed, direction and control data, which neighboring agents use to compute collision risks and safe maneuvers. The flow also depends on emerging communication technologies such as 5G and Beyond 5G, which introduce network slicing, separation between control and data planes, low latency and enhanced authentication procedures. The B5GCyberTestV2X work demonstrates how the inclusion of the 5G control plane reinforces security by ensuring that unauthorized devices cannot participate in the network. These systems depend on continuous exchanges among multiple layers, including in vehicle networks such as CAN, external interfaces such as OBD ports and wireless channels including DSRC, C-V2X, Wi Fi and cellular networks. The cybersecurity in autonomous vehicles literature shows that each step in the information flow may introduce vulnerabilities depending on how data is collected, processed, validated and transmitted.

C. Spoofing attacks in cooperative and vehicular sensing systems

Spoofing is a significant threat to V2X communication and vehicular sensing, with the potential to



(a) Isolated perception: the vehicle relies exclusively on its own sensors, resulting in a limited field of view.



(b) Cooperative perception: V2X information expands the understanding of the environment, reducing blind spots and increasing robustness.

Figure 1: Conceptual comparison between isolated perception and cooperative perception in V2X systems.

manipulate transmitted data, distort environmental awareness, and disrupt cooperative behavior among connected vehicles [5], [4]. Such attacks involve the deliberate injection of falsified information into sensing pipelines or communication channels. According to [3] , spoofing may target on-board sensors such as GPS, lidar, and cameras, generating deceptive data that manipulates perception modules and downstream decision-making. GPS spoofing can cause persistent localization errors, lidar spoofing can introduce phantom objects or hide real ones, and camera spoofing may exploit adversarial patterns or light-based interference.

In V2X contexts, message spoofing represents a major attack vector, as described in [5] . Adversaries may broadcast fabricated Basic Safety Messages, forge identities, or impersonate legitimate vehicles, leading to incorrect cooperative perception results and corrupted situational awareness. Spoofing is also discussed in studies on V2X security and jamming resilience, where attackers inject falsified warning messages or misbehavior signals to test the robustness of communication and control mechanisms. These attacks exploit implicit trust within cooperative systems, enabling erroneous information to propagate rapidly across multiple agents.

The B5GCyberTestV2X work [7] shows that emerging 5G control-plane procedures can restrict unauthorized devices from joining the network, yet they do not mitigate risks associated with malicious insiders or compromised nodes. Consequently, spoofing remains a central concern for any perception architecture that relies on shared or cooperative data.

D. Related work

Safeguarding the V2X Pathways [5] organizes V2X communication threats into spoofing, jamming, denial of service and eavesdropping, and highlights how each one affects safety and message integrity. Cybersecurity in Autonomous Vehicles [3] expands this categorization by including sensor spoofing, malware injection, man in the middle attacks, message replay, unauthorized access to in vehicle networks and physical compromise. Threat Modelling in Automotive Cybersecurity [8] describes structured methodologies such as STRIDE, DREAD, EVITA, HEAVENS and TVRA, providing systematic procedures for identifying weaknesses, estimating risk and guiding mitigation. TARA 2.0 for Connected and Automated Vehicles [9] aligns threat analysis with ISO SAE 21434 and UNECE WP29, emphasizing the importance of formal processes for risk assessment across connected and automated platforms.

Studies focusing on spoofing and adversarial manipulation provide additional insight into how attacks compromise cooperative perception. LiDAR point cloud transmission Adversarial perspectives of spoofing attacks [10] shows that injected or altered point clouds can create ghost objects or hide real ones, and that these perturbations remain effective across multiple frames. On the Realism of LiDAR Spoofing Attacks [11] demonstrates that both hardware based and algorithmic spoofing can mislead detectors by producing geometrically plausible patterns. Seeing is Deceiving Mirror Based LiDAR Spoofing reveals that passive optical elements can redirect beams and generate stealthy illusions without active emission. PhyScout Detecting

Sensor Spoofing Attacks [12] proposes cross sensor consistency checks but also shows that single sensor validation is insufficient when data is later fused. Radio Jamming in Vehicle to Everything Communication Systems [6] highlights that interference and message disruption can amplify spoofing effects by preventing proper cross validation between agents.

Taken together, these documents present a consistent view of cooperative and connected systems as environments composed of multiple interdependent attack surfaces, where manipulated signals or forged messages can propagate between agents and influence distributed decision making. Although these works provide extensive coverage of cyber physical threats in autonomous vehicles and V2X systems, they do not address spoofing within cooperative perception pipelines as a unified process nor explain how deceptive information evolves through sensing, communication, representation and fusion stages. This absence of a structured perspective reinforces the need for a conceptual framework capable of describing how spoofed information interacts with each component of cooperative perception and how such manipulation affects multi-agent situational awareness. Beyond providing such a framework, this work also offers an initial dataset-driven evaluation of annotation-level spoofing on a real cooperative perception benchmark.

III. Conceptual Model of Cooperative Perception

This section introduces a conceptual model of cooperative perception that organizes the main elements of the perception pipeline and highlights the vulnerabilities and inter-agent dependencies inherent to V2X based sensing and information sharing. The objective is to provide a unified structure that supports the analysis of spoofing attack surfaces and forms the foundation for the evaluation framework developed in later sections.

A. Overview of the perception pipeline

Modern cooperative perception systems follow a pipeline composed of several sequential stages, beginning with local sensing and culminating in a fused situational understanding shared across multiple agents. Although the implementation details vary across specific architectures and collaboration strategies, the fundamental stages remain consistent across the surveyed literature.

1) Sensing
   Each agent collects observations from onboard sensors such as LiDAR, cameras, radar, and GNSS, which provide complementary information for building the local perception state. LiDAR supplies high resolution three dimensional point clouds, cameras contribute semantic and appearance cues, radar provides long range and velocity measurements, and GNSS and IMU sensors support pose estimation. Studies such as [13] describe how the quality of perception depends on the reliability of these sensing modalities and note that occlusions, adverse weather, calibration errors, and hardware limitations introduce uncertainty that propagates through the entire cooperative perception pipeline.
2) Pre-processing
   Raw sensor outputs undergo modality specific preprocessing, including point cloud filtering, voxelization, BEV projection, feature extraction and timestamp synchronization. This stage directly affects the fidelity of downstream collaboration, since any distortion or delay introduced here propagates to later steps of the cooperative pipeline. Voxel based and BEV representations are commonly adopted because they reduce computational load while preserving the spatial structure required for fusion. Studies [13], [14] on cooperative perception pipelines also indicate that pose inaccuracies must be addressed during preprocessing, as misalignment between agents leads to inconsistent feature maps and degraded fusion quality.
3) Representation generation
   After preprocessing, each agent builds intermediate representations. Depending on the collaboration strategy, these may be raw sensor frames (early collaboration), feature maps such as BEV grids or encoded camera features (intermediate collaboration), or object-level detections (late collaboration). Feature extraction is typically performed by convolutional backbones or transformer-based encoders. Recent work [14] emphasizes that representation choices strongly influence robustness to communication constraints and latency, as well as vulnerability to adversarial perturbations.
4) Communication
   Agents exchange perception information over V2X channels such as DSRC, C-V2X, or NR-V2X. Communication may involve raw data, intermediate features, or final detections. Collaborative effectiveness depends on message frequency, timestamp accuracy, agent selection, and bandwidth availability. Studies on communication constraints [14], [6], [7], [15] show that delays, data loss, and heterogeneous link quality distort the temporal and spatial consistency of the shared

Table I: Comparison of selected studies in cooperative perception and V2X security.

Study / Feature	Spoofing Focus	CP Spec.	Attack–Pipeline	Stage Vuln.	Multi-Agent	Taxon.	Scen.	Experim. Method	Agn.
[2]	x	✓	x	~	✓	x	x	x	✓
[3]	~	x	x	x	x	x	x	x	✓
[4]	~	x	x	~	x	~	x	x	✓
[9]	x	x	x	✓	x	x	x	✓	x
[5]	~	~	x	x	✓	x	x	x	~
This Work	✓	✓	✓	✓	✓	✓	✓	✓	✓

Legend: ✓ = covered; x = not covered; ~ = partially covered.

perception space.

5) Fusion
   Fusion aligns and integrates self-perception with information received from other agents. Fusion methods vary from simple spatial merging to learned attention-based aggregation. Early fusion integrates raw data, intermediate fusion aligns and merges feature maps, and late fusion combines detection results. Multiple works [13], [14], [15] highlight that accurate data alignment, especially under pose uncertainty, remains one of the most challenging aspects of cooperative perception. Misaligned fusion can magnify errors rather than reduce them.
6) Final perception
   The fusion stage produces a consolidated world model, including object detection, segmentation, tracking, and scene inference. This final perception output is then used for downstream tasks such as planning and control. The resulting shared perception is only as reliable as the weakest stage in the pipeline, making the entire workflow vulnerable to spoofing and misinformation propagation.

B. Identification of intrinsic vulnerabilities

A coherent examination of the cooperative perception pipeline reveals a set of structural vulnerabilities that arise from the system's inherent interdependencies. These vulnerabilities emerge across sensing, preprocessing, representation generation, communication, and fusion, reflecting limitations in how agents acquire, align, and combine information. Each stage introduces opportunities for distortions, inconsistencies, or manipulation, which can be exploited by spoofing attacks or amplify internal errors already present in the system.

- Sensor-level vulnerabilities: LiDAR spoofing, camera blinding, radar interference, and GNSS manipulation can alter raw observations before any collaborative checks occur. Physically injected point clouds can bypass early-stage detection logic.
- Representation vulnerabilities: Intermediate representations such as BEV feature maps or voxel grids may lack physical constraints that distinguish legitimate and spoofed signals. Occlusion patterns and depth consistency are not always preserved in feature extraction, creating opportunities for attack injection.
- Pose alignment and synchronization vulnerabilities: Even small pose errors propagate through feature alignment, enabling attackers to inject misleading spatial information. Communication latency further amplifies temporal inconsistencies.
- Communication vulnerabilities: V2X messages can be delayed, lost, or tampered with. Because cooperative perception often assumes honest agents, false or manipulated CPMs or feature maps can directly influence fused perception results.
- Fusion vulnerabilities: Misaligned or conflicting information from multiple agents introduces ambiguity. Fusion algorithms may overweight erroneous foreign data or fail to detect inconsistencies between local and shared observations.

C. Discussion on inter-agent dependencies

Cooperative perception relies heavily on inter-agent dependencies created at multiple levels of the pipeline.

First, sensing dependencies arise because agents depend on each other to overcome occlusion, limited field of view, and range constraints. This creates a trust requirement: each agent implicitly assumes that neighboring agents provide accurate and reliable information.

Second, temporal dependencies result from asynchronous sensing and communication. Even minor timestamp mismatches can generate spatial inconsistencies, duplicate objects, or phantom detections.

Third, spatial dependencies are introduced through shared coordinate systems and pose transformations. Minor calibration errors propagate across all agents and may accumulate during fusion.

Fourth, semantic dependencies emerge in feature-level collaboration. Because agents combine abstract representations rather than raw data, each agent must

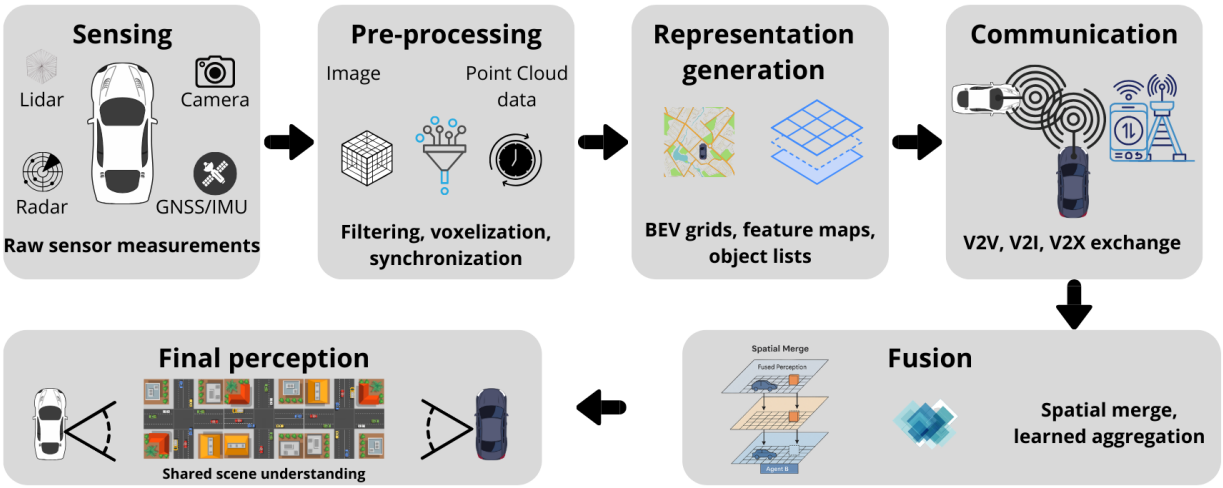


Figure 2: Unified cooperative perception pipeline, illustrating the sequence from sensing to final perception.

Table II: Intrinsic vulnerabilities across the cooperative perception pipeline.

Pipeline Stage	**Key Vulnerabilities**	**Underlying Causes**
Sensing	Occlusions; environmental sensitivity; physical spoofing	Line-of-sight dependence; sensor physics limitations
Pre-processing	Pose drift; misalignment; calibration errors	Synchronization delays; imperfect ego-motion estimation
Representation Generation	Feature corruption; semantic distortion	Abstraction layers; compression losses; reduced consistency checks
Communication	Delay; packet loss; inconsistent timestamps	Wireless channel variability; heterogeneous link quality
Fusion	Error amplification; conflict resolution bias	Inter-agent dependencies; reliance on shared features
Final Perception	False positives/negatives; unstable tracks	Accumulated distortions propagated from previous stages

assume that the feature extractor of every other agent is consistent, well-calibrated, and robust.

Finally, decision dependencies arise when cooperative perception is used for control tasks such as platooning, intersection coordination, or cooperative planning. In these contexts, errors introduced by one compromised agent can directly influence the behavior of the entire system.

Taken together, these dependencies create fertile conditions for spoofing attacks: any manipulated or inconsistent information injected at one agent can influence several others through the collaboration mechanism, ultimately affecting global perception and decision-making. Cooperative perception therefore amplifies both the benefits and the risks of interconnected sensing.

IV. Spoofing Attack Surfaces in Cooperative Perception

Spoofing attacks in cooperative perception exploit the distributed structure of sensing, communication, and fusion across multiple agents. Because cooperative perception integrates heterogeneous data sources into a shared situational representation, any deceptive information injected at local or remote nodes can propagate through the network. Analysis of LiDAR spoofing techniques, passive and active optical manipulation, radio interference, message falsification, and automotive threat models indicates four primary classes of attack surfaces in cooperative perception.

A. Attacks at the sensor level

Sensor level spoofing manipulates physical measurements before they enter the cooperative perception pipeline. This makes it one of the most critical attack surfaces, since downstream stages generally assume that raw sensor inputs are trustworthy. Such attacks exploit the exposed nature of sensing pathways and the absence of verification mechanisms at the earliest stages of processing, creating opportunities to inject artificial points, conceal real objects, or distort the perceived geometry of the scene. Because these manipulations occur before any alignment, fusion or cross checking takes place, their effects can propagate directly into all subsequent stages of cooperative perception.

LiDAR spoofing is one of the most widely studied threats. Active spoofing involves injecting counterfeit

pulses into the receiver, creating phantom obstacles or triggering mislocalization. Passive spoofing, as demonstrated in [16] , exploits specular reflections to redirect valid beams, creating object addition and object removal effects. These attacks generate plausible point clouds without requiring electronic emitters, making them highly stealthy. Studies on LiDAR transmission attacks further show how an adversary can intercept or modify returning pulses to distort range measurements.

Camera-based spoofing can include projector-based pattern injection, adversarial signage, or reflective materials designed to confuse lane detection or object recognition. Radar systems are vulnerable to replayed echoes or deceptive Doppler signatures. GNSS spoofing and jamming can corrupt localization, causing downstream misalignment in collaborative mapping and fusion.

These sensor-level attacks are particularly relevant because cooperative perception systems often assume that remote agents will detect objects reliably when line of sight is occluded. If a compromised agent provides manipulated sensor data, the deception spreads across multiple vehicles.

B. Attacks on intermediate representations

Intermediate representations such as BEV feature maps, voxel grids, object lists, and semantic embeddings are increasingly exchanged in cooperative perception systems. While these abstractions reduce communication bandwidth, they also introduce vulnerabilities because the receiving agents lack access to raw data for verification.

Feature level spoofing is often easier to perform than raw signal manipulation, since the attacker does not need to reproduce sensor physics and can instead introduce features that appear structurally consistent within the representation space. As shown in [10], small but plausible modifications in intermediate representations are sufficient to generate artificial free space, phantom objects, or distorted road geometry. Manipulation at the object list level can also introduce false detections or suppress real ones, and these distortions propagate through fusion mechanisms that rely on synchronized and calibrated remote features.

Automotive threat modeling, as outlined in [9], shows that abstraction layers can create blind spots for security monitoring. When intermediate data replaces raw measurements, the system loses the ability to verify physical consistency across sensing modalities. This enables semantic spoofing, in which an attacker manipulates the representation itself rather than the underlying sensor signal.

C. Attacks on communication channels

Communication channels represent a distinct attack surface where spoofing does not target sensors directly but instead manipulates the data exchanged between agents. Cooperative perception relies on V2X technologies such as DSRC, C-V2X, and emerging 5G and 6G networks. These links carry Cooperative Perception Messages, sensor feature maps, and motion predictions.

Communication spoofing can be broadly divided into two classes.

First, message injection or falsification involves crafting counterfeit V2X packets that appear authentic. These messages can introduce false object reports or misleading situational information. Studies on V2X security [3], [5], [17], [7] note that DSRC and early C-V2X implementations are susceptible to unauthorized transmitters, allowing an attacker to impersonate a legitimate vehicle. Even in 5G enabled systems, spoofers may attempt to exploit gaps in registration or handover procedures, though more advanced control planes make these attacks harder.

Second, message modification involves altering the content of legitimate V2X packets during transmission. Instead of forging entirely new messages, the attacker manipulates fields such as object lists, kinematic attributes, environmental descriptors, or situational updates while preserving the original packet structure, timing, and authentication metadata. Because the message appears valid at the communication layer, the receiving agent often integrates the altered information into its cooperative perception state without triggering lower level alarms. This type of semantic manipulation can distort shared situational awareness even when the communication channel itself remains fully operational[3], [17].

Emerging work on Beyond 5G simulation frameworks [7] demonstrates that future V2X architectures with stronger control plane enforcement can mitigate certain spoofing attempts, although message level manipulation remains an active threat.

D. Multi-agent coordinated spoofing

Cooperative perception introduces unique risks related to coordinated and distributed spoofing across multiple agents. Because vehicles fuse information from several peers, deceptive data injected at one node can influence many others, and inconsistencies may not be detected if the forged data appears self consistent.

Multi-agent spoofing encompasses attacks where one or more compromised agents collude or where a single attacker strategically influences the network to manipulate collective perception. For instance, an

attacker could deploy multiple spoofing devices to create a coordinated illusion of an obstacle across several agents, thereby increasing the credibility of the fake detection. Another possibility is cross agent feature manipulation, where an adversary exploits asynchronous sensing and communication to introduce spatial temporal inconsistencies that appear legitimate when combined.

These coordinated attacks are especially critical because they undermine the redundancy and trust assumptions that cooperative systems depend on. When architectures rely on majority voting or consensus mechanisms, a well-timed set of spoofed inputs can shift the collective decision toward an incorrect perception state. In operational scenarios such as platooning, intersection negotiation, or collaborative planning, corrupted perception originating from a single manipulated agent can propagate through the group and induce unsafe behavior among otherwise uncompromised vehicles.

Overall, coordinated spoofing represents a systemic threat that exploits the distributed architecture of cooperative perception. Its effects are magnified as the number of interconnected agents increases.

While the taxonomy spans sensor-level, representation-level, communication-level, and multi-agent coordinated spoofing, the experimental evaluation in the following sections focuses specifically on representation-level attacks. More precisely, the experiments target the cooperative object-layer provided by the dataset, enabling controlled manipulation of shared annotations without altering raw sensor data or the communication stack. This scoped evaluation complements the conceptual model by demonstrating how a subset of spoofing surfaces manifests in practice.

V. Experimental Setup and Implementation

To illustrate how spoofing affects the integrity of shared perception, this section presents a proof-of-concept experiment using the TUMTraf-V2X Mini dataset. The goal is not to implement a full cooperative fusion pipeline, but rather to evaluate how manipulations applied directly to the cooperative perception annotations alter the final BEV representation. The dataset already provides a pre-assembled cooperative view, and the experiments operate exclusively on this layer. This design allows us to isolate the effects of spoofing on the shared perception output without modifying any sensor-level data or fusion algorithms.

A. Dataset

Experiments were conducted using a subset of the TUMTraf-V2X dataset, specifically the Mini version made available for lightweight experimentation [https://tum-traffic-dataset.github.io/tumtraf-v2x/]. The subset includes LiDAR-derived object annotations stored in OpenLABEL JSON format and corresponding .pcd point cloud files.

Each frame contains:

- A list of 3D bounding boxes representing objects detected in the scene.
- Metadata such as position, orientation, dimensions, and timestamps.
- References to the raw LiDAR point cloud used to generate the bounding boxes.

It is important to note that the TUMTraf-V2X Mini subset does not provide explicit flags indicating whether an object was originally detected by a vehicle (ego) or by infrastructure-mounted sensors. For the purposes of this experiment, and given the simplified objective of evaluating spoofing propagation, all objects are treated uniformly as part of the local pre-fusion perception state.

B. Data Processing Pipeline

The implemented pipeline follows the conceptual stages described earlier but in a simplified and computationally lightweight form. Figure 2 (conceptual pipeline) remains valid; however, the experimental steps materialize as follows.

1) Frame loading and object extraction: For each frame, bounding boxes are loaded from the OpenLABEL JSON file. A custom parsing function (`load_openlabel_boxes_tumtraf`) extracts:

- box center coordinates (x, y, z),
- dimensions (l, w, h),
- object yaw angle,
- the optional `sensor_id` field, when available.

These bounding boxes constitute the baseline perception for that frame.

2) Transformation into infrastructure-centric coordinates: Although the conceptual model in Section III includes transformations between ego-centric and global frames, the experimental setup does not apply additional coordinate transformations. Instead, objects are analyzed directly in the BEV coordinate system provided by the dataset, which already expresses positions in a unified frame suitable for ROI analysis and metric computation. This simplifies the pipeline and ensures that the evaluation focuses solely on the effects of annotation-level spoofing.

3) Pre-cooperative representation: In this work, no fusion stage is implemented. Instead, we rely directly on the cooperative perception already provided by the TUMTraf-V2X Mini dataset. Each OpenLABEL JSON file contains a set of objects that represent the combined

Table III: Spoofing attack surfaces mapped to key points of the cooperative perception pipeline.

Attack Class	Entry Point	Example Effects	Exploited Assumptions
Sensor-level spoofing	Raw sensor measurements	Phantom points; object masking; geometry distortion	Trust in physical sensor integrity
Intermediate representation spoofing	BEV maps; feature tensors; object lists	Artificial free space; false detections; semantic shifts	Reliability of preprocessed features
Communication-level spoofing	V2X message exchange	False object reports; misleading situational updates	Authenticity of transmitted packets
Multi-agent coordinated spoofing	Distributed perception nodes	Consensus bias; reinforced misperception; emergent degradation	Trust in redundancy and agreement among agents

perception output of multiple sensing sources (e.g., vehicle-mounted and infrastructure LiDARs).

Objects are therefore not merged, aligned, or associated; they are consumed exactly as they appear in the dataset. Our manipulations operate directly on this cooperative perception layer.

For consistency of analysis, each object is represented internally as an Oriented Bounding Box (OBB), from which the following attributes are extracted:

- spatial footprint (center coordinates),
- orientation (yaw),
- height and dimensions,
- distance computed directly in the global BEV coordinate system provided by the dataset,
- class-agnostic identifiers.

This representation is intentionally simple because the goal is not to evaluate detection or classification accuracy, nor to reconstruct a fusion system. Instead, we analyze how spoofing attacks affect an existing cooperative perception output. All modifications—phantom insertion, random removal, and ROI-based removal—are applied directly on this cooperative annotation layer.

C. Spoofing Attacks Implemented

In addition to the conceptual model, three spoofing attacks were implemented to evaluate how adversarial manipulations affect the cooperative perception already provided by the TUMTraf-V2X Mini dataset. No fusion module is implemented in this work. Instead, all attacks operate directly on the cooperative annotation layer contained in each OpenLABEL JSON file, which represents the perception result already aggregated from vehicle and infrastructure sensors.

The attacks therefore modify only the shared cooperative perception data, without altering the original LiDAR point cloud or any upstream sensing processes. This reflects threat scenarios in which an adversary tampers with perception messages after sensing has occurred, such as through V2X message manipulation or annotation-layer corruption.

1) Phantom Object Injection: This attack inserts artificial objects directly into the cooperative perception output provided by the TUMTraf-V2X Mini dataset. Each phantom is added to the OpenLABEL annotation as if it were a legitimate detection produced by the cooperative sensing infrastructure. Phantoms are placed at plausible coordinates within the BEV reference frame and are assigned realistic dimensions and yaw values to resemble true road participants.

Key characteristics:

- no real object is modified or overwritten;
- the attack operates on the cooperative BEV annotation layer, not on sensor data;
- the injected phantom appears structurally consistent with legitimate detections.

Rationale: This attack emulates an adversary who forges cooperative perception messages or tampers with the annotation layer, injecting false objects into the shared situational representation. Because the dataset already provides a fused cooperative view, the phantom attack demonstrates how the final perception map may be corrupted even without modifying raw sensor measurements.

2) Random Object Removal: In this attack, a subset of cooperative detections is removed at random from the BEV annotation. The removal ratio is configurable, allowing simulation of partial corruption or deletion of perception data. The purpose is to evaluate how non-targeted manipulation degrades the completeness of the cooperative scene description.

This attack models scenarios where an adversary:

- suppresses arbitrary detections, creating incomplete situational awareness;
- degrades the cooperative perception quality by reducing object density;
- introduces uncertainty by randomly removing elements of the shared map.

Rationale: Random removal corresponds to message tampering, partial annotation corruption, or data loss induced by interference or adversarial manipulation. Because no fusion step exists in our experimental setup, the removed objects simply disappear from the cooperative BEV layer, directly impacting the scene interpretation.

3) Region-of-Interest (ROI) Object Removal: The ROI removal attack selectively deletes objects located within a predefined spatial region of the cooperative BEV map. The ROI is defined directly in the dataset's coordinate system and typically corresponds to zones of higher safety relevance, such as the central forward area or cross-traffic regions.

This attack simulates an adversary who:

- intentionally suppresses detections in safety-critical areas;
- hides obstacles or road users that would normally only be observed cooperatively;
- manipulates the cooperative BEV representation to induce dangerous misperceptions.

Rationale: Compared to random removal, ROI-based deletion represents a targeted and more sophisticated spoofing strategy. By manipulating only the most critical region of the BEV map, the attacker can produce disproportionate harm, demonstrating how localized annotation-layer attacks can severely distort the perceived traffic environment.

D. Evaluation Metrics

Since no fusion module is implemented in this work, all metrics are computed directly from the cooperative perception annotations provided by the dataset and their manipulated (spoofed) counterparts. For each frame, the baseline JSON file and the spoofed JSON file are compared to quantify the effect of the applied attack. The following metrics were extracted:

- **Object Count Difference (Δn):** The change in the number of annotated objects after spoofing, computed as
$$\Delta n = n_{\text{spoof}} - n_{\text{base}}.$$
- **Phantom Object Count:** The number of injected objects explicitly labeled as `PHANTOM` in the spoofed annotation.
- **Phantom Influence Index (PII):** Measures the proportion of injected objects that remain present in the spoofed cooperative perception,
$$PII = \frac{n_{\text{phantoms}}}{\max(n_{\text{spoof}} - n_{\text{base}}, 1)}.$$
- **Scene Divergence Score (SDS):** Quantifies how much the overall perception changes after manipulation,
$$SDS = \frac{|n_{\text{spoof}} - n_{\text{base}}|}{\max(n_{\text{base}}, 1)}.$$
- **ROI-Based Metrics:** For a predefined spatial region of interest (ROI) in the BEV coordinate system, we compute:
  - $n_{\text{base\_roi}}$ — number of baseline detections inside the ROI,
  - $n_{\text{spoof\_roi}}$ — number of detections inside the ROI after spoofing,
  - $roi_{\text{removed}} = n_{\text{base\_roi}} - n_{\text{spoof\_roi}}$,
  - ROI Impact Score:
$$ROI_{\text{impact}} = \frac{roi_{\text{removed}}}{\max(n_{\text{base\_roi}}, 1)}.$$

These metrics collectively quantify how spoofing alters the cooperative perception layer, either by inserting artificial objects, removing legitimate detections or selectively degrading critical regions of the BEV map.

E. Implementation Environment

All experiments were executed in Google Colab, using:

- Python 3.12.12,
- Open3D for geometric manipulation and visualization,
- NumPy for matrix operations,
- Matplotlib for heatmap generation.

The implementation was CPU-only, demonstrating that the TUMTraf-V2X dataset and the simplified cooperative pipeline can be executed without GPU acceleration.

VI. Implementation in Python

This section presents the core Python functions used to generate the spoofed cooperative perception annotations and to compute the metrics reported in Section VII. The implementation operates directly on TUMTraf-V2X OpenLABEL JSON files, injecting or removing objects and then comparing baseline and spoofed annotations on a per-frame basis.

A. Phantom Injection Function

The function `create_spoofed_json_openlabel` injects a configurable number of phantom objects into a TUMTraf OpenLABEL file. Phantoms are encoded as full 3D cuboids with type `PHANTOM`, allowing the official DevKit to render them in a distinct color.

```
def create_spoofed_json_openlabel(
    original_json_path,
    output_json_path,
    n_phantoms=3,
    x_range=(10, 30),
    y_range=(-5, 5),
    z_fixed=-1.5,
    lwh=(4.0, 1.8, 1.6),
    yaw_range=(-3.14, 3.14)
):
    """
    Creates a spoofed OpenLABEL JSON by adding PHANTOM
          objects.
    """

    with open(original_json_path, "r") as f:
        data = json.load(f)

```

```
    if "openlabel" not in data:
        raise RuntimeError("JSON is not in OpenLABEL format.")

    frames = data["openlabel"]["frames"]
    if len(frames) != 1:
        raise RuntimeError("Expected exactly one frame in JSON.")

    frame_key = list(frames.keys())[0]
    frame_data = frames[frame_key]

    original_objects = frame_data.get("objects", {})
    spoofed_objects = copy.deepcopy(original_objects)

    for i in range(n_phantoms):
        phantom_id = str(uuid.uuid4())

        x = np.random.uniform(*x_range)
        y = np.random.uniform(*y_range)
        z = z_fixed
        yaw = np.random.uniform(*yaw_range)

        length, width, height = lwh

        spoofed_objects[phantom_id] = {
            "object_data": {
                "type": "PHANTOM",
                "name": f"PHANTOM_{i}",
                "cuboid": {
                    "name": "shape3D",
                    "val": [
                        float(x),
                        float(y),
                        float(z),
                        0.0, 0.0,
                        np.sin(yaw/2), np.cos(yaw/2),
                        float(length),
                        float(width),
                        float(height)
                    ]
                }
            }
        }

    frames[frame_key]["objects"] = spoofed_objects

    with open(output_json_path, "w") as f:
        json.dump(data, f, indent=2)
```

B. Concealment Attack: Object Removal

The function `create_removed_json_openlabel` implements three concealment modes: random removal, ROI-based removal, and class-based removal. This supports experiments with both non-targeted and targeted deletion of cooperative detections.

```
def create_removed_json_openlabel(
    original_json_path,
    output_json_path,
    remove_mode="random",   # "random", "roi", "class"
    n_remove=3,
    roi_x=(10, 40),
    roi_y=(-10, 10),
    class_to_remove="CAR"
):
    """
    Removes legitimate objects from a TUMTraf OpenLABEL JSON
    to simulate concealment / blind-spot induction.
    """

    with open(original_json_path, "r") as f:
        data = json.load(f)

    frames = data["openlabel"]["frames"]
    frame_key = list(frames.keys())[0]
    frame_data = frames[frame_key]

    original_objects = frame_data.get("objects", {})
    new_objects = copy.deepcopy(original_objects)

    obj_list = []
    for oid, obj in original_objects.items():
        try:
            cuboid = obj["object_data"]["cuboid"]["val"]
            class_name = obj["object_data"]["type"]
            x, y, z = cuboid[0], cuboid[1], cuboid[2]
            obj_list.append((oid, x, y, z, class_name))
        except:
            continue

    objects_to_remove = []

    if remove_mode == "random":
        if isinstance(n_remove, str) and n_remove.lower() == "random":
            n_remove = np.random.randint(1, 10)
        if n_remove > 0 and len(obj_list) > 0:
            selected = np.random.choice(
                len(obj_list),
                size=min(n_remove, len(obj_list)),
                replace=False
            )
            objects_to_remove = [obj_list[i][0] for i in selected]

    elif remove_mode == "roi":
        for oid, x, y, z, cls in obj_list:
            if roi_x[0] <= x <= roi_x[1] and roi_y[0] <= y <= roi_y[1]:
                objects_to_remove.append(oid)

    elif remove_mode == "class":
        for oid, x, y, z, cls in obj_list:
            if cls.upper() == class_to_remove.upper():
                objects_to_remove.append(oid)

    for oid in objects_to_remove:
        if oid in new_objects:
            del new_objects[oid]

    frames[frame_key]["objects"] = new_objects

    with open(output_json_path, "w") as f:
        json.dump(data, f, indent=2)

    return objects_to_remove
```

C. Batch Processing of a Dataset Folder

The function `spoof_all_json_in_folder` applies one of the attacks to all JSON files in a TUMTraf

label directory, generating an attacked version of the cooperative perception dataset.

```python
def spoof_all_json_in_folder(
    input_json_dir,
    output_json_dir,
    attack_type="phantom",
    n_phantoms=3,
    x_range=(10, 30),
    y_range=(-5, 5),
    z_fixed=-1.5,
    lwh=(4.0, 1.8, 1.6),
    yaw_range=(-3.14, 3.14),
    n_remove=3,
    roi_x=(10, 40),
    roi_y=(-10, 10),
    class_to_remove="CAR"
):
    """
    Applies a spoofing attack to all OpenLABEL JSON files in a folder.
    """

    input_json_dir  = Path(input_json_dir)
    output_json_dir = Path(output_json_dir)
    output_json_dir.mkdir(parents=True, exist_ok=True)

    json_files = sorted(input_json_dir.glob("*.json"))

    for jpath in json_files:
        out_path = output_json_dir / jpath.name

        if attack_type == "phantom":
            create_spoofed_json_openlabel(
                original_json_path=jpath,
                output_json_path=out_path,
                n_phantoms=n_phantoms,
                x_range=x_range,
                y_range=y_range,
                z_fixed=z_fixed,
                lwh=lwh,
                yaw_range=yaw_range
            )

        elif attack_type in ("remove_random", "remove_roi", "remove_class"):
            if attack_type == "remove_random":
                remove_mode = "random"
            elif attack_type == "remove_roi":
                remove_mode = "roi"
            else:
                remove_mode = "class"

            create_removed_json_openlabel(
                original_json_path=jpath,
                output_json_path=out_path,
                remove_mode=remove_mode,
                n_remove=n_remove,
                roi_x=roi_x,
                roi_y=roi_y,
                class_to_remove=class_to_remove
            )
```

D. Object Extraction from OpenLABEL

To support metric computation, objects are extracted from each OpenLABEL file into a simplified Python structure containing ID, class, and position.

```python
def load_openlabel_objects(json_path):
    """
    Reads a TUMTraf OpenLABEL JSON and returns a list of objects:
    {"id": "...", "cls": "CAR", "x": ..., "y": ..., "z": ...}
    """
    json_path = Path(json_path)
    with open(json_path, "r") as f:
        data = json.load(f)

    frames = data["openlabel"]["frames"]
    frame_key = list(frames.keys())[0]
    objs_raw = frames[frame_key].get("objects", {})

    objs = []
    for oid, obj in objs_raw.items():
        try:
            odata = obj["object_data"]
            cls = odata.get("type", "UNKNOWN").upper()

            val = odata["cuboid"]["val"]
            x, y, z = val[0], val[1], val[2]

            objs.append({
                "id": oid,
                "cls": cls,
                "x": float(x),
                "y": float(y),
                "z": float(z),
            })
        except:
            continue

    return objs
```

E. Metric Computation per Frame

The function `compute_frame_metrics` compares a baseline JSON and a spoofed JSON, returning all metrics used in the experimental analysis, including object-count variation, phantom influence, scene divergence, and ROI impact.

```python
def compute_frame_metrics(
    json_base,
    json_spoof,
    roi_x=(10, 40),
    roi_y=(-10, 10)
):
    """
    Compares baseline vs. spoofed cooperative perception
    and returns a dictionary of metrics per frame.
    """

    base_objs  = load_openlabel_objects(json_base)
    spoof_objs = load_openlabel_objects(json_spoof)

    n_base  = len(base_objs)
    n_spoof = len(spoof_objs)
    delta_n = n_spoof - n_base

    cnt_base  = Counter([o["cls"] for o in base_objs])
    cnt_spoof = Counter([o["cls"] for o in spoof_objs])

    n_phantoms = cnt_spoof.get("PHANTOM", 0)
    total_phantoms_inserted = n_spoof - n_base if delta_n > 0 else 0
```

```
    PII = (n_phantoms / total_phantoms_inserted
           if total_phantoms_inserted > 0 else 0)

    SDS = abs(n_spoof - n_base) / max(n_base, 1)

    def inside_roi(o):
        return (roi_x[0] <= o["x"] <= roi_x[1] and
                roi_y[0] <= o["y"] <= roi_y[1])

    base_roi  = [o for o in base_objs if inside_roi(o)
        ]
    spoof_roi = [o for o in spoof_objs if inside_roi(o
        )]

    n_base_roi  = len(base_roi)
    n_spoof_roi = len(spoof_roi)
    removed_roi = n_base_roi - n_spoof_roi

    ROI_impact = removed_roi / max(n_base_roi, 1)

    return {
        "frame": Path(json_base).name,
        "n_base": n_base,
        "n_spoof": n_spoof,
        "delta_n": delta_n,
        "n_phantoms": n_phantoms,
        "phantom_infl_idx": PII,
        "scene_divergence": SDS,
        "n_base_roi": n_base_roi,
        "n_spoof_roi": n_spoof_roi,
        "roi_removed": removed_roi,
        "roi_impact": ROI_impact,
        "cnt_base": cnt_base,
        "cnt_spoof": cnt_spoof
    }
```

F. Batch Metric Aggregation

Finally, the `batch_process` function iterates over all frames in a baseline directory, pairs them with the corresponding spoofed files, and aggregates metrics into a single table (Pandas DataFrame) used in the analysis.

```
def batch_process(
    baseline_dir,
    spoof_dir,
    roi_x=(10, 40),
    roi_y=(-10, 10)
):
    baseline_dir = Path(baseline_dir)
    spoof_dir = Path(spoof_dir)

    json_base_files = sorted(baseline_dir.glob("*.json
        "))
    results = []

    for jbase in json_base_files:
        spoof_path = spoof_dir / jbase.name

        if not spoof_path.exists():
            print(f"[WARN] Spoof not found for {jbase.
                name}")
            continue

        metrics = compute_frame_metrics(
            jbase, spoof_path, roi_x, roi_y
        )
        results.append(metrics)

    return pd.DataFrame(results)
```

VII. Experimental Results

This section evaluates the effects of the three implemented spoofing attacks on the cooperative perception annotations of the TUMTraf-V2X Mini dataset. A total of 400 frames were processed for each attack type. Metrics were computed by comparing the original cooperative annotations (baseline) with their manipulated counterparts, as described in Section VI.

We present both quantitative results (Tables IV–VI) and qualitative visualizations (Figures 3–7). The analysis focuses on object-count divergence, spatial degradation, and the effect of each attack on critical regions of the BEV map.

A. Phantom Object Injection

The phantom attack inserts three artificial objects into every frame of the cooperative perception annotation. Since the dataset already provides a fused multi-sensor view, the injected phantoms directly modify the cooperative BEV map without affecting upstream sensing.

Table IV: Phantom Injection Metrics (400 Frames)

Metric	Mean	Std	Range
Δn	3.000	0.000	[3.0, 3.0]
Scene Divergence	0.1159	0.0356	[0.0833, 0.2143]
ROI Impact	-1.7615	0.9952	[-3.0, -0.3]

Interpretation:

- Every frame gains exactly three new objects, producing a stable object-count offset.
- Divergence increases moderately due to clutter inflation.
- ROI Impact is strongly negative, meaning phantoms frequently appear inside the ROI, increasing object density in the most critical region.

Figure 3 shows an illustrative frame with injected phantoms, while Figure 4 presents the divergence distribution across all frames.

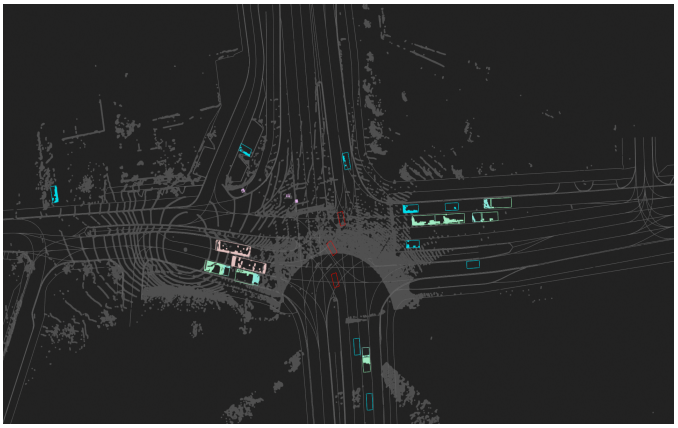


Figure 3: Example of phantom object injection on the cooperative BEV annotation.

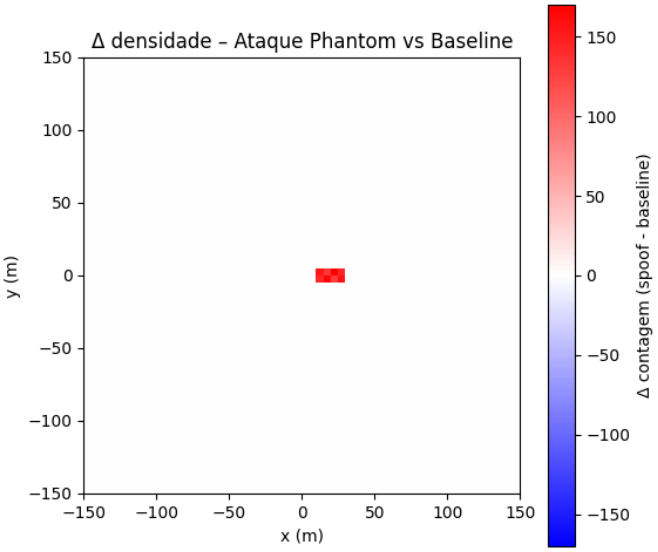


Figure 4: Scene divergence heatmap under phantom injection.

The phantom attack consistently produces *additive distortion*, increasing scene density and generating false obstacles that would influence downstream planning modules in a real system.

B. Random Object Removal

In this attack, a random subset of objects is removed from each cooperative annotation. Removal is uniformly sampled, simulating non-targeted corruption of cooperative perception messages.

Table V: Random Removal Metrics (400 Frames)

Metric	Mean	Std	Range
Δn	-4.8725	2.5793	[-9.0, -1.0]
Scene Divergence	0.1903	0.1247	[0.0286, 0.6000]
ROI Impact	0.1641	0.2808	[0.0, 1.0]

Interpretation:

- On average, 4–5 objects disappear per frame.
- Scene divergence is *higher* than in phantom injection due to variability in the number of removed objects.
- ROI Impact is generally low, reflecting that random deletion rarely targets the most critical areas consistently.

Figure 5 illustrates divergence patterns, which vary significantly due to the stochastic nature of the removal process.

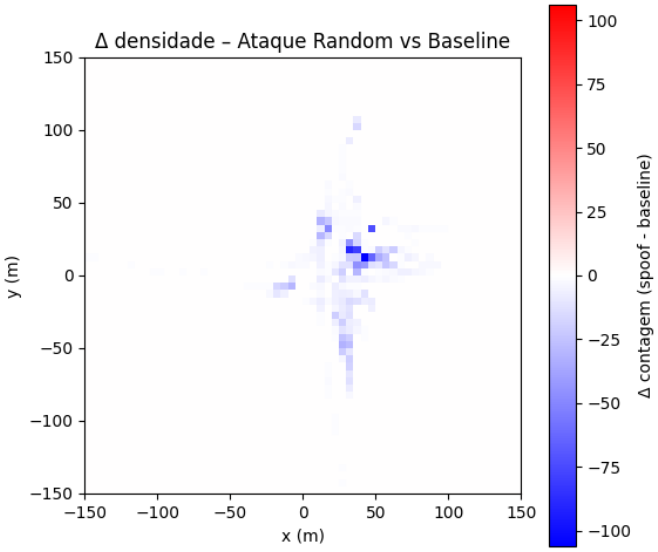


Figure 5: Scene divergence heatmap under random removal.

Random removal produces *non-systematic perception degradation*, which may confuse trackers or planning modules that rely on object persistence.

C. ROI-Based Object Removal

In this targeted attack, objects are removed only if they lie within a predefined spatial Region of Interest (ROI) in the BEV map. This simulates adversarial suppression of safety-critical detections.

Table VI: ROI Removal Metrics (400 Frames)

Metric	Mean	Std	Range
Δn	-0.6875	0.6488	[-3.0, 0.0]
Scene Divergence	0.0279	0.0302	[0.0, 0.1875]
ROI Impact	0.2732	0.3055	[0.0, 1.0]

Interpretation:

- The total number of removed objects is small, producing minimal divergence.
- However, ROI Impact is the highest among all attacks, demonstrating focused removal of critical detections.

- Even a small number of deletions inside the ROI can meaningfully affect cooperative situational awareness.

Figure 6 shows an example where objects inside a central ROI are removed.

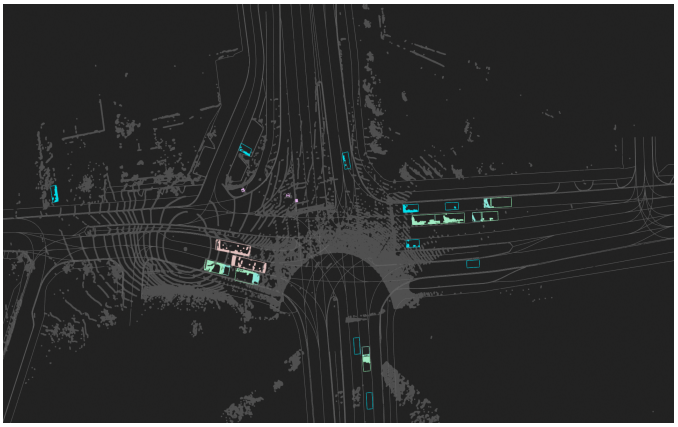


Figure 6: ROI-based removal: baseline vs. manipulated cooperative perception.

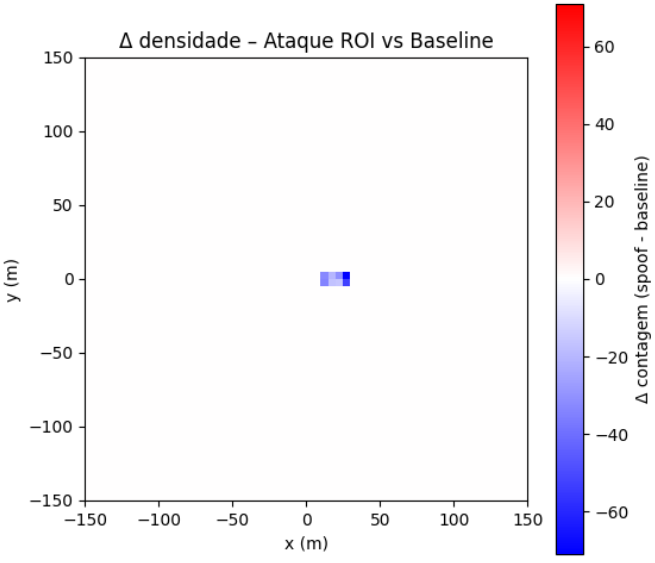


Figure 7: ROI impact across all frames for ROI removal.

D. Comparative Discussion

Figure 8 summarizes the relative impact of the three attacks.

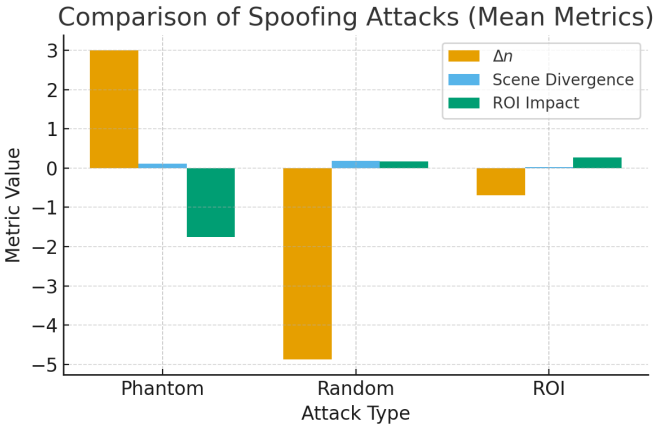


Figure 8: Comparison across spoofing attacks based on mean metrics.

Key insights:

- **Phantom Injection** introduces false obstacles, increasing scene clutter and significantly altering ROI density.
- **Random Removal** creates the largest overall divergence, degrading perception completeness in an unpredictable manner.
- **ROI Removal** has the highest safety impact per removed object, demonstrating that *small but targeted changes* are more dangerous than broad random manipulation.

Together, these findings illustrate how cooperative perception remains vulnerable to both additive and subtractive spoofing, even when no sensor-level manipulation occurs.

VIII. Discussion

The experimental results demonstrate that different categories of spoofing attacks affect the cooperative perception layer in distinct ways, even in the absence of sensor-level manipulation or a fusion module. Because the TUMTraf-V2X Mini dataset provides a pre-assembled cooperative BEV representation, all attacks directly modify the shared annotation layer—highlighting how vulnerable the final perception output becomes once its integrity is compromised.

A. Additive Manipulation: Phantom Injection

Phantom insertion produces a predictable and purely additive distortion: exactly three objects are added to every frame, leading to a stable increase in object-count divergence ($\Delta n = +3$). The scene divergence remains moderate, but the ROI impact becomes strongly negative, indicating that the injected objects frequently appear inside regions that are normally safety-critical.

This attack illustrates that, when cooperative perception is treated as a trusted data product, even simple

annotation-level injection can create false obstacles that would likely trigger unnecessary braking or avoidance maneuvers in downstream planning. Since the dataset does not include additional redundancy checks, every phantom persists (*PII* = 1.0), reinforcing the need for strong authentication and integrity validation of cooperative perception messages.

B. Non-targeted Subtractive Manipulation: Random Removal

Random removal induces the highest overall scene divergence among all evaluated attacks. With an average reduction of nearly five objects per frame, this attack directly degrades the completeness of the cooperative perception space. However, because removed objects are sampled uniformly, the ROI impact remains comparatively low.

From a resilience perspective, this attack highlights the fragility of systems that rely on the continuity and redundancy of cooperative detections. Although random removal is untargeted, its inconsistency across frames may impair tracking, temporal reasoning, and any module that depends on object permanence. The variability in Δn also results in unstable perception quality, which may produce unreliable behavior in downstream prediction modules.

C. Targeted Subtractive Manipulation: ROI Removal

Although ROI-based removal produces the smallest overall divergence of all attacks, its impact is more strategically harmful. The mean object-count change is small ($\Delta n \approx -0.69$), and global divergence remains near zero, which makes the attack difficult to detect using naive scene-level statistics. However, the ROI Impact Score is the highest among all manipulation types, confirming that the removal is disproportionately focused on the most important region of the BEV map.

This attack effectively suppresses safety-critical information such as vehicles directly ahead or cross-traffic in conflict zones. Because even small changes in such regions can dramatically alter risk assessments, this attack demonstrates how selective manipulation is far more dangerous than large-scale untargeted distortion.

D. Comparative Observations

Taken together, the three attacks reveal a spectrum of adversarial behaviors:

- **Phantom Injection** increases scene clutter and produces consistent false positives, which may cause unnecessary evasive maneuvers.
- **Random Removal** yields the largest global distortion, often degrading scene completeness in unpredictable ways.
- **ROI Removal** presents the highest safety relevance: despite minimal global divergence, it selectively deletes the objects that matter most for collision avoidance.

These results indicate that cooperative perception pipelines must be protected not only against additive false information but also against subtractive and spatially targeted manipulations. The findings reinforce that integrity violations at the annotation or communication layer can meaningfully compromise situational awareness, even if the underlying sensors remain uncompromised.

Overall, the experiments highlight a key vulnerability of cooperative perception: once the shared perception layer is assumed trustworthy, annotation-level attacks can significantly alter how the environment is interpreted. Robust defenses will therefore require combining cryptographic integrity, redundancy checks, spatial consistency reasoning, and anomaly detection tailored to the semantics of cooperative BEV representations.

IX. Conclusion and Future Work

This work presented a systematic evaluation of spoofing attacks applied directly to the cooperative perception layer of the TUMTraf-V2X Mini dataset. Unlike studies that focus on sensor-level tampering or on complex fusion architectures, our analysis targeted the integrity of the final cooperative annotation layer, demonstrating how simple manipulations can meaningfully distort the shared BEV representation. The developed attacks—phantom object injection, random removal, and ROI-based removal—were applied over 400 frames each, enabling a quantitative assessment of how additive, subtractive, and spatially targeted modifications affect the structure and reliability of the cooperative scene.

The results reveal that even small perturbations at the annotation level can significantly alter perceived object density, spatial layout, and the visibility of critical participants. Phantom injection consistently increased clutter and introduced plausible but nonexistent obstacles. Random removal produced the largest global divergence, degrading scene completeness in an unpredictable manner. ROI-based removal, while subtle in terms of global metrics, produced the strongest safety-relevant impact by selectively suppressing detections in high-risk regions. Together, these findings highlight that cooperative perception systems remain vulnerable to attacks that modify shared messages or

perception-layer artifacts, even when underlying sensors are uncompromised.

Future work will extend this analysis in several directions. First, evaluating the effects of these attacks on downstream modules—such as tracking, trajectory prediction, risk assessment, and decision-making—would allow a more complete understanding of their operational consequences. Second, integrating temporal consistency checks, redundancy across sensing agents, and physics-aware anomaly detection may provide effective defenses that can be benchmarked against the attacks proposed in this study. Third, applying similar manipulations to larger-scale versions of the TUMTraf dataset or to other cooperative perception frameworks would help validate the generality of the observed trends. Finally, incorporating authenticated V2X communication models or cryptographically verifiable perception pipelines may support the development of more robust cooperative systems capable of detecting or mitigating annotation-level spoofing.

Overall, this work serves as a proof-of-concept demonstration that the cooperative perception layer itself constitutes an attack surface and that even lightweight manipulations can meaningfully alter the perceived traffic environment. The findings reinforce the need for end-to-end integrity mechanisms and structured resilience evaluations in future cooperative and connected autonomous vehicle ecosystems.

References

[1] T. Huang *et al.*, "Vehicle-to-everything cooperative perception for autonomous driving," *Proceedings of the IEEE*, pp. 1–35, 2025.

[2] G. Yu, H. Li, Y. Wang, P. Chen, and B. Zhou, "A review on cooperative perception and control supported infrastructure-vehicle system," *Green Energy and Intelligent Transportation*, vol. 1, no. 3, p. 100023, Dec. 2022.

[3] I. Durlik, T. Miller, E. Kostecka, Z. Zwierzewicz, and A. Łobodzińska, "Cybersecurity in Autonomous Vehicles—Are We Ready for the Challenge?" *Electronics*, vol. 13, no. 13, p. 2654, Jul. 2024. [Online]. Available: https://www.mdpi.com/2079-9292/13/13/2654

[4] A. Yousseef *et al.*, "Autonomous vehicle security: Hybrid threat modeling approach," *IEEE Open Journal of Vehicular Technology*, vol. 6, pp. 1774–1795, 2025.

[5] K. H. M. Gularte *et al.*, "Safeguarding the v2x pathways: Exploring the cybersecurity landscape through systematic review," *IEEE Access*, vol. 12, pp. 72 871–72 895, 2024.

[6] A. S. Da Silva, J. P. J. Da Costa, G. A. Santos, Z. Miri, M. I. B. M. Fauzi, A. Vinel, E. P. De Freitas, and K. Kastell, "Radio jamming in vehicle-to-everything communication systems: Threats and countermeasures," in *2023 23rd International Conference on Transparent Optical Networks (ICTON)*. IEEE, pp. 1–4. [Online]. Available: https://ieeexplore.ieee.org/document/10207422/

[7] G. A. Santos, J. P. J. D. Costa, and A. A. S. D. Silva, "Towards to beyond 5g virtual environment for cybersecurity testing in v2x systems," in *2023 Workshop on Communication Networks and Power Systems (WCNPS)*. Brasilia, Brazil: IEEE, Nov. 2023, pp. 1–7.

[8] S. S. Raghavan, "Threat modelling in automotive cybersecurity: A comprehensive study of frameworks and risk mitigation," *IJCET*, vol. 13, no. 2, pp. 185–205, May 2022.

[9] M. Benyahya, A. Collen, T. Lenard, and N. A. Nijdam, "Tara 2.0 for connected and automated vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 26, no. 8, pp. 11 864–11 878, Aug. 2025.

[10] T. Hussain, M. N. Khan, B. Yang, R. W. Attar, and A. Alhomoud, "Lidar point cloud transmission: Adversarial perspectives of spoofing attacks in autonomous driving," *Computers & Security*, vol. 157, p. 104544, Oct. 2025.

[11] T. Sato *et al.*, "On the realism of lidar spoofing attacks against autonomous driving vehicle at high speed and long distance," in *Proceedings of the 2025 Network and Distributed System Security Symposium (NDSS)*. San Diego, CA, USA: Internet Society, 2025.

[12] Y. Xu, G. Deng, X. Han, G. Li, H. Qiu, and T. Zhang, "Physcout: Detecting sensor spoofing attacks via spatio-temporal consistency," in *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. Salt Lake City, UT, USA: ACM, Dec. 2024, pp. 1879–1893.

[13] T. Huang *et al.*, "Vehicle-to-everything cooperative perception for autonomous driving," *Proceedings of the IEEE*, pp. 1–35, 2025.

[14] J. Wang and T. Nordstrom, "Latency robust cooperative perception using asynchronous feature fusion," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2025.

[15] E. Yazdani Bejarbaneh, H. Du, and F. Naghdy, "Exploring shared perception and control in cooperative vehicle-intersection systems: A review," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 11, pp. 15 247–15 272, Nov. 2024.

[16] S. Yahia, I. Alla, G. B. Mohan, D. Rau, M. Singh, and V. Loscri, "Seeing is deceiving: Mirror-based lidar spoofing for autonomous vehicle deception," *arXiv*, Sep. 2025.

[17] V. Rishiwal, U. Agarwal, A. Alotaibi, S. Tanwar, P. Yadav, and M. Yadav, "Exploring secure v2x communication networks for human-centric security and privacy in smart cities," *IEEE Access*, vol. 12, pp. 138 763–138 788, 2024.