Learning Mutual View Information Graph for Adaptive Adversarial Collaborative Perception

Supplementary Material

Contents

A. Analysis of MVIG Representation

In this section, we provide theoretical analysis of why the unified MVIG representation can effectively reveal vulnerabilities exposed by different CP defense systems.

Overview. Our analysis consists of two complementary parts. Section A.1 establishes the *system-level* foundation through spectral graph analysis, proving that vulnerabilities manifest as specific spectral signatures in MVIG. Section A.2 develops a *region-level* quantification framework using entropy analysis, enabling precise vulnerability measurement at individual spatial locations and adaptation to different defense mechanisms.

A.1. System-Level Vulnerability Characterization

To understand why MVIG can effectively detect vulnerabilities across different CP defenses, we establish a unified spectral framework. This framework combines information theory with graph spectral analysis to prove that vulnerabilities manifest as specific spectral signatures—information flow capacity and consensus fragility—that can be systematically identified through MVIG's structure.

Unified Information-Spectral Framework. Let O_i denote the occupancy matrix observed by CAV i. Following the MVIG construction in the main text, the mutual information between CAVs i and j is:

$$\mathcal{I}(O_i; O_j) = \sum_{a,b=0}^{2} p_{ij}(a,b) \log \frac{p_{ij}(a,b)}{p_i(a)p_j(b)}, \tag{11}$$

where $p_{ij}(a,b)$ is the joint probability and $p_i(a), p_j(b)$ are marginal probabilities. The MVIG edge weights aggregate this across all spatial positions:

$$\mathbf{W}_{ij} = \mathbb{E}_{(x,y)}[\mathcal{I}(O_i; O_j)]. \tag{12}$$

The spectral decomposition $\mathbf{W} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$ reveals the vulnerability structure. We define the *information flow capacity* as:

$$C_{flow} = \frac{\lambda_1}{n-1}, \tag{13}$$

and the *consensus fragility* as:

$$F_{frag} = \frac{1}{\lambda_2 + \epsilon}, \tag{14}$$

where λ_1 is the largest eigenvalue of $\mathbf{W}$ and λ_2 is the second smallest eigenvalue of the Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{W}$.

Vulnerability Characterization Theorem. We establish the fundamental relationship between information asymmetry and attack success:

Theorem 1 (Vulnerability-Spectral Correspondence): A CP system is vulnerable to fabrication attacks if and only if $C_{flow} < \tau_1$ and $F_{frag} > \tau_2$ for system-dependent thresholds τ_1, τ_2.

Proof: (*Necessity*) If a CP system can be successfully attacked, then there must exist information gaps among CAVs (manifested as low C_{flow}) and weak consensus validation mechanisms (manifested as high F_{frag}). Without information gaps, all CAVs share complete knowledge and can detect fabrications through perfect cross-validation. Without weak

consensus, the system can reliably identify inconsistent observations.

(*Sufficiency*) We construct an attack strategy based on spectral analysis. When C_{flow} is low, the principal eigenvector of $\mathbf{W}$ identifies CAVs with weak information coupling, indicating regions where different CAVs have asymmetric views. When F_{frag} is high, the Fiedler vector (eigenvector of the second smallest eigenvalue of Laplacian $\mathbf{L}$) reveals the optimal bipartition of the CAV network, identifying which CAVs can be isolated or misled. By targeting these vulnerable configurations, we can construct fabrications that exploit the identified spectral weaknesses.

The attack success probability can be modeled as:

$$P_{success} = 1 - P_{detection} \approx 1 - \exp\left(-\alpha F_{frag}(1 - C_{flow})\right), \tag{15}$$

where $\alpha > 0$ is a system-dependent constant. This exponential form captures the intuition that detection probability decays exponentially as vulnerabilities increase: higher consensus fragility F_{frag} weakens collective validation, while lower information flow capacity C_{flow} (i.e., higher $1 - C_{flow}$) indicates greater information asymmetry. The exponential relationship models the multiplicative effect where these two factors jointly determine system vulnerability. Note that $C_{flow} \in [0, 1]$ by construction as a normalized metric. □

A.2. Region-Level Vulnerability Quantification

Theorem 1 identifies system-level vulnerability conditions but cannot pinpoint vulnerable regions at specific spatial locations. To enable practical vulnerability assessment, we introduce an entropy-based quantification framework. This framework measures local vulnerabilities through the *entropy deficit* between collective consensus and individual observations, and we prove its effectiveness across different defense mechanisms.

Unified Entropy-Deficit Framework. For a spatial region R, we define the *information entropy deficit* as the fundamental vulnerability measure. The collective uncertainty is:

$$H_c(R) = -\sum_{s=0}^{2} p_c(s|R) \log p_c(s|R), \tag{16}$$

where $s \in \{0, 1, 2\}$ represents occupancy states (free, occupied, unknown) and $p_c(s|R)$ is the consensus probability. The individual entropy of CAV i is:

$$H_i(R) = -\sum_{s=0}^{2} p_i(s|R) \log p_i(s|R). \tag{17}$$

The vulnerability score quantifies the entropy deficit:

$$V(R) = H_c(R) - \frac{1}{n} \sum_{i=1}^{n} H_i(R), \tag{18}$$

which captures the fundamental trade-off between collective uncertainty and individual confidence. When $V(R) > 0$, the collective consensus exhibits higher uncertainty than individual observations, indicating vulnerable regions where attackers can exploit disagreements among CAVs. Conversely, when $V(R) \leq 0$, individual CAVs have high uncertainty while the collective consensus is more certain, typically occurring in regions with sufficient multi-view coverage where attacks are less effective.

Theoretical Analysis and Bounds. We establish rigorous theoretical guarantees for vulnerability detection:

Theorem 2 (Entropy-Deficit Vulnerability Theorem): A region R is vulnerable to fabrication attacks if and only if $V(R) > \tau$ for some threshold $\tau > 0$. Moreover, the attack success probability satisfies:

$$P_{attack}(R) \geq \sigma(\beta V(R) - \tau), \tag{19}$$

where $\sigma(\cdot)$ is the sigmoid function and $\beta > 0$ is a scaling parameter. The sigmoid function is chosen to provide smooth, bounded probability estimates in $[0, 1]$ and ensure differentiability for gradient-based optimization. The threshold τ is empirically determined based on the defense system's validation mechanism and can be estimated from training data by analyzing the entropy deficit distribution in successful attack regions.

Proof: The necessity follows from information-theoretic arguments: successful attacks require exploiting regions where collective consensus is weak relative to individual confidence. For sufficiency, we construct an explicit attack that targets the entropy deficit. The bound follows from the relationship between entropy deficit and the probability of consensus failure, where the sigmoid function models the smooth transition between secure and vulnerable regions as the entropy deficit increases. □

Defense-Adaptive Information Aggregation. The MVIG framework adapts to different defense mechanisms through information-theoretic principles:

For occupancy-based defenses (e.g., CAD), the mutual information computation becomes:

$$\mathcal{I}_{CAD}(O_i; O_j) = \sum_{(x,y) \in R} \mathcal{I}(O_i(x, y); O_j(x, y)), \tag{20}$$

providing fine-grained spatial analysis. For feature-based defenses, we employ Blind Region Segmentation with bounded estimation error:

$$\mathcal{I}_{BRS}(\hat{O}_i; \hat{O}_j) = \mathcal{I}(\hat{O}_i; \hat{O}_j) + \epsilon, \tag{21}$$

where $|\epsilon| \leq \delta$ and δ depends on the BRS algorithm accuracy.

Corollary 1 (Defense-Invariant Vulnerability): Despite estimation errors, the vulnerability detection remains robust: if $V_{true}(R) > \tau + 2\delta$, then $V_{est}(R) > \tau$ with probability at least $1 - \exp(-\gamma\delta^2)$ for some $\gamma > 0$.

B. Loss Function

B.1. MVIGNet Loss Function

We train the MVIGNet using a multi-objective loss function that evaluates the effectiveness of the mask positions:

$$\mathcal{L}(\theta;\mathcal{D}) = \alpha\mathcal{L}_{\text{a}}(\theta;\mathcal{D}) + \beta\mathcal{L}_{\text{b}}(\theta;\mathcal{D}) + \gamma\mathcal{L}_{\text{d}}(\theta;\mathcal{D}), \quad (22)$$

where θ represents MVIGNet parameters, $\mathcal{D}$ is the training dataset, and $\alpha, \beta, \gamma \in \mathbb{R}^+$ are balancing coefficients. The three loss components address distinct aspects of attack optimization:

Attack Effectiveness Loss. Let $\mathcal{S}$ and $\mathcal{R}$ denote spoofing and removal attacks respectively. The attack effectiveness loss $\mathcal{L}_{\text{a}}$ is defined as: For spoofing attacks ($\mathcal{A} = \mathcal{S}$), the attack effectiveness loss maximizes detection confidence:

$$\mathcal{L}_{\text{a}}(\mathcal{S}) = \sum_{b\in B'} \text{IoU}(b, b_t) \cdot \log(1 - b_\sigma) + \lambda_d d(b, v), \quad (23)$$

For removal attacks ($\mathcal{A} = \mathcal{R}$), the attack effectiveness loss minimizes detection confidence:

$$\mathcal{L}_{\text{a}}(\mathcal{R}) = -\sum_{b\in B'} \text{IoU}(b, b_t) \cdot \log(1 - b_\sigma) + \lambda_d d(b, v), \quad (24)$$

where B' denotes the set of bounding box proposals after applying the perturbation, b_σ is the confidence score associated with proposal b, b_t represents the target region to attack, and $d(b, v)$ is the spatial distance between the proposed box and the victim CAV position with λ_d as a weighting factor. This objective function either maximizes or minimizes the confidence scores of proposals overlapping with the target region, depending on whether we're performing a spoofing or removal attack, while penalizing attacks that are too distant from the victim CAV to ensure attacks occur within effective perception range.

Box Differentiation Loss. To ensure that spoofed objects appear distinct from existing objects, we apply a box differentiation loss:

$$\mathcal{L}_{\text{b}}(\mathcal{A}) = \begin{cases} \max_{b_i\in B} \text{IoU}(b_t, b_i), & \mathcal{A} = \mathcal{S} \\ 0, & \mathcal{A} = \mathcal{R} \end{cases} \quad (25)$$

where B represents the set of all detected bounding boxes. This loss penalizes spoofed objects that overlap significantly with existing objects, encouraging the creation of distinct new objects rather than modifications to existing ones. For removal attacks, this loss is not applicable as the goal is to remove existing objects rather than create new ones.

Defense Evasion Loss. To evade defense mechanisms that rely on occupancy map cross-verification, we implement a defense-aware loss component:

$$\mathcal{L}_{\text{d}} = \sigma(\eta(\tau - d_{\min}(E_{\text{conflict}}, b_t))) \quad (26)$$

where σ is the sigmoid function ensuring smooth gradients, η is a scaling factor controlling the steepness of the penalty, τ is a safety threshold distance, $d_{\min}(E_{\text{conflict}}, b_t) = \min_{e\in E_{\text{conflict}}} \|e - b_t\|_2$ computes the minimum Euclidean distance between the target box center b_t and any conflict region e, and E_{conflict} represents the identified conflict areas. This loss encourages attacks to maintain sufficient distance from regions that would trigger defense systems' inconsistency detection.

Visualization of MVIGNet Loss Curves. Figure 7 shows the validation loss curves of different components during MVIGNet training. For the MVIG-Spoof attack (left), we observe that the attack loss stabilizes around 0.35 after approximately 20 epochs, while defense loss rapidly decreases to near 0.05, indicating effective evasion of defense mechanisms. The box difference loss remains consistently low throughout training, ensuring spatial accuracy of injected objects. In contrast, the MVIG-Removal attack (right) exhibits a different pattern where defense loss dominates (around 0.2), suggesting that evading detection of removed objects is more challenging than spoofing new ones. The attack loss converges to a lower value (approximately 0.07), demonstrating that the model efficiently learns to remove existing objects. In both attack types, the total loss shows initial volatility but stabilizes after 20 epochs, with the removal attack requiring slightly more training iterations to converge. The shaded regions represent confidence intervals across multiple training runs, confirming the consistency and reproducibility of our approach.

B.2. PGD Loss Function

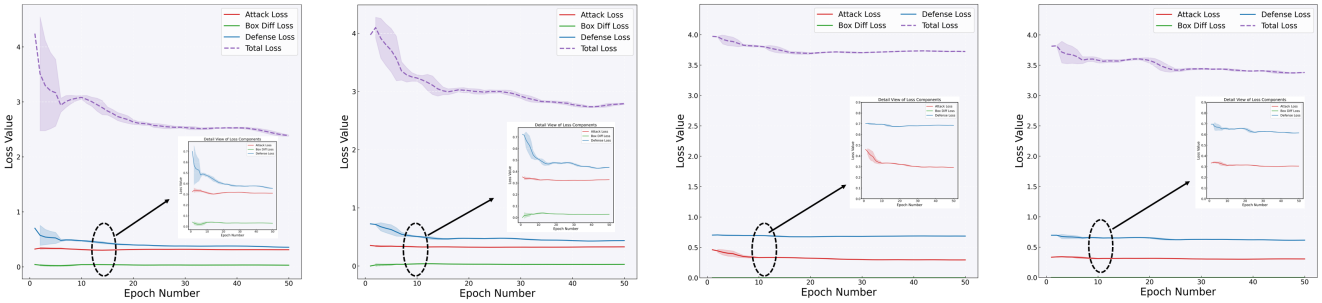
The three loss components described above ($\mathcal{L}_{\text{a}}$, $\mathcal{L}_{\text{b}}$, and $\mathcal{L}_{\text{d}}$) train our MVIGNet to predict optimal mask locations. While PGD optimization uses a loss function functionally similar to $\mathcal{L}_{\text{a}}$, the two procedures operate independently. Once MVIGNet identifies the optimal attack mask M_t^*, we generate the feature perturbation through two steps. First, we initialize it based on the predicted mask:

$$\delta_t^{(0)} = \Phi_\psi(M_t^*), \quad (27)$$

where $\Phi_\psi : \{0, 1\}^{H\times W} \to \mathbb{R}^d$ maps the binary mask to feature space. Then, we apply PGD to refine the perturbation:

$$\delta_t^{(k+1)} = \Pi_{\Omega_{M_t^*}}(\delta_t^{(k)} - \alpha \cdot \nabla_{\delta_t^{(k)}}\phi(\delta_t^{(k)})), \quad (28)$$

where α is the step size, $\nabla_{\delta_t^{(k)}}\phi(\delta_t^{(k)})$ is the gradient of PGD loss, and $\Pi_{\Omega_{M_t^*}}$ projects the perturbation onto the feasible region. Note that while PGD typically performs gradient ascent for maximization, we use gradient descent here because the PGD loss ϕ is formulated as a minimization objective that aligns with $\mathcal{L}_{\text{a}}$, i.e., minimizing ϕ corresponds to maximizing detection confidence for spoofing or minimizing it



(a) MVIG Spoof ($d_{max} = 15$) (b) MVIG Spoof ($d_{max} = 25$) (c) MVIG Remove ($d_{max} = 25$) (d) MVIG Remove ($d_{max} = 50$)

Figure 7. Validation loss curves of MVIGNet training on OPV2V dataset.

for removal attacks. The final perturbation δ_t is added to the original feature map $\mathbf{F}_a^t$ to produce the manipulated feature $\mathbf{F}_a^t + \boldsymbol{\delta}$ sent to victim CAVs.

C. More Implementation Details

C.1. Evaluation Metrics

Attack Success Rate (ASR): This metric quantifies the effectiveness of an attack in a scenario without any defense mechanism in place. It is calculated as the percentage of attack attempts that successfully mislead the ego vehicle's perception, for instance, by causing it to detect a non-existent object (spoofing) or miss a real one (removal). A higher ASR indicates a more potent attack.

Defense Success Rate (DSR): Conversely, in scenarios where a defense mechanism is active, DSR measures the percentage of incoming attacks that are correctly identified and neutralized by the defender. From the attacker's perspective, a lower DSR is desirable as it signifies that the attack is more successful at evading detection.

ΔAP@50: This metric measures the degradation in the victim's object detection performance caused by an attack. AP (Average Precision) is a standard metric for detection accuracy, and the IoU (Intersection over Union) threshold is set to 50%. ΔAP@50 represents the change (typically a drop) in AP, providing a clear measure of the attack's impact on the perception system's reliability.

True Positive Rate (TPR): Also known as sensitivity or recall, TPR measures the proportion of actual malicious data transmissions that are correctly flagged as attacks by the defense system. A high TPR is crucial for a robust defense.

False Positive Rate (FPR): This measures the proportion of legitimate, benign data transmissions that are incorrectly identified as malicious by the defense system. A high FPR is undesirable as it leads to the unnecessary rejection of valid data, which can degrade the overall performance and benefits of collaborative perception.

Frames Per Second (FPS): This metric indicates the computational efficiency of an attack or defense algorithm by measuring how many frames of data it can process per second. For real-world autonomous driving applications, a high FPS is essential to ensure that the system can operate in real-time without introducing dangerous latency.

C.2. Baselines

Basic Feature Attack [28]: This is a foundational, untargeted attack method that directly perturbs the feature maps shared between vehicles. It acts as a baseline for adversarial capability but lacks sophistication, as the perturbations are not optimized for stealth or for specific regions, making it relatively easy to detect.

Blind Area Confusion (BAC) Attack [26]: A more advanced untargeted attack, BAC leverages knowledge of the victim's field of view to constrain its perturbations primarily to the victim's blind spots. This spatial constraint increases the attack's stealthiness compared to the basic feature attack.

Ray-Casting (RC) Attack [35]: A targeted fabrication attack that aims to create a specific "ghost" vehicle at a chosen location. While it can generate subtle and effective perturbations, its primary limitation is the lack of a strategic mechanism to optimize the attack's timing and target region, which our `MVIG` attack is designed to overcome.

CAD (Collaborative Anomaly Detection) [35]: A defense system that establishes a consensus by comparing the occupancy maps from different collaborating vehicles. Anomalies, indicated by significant disagreements between maps, are flagged as potential attacks. Its effectiveness relies on having sufficient overlapping views.

ROBOSAC [21]: A defense framework built on the robust RANSAC (Random Sample Consensus) algorithm, which treats malicious data as outliers to a consensus model derived from a random subset of collaborators. The enhanced variant, ROBOSAC+, incorporates temporal verification, checking for object consistency across consecutive frames to counter transient attacks.

CP-Guard [14]: This defense system uses PASAC (Probability-Agnostic Sample Consensus), an improvement over traditional RANSAC that does not require prior assumptions about the distribution of malicious agents. This makes it more adaptive and robust in scenarios with varying

Table 3. Defense success rates (%) across different CP architectures. Lower values indicate better attack effectiveness.

CP Architecture	ROBOSAC	CP-Guard	GCP	CAD
AttFuse (Full)	14.8	17.2	13.0	32.0
V2VNet (Full)	14.9	17.1	13.5	32.2
Where2comm (Sparse)	15.2	18.1	13.6	33.1

numbers of attackers.

GCP (Guarded Collaborative Perception) [26]: A sophisticated defense mechanism that performs rigorous spatial-temporal consistency checks across multiple frames. It verifies the plausibility of detected objects by analyzing their position and movement over time, making it challenging for fabricated objects that lack realistic trajectories to remain undetected.

D. More Experimental Results

D.1. Visualization of Feature Maps

Figure 8 visualizes feature maps under different attacks. Basic feature attack and BAC attack generate untargeted perturbations covering widespread areas, making them easily detectable. In contrast, RC attack and MVIG attack focus perturbations within specific target regions. MVIG attack tends to identify and exploit LiDAR-sparse regions near obstacles for spoofing, as these locations are likely mutual blind spots to all benign CAVs.

D.2. Results with Different CP Architectures

To evaluate the generalizability of our MVIG attack across different CP architectures, we conducted additional experiments on mainstream CP models including AttFuse [35], V2VNet [30], and Where2comm [15]. Table 3 shows the defense success rates of different defenders against MVIG spoof attack under various CP architectures. The results demonstrate that our MVIG attack maintains consistent effectiveness across different CP architectures, with only minor variations in performance. This indicates that our attack framework is not sensitive to specific architectural choices and can generalize well to various feature fusion strategies, including both full and sparse feature transmission methods.

D.3. Scalability Analysis

We evaluated the scalability of our MVIG attack by varying the number of benign CAVs in the CP network. Table 4 shows the defense success rates against MVIG spoof attack as the number of participating benign CAVs increases. As expected, we observe a slight degradation in MVIG attack performance as the number of benign CAVs increases, particularly against the CAD defender which benefits from more comprehensive occupancy map validation. However, our MVIG attack still maintains good effectiveness (less than 37% defense success rate against CAD) within current dataset limitations, where most existing CP datasets contain at most 2-5 connected agents.

Table 4. Defense success rates (%) with varying numbers of benign CAVs. Lower values indicate better attack effectiveness.

No. Benign CAVs	ROBOSAC	CP-Guard	GCP	CAD
1	13.6	16.4	12.5	15.1
2	14.8	17.2	13.0	32.0
3	14.9	18.0	13.2	36.2

Table 5. Defense success rates (%) under different BRS error rates. Lower values indicate better attack effectiveness.

BRS Error Rate	ROBOSAC	CP-Guard	GCP	CAD
10%	14.8	17.0	13.0	34.2
20%	15.1	17.4	13.1	40.5
30%	15.2	17.3	13.2	52.1

D.4. Robustness to Occupancy Map Errors

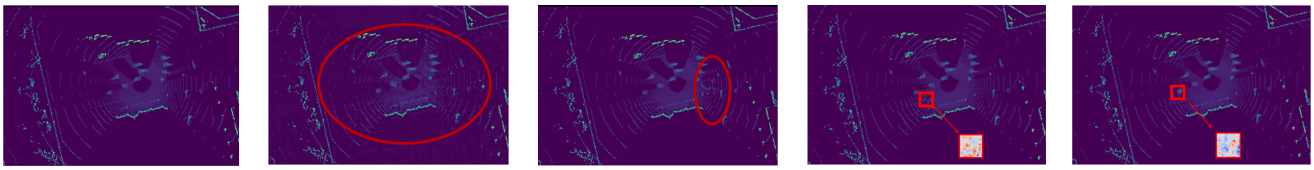
We conducted experiments to evaluate MVIG's sensitivity to occupancy map corruption caused by blind region segmentation (BRS) estimation errors. This is particularly relevant when attackers must estimate occupancy information from feature maps rather than accessing precise occupancy maps directly. Table 5 shows the defense success rates under different BRS error rates. The results show that our MVIG attack is not sensitive to occupancy map corruption caused by BRS estimation errors for most defenders (ROBOSAC, CP-Guard, GCP). This robustness stems from the fact that these defenders rely on holistic detection map similarity rather than fine-grained occupancy validation. For CAD defense, which requires precise occupancy maps, our attack can still tolerate estimation error rates up to 30% while maintaining reasonable effectiveness. However, it is worth noting that CAD defense inherently requires CAVs to transmit precise occupancy maps for validation, meaning attackers would not encounter such high estimation errors in practice. Therefore, our MVIG attack remains robust against all current CP defensive systems.

E. More Discussions

E.1. Attack Effectiveness Analysis

We provide deeper analysis of why MVIG spoof attacks demonstrate superior performance compared to other attack methods and removal attacks:

MVIG vs. Baseline Attacks: MVIG's superior performance against defenses stems from its strategic utilization of implicit view confidence information in CP messages to target



(a) Unperturbed Feature Map (b) Basic Feature Attack (c) BAC Attack (d) RC Attack (e) MVIG Attack

Figure 8. Feature map comparison of different attack methods on Adv-OPV2V dataset.

mutual vulnerable regions of multiple CAVs. Unlike baseline attacks that act as "blind strikes" without considering when and where to attack, MVIG operates like a "strategic sniper" that first identifies shared uncertain areas among neighboring benign CAVs through feature map or occupancy data analysis, then strategically guides perturbations towards these mutual vulnerabilities.

MVIG Spoof vs. Removal: MVIG spoof attacks show better performance than removal attacks due to larger optimization search space of candidate fabrication regions. Removal attacks must target existing objects, making the candidate fabrication region quite limited, while spoof attacks can explore broader areas for object placement.

CAD Defense Analysis: CAD shows better defense performance because it performs fine-grained grid-by-grid validation using additional precise occupancy maps that are required to be exchanged among CAVs. In contrast, other defenses like ROBOSAC and CP-Guard do not require exchanging such validation data and only rely on comparing holistic detection map similarity between neighboring CAVs and ego CAV to identify anomalies.

E.2. Potential Countermeasures Discussion

Based on our analysis of `MVIG` attack mechanisms, we identify several potential countermeasure directions:

Confidence Pattern Obscuring: A potential defense could involve obscuring confidence patterns in transmitted data using differential privacy-based methods to inject calibrated noise on feature maps or occupancy maps, or employing confidence quantization techniques to disrupt MVIG's ability to identify mutual vulnerable regions.

Randomized View Sharing: Implementing randomized or encrypted view sharing protocols could make it more difficult for attackers to consistently analyze mutual view information across multiple frames.

Temporal Anomaly Detection: Enhanced temporal consistency checks that monitor for sudden changes in collaboration patterns or unexpected persistent objects could help detect `MVIG` attacks.

However, these countermeasures create a fundamental trade-off between collaborative perception accuracy and attack resilience, representing an interesting direction for future research in secure collaborative perception systems.

E.3. Communication Delay

In our work, we follow the standard assumption adopted by prior CP security literature [21, 26, 35] by focusing on low-latency CP systems where feature map transmission and caching can be accomplished within one frame (typically less than 100ms). An ego CAV collects feature maps from neighboring CAVs based on timestamps that fall into a certain frame time window.

For high-latency scenarios, prior works focusing on communication issues in CP systems [17, 31] have demonstrated that latency can be addressed through feature prediction and compensation techniques. Our `MVIG` attack can be readily adapted to such high-latency scenarios through two approaches: (1) incorporating feature prediction and compensation into our attack optimization process, or (2) strategically waiting for better channel conditions to initiate attacks when the latency is reduced.

Synchronization Considerations. In our threat model, we consider an internal attacker scenario where the attacker has successfully compromised a legitimate CAV's data access rights. In this scenario, the attacker can normally interact with neighboring benign CAVs to exchange feature maps and occupancy maps (if shared) until it is identified as a malicious agent by defenders. The attacker leverages the same synchronization mechanisms as legitimate CAVs, accessing the same timestamp-based frame windows used by the CP system. This eliminates the need for additional synchronization capabilities beyond what the CP system already provides.

Real-World Deployment. Modern V2X communication systems typically achieve latencies of 20-100ms under normal conditions, which aligns well with our assumption. For scenarios where network conditions degrade, the attacker can employ adaptive strategies such as monitoring channel quality and postponing attacks until conditions improve, or adjusting the prediction horizon m to accommodate longer transmission delays. These practical considerations do not fundamentally change our attack framework but rather represent operational parameters that can be tuned based on deployment conditions.

Table 6. Runtime comparison across different attack (FPS).

Methods	0-frame	1-frame	3-frame
Basic Attack [28]	24.4	24.4	24.4
RC Attack [35]	17.9	24.2	33.8
BAC Attack [26]	10.8	15.0	18.4
MVIG Attack (Ours)	15.6	21.3	29.9

F. Overhead Analysis

We analyze the computational overhead of our MVIG attack and compare it with baseline methods. All experiments are conducted on a server with Intel Xeon Silver 4410Y CPU and NVIDIA RTX A5000 GPU.

F.1. Runtime Performance

Table 6 shows the frame rates (FPS) achieved by different attack methods under various persistence settings. The persistence parameter p represents the number of consecutive frames covered by a single optimization, with higher values indicating more efficient resource utilization. Our MVIG attack achieves 15.6-29.9 FPS, demonstrating real-time feasibility for practical deployment.

F.2. Resource Requirements

Our MVIG attack requires modest computational resources:

- **GPU memory**: 5.6 GB (2.6 GB for mask generation + 3.0 GB for PGD optimization)
- **Processing time**: 115 ms total (14 ms MVIGNet + 88 ms PGD + 13 ms others)
- **CPU operations**: Feature mapping and coordinate transformations

This makes our attack feasible on automotive-grade hardware like NVIDIA DRIVE AGX Thor (250 TFLOPS, 16GB memory) and even mainstream edge devices like Jetson Orin NX (3.8 TFLOPS, 16GB memory).

F.3. Prediction Horizon Selection

To ensure online attack feasibility, we must complete all preparation work before the target frame arrives. As shown in Table 7, a single MVIG attack instance comprises four operations: mask generation, PGD optimization, attack transformation, and feature mapping. For the OPV2V dataset operating at 10 FPS (100 ms per frame), setting the prediction horizon to $m = 2$ provides a 200 ms time buffer, which is sufficient to complete all attack operations before the target frame arrives. Specifically, with $m = 2$:

1. The MVIGNet processes the current frame and predicts attack positions in 14 ms.
2. PGD optimization for initial mask generation requires 88 ms.
3. Attack transformation takes 3 ms.
4. Other operations (feature mapping, coordinate transformation, etc.) require 10 ms.

The total processing time of 115 ms is well within the 200 ms time window provided by a 2-frame prediction horizon, ensuring our attack can operate in real-time while maintaining effectiveness. Larger values of m would increase prediction uncertainty without providing significant timing benefits, while $m = 1$ would be insufficient to complete all operations before the target frame arrives.

F.4. Component-wise Analysis

Table 7 provides detailed breakdown of computational costs for different attack components across methods.

G. Fabrication Risk Map Generation

During inference, we employ Gaussian kernel-based contrast enhancement as test-time augmentation to improve attack position decision robustness. This process refines the masked score map $\hat{\mathbf{S}}_{t+m} \in \mathbb{R}^{H\times W}$ into a risk representation $\tilde{\mathbf{S}}_{t+m} \in [0,1]^{H\times W}$ that highlights vulnerability regions while suppressing noise. A simple normalization approach would define the risk map as:

$$\mathbf{R}_d = \frac{\hat{\mathbf{S}}_{t+m}}{\|\hat{\mathbf{S}}_{t+m}\|_\infty}, \tag{29}$$

where $\|\cdot\|_\infty$ denotes the maximum norm. However, this elementary transformation fails to capture intrinsic spatial correlations and vulnerability variations, potentially leading to inconsistent attack decisions. Our enhanced processing pipeline consists of three steps. First, we apply Gaussian kernel smoothing:

$$\begin{aligned}\mathbf{S}_s &= (\mathcal{G}_\sigma * \hat{\mathbf{S}}_{t+m})(x,y) \\ &= \iint_{\mathbb{R}^2} \hat{\mathbf{S}}_{t+m}(u,v)\cdot G_\sigma(x-u, y-v)\,du\,dv,\end{aligned} \tag{30}$$

where the Gaussian kernel $G_\sigma(x,y)$ is defined as:

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right), \tag{31}$$

where $\sigma = 4.0$ controls the spatial correlation scale. This value corresponds to approximately 2 grid cells $(4.0/2.0)$ in our BEV representation (grid resolution 0.4m), capturing local vulnerability patterns while filtering high-frequency noise from MVIGNet predictions. Next, we perform contrast enhancement:

$$\mathbf{S}_c = \mathcal{T}_\gamma(\mathbf{S}_s) = \left(\frac{\mathbf{S}_s}{\|\mathbf{S}_s\|_\infty}\right)^{1/\gamma}, \tag{32}$$

with $\gamma = 2.5$. This power-law transformation with $\gamma > 1$ compresses high values and expands low values, enhancing

Table 7. Runtime comparison of different attack methods. The persistence parameter p represents the number of consecutive frames covered by a single optimization. N/A indicates the process is not part of the attack, while Offline indicates the process is pre-computed and not included in real-time application.

Method	**Component Runtime**					**Avg. FPS**		
	Mask Gen.	Ray-Cast	PGD Opt.	Attack Trans.	Others	$p=0$	$p=1$	$p=3$
Basic Attack [28]	N/A	N/A	40 ms	N/A	1 ms	24.4	24.4	24.4
BAC Attack [26]	49/(1+p) ms	N/A	42 ms	N/A	2 ms	10.8	15.0	18.4
RC Attack [35]	N/A	Offline	88/(2+p) ms	3 ms	9 ms	17.9	24.2	33.8
MVIG Attack (Ours)	14/(2+p) ms	Offline	88/(2+p) ms	3 ms	10 ms	15.6	21.3	29.9

the contrast between vulnerable and safe regions—critical for distinguishing subtle vulnerability gradients in complex urban driving scenarios. Finally, we apply a threshold operation:

$$\tilde{\mathbf{S}}_{t+m} = \Psi_\tau(\mathbf{S}_c) = \frac{\max(\mathbf{S}_c - \tau, 0)}{\|\max(\mathbf{S}_c - \tau, 0)\|_\infty + \epsilon}, \tag{33}$$

where $\tau = 0.3$ is the risk threshold and ϵ is a small constant for numerical stability. This threshold filters out low-confidence regions, ensuring attacks target only high-vulnerability areas where collective perception is genuinely weak. The complete transformation is expressed as:

$$\tilde{\mathbf{S}}_{t+m} = (\Psi_\tau \circ \mathcal{T}_\gamma \circ \mathcal{G}_\sigma)(\hat{\mathbf{S}}_{t+m}). \tag{34}$$

By applying this enhancement during inference, we achieve three key properties: (1) spatial coherence through Gaussian diffusion, (2) dynamic range expansion via contrast enhancement, and (3) noise suppression through thresholding. The parameters (σ, γ, τ) were optimized to maximize mutual information between original and enhanced maps while minimizing noise entropy, making our attack decisions more reliable in complex urban environments.

H. Entropy-Aware Vulnerability Search

To efficiently maintain attack persistence across frames while ensuring temporal consistency, we develop an entropy-aware vulnerability search strategy that identifies optimal attack transformations across spatiotemporal dimensions by maximizing the information entropy deficit between collective consensus and individual CAV observations. Given the current mask position $\mathbf{M}_{t+m+j-1}$ and the fabrication risk map $\tilde{\mathbf{S}}_{t+m}$ at the initial prediction time $t+m$, our transformation operator $\mathcal{T}_j$ projects the mask to positions for each of the j subsequent frames while ensuring temporal consistency.

Entropy-Driven Search Direction. The fabrication risk map $\tilde{\mathbf{S}}_{t+m}$ is a learned representation of the entropy deficit $V(R)$ quantified in Theorem 2, where higher scores indicate regions with greater collective uncertainty relative to individual confidence. The transformation process begins by identifying the center of mass of the current mask $\mathbf{M}_{t+m+j-1}$, denoted as (x_c, y_c). We compute an entropy-aware search direction by combining two critical vector fields: the current velocity field $\mathbf{v}_{\text{cur}} \in \mathbb{R}^2$ derived from historical positions, and the entropy gradient field $\nabla\tilde{\mathbf{S}}_{t+m}$ that points toward regions of increasing vulnerability (i.e., higher entropy deficit). The ideal search direction $\mathbf{d}_{\text{ideal}} \in \mathbb{R}^2$ is expressed as:

$$\mathbf{d}_{\text{ideal}} = \frac{\alpha\mathbf{v}_{\text{cur}} + (1-\alpha)\nabla\tilde{\mathbf{S}}_{t+m}(x_c, y_c)}{||\alpha\mathbf{v}_{\text{cur}} + (1-\alpha)\nabla\tilde{\mathbf{S}}_{t+m}(x_c, y_c)||}, \tag{35}$$

where the blending coefficient $\alpha \in [0.7, 0.95]$ is adaptively determined based on the angular coherence between velocity and gradient vectors:

$$\alpha = f\left(\cos^{-1}\left(\frac{\mathbf{v}_{\text{cur}} \cdot \nabla\tilde{\mathbf{S}}_{t+m}(x_c, y_c)}{||\mathbf{v}_{\text{cur}}|| \cdot ||\nabla\tilde{\mathbf{S}}_{t+m}(x_c, y_c)||}\right)\right). \tag{36}$$

The function $f : [0, \pi] \to [0.7, 0.95]$ maps angular differences to blending coefficients, with smaller angular differences resulting in lower α values (allowing the entropy gradient field to exert greater influence), while larger angles favor preserving motion momentum. The GradS function samples the fabrication risk map in the forward-facing sector around (x_c, y_c) to approximate the gradient direction that maximizes the entropy deficit, effectively identifying the steepest ascent toward vulnerable regions.

Entropy-Maximizing Position Selection. Within a trajectory-constrained search space $\Omega_{\mathbf{d}}(x_c, y_c, \delta) \subset \mathbb{R}^2$ defined along the ideal direction, where $\delta \in \mathbb{R}^+$ represents the maximum search radius, we identify the optimal position that maximizes the entropy deficit while respecting physical motion constraints:

$$(x^*, y^*) = \underset{(x,y)\in\Omega_{\mathbf{d}}(x_c,y_c,\delta)}{\arg\max} \left[\tilde{\mathbf{S}}_{t+m}(x, y) + \mathcal{R}(x, y)\right], \tag{37}$$

where $\tilde{\mathbf{S}}_{t+m}(x, y)$ quantifies the entropy deficit at position (x, y) (i.e., the vulnerability score), and the reward function $\mathcal{R} : \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ captures directional coherence, velocity consistency, and historical motion alignment with respect to the ideal direction $\mathbf{d}_{\text{ideal}}$ and current velocity $\mathbf{v}_{\text{cur}}$. To maintain physical realism in autonomous driving scenarios, the reward function penalizes attack transformations that

violate motion constraints of typical traffic participants:

$$\mathcal{R}(x,y) = \beta_d \mathcal{D}(x,y) + \beta_v \mathcal{V}(x,y) + \beta_h \mathcal{H}(x,y). \quad (38)$$

Physical Interpretation. Each reward component enforces realistic motion constraints typical of urban vehicle traffic:

The directional consistency component $\mathcal{D}$ penalizes abrupt heading changes that would be physically implausible for vehicles. In autonomous driving, vehicles cannot instantaneously change direction; they follow smooth trajectories constrained by steering mechanics and lane geometry:

$$\mathcal{D}\left((x,y), \mathbf{d}_{\text{ideal}}\right) = 1.0 - \frac{|\angle(\mathbf{d}_{\text{ideal}}, (x - x_c, y - y_c))|}{\theta_{\max}}, \quad (39)$$

where $\angle(\cdot,\cdot)$ computes the angle between vectors and $\theta_{\max}$ is the maximum allowed angular deviation. This enforces smooth steering behavior, preventing unrealistic sharp turns that would immediately raise suspicion.

The velocity consistency component $\mathcal{V}$ enforces speed coherence across frames, mimicking constant or gradually changing velocities typical of real vehicles rather than erratic speed fluctuations:

$$\mathcal{V}\left((x,y),(x_c,y_c)\right) = 1.0 - \frac{|d_{\text{ideal}} - \|(x - x_c, y - y_c)\|_2|}{d_{\max}}, \quad (40)$$

where d_{ideal} represents the expected displacement based on current velocity (typically $v \cdot \Delta t$, where Δt is the inter-frame time) and $d_{\max}$ bounds acceptable acceleration/deceleration ranges. This prevents teleportation-like jumps that violate physics.

The historical motion alignment component $\mathcal{H}$ ensures momentum conservation, rewarding trajectories that continue in directions consistent with recent motion history—reflecting vehicle inertia and driver intent continuity:

$$\mathcal{H}\left((x,y), \mathbf{v}_{\text{cur}}\right) = \frac{\mathbf{v}_{\text{cur}} \cdot (x - x_c, y - y_c)}{\|\mathbf{v}_{\text{cur}}\|_2 \cdot \|(x - x_c, y - y_c)\|_2}, \quad (41)$$

where $\mathbf{v}_{\text{cur}}$ is the normalized velocity vector from previous frames. High $\mathcal{H}$ values indicate forward motion along the established trajectory, while negative values would suggest unnatural backward motion.

The coefficients $\beta_d, \beta_v, \beta_h \in \mathbb{R}^+$ control the relative importance of each component, balancing physical plausibility with attack effectiveness. The mask is then translated to this new position while preserving its shape through the $\mathcal{T}_M$ function:

$$\mathbf{M}_{t+m+j} = \mathcal{T}_M(\mathbf{M}_{t+m+j-1}, x^* - x_c, y^* - y_c), \quad (42)$$

where $\mathcal{T}_M : \{0,1\}^{H\times W} \times \mathbb{R} \times \mathbb{R} \to \{0,1\}^{H\times W}$ implements a rigid translation of the binary mask. For each subsequent frame, we evaluate whether to continue the attack based on the expected risk score in a neighborhood around the current mask center:

$$\mathcal{C}_{t+m+j} = \mathbb{I}\left[\mathbb{E}_{(x,y)\in\mathcal{N}(x_c,y_c)}[\tilde{\mathbf{S}}_{t+m}(x,y)] \geq \eta\right], \quad (43)$$

where $\mathbb{I}[\cdot]$ is the indicator function, $\mathbb{E}[\cdot]$ denotes the expectation operator, and η is the continuation threshold. The final attack mask becomes $\mathbf{M}^{\text{final}}_{t+m+j} = \mathcal{C}_{t+m+j} \cdot \mathbf{M}_{t+m+j}$.

Algorithm 1 Entropy-Aware Vulnerability Search for Attack Transformation

Require: Current mask $\mathbf{M}_{t+m+j-1}$, risk map $\tilde{\mathbf{S}}_{t+m}$, velocity $\mathbf{v}_{\text{cur}}$
Ensure: Transformed mask $\mathbf{M}^{\text{final}}_{t+m+j}$

$(x_c, y_c) \leftarrow \text{CoM}(\mathbf{M}_{t+m+j-1})$ ▷ `Compute center`
$\nabla\tilde{\mathbf{S}} \leftarrow \text{GradS}(\tilde{\mathbf{S}}_{t+m}, x_c, y_c)$ ▷ `Entropy gradient estimation`
$\alpha \leftarrow \text{AdaptiveBlend}(\mathbf{v}_{\text{cur}}, \nabla\tilde{\mathbf{S}})$ ▷ `Adaptive blending`
$\mathbf{d}_{\text{ideal}} \leftarrow \text{Normalize}(\alpha\mathbf{v}_{\text{cur}} + (1-\alpha)\nabla\tilde{\mathbf{S}})$
Calculate reward components $\mathcal{D}, \mathcal{V}, \mathcal{H}$ ▷ `Motion coherence terms`
Calculate composite reward $\mathcal{R}$ using Eq. 38
$(x^*, y^*) \leftarrow \arg\max_{(x,y)}\{\tilde{\mathbf{S}}_{t+m}(x,y) + \mathcal{R}(x,y)\}$ ▷ `Maximize entropy deficit`
$\mathbf{M}_{t+m+j} \leftarrow \text{TranslateMask}(\mathbf{M}_{t+m+j-1}, x^* - x_c, y^* - y_c)$
$\mathcal{C}_{t+m+j} \leftarrow \mathbb{I}[\mathbb{E}_{(x,y)\in\mathcal{N}(x_c,y_c)}[\tilde{\mathbf{S}}_{t+m}(x,y)] \geq \eta]$
$\mathbf{M}^{\text{final}}_{t+m+j} \leftarrow \mathcal{C}_{t+m+j} \cdot \mathbf{M}_{t+m+j}$
return $\mathbf{M}^{\text{final}}_{t+m+j}$

This entropy-aware vulnerability search strategy fundamentally differs from conventional greedy approaches by explicitly leveraging the information-theoretic vulnerability quantification established in Theorem 2. By maximizing entropy deficit $V(R) = H_c(R) - \frac{1}{n}\sum_{i=1}^{n} H_i(R)$ through the learned risk map $\tilde{\mathbf{S}}_{t+m}$, our approach systematically identifies optimal attack transformations across spatiotemporal dimensions that exploit regions where collective consensus uncertainty exceeds individual observation confidence—precisely the conditions under which CP defense mechanisms are most vulnerable to fabrication attacks.

After obtaining the final mask, we determine whether to perform full optimization or simply warp the perturbation. For frames where $\mathcal{C}_{t+m+j} = 1$ and we're within the persistence window p, we employ a feature-space homeomorphism $\mathcal{W} : \mathbb{R}^d \times \{0,1\}^{H\times W} \times \{0,1\}^{H\times W} \to \mathbb{R}^d$ that transfers the perturbation between frames:

$$F^p_j = \mathcal{W}(F^p_{j-1}, \mathbf{M}_{j-1}, \mathbf{M}_j), \quad (44)$$

where F^p_j represents the perturbed features at frame $t+m+j$, and $\mathbf{M}_j$ is shorthand for $\mathbf{M}^{\text{final}}_{t+m+j}$. This approach leverages the quasi-stationary nature of feature representations

across consecutive frames, enabling efficient attack persistence without computationally expensive re-optimization.

I. Occupancy Map Estimation

When occupancy maps are not explicitly shared in CP systems, we estimate them from available perception data by adapting the Blind Region Segmentation (BRS) algorithm proposed by Tao *et al.* [26]. The process begins with differential detection, where the attacker compares independent and collaborative perception results to infer each CAV's blind spots. For each collaborating CAV j, the attacker generates two perception outputs:

$$\mathbf{Y}_s^j = \Phi_{\texttt{dec}}(\mathbf{F}_{j\to i}), \quad \mathbf{Y}_c^j = \Phi_{\texttt{dec}}(f_{\texttt{agg}}(\mathbf{F}_{i\to i}, \mathbf{F}_{j\to i})), \tag{45}$$

where $\mathbf{Y}_s^j$ represents single perception results using only CAV j's features $\mathbf{F}_{j\to i}$, and $\mathbf{Y}_c^j$ represents collaborative perception results using both local features $\mathbf{F}_{i\to i}$ and CAV j's features. The function $\Phi_{\texttt{dec}}$ denotes the decoder network and $f_{\texttt{agg}}$ is the feature aggregation function. By computing the intersection $\mathbf{Y}_s^j \cap \mathbf{Y}_c^j$ and set difference operations, we identify:

$$\mathbf{Y}_{j\text{-only}} = \mathbf{Y}_s^j, \quad \mathbf{Y}_{non\text{-}j} = \mathbf{Y}_c^j \setminus (\mathbf{Y}_s^j \cap \mathbf{Y}_c^j), \tag{46}$$

where $\mathbf{Y}_{j\text{-only}}$ contains CAV j's unique detections and $\mathbf{Y}_{non\text{-}j}$ contains detections not attributable to CAV j. This differential analysis reveals the spatial distribution of perception capabilities. The BRS algorithm partitions the BEV detection map into confident areas (CA) and blind areas (BA) through an adaptive region growing process. The algorithm employs a spatially-aware neighbor selection function that adapts to the distance from CAV j's center position $\mathbf{p}_j$:

$$\kappa(\mathbf{s}, \mathbf{p}_j) = \left\lceil \kappa_0 \cdot \exp\left(-\lambda \cdot \frac{\|\mathbf{s} - \mathbf{p}_j\|_2}{\sqrt{H^2 + W^2}}\right)\right\rceil, \tag{47}$$

where $\kappa(\mathbf{s}, \mathbf{p}_j)$ determines the number of neighbors to consider at grid position $\mathbf{s}$, $\kappa_0 = 6$ is the base connectivity parameter, $\lambda = 0.3$ controls the exponential decay rate, $\|\mathbf{s} - \mathbf{p}_j\|_2$ is the Euclidean distance between position $\mathbf{s}$ and CAV j's center $\mathbf{p}_j$, and $\sqrt{H^2 + W^2}$ normalizes the distance based on the BEV map dimensions $H \times W$. This adaptive connectivity ensures denser region growing near the CAV and sparser expansion in distant regions, modeling the natural degradation of perception reliability with distance. The key step for our occupancy map estimation is converting the resulting binary mask $\mathbf{M}_j \in \{0, 1\}^{H\times W}$ into an occupancy grid map $\mathbf{O}_j \in \{0, 1, 2\}^{H\times W}$. For each grid cell position $\mathbf{p} = (x, y)$, we apply the following mapping:

$$\mathbf{O}_j(\mathbf{p}) = \begin{cases} 1, & \text{if } \mathbf{M}_j(\mathbf{p}) = 1 \text{ and } \mathbf{p} \in \mathcal{D}_j \\ 0, & \text{if } \mathbf{M}_j(\mathbf{p}) = 1 \text{ and } \mathbf{p} \notin \mathcal{D}_j \\ 2, & \text{if } \mathbf{M}_j(\mathbf{p}) = 0 \end{cases} \tag{48}$$

where $\mathcal{D}_j$ represents the set of grid positions containing detected objects from CAV j. In confident areas ($\mathbf{M}_j(\mathbf{p}) = 1$), grid cells containing detected objects are marked as occupied (1), while empty cells are marked as free (0). Blind areas ($\mathbf{M}_j(\mathbf{p}) = 0$) are marked as unknown (2). This process generates approximate occupancy grid maps for each collaborating CAV that, while not as precise as explicitly shared maps in CAD-enabled systems, provide sufficient information for constructing the MVIG and optimizing our attack strategy.

J. More Background and Related Work

J.1. Robust Collaborative Perception

Single-agent perception systems are constrained by limited field-of-view (FoV), which collaborative perception (CP) addresses through multi-agent data fusion [10, 12]. CP architectures have evolved from raw-data-level fusion [4] and output-level fusion [33] to intermediate-level feature fusion, with DiscoNet [18] and V2VNet [30] demonstrating the effectiveness of this approach. For practical deployment of CP systems, robustness against various challenges becomes crucial. These challenges can be broadly categorized into systematic robustness issues and malicious agent threats. Systematic robustness concerns include synchronization challenges [17, 31], communication interruption [24], data corruption [34], and camera sensor failures [29]. While these system-level issues can affect perception performance, they are often predictable and can be mitigated through proper system design and error handling mechanisms. In contrast, threats from malicious agents represent a more severe challenge to CP systems. Unlike systematic issues that occur randomly and independently, malicious agents can launch targeted attacks by deliberately manipulating shared information and exploiting system vulnerabilities.

J.2. Attacks on Collaborative Perception

The security challenges in CP systems have become increasingly complex due to the emergence of various adversarial attacks, which have evolved alongside advances in perception architectures. Initially, security concerns primarily focused on physical attacks against individual CAVs, including GPS spoofing [19], LiDAR spoofing [2, 9], and the deployment of adversarial objects in the physical environment [27]. As CP systems adopted collaborative frameworks, new vulnerabilities emerged. Late-fusion proved particularly susceptible to attacks, as their direct sharing of object detection results [7, 8] made it straightforward for attackers to manipulate the perception outcomes [1, 22]. Early attempts to attack these systems by corrupting raw sensor data proved ineffective against the more robust feature-level fusion systems that were later developed.

The evolution of CP systems towards feature-level fu-

Table 8. MVIGNet Architecture and Parameters

Component	Shape/Value	Description
Input Data Structure		
Temporal Graphs	`List of Dictionaries`	Sequence of graph structures
Node Features Composition		
Basic Features	`[3]`	Normalized occupancy frequencies
Position-Pose	`[6]`	CAV position and orientation
Spatial Features	`[91]`	Multi-scale pooled features
Edge Features Composition		
Mutual Information	`[1]`	Average MI between occupancy maps
Backbone		
Input	`[batch, seq_len, num_nodes, 100]`	Sequence of graph node features
MVIG-Conv ($\times 3$)	`[batch, seq_len, num_nodes, 64]`	Specialized graph convolution layers
Mean Pooling	`[batch, seq_len, 64]`	Aggregates node features per-frame
GRU	`[batch, seq_len, 64]`	Processes temporal data across frames
Last Hidden	`[batch, 64]`	Extracts final hidden state from GRU
Score Head Component		
FCN Layer 1	`[batch, 64]`	Fully connected layer
ReLU	`[batch, 64]`	Activation function
FCN Layer 2	`[batch, 40000]`	Fully connected layer
Softmax	`[batch, 40000]`	Converts to probability distribution
Grid Map	`[batch, 200, 200]`	Reshapes scores to 2D grid
Position Selection	`[batch, 2]`	Selects grid position
Output	`[batch, 7]`	bounding box parameters
Hyperparameters		
node_dim	`100`	Dimension of input node features
edge_dim	`1`	Dimension of edge features
hidden_dim	`64`	Dimension of hidden layers
num_layers	`3`	Number of graph convolution layers
grid_size	`(200, 200)`	Size of the output grid map
range_limit	`20`	Range limit for x and y coordinates
attack_type	`"spoof", "remove"`	Type of attack to perform

sion prompted attackers to develop more sophisticated strategies. Tu *et al.* [28] pioneered an untargeted adversarial attack framework by demonstrating how CP systems could be compromised through feature map perturbations, achieving high success rates. However, their approach produced obvious modifications of bounding boxes in perception results, making it easily detectable by threshold-based outlier anomaly detection methods [21, 36]. Tao *et al.* [26] improved upon this by introducing the blind area confusion (BAC) attack that utilizes victim CAV's view information through differential detection and blind area estimation to generate a rough mask, reducing ineffective attack boxes while maintaining high success rates. Nevertheless, their approach relies solely on victim CAV's knowledge for mask estimation, failing to consider that benign agents can collectively validate detection results through their overlapping view regions. Zhang *et al.* [35] advanced this work by developing a targeted attack framework that uses specialized loss functions and masking techniques to generate fabricated detection boxes at specific locations. Their ray-casting

Table 9. Comparison of existing attack methods in collaborative perception. ✓ indicates the feature is supported, while ✗ indicates it is not.

Method	System	Prerequisites	Real-time	Targeted	Attack Type		Stealthiness	
					Spoof	Remove	Timing	Region
GPS/LiDAR spoofing [2, 19]	Single	Laser emitters	✓	✓	✓	✓	✗	✗
Physical attack [27]	Single	Physical access	✓	✓	✗	✓	✗	✗
Output manipulation [7, 8]	Late fusion	None	✓	✓	✓	✓	✗	✗
Basic feature attack [28]	Int. fusion	Local computing	✗	✗	✓	✗	✗	✗
RC attack [35]	Int. fusion	Local computing	✓	✓	✓	✓	✗	✗
BAC attack [26]	Int. fusion	Local computing	✗	✗	✓	✗	✗	✓
MVIG attack (Ours)	Int. fusion	Local computing	✓	✓	✓	✓	✓	✓

(RC) attack achieved more stealthy and real-time attacks by leveraging LiDAR ray-casting to accelerate perturbation generation and enhance attack stealthiness. However, their method lacked systematic consideration of attack timing ("when to attack") and attack region selection ("where to attack"), relying instead on random selection of these critical parameters. This randomness made their attacks vulnerable to occupancy grid-based collaborative anomaly detection (CAD) systems. These limitations motivate our proposed MVIG attack, which systematically analyzes and exploits the dynamic mutual view relationships to achieve both effective and stealthy attacks through optimized attack timing and region control. A comparison of the MVIG attack with existing attack methods is shown in Table 9.

J.3. Defensive Collaborative Perception

The growing security threats in CP systems have driven the development of various defense mechanisms. Initial defense approaches, including ROBOSAC [21], CP-Guard [14], and MADE [36], primarily focused on detecting output-level perturbations through consensus-based verification, where perception outputs are cross-validated among CAVs using Hungarian matching and reconstruction loss. Building upon these foundations, Tao *et al.* [26] enhanced the defense capabilities by introducing spatio-temporal anomaly detection to examine perception patterns across both spatial and temporal dimensions. While these methods demonstrated effectiveness against Tu's classical feature attack framework and naive attacks, they face significant challenges when confronted with sophisticated perturbations that maintain output-level consistency.

In response to more sophisticated attacks, Zhang *et al.* [35] proposed a collaborative anomaly detection (CAD) system that implements a rule-based defense mechanism through occupancy map exchange. In their approach, the occupancy map is categorized into three states: 0 (free), 1 (occupied), and 2 (unknown). The defense operates on two principles: (1) in ego-known regions, any inconsistency with the ego CAV's occupancy map is immediately flagged as suspicious and evaluated against a conflict threshold to determine if an attack is present; (2) in unknown regions, the system relies on consistency checking among other collaborative CAVs' occupancy maps. However, this defense strategy reveals critical vulnerabilities. The CAD system becomes ineffective in regions that are mutually unknown to benign CAVs or in scenarios where malicious agents have advantageous viewpoints. More importantly, the sharing of occupancy maps inadvertently provides attackers with valuable information about the collective perception coverage, allowing them to carefully optimize their attack timing and regions to bypass the defense mechanism.

K. Structure of MVIGNet

The structure of the MVIGNet is shown in Table 8.