Sensor Fusion Desynchronization Attacks

Andreas Finkenzeller
School of Computation, Information and Technology, Technical University of Munich, Germany

Andrew Roberts
FinEst Centre for Smart Cities, Tallinn University of Technology, Estonia

Mauro Bellone
FinEst Centre for Smart Cities, Tallinn University of Technology, Estonia

Olaf Maennel
School of Computer and Mathematical Sciences, The University of Adelaide, Australia

Mohammad Hamad
School of Computation, Information and Technology, Technical University of Munich, Germany

Sebastian Steinhorst
School of Computation, Information and Technology, Technical University of Munich, Germany

Abstract

Environmental perception and 3D object detection are key factors for advancing autonomous driving and require robust security measures to ensure optimal performance and safety. However, established methods often focus only on protecting the involved data and overlook synchronization and timing aspects, which are equally crucial for ensuring profound system security. For instance, multi-modal sensor fusion techniques for object detection can be affected by input desynchronization resulting from random communication delays or malicious cyber attacks, as these techniques combine various sensor inputs to extract shared features present in their data streams simultaneously. Current research acknowledges the importance of temporal alignment in this context. However, the presented studies typically assume genuine system behavior and neglect the potential threat of malicious attacks, as the suggested solutions lack strategies to prevent intentional data misalignment. Additionally, they do not adequately address how sensor input desynchronization affects fusion performance in depth. This paper investigates how desynchronization attacks impact sensor fusion algorithms for 3D object detection. We evaluate how varying sensor delays affect the detection performance and link our findings to the internal architecture of the sensor fusion algorithms and the influence of specific traffic scenarios and their dynamics. We compiled four datasets covering typical traffic scenarios for our empirical evaluation and tested them on four representative fusion algorithms. Our results show that all evaluated algorithms are vulnerable to input desynchronization, as the performance declines with increasing sensor delays, highlighting the existing lack of resilience to desynchronization attacks. Furthermore, we observe that the Light Detection and Ranging (LiDAR) sensor is significantly more susceptible to delays than the camera. Finally, our experiments indicate that the chosen fusion architecture correlates with the system's resilience against desynchronization, as our results demonstrate that the early fusion approach provides greater robustness than others.

2012 ACM Subject Classification Security and privacy → Systems security; Hardware → Safety critical systems

Keywords and phrases Security, Time Synchronization, Sensor Fusion, Autonomous Driving, Delay Attack

Digital Object Identifier 10.4230/LIPIcs.ECRTS.2025.6

Supplementary Material *Software (Source Code)*:
`https://osf.io/x2yrq/files/osfstorage?view_only=f148ad01c4104699b214e76d3cad7388` [11]
Software (ECRTS 2025 Artifact Evaluation approved artifact):
`https://doi.org/10.4230/DARTS.11.1.1`

Funding The authors acknowledge the financial support by the Federal Ministry of Education and Research of Germany in the programme of “Souverän. Digital. Vernetzt.”. Joint project 6G-life, project identification number: 16KISK002.
This work is supported by the European Union-funded project CyberSecDome (Agreement No.: 101120779).

1 Introduction

Autonomous driving has gained much attention in the last few years, as has the research on overcoming the remaining obstacles to full autonomy. Common methods for enabling vehicles to drive autonomously can be categorized into three main sub-tasks: scene recognition, path planning, and vehicle control [20]. All parts must intertwine smoothly, and the individual tasks, as well as the exchanged data, must be trustworthy to ensure reliable operation. Ensuring reliability is crucial in this context because autonomous vehicles are safety-critical systems. Numerous measures to establish a trusted system environment are already in place, including protection of visual sensors such as cameras and their associated deep learning algorithms from malicious adversarial samples [53, 48, 54]. Furthermore, Intrusion Detection and Prevention Systems (IDPSs) and Intrusion Response Systems (IRSs) detect and mitigate packet anomalies in exchanged data traffic and, hence, support securing in-vehicle communications [47]. Despite covering many critical security aspects, implemented methods are often data-centric and mainly focus on the data’s authenticity and integrity. However, data protection alone is insufficient in certain situations, as timing aspects are equally important. These situations include object detection for environmental sensing and scene recognition, for example.

Recently, object detection has been increasingly based on sensor fusion approaches, where the data of multiple sensor modalities is combined to improve the detection performance. Common approaches include the combination of camera and Light Detection and Ranging (LiDAR) [44, 18], camera and Radio Detection and Ranging (RADAR) [31, 19, 24], and most lately also LiDAR and RADAR [36, 49] sensors. We can further distinguish sensor fusion approaches based on the fusion type applied. In this work, we distinguish *early fusion*, *deep fusion*, *late fusion*, and *multi-stage fusion* approaches, depending on when the fusion occurs (see Section 2.1 for more details). The involved sensors capture the environmental state at a specific time, so both the sensor data and timing are essential for successful fusion results. In particular, when more than one snapshot is taken from the same environment, ensuring the temporal alignment of all snapshots is inevitable. If the temporal alignment cannot be guaranteed, for example, due to random delays, the lack of proper synchronization, or malicious cyber attacks, the sensor fusion performance can be heavily affected.

Current literature on sensor fusion methods emphasizes the importance of temporal input synchronization [5, 23, 25]. However, there is a lack of additional analysis regarding the effects of misalignment or comprehensive empirical results to support this statement. Some studies address temporal misalignment by proposing synchronization strategies to improve model performance [33, 45, 10, 22, 28, 39, 37, 9]. However, these approaches typically assume genuine system behavior and overlook the potential threat of malicious actions. Others cover the topic as part of broader benchmarking experiments and only conclude the general significance of input synchronization [7, 15]. The existing literature does not adequately address how sensor input desynchronization affects the fusion performance, nor does it explore the effects of other factors, such as the sensor fusion type or scene dynamics. With this work, we aim to fill this remaining research gap.

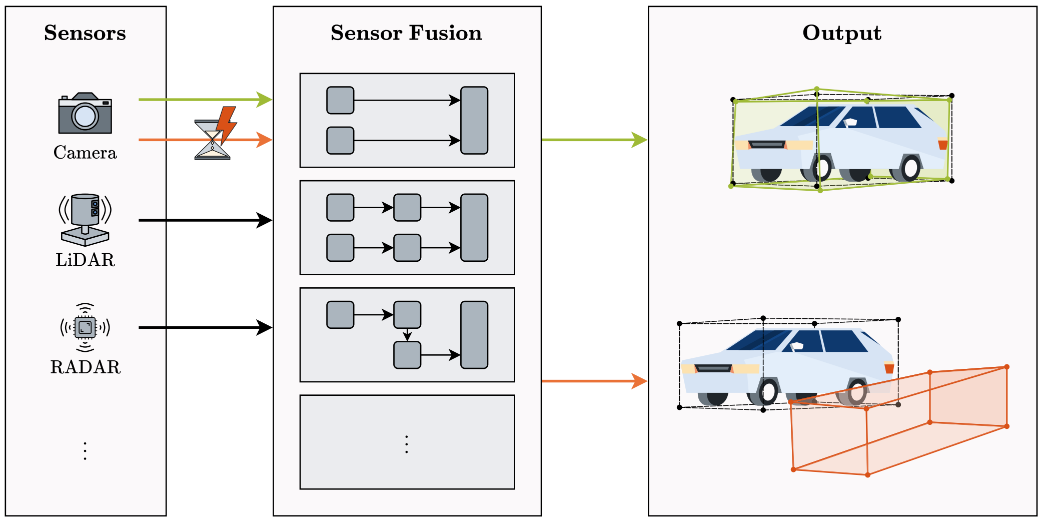


Figure 1 Typical sensor fusion approach for 3D object detection. All input data is synchronized in the default scenario (green), and the detection results meet the expectations. However, if the inputs are not properly aligned (red), the detection performance is heavily affected depending on the chosen fusion algorithm and traffic scenario.

Contributions. This study thoroughly analyzes the impact of temporally misaligned sensor inputs provided to common sensor fusion algorithms for 3D object detection as visualized in Figure 1. In this context, we investigate the effects of individual sensor delays, traffic scenarios, and implemented fusion methods on the detection performance. We first study the structure of different sensor fusion techniques (Section 2.1) and derive a theoretical model to formally describe the temporal misalignment of provided input data (Section 2.2). Furthermore, we propose a new metric to describe traffic scenario dynamics to account for this effect in our evaluation (Section 2.3). For evaluation, we introduce four datasets covering typical traffic scenarios (Section 4) to measure the performance degradation for representative algorithms of each sensor fusion category (Section 5). Moreover, we develop reasoning for the experienced desynchronization effects on the respective fusion performance (Section 6) and compare our results to related work (Section 7). Finally, we point out potential attack surfaces that can lead to the addressed synchronization violations (Section 3) and propose related mitigation approaches (Section 8). We provide open-source access to our compiled datasets and the associated source code for running our experiments, enabling the community to reproduce our results.

2 System Model

2.1 Sensor Fusion Approaches

Over the last few years, several sensor fusion techniques have evolved. A common taxonomy is based on when the fusion step occurs, as outlined by [41]. Wang et al. [42] extend this classification to include additional categories, such as feature representation and alignment. This work mainly distinguishes four categories: *early fusion*, *deep fusion*, *late fusion*, and *multi-stage fusion* approaches. However, categorizing existing algorithms is generally difficult due to dynamic boundaries, as some algorithms encompass features from multiple categories. Nonetheless, a classification remains valuable for connecting observed performance evaluation trends to the algorithm's internal structure. Figure 2 illustrates the various algorithmic structures and their corresponding classifications.

Early Fusion (or low-level fusion)

Early fusion approaches directly fuse raw sensor data, usually by leveraging deep learning techniques. The main advantage of early fusion is that the fusion model benefits from all available data without losing information due to prior processing steps. However, this also requires similar input data structures, e.g., using the bird's eye view (BEV) representation for LiDAR data to allow for proper fusion with 2D camera images. Additionally, raw data is usually extensive and requires more processing and memory resources. This can significantly impact applications with real-time constraints, such as automotive systems, for which reliable 3D object detection is crucial to enable autonomous driving.

Deep/Late Fusion (or high-level fusion)

To overcome the performance drawbacks of early sensor fusion, we can first extract more high-level elements from the individual data sources. This significantly reduces the fusion algorithm's input data complexity as extracted elements can be stored and processed more efficiently. The extraction step implicitly converts heterogeneous sensor data formats to a uniform representation, simplifying the support for diversified sensor setups. We differentiate between deep and late fusion based on the abstraction level of the extraction step's output. While deep fusion combines intermediary features, late fusion fuses final object proposals [17]. In both cases, the preceding preprocessing step focuses on a single sensor. Therefore, it disregards the available data of other sensor modalities as a trade-off, contrasting the original idea of sensor fusion. If one of the individual extractions performs poorly, we lose valuable information in the preprocessing step, potentially decreasing the overall high-level fusion performance.

Multi-Stage Fusion

Unlike the other two approaches, which process the provided sensor data in parallel, we can also fuse the available data in multiple sequential stages. The data evaluation of the sensor input in the first stage then affects the processing in the second stage by acting as a filter on the second sensor's raw data, for example. In this way, the input data for each stage can be optimized to either increase the fusion performance or reduce the model complexity and required resources. For instance, a LiDAR point cloud can be significantly reduced before evaluation by only considering the points in the expected object region determined by a preceding 2D object detection step based on a camera frame [44]. The overall fusion performance again heavily depends on the filter step results, similar to feature extraction and object proposals in deep and late fusion approaches.

2.2 Sensor Data Sampling

To characterize sensor desynchronization effectively, we first need a formal description of sensor data sampling. For that, we follow an approach similar to the *reference-point-based temporal robustness* model introduced by Kuhse et al. [25]. In our model, we consider n independent sensors S_j, $j \in [1, n]$ that provide input data for a sensor fusion function $F(s_1, \ldots, s_n)$, where s_i is the data stream of sensor S_i. There is a mutual sampling rate r at which all sensor readings shall be captured to be ready simultaneously at (discrete) time steps t_i, $i \in \mathbb{N}_0$ with $t_{i+1} = t_i + \delta$, where $\delta = \frac{1}{r}$ is the defined interval time. In a perfect setup, each sensor S_j samples the data strictly at time t_i. However, this is impossible in practice, which results in the actual sampling times of the individual sensors S_j:

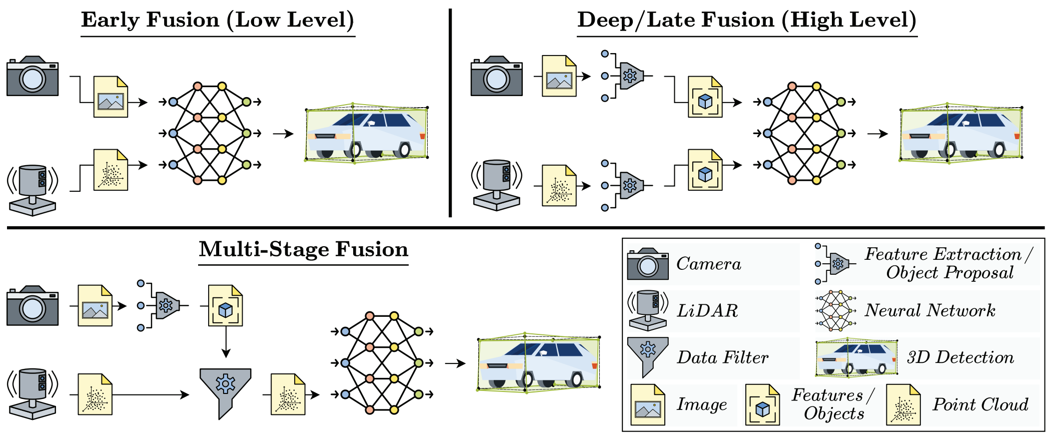


Figure 2 Visualization of different sensor fusion approaches. The methods primarily differ regarding when the fusion occurs during the processing. Early fusion techniques work with raw sensor data, while deep and late fusion approaches include preceding feature extraction and object proposal steps, respectively, to fuse higher-level data. In multi-stage fusion methods, the data evaluation from one sensor serves as a sequential filter for the next.

$$t_i^{(S_j)} = t_i + \epsilon_j(t_i) \tag{1}$$

where $\epsilon_j(t_i)$ denotes the offset from the perfect moment t_i. This offset typically results from random errors due to varying processing loads and environmental settings (heat, humidity, etc.) or deliberate, malicious attacks (see Section 3). In the following, we assume the offset $\epsilon_j(t_i)$ mainly consists of deliberate delays caused by malicious attacks and neglect comparatively small and nondeterministic delays resulting from random errors. Figure 3 visualizes the sampling process and associated timings. The main goal of this work is to evaluate the impact of $\epsilon_j(t_i) \neq 0$ on the sensor fusion performance dependent on the chosen algorithm $F(s_1, \ldots, s_n)$ and other system parameters, such as the scene dynamics.

Sensor fusion algorithms are typically trained on prerecorded, sequentially numbered, labeled data such as the KITTI dataset [16]. As a result, they expect the input data to be sampled simultaneously and available immediately. In other words, the algorithms themselves follow the best-effort approach, assuming coherent sampling rates at the individual sensors and instant data availability. Particularly, the algorithms do not follow specific strategies to consider or prevent data misalignment. This behavior is inherently error-prone as synchronicity can generally not be guaranteed in externally provided data. Hence, external synchronization mechanisms must be implemented to ensure proper alignment of the data provided to the sensor fusion algorithms. One approach is to add accurate timestamps to each record to maintain the data's temporal reference. However, this solution requires precise clocks with appropriate clock synchronization at the sensors to properly label the sensor readings directly when captured. While enhancing general data alignment, this method remains vulnerable to potential attacks targeting the clock synchronization mechanism, as discussed in Section 3.

2.3 Traffic Scenario Dynamics

Besides the chosen sensor fusion algorithm, the specific traffic scenario is also an essential factor to consider for evaluation. Understanding the dynamics of traffic scenarios is crucial, as the temporal misalignment among different sensor modalities only becomes apparent

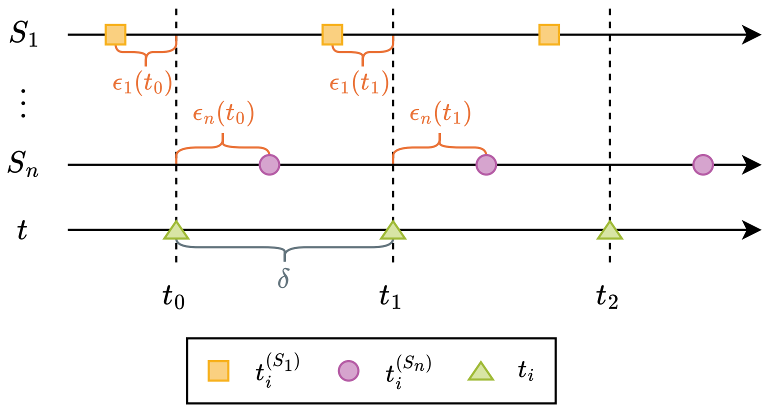


Figure 3 Illustrating the timings involved in a typical data sampling process for sensor fusion. Usually, the sensors S_j do not accurately sample at the expected sampling points t_i, resulting in an undesirable offset $\epsilon_j(t_i)$. The reasons for such delays can range from random errors to malicious attacks.

during actual environmental changes over time. Thus, we specifically examine the movement of individual objects of interest, such as cars, within the sensors' field of view (FoV) during the scenario. For cameras, for example, each object has a unique 2D bounding box containing all visible object pixels in a given frame. Hence, the object's motion within two consecutive frames can be described as the relative movement of its enclosing bounding box. We introduce the dynamics vector $\vec{d}$ describing the aggregated movements of all M objects of interest in two consecutive camera frames at t_i and t_{i+1} as:

$$\vec{d} = \sum_{j}^{M} |\vec{c}_{j,t_{i+1}} - \vec{c}_{j,t_i}| \tag{2}$$

where $\vec{c}_{j,t_i}$ and $\vec{c}_{j,t_{i+1}}$ denote the centers of object j in the frames taken at t_i and t_{i+1}, respectively. To normalize the vector $\vec{d}$ across scenarios with different object occurrences, we calculate the L1-norm and divide it by the number of objects present in a single frame. Therefore, the final scalar metric for the traffic scenario dynamics d_{norm} is computed (per frame) as:

$$d_{norm} = \frac{\|\vec{d}\|_1}{M} \tag{3}$$

Figure 4 illustrates the proposed dynamics score computation principle. Using this metric, we can categorize our selected traffic scenarios according to their dynamics, taking this effect into account in the attack evaluation.

3 Desynchronization Attacks

As mentioned in Section 2.2, the sensor fusion algorithms typically require the input data to be sampled simultaneously. If this synchronization cannot be guaranteed by design, external solutions such as data timestamping are necessary to ensure temporally aligned inputs for the algorithms. However, even external synchronization mechanisms cannot fully guarantee proper data alignment due to varying processing delays before the time-stamping event or malicious attacks that break the required clock synchronization.

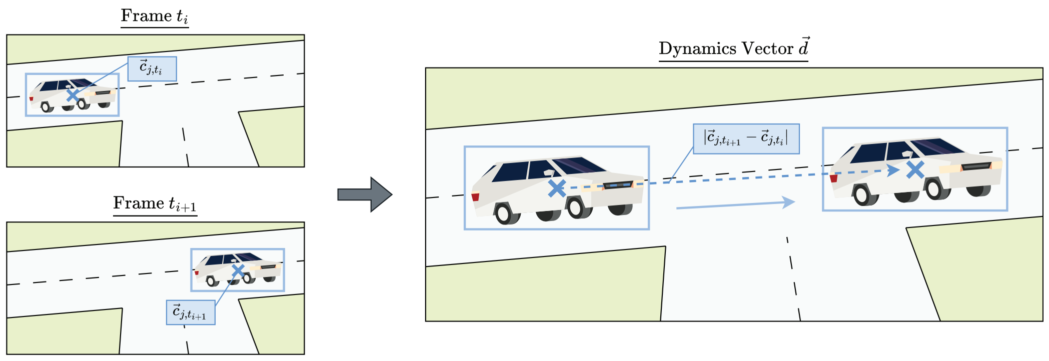

Frame t_i
$\vec{c}_{j,t_i}$
Frame t_{i+1}
$\vec{c}_{j,t_{i+1}}$
Dynamics Vector $\vec{d}$
$|\vec{c}_{j,t_{i+1}} - \vec{c}_{j,t_i}|$

Figure 4 Visualization of the introduced scenario dynamics computation. The movements of the objects are represented by the pixel shifts of the corresponding bounding boxes between two consecutive camera frames. The final dynamics score is calculated as the normalized sum of all L1-norms of the shift vectors for all objects in the camera frame.

3.1 Desynchronization Origin

In general, we can distinguish two sources of desynchronization that ultimately result in the same effect of temporal misalignment of sensor input data, namely, random jitter and malicious attacks. Random jitter typically follows a stochastic model, such as a Gaussian distribution, and can be considered noise. Hence, jitter refers to variations in a system that create disturbances without deterministic behavior. This noise can result from varying system and network loads, as well as environmental differences like temperature fluctuations. In contrast, attacks are malicious actions that intentionally harm a system by exploiting its vulnerabilities. These deliberate actions usually allow for a much more fine-grained influence as the attackers can control parameters such as the duration of the attack and the level of desynchronization introduced. This results in various arbitrary desynchronization patterns that can be applied. Since the attack strategies can also follow random distributions, attacks can be seen as a more general way to model system disturbances, as they extend the jitter by arbitrary desynchronization behaviors. For the remainder of this work, we will focus on malicious desynchronization attacks, as they represent a more general model for temporal misalignment.

3.2 Threat Model

In our threat model for desynchronization attacks, we consider two cases: the best-effort case without any external synchronization mechanism and the case with an established time synchronization mechanism and timestamping of the individual sensor readings. In both scenarios, we assume that data authenticity and integrity are maintained. Authenticity and integrity can be ensured, for example, through cryptographic measures, such as Message Authentication Codes (MACs)[1]. Allowing the modification of packet contents enables attackers to launch more severe attacks than temporal misalignment by directly altering sensor data, which belongs to the research about adversarial samples and is beyond the scope of this work. Instead, we focus on delay attacks, i.e., delaying the packet transmission.

[1] The feasibility of particular solutions heavily depends on the available computational resources on the usually constrained sensor devices. Nevertheless, from a technical point of view, this problem has been solved, and finding appropriate solutions is beyond this work's scope.

For this, the attackers can access the communication network and its hardware components, allowing them to introduce delays to transmitted messages. This is possible as several studies have shown, which will be further discussed in Section 3.3. One strength of this attack model is its stealthiness. Unlike data modifications or packet introduction/deletions, which are quite intrusive, malicious packet timings are hard to monitor and detect. This is because packet-switched networks, by default, have non-deterministic transmission latencies due to their stochastic nature. In the best-effort case, the attack impact is evident, as any introduced packet delay directly affects the timing when the fusion algorithm consumes the input data. If a timestamping mechanism exists, attackers can use more sophisticated methods (see Section 3.3) to manipulate the clocks and invalidate the timestamps.

3.3 Attack Implementation

Typical sensor fusion systems generate large amounts of data that require high-bandwidth communication for data transfer. Therefore, packet-switched networks, such as Automotive Ethernet, are commonly used. In these networks, attackers can provoke packet delays by being in a Machine-in-the-Middle (MitM) position and directly delaying the passing packets on the fly. Figure 5 depicts a simplified automotive network visualizing the attack. To reach a MitM position, attackers must first get access to the network, for example, by compromising an involved Electronic Control Unit (ECU). Numerous attack vectors to hijack an ECU have been documented in the literature, including [30, 2, 21, 32]. Note that the ECU can be arbitrary and is not required to participate in the sensor fusion process. We can directly perform the delay attack if the compromised ECU is already positioned in a MitM setup. Otherwise, we can leverage further attack strategies to gain this position, as described in [52]. Even if a MitM position is unreachable, attackers can still create network congestion that similarly affects packet timings by flooding the network with excessive traffic.

If timestamping mechanisms are in place, simply delaying the data packets is not sufficient anymore, as the delay will be detected by analyzing the timestamp captured before network transmission. Nevertheless, attackers can manipulate the clocks to invalidate the captured timestamps, ultimately resulting in desynchronization as well. Several studies have discussed the feasibility of such clock manipulation attacks. For example, Finkenzeller et al. [14] propose different attack strategies that disturb Precision Time Protocol (PTP) clock synchronization via Ethernet-based communication. To execute the presented delay attack, either a random network position or at most a MitM position is sufficient. Another approach is presented by Barreto et al. [1], which disrupts PTP synchronization by physically extending the optical fiber using a special delay box.

For the remainder of this work, we assume that delaying individual sensor inputs is possible, e.g., by using one of the presented methods. Because sensor fusion algorithms are typically tested with prerecorded and labeled datasets, such as the KITTI dataset [16], we can use index shifting to virtually attack them in our evaluation. To desynchronize a specific sensor input, we modify the index of the corresponding data sample to reflect the desired delay Δ in relation to the other sensor readings. The new index idx_{attack} can be computed with the following formula:

$$idx_{attack} = idx_{genuine} + \lfloor \frac{\Delta}{\delta} \rfloor \tag{4}$$

where δ denotes the interval time for sensor sampling. Note that this approach allows only integral multiples of the sampling interval time to be set as delays, which is acceptable if the interval time is sufficiently low.

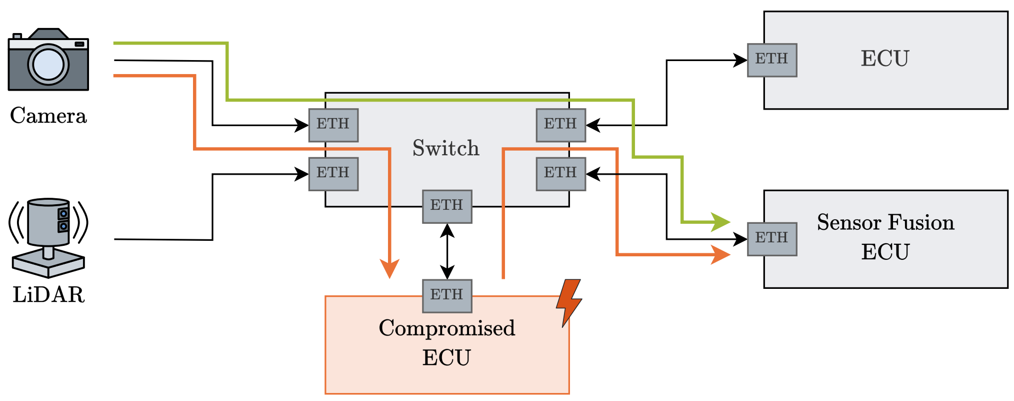


Figure 5 A simplified automotive network illustrating Ethernet communication between sensors and the sensor fusion ECU. A successful MitM attack enables a compromised ECU to reroute genuine traffic (green) through itself, allowing it to perform delay attacks on packets originally not addressed to it (red) [52].

4 Test Setup

We conduct thorough tests to evaluate the effects of temporary misalignment on sensor fusion algorithms caused by the desynchronization of the input data. To achieve this, our test environment includes implementations of representative algorithms from the sensor fusion classes outlined in Section 2.1. We evaluate the selected algorithms on four typical traffic scenarios using custom datasets that we compiled from the KITTI raw datasets [16].

4.1 3D Object Detection Algorithms

In recent years, 3D object detection algorithms have increasingly relied on sensor fusion techniques. Therefore, we selected several representative algorithms encompassing all the fusion types presented in Section 2.1 and executed the publicly available reference implementations in individual Docker containers. The selection was based on a mixture of both technical aspects, such as the required open source access to the proposed model and a high ranking in the KITTI 3D object detection leaderboard[2] at the time of publication. Table 1 summarizes all the selected algorithms we evaluate in our tests. We utilize the available pre-trained models provided by the authors for all algorithms. Remember that this study focuses on the changes in performance caused by the desynchronization attack rather than the optimal performance of the algorithm. Thus, the used model plays only a subordinated role here and the algorithmic structure is much more important.

4.2 Data Preparation

We introduce four datasets covering typical traffic scenarios to evaluate the chosen algorithms. The creation process was guided by several criteria we require for our datasets. First, our sets must include data from all sensor modalities relevant to the fusion algorithms, i.e., camera and LiDAR, and appropriate calibration information in the required format to match the algorithm's interface. Here, we discovered that while some tools, such as the *CARLA*

[2] `https://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=3d`

Table 1 Evaluated sensor fusion algorithms.

Algorithm	Year	Sensors	Fusion Method	Model
Frustum ConvNet [44]	2019	LiDAR, Camera	Multi-Stage	pre-trained (KITTI)
EPNet [18]	2020	LiDAR, Camera	Deep Fusion	pre-trained (KITTI)
CLOCs [34]	2020	LiDAR, Camera	Late Fusion	pre-trained (KITTI)
VirConv-L [46]	2023	LiDAR, Camera	Early Fusion	pre-trained (KITTI)

simulator [8] and the *Simutack* framework [13] are capable of generating the desired data in principle, the quality of the generated LiDAR data is particularly poor. Based on their specific implementation, some algorithms consider the geometric LiDAR information and the reflectivity for detection tasks. However, since reflectivity is not accurately modeled in these simulations [6], it is not usable for our evaluation. Second, we need correct ground truth information to compare the detection results we obtained and evaluate the performance variations caused by the attacks. By using a prepared and prerecorded dataset, we can test different delays for each algorithm with the same data basis, ensuring reproducibility and fair evaluation.

Furthermore, we need the data to be consecutively aligned in order to implement our delay attack by shifting the data indices. Therefore, the default KITTI dataset for 3D object detection[3] is not suitable due to its scattered data distribution. Hence, we assessed the recorded scenes available in the time-aligned KITTI raw data[4] and checked whether some of the frames are also present in the 3D object detection training dataset to obtain the necessary ground truth information. Using this method, we were able to compile four datasets that meet all the criteria stated above. The data was sampled with a mutual update rate of $r = 5\,\text{Hz}$, resulting in an interval time of $\delta = 200\,\text{ms}$ between two data samples with consecutive indices. While the KITTI raw dataset was sampled initially with a rate of 10 Hz, we encountered limitations due to the available ground truth information for consecutively aligned data, which resulted in a lower sampling rate. This sampling rate also limits the resolution of our delay attack, which we employ by shifting the sensor reading index as discussed in Section 3. While the available 200 ms interval does not allow for very fine-grained delay analysis, it is sufficient for observing the general trends we aim for. A custom dataset would have to be generated to facilitate a more detailed analysis. We provide public access to the generated datasets in our supplementary materials, allowing other researchers to replicate our findings.

4.3 Traffic Scenarios

As discussed in Section 2.3, the scenario dynamics can significantly impact the attack performance. Therefore, we selected four continuous driving scenarios that include sufficient variations in dynamics to more effectively evaluate their impact while also covering typical traffic situations. The derived scenarios are depicted in Figure 6, while Figure 7 compares the related scenario dynamics calculated according to (3). Recall that d_{norm} is related to object movements within the image plane. Thus, spikes in the plots indicate significant movements occurring quickly, often due to vehicles moving at close distances, for example. The complete list comprises the following four scenarios:

[3] `https://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=3d`
[4] `https://www.cvlibs.net/datasets/kitti/raw_data.php`

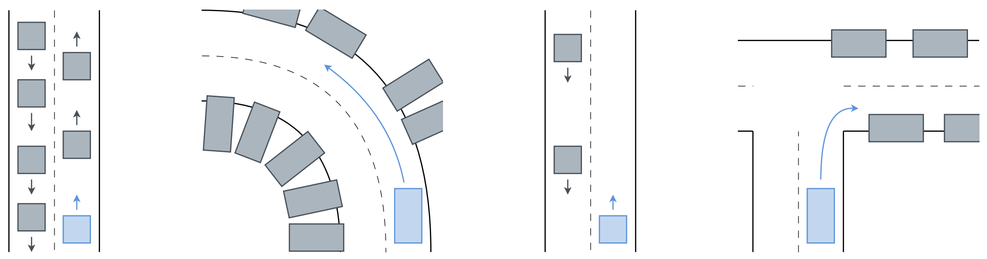

(a) S1: Urban Traffic. **(b)** S2: Left Turn. **(c)** S3: Highway. **(d)** S4: Resident. Traffic.

Figure 6 Visualization of the four evaluated traffic scenarios. The blue box represents the test vehicle running the 3D object detection algorithms, whereas the gray boxes depict other vehicles that shall be detected. The arrows indicate the respective driving directions.

Scenario 1: Urban Traffic. The test vehicle follows a line of cars in an urban traffic scenario without any lane changes or turns. Many vehicles are oncoming and passing the test vehicle on the neighboring lane. The scenario can be considered more dynamic as many vehicles are involved and constantly moving at approximately 50 $\frac{\text{km}}{\text{h}}$.

Scenario 2: Left Turn. In this scenario, the test vehicle takes a long left-hand turn, passing many parking vehicles on the side of the road. The car drives at approximately 30 $\frac{\text{km}}{\text{h}}$, and no other vehicles are moving in this scenario. Because of the turn and the numerous cars involved that move throughout the test vehicle's FoV, this scenario is also quite dynamic.

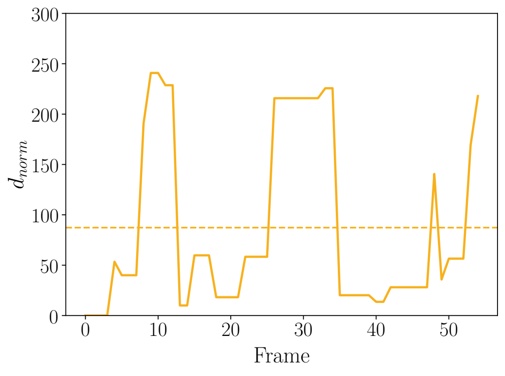
Scenario 3: Highway. The test vehicle drives down an empty lane on the highway, passing by several approaching cars. All cars drive at high speeds, approximately 100 $\frac{\text{km}}{\text{h}}$. Despite all vehicles moving fast, this scenario is less dynamic as the road is straight and only a few vehicles are passing by. The peaks in Figure 6c arise from the moments when the approaching cars pass the test vehicle.

Scenario 4: Residential Traffic. In the last scenario, the test vehicle drives in a residential area at low speeds below 30 $\frac{\text{km}}{\text{h}}$. Starting with a tight right turn, the vehicle drives on a narrow street, passing many parking cars. Because of the low driving speed, this scenario is also considered less dynamic.

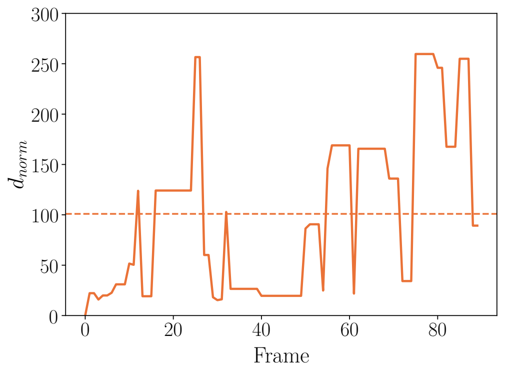
5 Evaluation

We conduct comprehensive experiments to evaluate the impact of sensor desynchronization attacks using the test setup as explained in Section 4. To establish a performance baseline, we first run all scenarios for each algorithm without any sensor delay. Subsequently, we gradually introduce specific delays and observe the resulting change in the performance of 3D object detection. The detection performance Φ_{IoU} is measured using the typical three-dimensional Intersection over Union (IoU) metric, which calculates the ratio of the intersection volume $V_{intersect}$ to the union volume V_{union} of the estimated and ground truth bounding boxes of a detected object. The formula is given as:

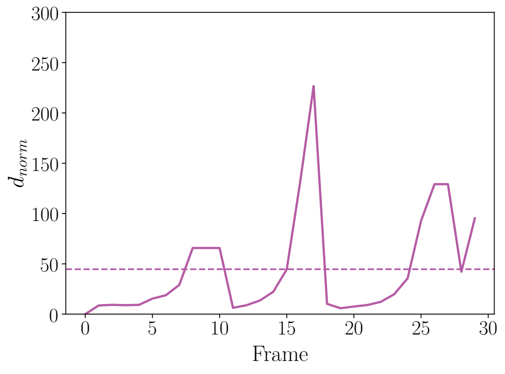
$$\Phi_{IoU} = \frac{V_{intersect}}{V_{union}} \tag{5}$$



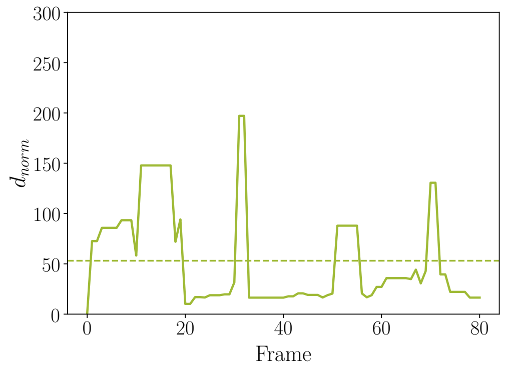
(a) Scenario 1: Urban Traffic.



(b) Scenario 2: Left Turn.



(c) Scenario 3: Highway.



(d) Scenario 4: Residential Traffic.

Figure 7 Illustrating the dynamics of the four traffic scenarios that were evaluated. A higher dynamics value suggests increased object movements within the test vehicle's FoV. The dashed line represents the average score for the entire scenario.

Note that $0 \leq \Phi_{IoU} \leq 1$, as the intersection can be at most the full union and zero for non-overlapping bounding boxes. It is also important to note that $\Phi_{IoU} = 0$ does not necessarily imply that no objects have been detected. Instead, it may indicate that an estimated 3D bounding box has no intersection with the ground truth. To identify good bounding box candidates, we define a detection threshold of $\theta = 0.5$ that must be exceeded for a detection to be considered successful. We distinguish the following cases:

$$detection \leftarrow \begin{cases} \text{true positive (tp)}, & \Phi_{IoU} > \theta, \text{ correct detection} \\ \text{false positive (fp)}, & \Phi_{IoU} > \theta, \text{ incorrect detection} \\ \text{true negative (tn)}, & \Phi_{IoU} \leq \theta, \text{ correct rejection} \\ \text{false negative (fn)}, & \Phi_{IoU} \leq \theta, \text{ incorrect rejection} \end{cases} \tag{6}$$

With these criteria, we can assess all objects and determine the overall algorithm performance based on the typical metrics *precision* and *recall*:

$$precision = \frac{tp}{tp + fp}, \qquad recall = \frac{tp}{tp + fn} \tag{7}$$

5.1 Desynchronization Attacks

After establishing the baseline, we gradually increase the delay for one sensor at a time. We reevaluate the object detection performance utilizing the IoU criterion and the related *precision* and *recall* metrics based on (6) and (7). Note that this work does not aim to optimize detection performance but rather to analyze the extent of performance degradation caused by temporal misalignment from desynchronization attacks. Consequently, in the absence of attacks, we expect the baseline performance to yield the best results. We gather data for each scenario, algorithm, and sensor delay combination. The obtained results are grouped by the evaluated algorithm and depicted in Tables 2-5 for the *recall* metric. The results for the *precision* metric can be found in Tables 6-9 in the appendix for completeness. Similar trends can also be observed in this context, which allows us to draw the same conclusions without adding more value to the discussion here. The colored bars visually emphasize the measurement results by partially filling the table cells according to the related numeric values. A performance result of $recall = 0.75$ would fill the respective cell by three-quarters, for example. We observe that across all scenarios and algorithms, the detection performance, on average, clearly decreases as the sensor delays increase. There are a few exceptions from this trend, which we will further discuss in Section 5.3. Additionally, we notice that delays in LiDAR have a higher impact on performance than delays in camera frames. This observation is particularly evident with the *EPNet* algorithm. Consequently, we state our first important finding:

Finding 1: Delays in individual sensor data can significantly reduce the effectiveness of sensor fusion algorithms, whereby longer delays generally result in more significant performance reductions. While both the camera and LiDAR sensors are vulnerable to the attack, delaying the LiDAR data shows a greater impact on the performance.

Table 2 Evaluation results of the *CLOCs* algorithm based on the recall with $\theta = 0.5$.

Delay	Scenario 1		Scenario 2		Scenario 3		Scenario 4	
	Camera	LiDAR	Camera	LiDAR	Camera	LiDAR	Camera	LiDAR
0 ms	0.84	0.84	0.98	0.98	0.87	0.87	0.98	0.98
200 ms	0.22	0.20	0.05	0.01	0.00	0.00	0.08	0.12
400 ms	0.21	0.20	0.08	0.09	0.00	0.00	0.00	0.00
800 ms	0.19	0.18	0.09	0.08	0.00	0.00	0.05	0.05
1600 ms	0.04	0.06	0.06	0.05	0.00	0.00	0.00	0.01

Table 3 Evaluation results of the *EPNet* algorithm based on the recall with $\theta = 0.5$.

Delay	Scenario 1		Scenario 2		Scenario 3		Scenario 4	
	Camera	LiDAR	Camera	LiDAR	Camera	LiDAR	Camera	LiDAR
0 ms	0.89	0.89	0.96	0.96	0.94	0.94	0.96	0.96
200 ms	0.80	0.23	0.91	0.02	0.85	0.04	0.92	0.11
400 ms	0.67	0.25	0.89	0.07	0.55	0.19	0.88	0.01
800 ms	0.62	0.23	0.88	0.07	0.62	0.11	0.83	0.05
1600 ms	0.58	0.09	0.87	0.05	0.79	0.04	0.80	0.02

Table 4 Evaluation results of the *Frustum-ConvNet* algorithm based on the recall with $\theta = 0.5$.

Delay	**Scenario 1**		**Scenario 2**		**Scenario 3**		**Scenario 4**	
	Camera	LiDAR	Camera	LiDAR	Camera	LiDAR	Camera	LiDAR
0 ms	0.67	0.67	0.85	0.85	0.57	0.57	0.93	0.93
200 ms	0.21	0.21	0.08	0.03	0.00	0.06	0.14	0.19
400 ms	0.22	0.20	0.08	0.10	0.00	0.02	0.01	0.03
800 ms	0.16	0.19	0.09	0.08	0.11	0.02	0.07	0.06
1600 ms	0.06	0.07	0.07	0.08	0.02	0.02	0.03	0.02

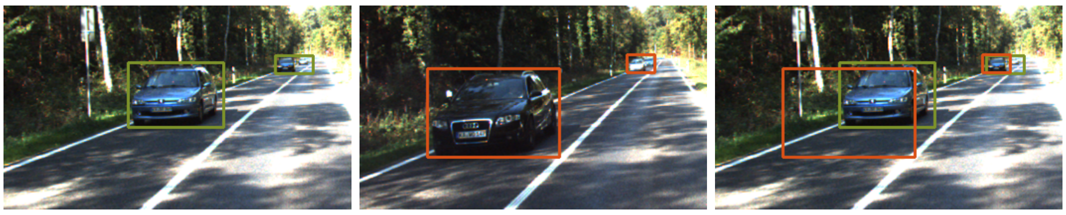
Table 5 Evaluation results of the *VirConv-L* algorithm based on the recall with $\theta = 0.5$.

Delay	**Scenario 1**		**Scenario 2**		**Scenario 3**		**Scenario 4**	
	Camera	LiDAR	Camera	LiDAR	Camera	LiDAR	Camera	LiDAR
0 ms	0.83	0.83	0.94	0.94	0.87	0.87	0.97	0.97
200 ms	0.81	0.76	0.94	0.91	0.72	0.66	0.95	0.94
400 ms	0.77	0.73	0.93	0.92	0.68	0.57	0.95	0.92
800 ms	0.76	0.74	0.92	0.87	0.72	0.64	0.95	0.94
1600 ms	0.75	0.74	0.93	0.85	0.77	0.66	0.94	0.92

5.2 Algorithm Evaluation

We further compare all algorithm implementations to each other to better understand how the fusion types introduced in Section 2.1 affect the attack performance. The data in Table 2 and Table 4 clearly show that both *CLOCs* and *Frustum ConvNet* experience significant performance losses even with slight delays. Both late fusion and multi-stage approaches include steps for object proposal and filtering that operate on individual sensor data before the actual fusion process occurs. If the quality of one raw input is poor due to a delay, the fusion model cannot compensate for it, as it only has access to the preprocessed data that reflects the poor quality of the raw data. Early fusion approaches, on the other hand, can somewhat compensate for that as the model benefits from the raw input in its decision-making, as observed in the *VirConv-L* data shown in Table 5. Although the effects of the attack are still noticeable, the decrease in performance is less severe. With the *EPNet* algorithm, we observe a mixed behavior. While deep fusion algorithms operate on extracted features rather than raw data, these are less processed than final object proposals. This allows the model to potentially achieve better performance compared to late fusion models. In Table 3, this is evident with the camera delays that can be partially compensated by the extracted LiDAR features, but not vice versa. Following this evaluation, we formulate our second finding:

Finding 2: The internal design of sensor fusion algorithms significantly affects their resistance to desynchronization attacks. As early fusion provides access to all available raw data, this approach allows related models to benefit from the existing redundancy and outperform other fusion types that rely on prior feature extraction and object proposals based on an isolated view of individual sensor modalities.



(a) Camera frame at t_0. **(b)** Camera frame at t_8. **(c)** Overlap after 8 frames.

Figure 8 Visualization of overlapping 2D detections at two distinct moments causing counter-intuitive performance increases despite the employed delay attack. The respective 2D regions for the independent detections in (a) and (b) partially overlap, as can be verified in (c), despite being derived from two distinct vehicles at two different points in time.

5.3 Scene Dynamics

Finally, we assess the effects of the scene dynamics. As discussed in Section 2.3, desynchronization attacks are effective only when the recorded data changes over time. Given that *Scenario 1* and *Scenario 2* are generally more dynamic, as outlined in Section 4.3, we expect higher impacts from delay attacks in these scenarios. However, this effect is not quite visible in our data. One possible reason for this result could be the insufficient variation in the dynamics of our selected scenarios, despite their stated differences. On the other hand, we observe a slight increase in performance with higher delays in certain scenarios. This behavior may seem counterintuitive, but it can be explained by examining the flow of the scenario. The algorithms are specifically designed to detect objects in a single-step manner without keeping track of past results. In the genuine scenario, both the camera and LiDAR detect the same object, which can be accurately identified. If we delay one sensor so that it misses the desired object, we create a mismatch of information, resulting in poorer detection results. However, if we wait long enough for a second vehicle to appear at the same spot, both sensors will detect an object, but not the same. In the performance evaluation, the differences between objects do not matter as there is no need to identify and track the object, and thus, the detection performance increases once again. Figure 8 visualizes this behavior.

6 Discussion

The evaluation results show that desynchronization attacks on sensor fusion algorithms significantly affect object detection performance. Although the actual severity might vary, we can conclude that all evaluated algorithms are vulnerable. It is also evident that, on average, greater delays in individual sensors lead to poorer overall detection performance. Exceptions to this trend can be explained by scenarios where different objects appear at the same place at different times. This shows that sensor fusion algorithms generally have no sense of time or any other intelligence that perceives vehicles as unique objects but they rather focus on the geometrical information instead. Considering these aspects would enhance the system's ability to understand its surroundings, ultimately making it more resilient to any disturbances and malicious attacks.

Moreover, we found that delaying the LiDAR data affects the detection performance more significantly, which is consistent with other studies' findings. Both [50] and [25] also conclude that errors in LiDAR data have a more significant impact on overall fusion performance compared to errors within camera frames. While our results support this conclusion, we additionally emphasize other important aspects and reason our analysis based on a larger data basis. This observation is particularly interesting for future efforts in developing defense mechanisms as it suggests prioritizing the security of the LiDAR sensor over that of the camera. Additionally, we can conclude that the chosen fusion approach also influences the detection performance. The employed fusion model cannot effectively address poor input data resulting from a delay attack if it lacks access to the raw data of other sensor modalities, as our experiments have demonstrated. We believe that further research into early fusion approaches could enhance the resilience of object detection systems against desynchronization attacks. By leveraging the redundancy in available sensor data, related models can effectively reduce errors and mitigate the impact of such attacks.

7 Related Work

The problem of 3D object detection performance, especially concerning multi-modal sensor fusion methods, has been thoroughly explored in the literature. The broad scope includes new algorithm proposals that excel over their predecessors in genuine scenarios and an analysis of common pitfalls in sensor fusion, such as data alignment and quality [15, 26, 38, 25, 5, 7]. Also, malicious attacks intentionally disrupting the detection process to impair the results have been covered [51, 40, 3, 27]. This study examines multi-sensor setups, emphasizing the significance of temporal data alignment and the attacks that compromise it.

A significant body of work [35, 51, 40, 43, 3] focuses on adversarial samples, which study how machine learning model performance changes due to malicious alterations in input data. This field emphasizes the integrity of the data concerning malicious attacks rather than the timing of data arrival. Hence, it differs from our defined threat model and is out of scope. The same applies to works addressing delay attacks on a single sensor modality, such as the SlowTrack framework [29], as we concentrate on desynchronization effects in multi-modal sensor setups.

In general, many papers recognize the necessity of temporal alignment in sensor fusion methods. For instance, [5] discusses the general robustness of sensor fusion, including the importance of temporal alignment in the input data. However, no additional analysis regarding the impact of misalignment or experimental results is provided to substantiate this claim. Some papers tackle this issue by proposing synchronization strategies to enhance the model performance. We can distinguish two types: active synchronization methods, which focus on clock synchronization and precise timestamp assignment for accurate timestamping [45, 10, 28, 39] and passive techniques, which do not involve any timestamping at all [33, 22, 37, 9]. All of these approaches assume genuine system behavior and overlook the threat of malicious actions, as discussed in Section 3. Consequently, the studies do not provide insights into the implications of sensor input desynchronization on the fusion performance. Kuhse et al. [25] investigate sensor fusion methods in the context of 3D object detection and analyze the temporal robustness of these algorithms from a theoretical perspective. They provide a formal definition for the misalignment phenomenon and suggest methods to establish latency boundaries or derive design criteria for more robust algorithms. Although the authors present foundational work on temporal alignment, they do not consider additional factors such as scene dynamics in their study. Song et al. [38] assess the effect of random LiDAR delays and introduce a new method to alleviate desynchronization effects. The method utilizes historical LiDAR frames to predict current states, enhancing the algorithm's temporal robustness against data lags. However, the experimental results are limited and primarily aim to justify the applied methodology for improved robustness rather than comprehensively analyzing the specific impact of the introduced sensor delays.

Dong et al. [7] conducted extensive benchmarking to assess the resilience of 3D object detection methods against various corruptions, including temporal misalignment. They evaluate various single-sensor algorithms and some sensor fusion candidates, testing the results on three different datasets. The input desynchronization is simulated by keeping the sensor input identical to the data at the previous timestamp, effectively causing data lag. Gao et al. [15] also present comprehensive benchmarking results on object detection that cover various influencing factors, including temporal aspects. The authors conduct experiments with five different levels (0.1 s to 0.5 s) of camera and LiDAR delays on two object tracking systems. Although both works conclude that sensor fusion methods are vulnerable to temporal misalignment, they do not specifically focus on input desynchronization in their benchmarking results, resulting in a lack of thorough analysis regarding its effects on the overall fusion performance. In [26] and [27], the authors investigate Simultaneous Localization and Mapping (SLAM) algorithms, which can also be based on multi-modal sensor fusion approaches. The experiments involved camera and Inertial Measurement Unit (IMU) data with introduced delays ranging from 0 ms to 20 ms [26], demonstrating performance degradation in the resulting SLAM trajectories.

In conclusion, while many papers acknowledge the issue of temporal alignment of input data, only a few specifically address this concern in their evaluations. However, the evaluations are relatively brief and only support the claim that temporal alignment is generally important in sensor fusion setups without providing further insight into how misalignment affects the algorithms in particular. In our work, we evaluate a wider selection of algorithms and sensor delays while considering additional evaluation criteria, such as fusion types and scene dynamics, to better understand their impact on the overall sensor fusion performance.

8 Countermeasures

Our experiments demonstrate that sensor fusion algorithms are susceptible to desynchronization attacks. To protect against the attack strategies mentioned in Section 3, we discuss two potential mitigation techniques: secure time synchronization and consistency checks.

8.1 Secure Time Synchronization

A significant attack vector in current systems is the missing or unreliable time synchronization that renders assigned timestamps ineffective. To address this issue, a countermeasure is to ensure secure time synchronization, making the timestamps reliable once again. For synchronization, the usage of PTP is recommended, providing accuracies at the sub-microsecond level, which should suffice for all sensor update rates. When using PTP, it is essential to implement proper authentication and integrity protection, such as MACsec and IPsec. To safeguard against delay attacks, extensions such as PTPsec [12] can be employed. Chen et al. [4] present an approach that resists uncertain delays and timestamp falsification attacks by estimating the likelihood of when sensor data was captured. For that, they propose a resilient method that uses probability intervals instead of fixed sampling, aiming to achieve consensus by outputting the overlap of intervals.

8.2 Consistency Checks

In addition to secure time synchronization, sensor fusion systems can also be hardened by introducing consistency checks to verify whether the individual sensor inputs are valid before fusing them. One approach presented by Zhang et al. [53] involves computing multiple depth maps from LiDAR or camera inputs to identify differences and postulate potentially attacked sensors. This approach can be extended to derive general consistency checks among all used sensor modalities to agree on a common perception state. Metrics such as vehicle speed can provide a common basis for comparison among data derived from various sensors. Once established, sensor data is considered genuine and trustworthy only if it aligns with the majority.

9 Conclusion & Future Work

This study investigates the phenomenon of input desynchronization in sensor fusion algorithms for 3D object detection. In particular, we examine four algorithms with distinct internal model architectures on four typical traffic scenarios to understand the impact of delaying the sensor inputs. From our experimental results, we conclude two important findings, namely that all evaluated algorithms are vulnerable to the employed delay attacks and that LiDAR has much more influence on successful detection than the camera. Moreover, we observed that early fusion methods are probably more resilient against these attacks as they benefit from all available raw data, being able to fully exploit the redundancy. More testing must be done with specifically recorded datasets to better understand the dynamics. Also, we need to develop effective countermeasures to prevent the presented delay attacks or, more specifically, to make the algorithms more robust and resilient.

References

1 Sergio Barreto, Aswin Suresh, and Jean-Yves Le Boudec. Cyber-attack on packet-based time synchronization protocols: The undetectable delay box. In *2016 IEEE International Instrumentation and Measurement Technology Conference Proceedings*, pages 1–6. Ieee, 2016. doi:10.1109/I2MTC.2016.7520408.

2 Zhiqiang Cai, Aohui Wang, Wenkai Zhang, Michael Gruffke, and Hendrick Schweppe. 0-days & mitigations: Roadways to exploit and secure connected bmw cars. *Black Hat USA*, 2019(39):6, 2019.

3 Yulong Cao, Ningfei Wang, Chaowei Xiao, Dawei Yang, Jin Fang, Ruigang Yang, Qi Alfred Chen, Mingyan Liu, and Bo Li. 3d adversarial object against msf-based perception in autonomous driving. In *Proceedings of the 3rd Conference on Machine Learning and Systems*, 2020.

4 Yanfeng Chen, Tianyu Zhang, Fanxin Kong, Lin Zhang, and Qingxu Deng. Attack-resilient fusion of sensor data with uncertain delays. *ACM Transactions on Embedded Computing Systems (TECS)*, 21(4):1–25, 2022. doi:10.1145/3532181.

5 Varuna De Silva, Jamie Roche, and Ahmet Kondoz. Robust fusion of lidar and wide-angle camera data for autonomous mobile robots. *Sensors*, 18(8):2730, 2018. doi:10.3390/S18082730.

6 Martin Dimitrievski, David Van Hamme, and Wilfried Philips. Physically accurate lidar simulation for automotive digital twins. In *2024 IEEE SENSORS*, pages 1–4. IEEE, 2024. doi:10.1109/SENSORS60989.2024.10784869.

7 Yinpeng Dong, Caixin Kang, Jinlai Zhang, Zijian Zhu, Yikai Wang, Xiao Yang, Hang Su, Xingxing Wei, and Jun Zhu. Benchmarking robustness of 3d object detection to common corruptions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1022–1032, 2023.

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

9 Yuchuan Du, Bohao Qin, Cong Zhao, Yifan Zhu, Jing Cao, and Yuxiong Ji. A novel spatio-temporal synchronization method of roadside asynchronous mmw radar-camera for sensor fusion. *IEEE transactions on intelligent transportation systems*, 23(11):22278–22289, 2021. doi:10.1109/TITS.2021.3119079.

10 Andrew English, Patrick Ross, David Ball, Ben Upcroft, and Peter Corke. Triggersync: A time synchronisation tool. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6220–6226. IEEE, 2015. doi:10.1109/ICRA.2015.7140072.

11 Andreas Finkenzeller. Sensor Fusion Desynchronization Attacks. Software (visited on 2025-06-23). URL: https://osf.io/x2yrq/files/osfstorage?view_only=f148ad01c4104699b214e76d3cad7388, doi:10.4230/artifacts.23663.

12 Andreas Finkenzeller, Oliver Butowski, Emanuel Regnath, Mohammad Hamad, and Sebastian Steinhorst. Ptpsec: Securing the precision time protocol against time delay attacks using cyclic path asymmetry analysis. In *IEEE INFOCOM 2024 - IEEE Conference on Computer Communications*, pages 461–470, 2024. doi:10.1109/INFOCOM52122.2024.10621345.

13 Andreas Finkenzeller, Anshu Mathur, Jan Lauinger, Mohammad Hamad, and Sebastian Steinhorst. Simutack-an attack simulation framework for connected and autonomous vehicles. In *2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)*, pages 1–7. IEEE, 2023. doi:10.1109/VTC2023-SPRING57618.2023.10200555.

14 Andreas Finkenzeller, Thomas Wakim, Mohammad Hamad, and Sebastian Steinhorst. Feasible time delay attacks against the precision time protocol. In *GLOBECOM 2022-2022 IEEE Global Communications Conference*, pages 3375–3380. IEEE, 2022. doi:10.1109/GLOBECOM48099.2022.10001666.

15 Xinyu Gao, Zhijie Wang, Yang Feng, Lei Ma, Zhenyu Chen, and Baowen Xu. Benchmarking robustness of ai-enabled multi-sensor fusion systems: Challenges and opportunities. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 871–882, 2023. doi:10.1145/3611643.3616278.

16 Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. doi:10.1177/0278364913491297.

17 Keli Huang, Botian Shi, Xiang Li, Xin Li, Siyuan Huang, and Yikang Li. Multi-modal sensor fusion for auto driving perception: A survey. *arXiv preprint arXiv:2202.02703*, 2022. arXiv:2202.02703.

18 Tengteng Huang, Zhe Liu, Xiwu Chen, and Xiang Bai. Epnet: Enhancing point features with image semantics for 3d object detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16*, pages 35–52. Springer, 2020. doi:10.1007/978-3-030-58555-6_3.

19 Jyh-Jing Hwang, Henrik Kretzschmar, Joshua Manela, Sean Rafferty, Nicholas Armstrong-Crews, Tiffany Chen, and Dragomir Anguelov. Cramnet: Camera-radar fusion with ray-constrained cross-attention for robust 3d object detection. In *European conference on computer vision*, pages 388–405. Springer, 2022. doi:10.1007/978-3-031-19839-7_23.

20 Shinpei Kato, Eijiro Takeuchi, Yoshio Ishiguro, Yoshiki Ninomiya, Kazuya Takeda, and Tsuyoshi Hamada. An open approach to autonomous vehicles. *IEEE Micro*, 35(6):60–68, 2015. doi:10.1109/MM.2015.133.

21 Daan Keuper and Thijs Alkemade. The connected car ways to get unauthorized access and potential implications. *Computest, Zoetermeer, The Netherlands, Tech. Rep*, 2018.

22 Peter Kim, Jason Orlosky, and Kiyoshi Kiyokawa. Ar timewarping: A temporal synchronization framework for real-time sensor fusion in head-mounted displays. In *Proceedings of the 9th Augmented Human International Conference*, pages 1–8, 2018. doi:10.1145/3174910.3174919.

23 Taewan Kim and Joydeep Ghosh. On single source robustness in deep fusion models. *Advances in Neural Information Processing Systems*, 32, 2019.

24 Youngseok Kim, Sanmin Kim, Jun Won Choi, and Dongsuk Kum. Craft: Camera-radar 3d object detection with spatio-contextual fusion transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 1160–1168, 2023. doi:10.1609/AAAI.V37I1.25198.

25 Daniel Kuhse, Nils Holscher, Mario Gunzel, Harun Teper, Georg Von Der Bruggen, Jian-Jia Chen, and Ching-Chi Lin. Sync or sink? the robustness of sensor fusion against temporal misalignment. In *2024 IEEE 30th Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 122–134. IEEE, 2024.

26 Ao Li, Han Liu, Jinwen Wang, and Ning Zhang. From timing variations to performance degradation: Understanding and mitigating the impact of software execution timing in slam. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 13308–13315. IEEE, 2022. doi:10.1109/IROS47612.2022.9981275.

27 Ao Li, Jinwen Wang, and Ning Zhang. Chronos: Timing interference as a new attack vector on autonomous cyber-physical systems. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 2426–2428, 2021. doi:10.1145/3460120.3485350.

28 Limeng Lu, Chaofan Zhang, Yong Liu, Wen Zhang, and Yingwei Xia. Ieee 1588-based general and precise time synchronization method for multiple sensors. In *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 2427–2432. IEEE, 2019. doi:10.1109/ROBIO49542.2019.8961658.

29 Chen Ma, Ningfei Wang, Qi Alfred Chen, and Chao Shen. Slowtrack: Increasing the latency of camera-based perception in autonomous driving using adversarial examples. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 4062–4070, 2024. doi:10.1609/AAAI.V38I5.28200.

30 Charlie Miller and Chris Valasek. Remote exploitation of an unaltered passenger vehicle. *Black Hat USA*, 2015(S 91):1–91, 2015.

31 Ramin Nabati and Hairong Qi. Centerfusion: Center-based radar and camera fusion for 3d object detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1527–1536, 2021.

32 Sen Nie, Ling Liu, and Yuefeng Du. Free-fall: Hacking tesla from wireless to can bus. *Briefing, Black Hat USA*, 25(1):16, 2017.

33 Edwin Olson. A passive solution to the sensor synchronization problem. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1059–1064. IEEE, 2010. doi:10.1109/IROS.2010.5650579.

34 Su Pang, Daniel Morris, and Hayder Radha. Clocs: Camera-lidar object candidates fusion for 3d object detection. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10386–10393, 2020. doi:10.1109/IROS45743.2020.9341791.

35 Won Park, Nan Liu, Qi Alfred Chen, and Z Morley Mao. Sensor adversarial traits: Analyzing robustness of 3d object detection sensor fusion models. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 484–488. IEEE, 2021. doi:10.1109/ICIP42928.2021.9506183.

36 Kun Qian, Shilin Zhu, Xinyu Zhang, and Li Erran Li. Robust multimodal vehicle detection in foggy weather using complementary lidar and radar signals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 444–453, 2021. doi:10.1109/CVPR46437.2021.00051.

37 Kejie Qiu, Tong Qin, Jie Pan, Siqi Liu, and Shaojie Shen. Real-time temporal and rotational calibration of heterogeneous sensors using motion correlation analysis. *IEEE Transactions on Robotics*, 37(2):587–602, 2020. doi:10.1109/TRO.2020.3033698.

38 Zhihang Song, Lihui Peng, Jianming Hu, Danya Yao, and Yi Zhang. Timealign: A multi-modal object detection method for time misalignment fusing in autonomous driving. *arXiv preprint arXiv:2412.10033*, 2024. doi:10.48550/arXiv.2412.10033.

39 Josef Steinbaeck, Christian Steger, Eugen Brenner, and Norbert Druml. A hybrid timestamping approach for multi-sensor perception systems. In *2020 23rd Euromicro Conference on Digital System Design (DSD)*, pages 447–454. IEEE, 2020. doi:10.1109/DSD51259.2020.00077.

40 James Tu, Huichen Li, Xinchen Yan, Mengye Ren, Yun Chen, Ming Liang, Eilyan Bitar, Ersin Yumer, and Raquel Urtasun. Exploring adversarial robustness of multi-sensor perception systems in self driving. *arXiv preprint arXiv:2101.06784*, 2021. `arXiv:2101.06784`.

41 Mohd Anas Wajid and Aasim Zafar. Multimodal fusion: a review, taxonomy, open challenges, research roadmap and future directions. *Neutrosophic Sets and Systems*, 45(1):8, 2021.

42 Li Wang, Xinyu Zhang, Ziying Song, Jiangfeng Bi, Guoxin Zhang, Haiyue Wei, Liyao Tang, Lei Yang, Jun Li, Caiyan Jia, et al. Multi-modal 3d object detection in autonomous driving: A survey and taxonomy. *IEEE Transactions on Intelligent Vehicles*, 8(7):3781–3798, 2023. `doi:10.1109/TIV.2023.3264658`.

43 Shaojie Wang, Tong Wu, and Yevgeniy Vorobeychik. Towards robust sensor fusion in visual perception. *arXiv preprint arXiv:2006.13192*, 2020. `arXiv:2006.13192`.

44 Zhixin Wang and Kui Jia. Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal 3d object detection. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1742–1749. IEEE, 2019.

45 Antje Westenberger, Tobias Huck, Martin Fritzsche, Tilo Schwarz, and Klaus Dietmayer. Temporal synchronization in multi-sensor fusion for future driver assistance systems. In *2011 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, pages 93–98. IEEE, 2011.

46 Hai Wu, Chenglu Wen, Shaoshuai Shi, Xin Li, and Cheng Wang. Virtual sparse convolution for multimodal 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21653–21662, 2023. `doi:10.1109/CVPR52729.2023.02074`.

47 Wufei Wu, Renfa Li, Guoqi Xie, Jiyao An, Yang Bai, Jia Zhou, and Keqin Li. A survey of intrusion detection for in-vehicle networks. *IEEE Transactions on Intelligent Transportation Systems*, 21(3):919–933, 2019. `doi:10.1109/TITS.2019.2908074`.

48 Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. *arXiv preprint arXiv:1711.01991*, 2017. `arXiv:1711.01991`.

49 Yanlong Yang, Jianan Liu, Tao Huang, Qing-Long Han, Gang Ma, and Bing Zhu. Ralibev: Radar and lidar bev fusion learning for anchor box free object detection systems. *IEEE Transactions on Circuits and Systems for Video Technology*, 2024.

50 Kaicheng Yu, Tang Tao, Hongwei Xie, Zhiwei Lin, Tingting Liang, Bing Wang, Peng Chen, Dayang Hao, Yongtao Wang, and Xiaodan Liang. Benchmarking the robustness of lidar-camera fusion for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3188–3198, 2023. `doi:10.1109/CVPRW59228.2023.00321`.

51 Youngjoon Yu, Hong Joo Lee, Byeong Cheon Kim, Jung Uk Kim, and Yong Man Ro. Investigating vulnerability to adversarial examples on multimodal data fusion in deep learning. *arXiv preprint arXiv:2005.10987*, 2020. `arXiv:2005.10987`.

52 Daniel Zelle, Timm Lauser, Dustin Kern, and Christoph Krauß. Analyzing and securing some/ip automotive services with formal and practical methods. In *Proceedings of the 16th International Conference on Availability, Reliability and Security*, pages 1–20, 2021. `doi:10.1145/3465481.3465748`.

53 Jindi Zhang, Yifan Zhang, Kejie Lu, Jianping Wang, Kui Wu, Xiaohua Jia, and Bin Liu. Detecting and identifying optical signal attacks on autonomous driving systems. *IEEE Internet of Things Journal*, 8(2):1140–1153, 2020. `doi:10.1109/JIOT.2020.3011690`.

54 Yuxuan Zhang, Bo Dong, and Felix Heide. All you need is raw: Defending against adversarial attacks with camera image pipelines. In *European conference on computer vision*, pages 323–343. Springer, 2022. `doi:10.1007/978-3-031-19800-7_19`.

A Appendix

The tables below present the *precision* data collected from our experiments. We observe the same trends as discussed for the *recall* data presented in Section 5.

Table 6 Evaluation results of the *CLOCs* algorithm based on the precision with $\theta = 0.5$.

Delay	Scenario 1		Scenario 2		Scenario 3		Scenario 4	
	Camera	LiDAR	Camera	LiDAR	Camera	LiDAR	Camera	LiDAR
0 ms	0.97	0.97	0.73	0.73	0.98	0.98	0.75	0.75
200 ms	0.27	0.25	0.04	0.01	0.00	0.00	0.06	0.09
400 ms	0.28	0.27	0.06	0.07	0.00	0.00	0.00	0.00
800 ms	0.28	0.29	0.07	0.06	0.00	0.00	0.04	0.04
1600 ms	0.07	0.11	0.04	0.04	0.00	0.00	0.00	0.01

Table 7 Evaluation results of the *EPNet* algorithm based on the precision with $\theta = 0.5$.

Delay	Scenario 1		Scenario 2		Scenario 3		Scenario 4	
	Camera	LiDAR	Camera	LiDAR	Camera	LiDAR	Camera	LiDAR
0 ms	0.77	0.77	0.46	0.46	0.75	0.75	0.46	0.46
200 ms	0.49	0.16	0.46	0.01	0.32	0.02	0.42	0.05
400 ms	0.42	0.15	0.44	0.03	0.20	0.07	0.39	0.01
800 ms	0.46	0.15	0.45	0.04	0.24	0.05	0.38	0.02
1600 ms	0.41	0.06	0.44	0.03	0.32	0.02	0.37	0.01

Table 8 Evaluation results of the *Frustum-ConvNet* algorithm based on the precision with $\theta = 0.5$.

Delay	Scenario 1		Scenario 2		Scenario 3		Scenario 4	
	Camera	LiDAR	Camera	LiDAR	Camera	LiDAR	Camera	LiDAR
0 ms	0.98	0.98	0.94	0.94	1.00	1.00	0.98	0.98
200 ms	0.47	0.48	0.09	0.04	0.00	0.16	0.15	0.22
400 ms	0.51	0.46	0.11	0.13	0.00	0.05	0.01	0.04
800 ms	0.36	0.37	0.15	0.12	0.20	0.06	0.09	0.11
1600 ms	0.19	0.14	0.11	0.12	0.03	0.07	0.05	0.05

Table 9 Evaluation results of the *VirConv-L* algorithm based on the precision with $\theta = 0.5$.

Delay	Scenario 1		Scenario 2		Scenario 3		Scenario 4	
	Camera	LiDAR	Camera	LiDAR	Camera	LiDAR	Camera	LiDAR
0 ms	0.95	0.95	0.76	0.76	0.91	0.91	0.65	0.65
200 ms	0.95	0.88	0.76	0.69	0.92	0.56	0.66	0.64
400 ms	0.95	0.86	0.77	0.68	0.97	0.63	0.67	0.61
800 ms	0.94	0.91	0.76	0.68	0.94	0.50	0.68	0.57
1600 ms	0.95	0.88	0.78	0.69	0.90	0.47	0.67	0.54