Cooperative Perception for Safe Control of Autonomous Vehicles under LiDAR Spoofing Attacks

Hongchao Zhang*
Washington University in St. Louis
hongchao@wustl.edu

Zhouchi Li*
Worcester Polytechnic Institute
zli4@wpi.edu

Shiyu Cheng
Washington University in St. Louis
cheng.shiyu@wustl.edu

Andrew Clark
Washington University in St. Louis
andrewclark@wustl.edu

***Abstract*—Autonomous vehicles rely on LiDAR sensors to detect obstacles such as pedestrians, other vehicles, and fixed infrastructures. LiDAR spoofing attacks have been demonstrated that either create erroneous obstacles or prevent detection of real obstacles, resulting in unsafe driving behaviors. In this paper, we propose an approach to detect and mitigate LiDAR spoofing attacks by leveraging LiDAR scan data from other neighboring vehicles. This approach exploits the fact that spoofing attacks can typically only be mounted on one vehicle at a time, and introduce additional points into the victim's scan that can be readily detected by comparison from other, non-modified scans. We develop a Fault Detection, Identification, and Isolation procedure that identifies non-existing obstacle, physical removal, and adversarial object attacks, while also estimating the actual locations of obstacles. We propose a control algorithm that guarantees that these estimated object locations are avoided. We validate our framework using a CARLA simulation study, in which we verify that our FDII algorithm correctly detects each attack pattern.**

I. Introduction

Autonomous driving systems rely on perception modules to understand their environments, including their own positions as well as the locations of pedestrians, vehicles, and other obstacles. Perception modules develop this understanding using data from sensors such as cameras, Global Positioning Systems, RADAR, and Light Detection and Ranging (LiDAR). LiDARs, which measure the distances from the LiDAR transceiver to obstacles, provide 360° view and 3D representation, namely point cloud, of the environment rather than a 2D image as from a camera, and thus are crucial sensors for perception in autonomous vehicles (AVs).

Since LiDAR perception has a significant impact on the safety-critical decisions of AVs, many prior research efforts have been made to investigate the security of LiDAR perception. The LiDAR perception can be compromised by relay attacks [6] [4] [18] and adversarial object attacks [7] [21]. In relay attacks, a relay spoofer injects adversarial points in the point cloud, disturbs the outputs of the object detection algorithms, and either creates the perception of a fake object or hides an existing objects. In adversarial object attacks, a well-designed object fools the object detection algorithms and makes itself undetectable. While sensor fusion based methods can partially mitigate the impact of LiDAR attacks [5], such methods can still be thwarted by an adversary who can target multiple sensor modalities simultaneously.

In this paper, we propose a new approach to detect and mitigate LiDAR spoofing attacks on AVs. Our approach leverages Connected and Autonomous Vehicle [2] paradigms to share LiDAR sensor data among neighboring vehicles. Indeed, while techniques such as Collective Perception Message (CPM) have been proposed to cooperatively perceive the environment [20], to the best of our knowledge this information sharing has not been used to detect and mitigate spoofing attacks. Our approach is based on two insights. First, current demonstrated LiDAR spoofing hardwares can only target a single LiDAR sensor, and hence simultaneously spoofing multiple vehicles in a believable manner would involve a burdensome level of coordination among multiple distributed spoofers. Second, relay-based LiDAR attacks are fundamentally based on introducing new points into the LiDAR point cloud. While these spurious points may fool the targeted sensor, they will be absent from the scans of neighboring vehicles. We make the following specific contributions:

- We propose a safe control system for LiDAR-perception-based AVs based on the point cloud from neighboring vehicles. In the proposed system, a Fault Detection, Identification, and Isolation (FDII) module detects and classifies the attacks, and updates the unsafe region for the vehicle. A safe controller guarantees the safety of the system based on the updated unsafe region.
- We analyze the correctness of the results from the FDII module. We show that the FDII module can detect and classify attacks correctly and output the unsafe region containing the projection of obstacles.
- Our results are validated through CARLA [9], in which we show that the proposed FDII procedure correctly detects multiple attack types and reconstructs

∗ : Authors contributed equally.
This work is sponsored by NSF grant CNS-1941670.

Symposium on Vehicles Security and Privacy (VehicleSec) 2023
27 February 2023, San Diego, CA, USA
ISBN 1-891562-88-6
https://dx.doi.org/10.14722/vehiclesec.2023.23066
www.ndss-symposium.org

the true unsafe region. We then show that, under our control algorithm, the vehicle reaches the given target while avoiding an obstacle.

The rest of the paper is organized as follows. Section II presents related work. Section III states the LiDAR observation model and threat model. Section IV presents the proposed method. Section V contains simulation results. Section VI concludes the paper.

II. Related Work

LiDAR sensors have been demonstrated to be vulnerable to spoofing attacks in [18]. Machine learning-based LiDAR detection was also shown to be vulnerable in [10]. LiDAR spoofing attacks focus on falsifying non-existing obstacles or hiding existing obstacles. Spoofing attacks that aim to create non-existing obstacles mainly use relay attack [6], in which an adversary fires laser to the LiDAR measurement unit with the same wavelength to inject false points. To hide an object from being detected by LiDAR sensor, methods include adversarial objects [7] and physical removal attacks [4]. Adversarial objects are synthesized such that deep neural network based detection modules fail to detect in a certain range of distance and angle. Physical removal attacks hide arbitrary objects, such as pedestrians, by relay attacks.

Countermeasures to LiDAR spoofing have been proposed in recent years. Defense approaches such as random sampling and randomizing waveforms focus on robust perception in a single sensor scenario. Random sampling proposed in [8] uses robust RANSAC method to randomly sample features from the point clouds. The approach presented in [12] randomizes the pulses' waveforms. However, a shortcoming in practical perception of these approaches is the increase in cost. RANSAC requires high computational capability to formulate the momentum model for adversarial detection [8]. The approach presented in [12] introduces extra modulation components into the lens system and may decrease the sensitivity of LiDAR [19]. A vehicle system is usually equipped with more than one sensors to estimate states and observe its surroundings. One cost-efficient method to increase robustness of estimation in faulty and adversarial environments is to use redundant information. Such a redundancy-based approach includes sensor fusion [25] and multiple sensor overlapping. However, existing work on fault-tolerant estimation with multiple sensor overlapping focus more on the case where an agent is equipped with redundant sensors, but leave the problem of cooperative robust perception less studied.

With the development of communication and smart city, the concept of Connected Vehicles (CVs), which connect with other vehicles, pedestrians, and infrastructures, are often realized with Autonomous Vehicles (AVs) simultaneously [2]. CVs provide cooperative perception that is necessary to the fully AVs which do not rely on human supervision [17]. The cooperative perception may also benefit the defense of LiDAR attack by providing redundant information to neighboring vehicles. However, existing research focuses on enhancing the vehicle itself by adding redundant LiDAR sensors, reducing the signal-receiving angle, transmitting pulses in random directions, and randomizing the pulses' waveforms [18] [13] [14], and less attention has been paid on utilizing cooperative perception in the CV environment.

Recent work such as [24] proposed a V2X framework that is robust to adversarial noise perturbation and collaboration attack. However, the potential of fault tolerance framework of V2X to mitigate LiDAR spoofing attacks is less studied.

III. Observation and Threat Models

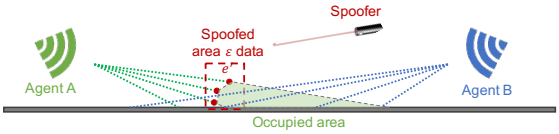
In this section, we introduce the LiDAR observation and threat model that we consider in this paper.

A. LiDAR Observation Model

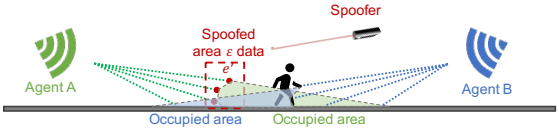
A LiDAR sensor fires and collects n_s laser beams and calculates the relative distance and angles to objects. For a laser beam indexed $i \in [1, n_s] \subseteq \mathbb{Z}^+$, we let $p_i := (s_i^r, s_i^a, s_i^\phi)$, where s_i^r denotes the range, s_i^a denotes the horizontal angle, and s_i^ϕ denotes vertical angle. A LiDAR sensor observes the environment by constructing a scan $S := \{p_i,\ 1 \leq i \leq n_s\}$. We denote the Cartesian translated LiDAR scan S measured at pose x as $\mathcal{O}(x, S) := \{(o_i^x, o_i^y, o_i^z), 1 \leq i \leq n_s\}$, where o_i^x, o_i^y, and o_i^z denote the x, y, and z coordinates of p_i. We assume that the vehicle has a default map, containing the locations of the infrastructure (for example, walls).

B. Threat Model

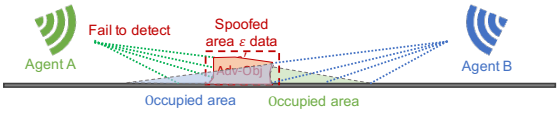
The purpose of the attacks is to disrupt the output of object detection algorithms of the LiDAR perception by either falsifying non-existing obstacles or hiding existing obstacles. Attacks on the positioning of the vehicle are out of scope. In this paper, we consider three attacks with different purposes and implementation methods, as shown in Fig. 1.



(a) Relay attacks falsify non-existing obstacles.



(b) Physical removal attacks hide the object from the LiDAR detector.



(c) Adversarial objects can hide from the LiDAR detector.

Fig. 1: Illustration of three attack types against LiDAR-based perception. Each attack type leaves a trace in the raw data that can be detected using our proposed approach.

Non-Existing Obstacle (NEO): The spoofer falsifies non-existing obstacles by introducing relay perturbations (Fig. 1a).

In a relay attack proposed in [6], the adversary fires laser beams to inject artificial points e' into a LiDAR scan S. The resulting scan has points $S \cup e'$. Due to the physical limitation of the spoofing hardware, the injected point can only be within a very narrow spoofing angle. Hence, in this paper, we assume that the relay adversary can only spoof one LiDAR sensor. As a result of the attack, the LiDAR detection algorithm incorrectly detects an obstacle between the spoofer and the LiDAR sensor.

Physical Removal Attack (PRA): As shown in Fig. 1b, the spoofer implements the physical removal attacks by sending a relay signal to the LiDAR receiver. The LiDAR detection algorithm believes that the true obstacle does not exist. In the scan S of the compromised LiDAR, there are artificial points e' between the true obstacle and the LiDAR. In order to realize the attack successfully, artificial points e' will reach the LiDAR receiver first and obscure the true obstacle. The true LiDAR signal reflected by the obstacle will be discarded by the LiDAR receiver. Due to the physical limitation of the spoofing hardware, only one LiDAR is compromised by the spoofer.

We further divide Attack PRA into three categories, based on whether the area containing the artificial points is fully observed by the uncompromised LiDAR sensor (PRA1), partially observed (PRA2), or not observed (PRA3).

Adversarial Object (AO): Adversarial objects are synthesized to be undetectable to the object detection algorithms of the LiDAR perception systems in a certain range of distance and angle (Fig. 1c). The adversarial objects introduce disturbance signal e' in the LiDAR scan S. More than one LiDAR sensor may be compromised by one adversarial object.

In this paper, we assume that at most one attack occurs. The attack could be any of attacks NEO, PRA, or AO, and the autonomous vehicle (AV) does not know the attack type a priori.

IV. Proposed Detection and Control Approach

In this section, we propose a detection and control approach to ensure safety of the vehicle in an adversarial environment. The proposed approach is illustrated as Fig. 2. The victim agent leverages LiDAR observations from neighboring agents to detect and identify faults in two steps, namely, occupied area identification and the FDII, which are described as follows.

A. Occupied Area Identification

In this subsection, we present our method to identify and extract the area that either contains obstacles or is not observable by the LiDAR, which we denote the occupied area. The detected occupied area will then be combined with the information from other vehicles to detect attacks and compute the unsafe region (Section IV-B).

We first prune the raw data of the observation $\mathcal{O}(x, S)$ by removing the points corresponding to the infrastructure (for example, walls) on a default map. For the rest of the paper, we use $\mathcal{O}(x, S)$ to denote the pruned observation. For Agent A, denote $L := \{j : \mathcal{I}_A \cap \mathcal{I}_j \neq \emptyset\}$ as the set of agents which have an overlapping scan-covered area with Agent A. We define the vertical projection operation $\mathcal{P}(\mathcal{O}(x, S)) \longrightarrow \mathcal{Y}$, which takes an observation $\mathcal{O}(x, S)$ as input and outputs the set $\mathcal{Y}$ of the projections of the points in $\mathcal{O}(x, S)$ onto the x-y coordinate plane. We define a non-obstacle scan-covered area $\mathcal{I}_j = \mathcal{P}(\bar{\mathcal{O}}(x, S))$ as the projection of the observation of Agent j, where $\bar{\mathcal{O}}(x, S)$ denotes the pruned observation within maximum LiDAR perception range.

We utilize the altitude coordinates of the points to distinguish the ground and the obstacle as described in [15] [3]. We let ζ^z denote the estimation error of the altitude coordinate. The detection algorithm identifies $p_i = (o_i^x, o_i^y, o_i^z)$ as belonging to the ground if $o_i^z \leq \zeta^z$ and belonging to an obstacle if $o_i^z > \zeta^z$. After ground removal, we use the 3D bounding box provided by the LiDAR-based 3D object detection algorithms of the autonomous vehicles (AVs) [26] to cluster the points into a collection of obstacles, denoted $O_k^b(x, S) \subseteq O(x, S)$. Each obstacle has a corresponding bounding box denoted $\mathcal{B}_k \subseteq \mathbb{R}^3$. Here, the index k ranges from 1 to n_o^j, where n_o^j denotes the number of obstacles detected by Agent j. Note that the set $O_1^b(x, S), \ldots, O_{n_o^j}^b(x, S)$ may contain fake obstacles introduced by an adversary. We regard the points that do not belong to any bounding box as the points introduced by Attack PRA or Attack AO. We further calculate the oblique projections $\mathcal{O}_k^p(x, S)$ of the points in $\mathcal{O}_k^b(x, S)$, which consists of the points where a straight line from the sensor to each point in $\mathcal{O}_k^b(x, S)$ intersects the ground. We have that $\mathcal{O}_k^p(x, S) \subseteq \mathbb{R}^3$.

We let $\mathcal{U}_{jk}$ denote the occupied area corresponding to Obstacle k observed by Agent j, which consists of the area that contains the obstacle as well as the area that is obscured by the obstacle. Formally, the occupied area $\mathcal{U}_{jk}$ is computed as the convex hull of $\mathcal{P}(\mathcal{O}_k^b(x, S)) \cup \mathcal{O}_k^p(x, S)$. The convex hull can be calculated via open-source tools such as Scipy [22]. In order to compute the convex hull, we first obtain a collection of pairs of points $\{(x_1^l, y_1^l), (x_2^l, y_2^l) : l = 1, \ldots, n_{jk}^h\}$ from the object detection and oblique projection algorithms, where n_{jk}^h is the number of the boundaries of Obstacle k observed by Agent j. For each pair indexed l, we compute a half-plane constraint $h_l^{jk}(x) \leq 0$ with $h_l^{jk}(x) = a_l^{jk} \times x + b_l^{jk}$ where a_l^{jk} and b_l^{jk} are defined as $a_l^{jk} = \frac{y_2^l - y_1^l}{x_2^l - x_1^l}$ and $b_l^{jk} = \frac{x_2^l \times y_1^l - x_1^l \times y_2^l}{x_2^l - x_1^l}$. The convex hull is equal to the intersection of the half-plane constraints, i.e.,

$$\mathcal{U}_{jk} = \bigcap_{l=1}^{n_{jk}^h} \{x : h_l^{jk}(x) \leq 0\}.$$

The occupied area computed by the above procedure may not fully contain the obstacle due to the fact that the obstacle location must be interpolated from a finite number of samples. We enhance the robustness of these sampling errors by changing the boundaries of the occupied area to $h_l^{jk}(x) - ||a_l^{jk}||\zeta_h \leq 0$, where $\zeta_h = \zeta_n + \zeta_r$, where ζ_n is the observation noise bound and ζ_r is a bound on the distance between neighboring sample points that can be obtained from the distance to the object and the angular resolution of the LiDAR. We assume that the LiDAR resolution is sufficiently large such that each point on the obstacle that is in the line-of-sight of the LiDAR is at most ζ_r distance away from at least one scan point.

In what follows, we will show that each obstacle visible to Agent j (including false or adversarial objects) is contained in

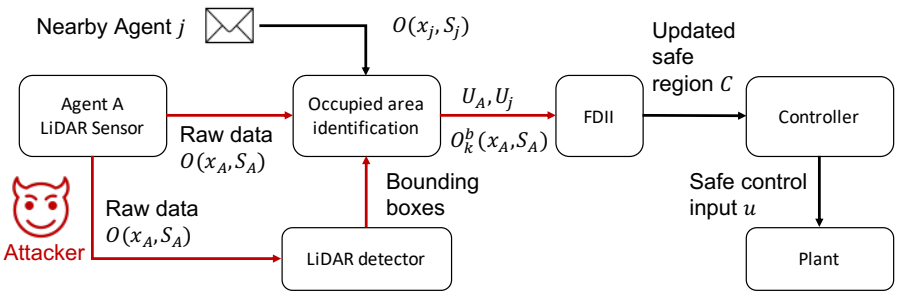


Fig. 2: Schematic illustration of the proposed approach: to identify attacks marked in red, Agent A requests point cloud from nearby agents, i.e., Agent j. Then, FDII module takes $\mathcal{U}_A, \mathcal{U}_j, \mathcal{O}_k^b(x_A, S_A)$ and outputs detected attack type and updated safe region $\mathcal{C}$. Finally, controller output safe control input u.

an occupied region that is computed according to the procedure described above. Let $P^{ob} \subseteq \mathbb{R}^3$ denote the location of an obstacle. We divide the obstacle into two parts. The first part is the set of points in P^{ob} that have line-of-sight with the LiDAR. We denote the first part as P_1. The second part, which is denoted as P_2, is the set of points in P^{ob} that do not have line-of-sight with the LiDAR.

Assumption 1: For Agent $j \in L$ with occupied areas $\mathcal{U}_{jk}, k \in \{1, \ldots, n_o^j\}$ and Obstacle $k' \in \{1, \ldots, n_o^j\}$ with the set of points $P_{k'}^{ob} \subseteq \mathbb{R}^3$, we have $\mathcal{P}(P^{ob}) \bigcap (\cup_{k \in \{1,\ldots,n_o^j\} \setminus \{k'\}} \mathcal{U}_{jk}) = \emptyset$.

Assumption 1 implies that for any obstacle visible to agent j, there is no overlap between the obstacle and the occupied area of any other obstacle detected by agent j.

Lemma 1: Suppose that Assumption 1 holds and we are given the observation of an obstacle $\mathcal{O}_k^b(x, S)$ in a bounding box and the set of oblique projections $\mathcal{O}^p(x, S)$. If occupied area $\mathcal{U}$ is computed as the convex hull of $\mathcal{P}(\mathcal{O}_k^b(x, S)) \cup \mathcal{O}^p(x, S)$, then $\mathcal{P}(P^{ob}) \subseteq \mathcal{U}$.

The proof is straight forward by proving $\mathcal{P}(P_1) \subseteq \mathcal{U}$ and $\mathcal{P}(P_2) \subseteq \mathcal{U}$ with details in [28] omitted here.

B. Fault Detection, Identification, and Isolation

The objective of the proposed fault detection, identification, and isolation (FDII) module is to detect the deviations between the observations of different agents, identify the spoofed agent, isolate the corrupted parts of the raw data, and restore the true unsafe region. The proposed FDII module is illustrated as Fig. 3. FDII first iterates over all detected obstacles to detect faults by leveraging LiDAR scan data from other neighboring vehicles. It then removes points contained in the bounding box and uses the residual point cloud for false data detection and identification.

We first describe how Agent A incorporates LiDAR information from other neighboring agents. At each time, Agent A collects the current observations of nearby agents $\mathcal{O}(x_j, S_j), \forall j \in L, j \neq A$, position x_j, and sensor information including observation noise bound and resolution bound. The corresponding observations $\mathcal{O}(x_A, S_j)$, the observations of obstacles $\mathcal{O}_k^b(x_A, S_j)$, and the occupied areas $\mathcal{U}_{jk}$ are calculated by Agent A by repeating the steps described in Section IV-A. In $\mathcal{O}(x_A, S_j)$, Agent A marks the points in its

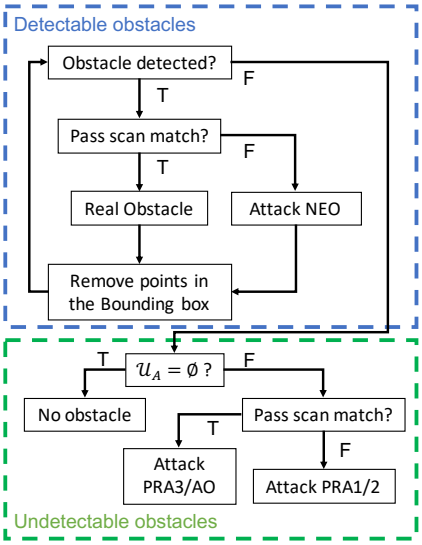


Fig. 3: Decision tree of the FDII module: in the blue box, we iterate over detectable obstacles to detect faults. Then we remove points contained in bounding boxes and pass the remaining point cloud to the green box to identify undetectable obstacles.

bounding boxes as $\mathcal{O}_k^b(x_A, S_j) = \mathcal{O}(x_A, S_j) \cap \mathcal{B}_k$. For each $\mathcal{O}_k^b(x_A, S_j)$, Agent A calculates the corresponding $\mathcal{O}_k^p(x, S)$. Agent A then calculates $\mathcal{U}_{jk}$ by computing the convex hull of $\mathcal{P}(\mathcal{O}_k^b(x, S)) \cup \mathcal{O}_k^p(x, S)$.

We define the scan points in observation $\mathcal{O}(x_A, S_A)$ that is not detected by Agent A as $\mathcal{O}_A^u(x_A, S_A) := \mathcal{O}(x_A, S_A) \setminus \bigcup_k \mathcal{O}_k^b(x_A, S_A)$ and the undetected points in observation $\mathcal{O}(x_A, S_j)$ as $\mathcal{O}_j^u(x_A, S_A) := \mathcal{O}(x_A, S_j) \setminus \bigcup_k \mathcal{O}_k^b(x_A, S_j)$. We then compute their corresponding occupied area $\mathcal{U}_A^u$ and $\mathcal{U}_j^u$ by calculating their convex hulls.

We now analyze how each of the attacks defined in Section III-B affects the LiDAR perception and occupied area identification introduced in Section IV-A. In what follows, we use A to denote the index of the victim agent and analyze whether the obstacle observed by Agent A could also be observed and validated by other agents, i.e. B under each of the attacks defined in Section III-B.

For Attack NEO, the artificial points e' introduce a fake obstacle with points $\mathcal{O}_k^b(x_A, S_A \cup e')$. Since there is no true obstacle, we have $\mathcal{U}_{Bk} = \emptyset$, and as a corollary, $\mathcal{P}(\mathcal{O}_k^b(x_A, S_A)) \nsubseteq \mathcal{U}_{Bk}$.

For Attack PRA1, the spoofed obstacle location does not overlap with the occupied area of agent B. Hence $\mathcal{P}(\mathcal{O}_A^u(x_A, S_A \cup e')) \nsubseteq \mathcal{U}_B^u$ and $\mathcal{P}(\mathcal{O}_A^u(x_A, S_A \cup e')) \cap \mathcal{U}_B^u = \emptyset$.

For Attack PRA2, there are also fake $\mathcal{O}_A^u(x_A, S_A \cup e')$ and non-empty $\mathcal{U}_B^u$. However, $\mathcal{P}(\mathcal{O}_A^u(x_A, S_A \cup e')) \nsubseteq \mathcal{U}_B^u$ and $\mathcal{P}(\mathcal{O}_A^u(x_A, S_A)) \cap \mathcal{U}_B^u \neq \emptyset$ in this case because Agent B could only observe the partial space of $\mathcal{O}_A^u(x_A, S_A \cup e')$.

For Attack PRA3 and AO, $\mathcal{P}(\mathcal{O}_A^u(x_A, S_A \cup e'))$ is fully covered by non-empty $\mathcal{U}_B^u$, which means $\mathcal{P}(\mathcal{O}_A^u(x_A, S_A \cup e')) \subseteq \mathcal{U}_B^u$.

For the true obstacle, there is non-empty $\mathcal{O}_k^b(x_A, S_A \cup e')$ and non-empty $\mathcal{U}_{Bk}$ corresponding to the true obstacle. According to Lemma 1, we have $\mathcal{P}(P^{ob}) \subseteq \mathcal{U}_{Bk}$. Since $\mathcal{O}_k^b(x_A, S_A \cup e') \subseteq \mathcal{P}(P^{ob})$, we have $\mathcal{O}_k^b(x_A, S_A \cup e') \subseteq \mathcal{U}_{Bk}$.

The following lemma uses the preceding analysis to describe how the scan data from agent B can be used to detect and identify the attack type.

Lemma 2: If $\mathcal{P}(\mathcal{O}_k^b(x_A, S_A \cup e')) \backslash \mathcal{U}_{Bk} \neq \emptyset$, then Agent A is being targeted by an attack of type NEO. If $\mathcal{P}(\mathcal{O}_k^b(x_A, S_A \cup e')) \setminus \mathcal{U}_{Bk} = \emptyset$, then Obstacle k is a true obstacle. If $\mathcal{P}(\mathcal{O}_A^u(x_A, S_A \cup e')) \backslash \mathcal{U}_B^u \neq \emptyset$, then Agent A is being targeted by PRA1 or PRA2. If $\mathcal{P}(\mathcal{O}_A^u(x_A, S_A \cup e')) \setminus \mathcal{U}_B^u = \emptyset$, then Agent A is under Attack PRA3 or AO.

The proof derived from previous analysis is omitted. For a detailed treatment, please refer [28].

We check whether $\mathcal{P}(\mathcal{O}_k^b(x_A, S_A)) \setminus \mathcal{U}_{Bk}$ is empty as follows. For each point $p \in \mathcal{P}(\mathcal{O}_k^b(x_A, S_A))$, we check to see whether p satisfies $h_l^{Bk}(p) - ||a_l^{jk}||\zeta_h \leq 0, \forall l \in \{1, \ldots, n_{Bk}^h\}$. If not, we put p into $\mathcal{P}(\mathcal{O}_k^b(x_A, S_A)) \setminus \mathcal{U}_{Bk}$. We name this operation a scan match.

We design the decision tree as shown in Fig. 3 to detect each attack. Agent A iterates the bounding boxes to check whether an object is detected. The points in the bounding box belong to either the non-existing obstacle from Attack NEO or the true obstacle. After detecting and identifying the attack or authenticating that it is the true obstacle, Agent A removes the points from the observation, then move on to the next bounding box. After traversing all bounding boxes iteratively, Agent A checks whether there is still observation of the obstacle. If not, there is no more attack or obstacle. If there is the observation of the obstacle, there is an attack and a scan match will be done to decide the classification of the attack.

After detecting the scan mismatch and identifying the spoofed agent, the proposed FDII algorithm isolates the corrupted parts of the raw data and updates the unsafe region by extracting the intersection of the unions of the occupied areas of the agents indexed in L.

The following theorem shows that the updated unsafe region $(\cup_{k=1,\ldots,n_o^A} \mathcal{U}_{Ak}) \bigcap (\cup_{m=1,\ldots,n_o^j} \mathcal{U}_{jm})$ contains $\mathcal{P}(P_k^{ob}), \forall k \in \{1, \ldots, n_o^A\}$. We note that Assumption 1 is not required in Theorem 1.

Theorem 1: Suppose we are given the occupied areas $\mathcal{U}_{Ak}$ of Agent A and $\mathcal{U}_{jk}$ of Agent j, $k \in \{1, \ldots, n_o^j\}$ in the area of $\mathcal{I}_A \cap \mathcal{I}_j$. If $\mathcal{P}(P_k^{ob}) \subseteq \mathcal{I}_A \cap \mathcal{I}_j$, then $\mathcal{P}(P_k^{ob}) \subseteq (\cup_{k=1,\ldots,n_o^A} \mathcal{U}_{Ak}) \bigcap (\cup_{m=1,\ldots,n_o^j} \mathcal{U}_{jm})$ for any of the attack types NEO, PRA, or AO.

We omit the proof due to space limit. For a detailed treatment, please refer [28].

To calculate $(\cup_{k=1,\ldots,n_o^A} \mathcal{U}_{Ak}) \bigcap (\cup_{m=1,\ldots,n_o^j} \mathcal{U}_{jm})$, which is equivalent to $\bigcup_{k'=1,\ldots,n_o^A} (\mathcal{U}_{Ak'} \bigcap (\cup_{k=1,\ldots,n_o^j} \mathcal{U}_{jk}))$, Agent A approximates the set $\mathcal{U}_{Ak'} \bigcap (\cup_{k=1,\ldots,n_o^j} \mathcal{U}_{jk})$ by computing the convex hull of all scan points whose projections are contained in $\mathcal{U}_{Ak'} \bigcap (\cup_{k=1,\ldots,n_o^j} \mathcal{U}_{jk})$. Then Agent A computes a set of half-plane constraints $h_l^{jk'}(x), l = \{1, \ldots, n_{jk'}^h\}$ based on the vertices, where $h_l^{jk'}(x) - ||a_l^{jk}||\zeta_h < 0$ describes the corresponding half-plane and $n_{jk'}^h$ is the number of half-planes forming the convex hull $\mathcal{U}_{jk'}$. We define $\bar{h}_{jk'}(x) = \max_l h_l^{jk'}(x)$.

C. Safe Control

In this subsection, we present the safe vehicle controller based on the unsafe region provided by the FDII. The kinematic bicycle model [16] adopted in this paper is defined as

$$\begin{bmatrix} \dot{\phi} \\ \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} \frac{v}{l} \sin \psi \\ v \cos(\phi + \psi) \\ v \sin(\phi + \psi) \end{bmatrix} \tag{1}$$

where ϕ is the heading angle between the orientation and the x-axis, (x, y) is the position of the reference point (at the middle of the front axle, between the front wheels), v is the forward velocity at the reference point, l is the wheelbase, and ψ is the steering angle.

For implementation, we discretize Eq. 1 using forward differencing method with the sample time dt to obtain

$$\begin{aligned} \mathbf{x}[t+1] &= \begin{bmatrix} \phi[t+1] \\ x[t+1] \\ y[t+1] \end{bmatrix} = f(\mathbf{x}[t], \mathbf{u}[t]) \\ &= \begin{bmatrix} \phi[t] + dt \frac{v[t]}{l} \sin \psi[t] \\ x[t] + dtv[t] \cos(\phi[t] + \psi[t]) \\ y[t] + dtv[t] \sin(\phi[t] + \psi[t]) \end{bmatrix} \end{aligned} \tag{2}$$

where $\mathbf{u}[t] = (v[t], \psi[t])'$.

In order to avoid collision between the vehicle and the unsafe region, we utilize Model Predictive Control with Discrete-Time Control Barrier Function (MPC-CBF) [27]. The controller solves the following finite-time optimization problem with prediction horizon T at each time step

$$\min_{\mathbf{u}[t:t+T-1]} \; (\mathbf{x}[t+T] - \mathbf{x}_r)^T F (\mathbf{x}[t+T] - \mathbf{x}_r) \tag{3a}$$

$$+ \sum_{t'=t}^{t+T-1} (\mathbf{x}[t'] - \mathbf{x}_r)^T Q (\mathbf{x}[t'] - \mathbf{x}_r) + \mathbf{u}[t']^T R \mathbf{u}[t']$$

$$\text{s.t. } \bar{h}_{jk'}(\mathbf{x}[t+1]) - \bar{h}_{jk'}(\mathbf{x}[t]) \geq -\gamma \bar{h}_{jk'}(\mathbf{x}[t]),$$

$$k' = 1, \ldots, n_o^A \tag{3b}$$

$$\mathbf{x}[t'+1] = f(\mathbf{x}[t'], \mathbf{u}[t']), t' = t, \cdots, t+T-1 \tag{3c}$$

$$\mathbf{x}[t'] \in X, t' = t, \cdots, t+T-1 \tag{3d}$$
$$\mathbf{u}[t'] \in U, t' = t, \cdots, t+T-1 \tag{3e}$$

where Eq. (3a) means that we would like the vehicle to converge to the midline of the lane (x_r) with minimal control effect, Eq. (3b) is the DT-CBF constraints ensuring that the vehicle avoids the unsafe regions provided by FDII, n_o^j is the number of the occupied areas of Agent j, Eq. (3c) is the system dynamics as shown in Eq. (2), Eq. (3d) is the admissible set of system state $\mathbf{x}$ (eg. stationary obstacles shown in the default map), and Eq. (3e) is the admissible set of control input $\mathbf{u}$ (eg. the upper bounds and lower bounds of the control inputs).

The optimal solution of Eq. (3) is a sequence of control inputs $\mathbf{u}^*[t], \ldots, \mathbf{u}^*[t+T-1]$. The first element $\mathbf{u}[t] = \mathbf{u}^*[t]$ will be executed. Since $\mathbf{u}[t] = \mathbf{u}^*[t]$ satisfies Eq. (3b), the vehicle will not collide with the obstacle at time step t. Then at time step $t+1$, Eq. (3) will be solved based on the new state $x[t+1]$. If FDII provides updated unsafe regions, they will be incorporated in Eq. (3b). This receding horizon control strategy guarantees collision avoidance between the vehicle and the obstacles.

V. Case Study

In this section, we evaluate our approach in CARLA [9] simulation environment. We first evaluate our proposed approach to FDII and obstacle detection under attacks. We then show that our safe controller ensures that the CARLA vehicle avoids obstacles using the proposed control strategy.

A. Augmented LiDAR FDII

We first introduce the simulation settings. We initialize two vehicles, denoted as Agents A and B, at locations $(-54.34, 137.05)$ and $(-34.34, 137.05)$, respectively. We evaluate our Augmented LiDAR FDII by simulating four scenarios: attack-free, attack NEO, PRA2 and PRA3.

We simulate the attack-free scenario as a reference group. As shown in Fig. 5a, a pedestrian spawns at $(-42.34, 137.05)$, 12 meters in front of Agent A.

In Attack NEO, the attacker injects false data to create a non-existing obstacle. As shown in Fig. 4a, the false data is injected into Agent A's LiDAR observation to falsify a cylinder 8 meters in front of agent A with radius 1 meter.

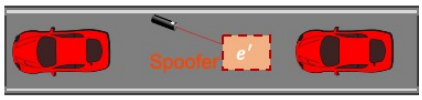
The PRA attacker spoofs LiDAR detection modules to hide obstacles in relay attacked area by injecting false data between victim agent and obstacles. We simulate this attack by replacing true measurement with Gaussian noise. In attack PRA2, we use the same basic setting as the aforementioned attack-free case. In addition, we let the attacker inject false data into Agent A's LiDAR observation to hide the pedestrian in the relay attacked area as shown in Fig. 5b. The injected false data, located 8 meters ahead of Agent A, is Gaussian noise distributed in a cylinder area with 1.3 meters radius.

In the previous PRA2 case, the cylinder area can be partially seen by Agent B. We further validate our approach on a scenario where the cylinder area is blocked by pedestrian. In Attack PRA3 case, we set the cylinder area with 0.4 meter radius, shown in Fig. 5c.

Finally, we present the simulation results and analyze them by comparing with the reference attack-free scenario. We list the corresponding point cloud of joint perception of two agents and the intersection of the occupied area in the second and third row of Fig. 5, respectively.

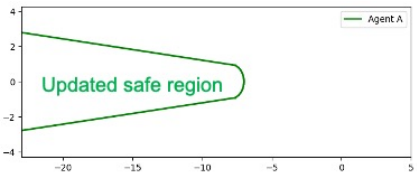
In the attack-free case, agents exchange point cloud data of the intersected area. Both agents can detect the obstacle and generate bounding boxes to contain those points. Agents identify occupied area and generate non-empty sets $\mathcal{U}_A$ and $\mathcal{U}_B$ with the contained points. The proposed FDII module takes these information and detects that there is no attack. By overlapping $\mathcal{U}_A$ and $\mathcal{U}_B$ according to agents' location, FDII further identifies the intersection shown in Fig. 5g as the candidate unsafe region.

In the attack NEO case, a fake obstacle is detected by Agent A annotated in Fig. 4b, which create an erroneous unsafe region. Due to the aforementioned limitation of relay attack, this fake obstacle can only be seen by Agent A. Therefore with the observation provided by Agent B, there is no occupied area identified with the points contained, i.e., $\mathcal{U}_B = \emptyset$. The proposed FDII module detects attack NEO and the obstacle is non-existing. Hence, the unnecessary unsafe region is removed, shown in Fig. 4c.



(a) Attack NEO: An obstacle falsified by spoofer is set in front of Agent A.

(b) Fake obstacle can be detected only by Agent A with $\mathcal{U}_A \neq \emptyset$ and $\mathcal{U}_B = \emptyset$.



(c) The FDII module detects NEO attack. The intersection of the occupied area $\mathcal{U}_A \cap \mathcal{U}_B = \emptyset$, denoting there is no unsafe region.

Fig. 4: FDII simulation settings and results of attack-NEO case

In attack PRA2 case, both attack signal and the pedestrian can be captured by Agent A and B respectively. With point cloud from Agent B, Agent A can observe both obstacles annotated in Fig. 5e and generate their corresponding occupied area with $\mathcal{U}_A \neq \emptyset$ and $\mathcal{U}_B \neq \emptyset$. Since some areas affected by injected false data can be observed by Agent B, we have the $\mathcal{P}(\mathcal{O}_k^b(x_A, S_A)) \nsubseteq \mathcal{U}_B$ and $\mathcal{P}(\mathcal{O}_k^b(x_A, S_A)) \cap \mathcal{U}_B \neq \emptyset$. The proposed FDII algorithm detects attack PRA2. By overlapping

$\mathcal{U}_A$ and $\mathcal{U}_B$ according to agents' location, FDII further identifies the intersection shown in Fig. 5h as the candidate unsafe region.

In attack PRA3 case, the false data affected area is relatively small as shown in Fig. 5f, and hence the area is fully blocked by the pedestrian from Agent B's perspective. In this case, we have the $\mathcal{P}(\mathcal{O}_k^b(x_A, S_A)) \subseteq \mathcal{U}_B$ and $\mathcal{P}(\mathcal{O}_k^b(x_A, S_A)) \cap \mathcal{U}_B \neq \emptyset$. The proposed FDII algorithm detects attack PRA3. By overlapping $\mathcal{U}_A$ and $\mathcal{U}_B$ according to agents' location, FDII further identifies the intersection shown in Fig. 5i as the candidate unsafe region.

B. Safe Control

In this case study, we show that our MPC controller ensures the vehicle to be safe. We consider CARLA vehicle Model-3 as our control object and use (2) as the simplified vehicle model with $l = 4$ and $dt = 0.03$. We further take standard feedback linearization approach to acquire linear model as

$$\begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_{k+1} = \begin{bmatrix} 1 & 0 & 0.03 & 0 \\ 0 & 1 & 0 & 0.03 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_k \tag{4}$$

$$+ \begin{bmatrix} 0.0045 & 0 \\ 0 & 0.0045 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \Delta v_x \\ \Delta v_y \end{bmatrix}_k, \tag{5}$$

in which v_x, v_y are velocity component of x-axis and y-axis, respectively, control input $u = [\Delta v_x, \Delta v_y]^T$ representing the corresponding changes.

We define an MPC controller according to (3) with F, Q, and R set to be identical matrices. We realize our controller with an open-source Python library do-mpc [11], which calls CasADi [1] and IPOPT [23] for nonlinear programming.

We next present the setting of the case. The vehicle is asked to perform reach-and-avoid task starting from location $(-14.34, 137.05)$ to $(-5.00, 135.25)$ without entering the unsafe region. We set the initial state to be $[-14.34, 137.05, 0, 0]^T$ and initial guess to be $[1, 0]^T$. Given the unsafe region detected by FDII module, controller restores the constraints on position $\boldsymbol{s} = [x, y]^T$ as

$$\begin{aligned} h(\boldsymbol{s})_1 &= [-0.35, 0.94]\boldsymbol{s} - 132.74 \\ h(\boldsymbol{s})_2 &= [-0.17, -0.99]\boldsymbol{s} + 132.5 \\ &\dots \\ h(\boldsymbol{s})_{14} &= [0.12, -0.99]\boldsymbol{s} - 134.62. \end{aligned}$$

Finally, we present the trajectory of the CARLA vehicle controlled by MPC in an urban street. As shown in Fig. 6, the vehicle drives from the location $(-14.34, 137.05)$ to $(-5.00, 135.25)$ without entering the unsafe region.

VI. Conclusion

This paper presented an approach for leveraging sensor data from neighboring vehicles to detect LiDAR spoofing attacks on autonomous vehicles. In our approach, vehicles exchange LiDAR scan data and identify spoofing attacks by checking for disparities between the detected obstacles under each scan. We further develop a decision tree to differentiate between non-existing obstacle, physical removal, and adversarial object attacks. We then construct an estimate of the unsafe region based on the joint scan data, and propose a control policy that avoids the unsafe region. We validated our framework using the CARLA simulation platform and showed that it can detect and identify LiDAR attacks as well as guarantee safe driving.

References

[1] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[2] P. Bansal and K. M. Kockelman, "Forecasting Americans' long-term adoption of connected and autonomous vehicle technologies," *Transportation Research Part A: Policy and Practice*, vol. 95, pp. 49–63, 2017.

[3] I. Bogoslavskyi and C. Stachniss, "Efficient online segmentation for sparse 3D laser scans," *PFG–Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, vol. 85, no. 1, pp. 41–52, 2017.

[4] Y. Cao, S. H. Bhupathiraju, P. Naghavi, T. Sugawara, Z. M. Mao, and S. Rampazzi, "You can't see me: physical removal attacks on LiDAR-based autonomous vehicles driving frameworks," *arXiv eprint archive*, 2022.

[5] Y. Cao, N. Wang, C. Xiao, D. Yang, J. Fang, R. Yang, Q. A. Chen, M. Liu, and B. Li, "Invisible for both camera and LiDAR: Security of multi-sensor fusion based perception in autonomous driving under physical-world attacks," in *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2021, pp. 176–194.

[6] Y. Cao, C. Xiao, B. Cyr, Y. Zhou, W. Park, S. Rampazzi, Q. A. Chen, K. Fu, and Z. M. Mao, "Adversarial sensor attack on LiDAR-based perception in autonomous driving," in *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, 2019, pp. 2267–2281.

[7] Y. Cao, C. Xiao, D. Yang, J. Fang, R. Yang, M. Liu, and B. Li, "Adversarial objects against LiDAR-based autonomous driving systems," *arXiv preprint arXiv:1907.05418*, 2019.

[8] D. Davidson, H. Wu, R. Jellinek, V. Singh, and T. Ristenpart, "Controlling UAVs with sensor input spoofing attacks," in *10th USENIX workshop on offensive technologies (WOOT 16)*, 2016.

[9] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

[10] J. Liu and J.-M. Park, ""Seeing is not always believing": Detecting perception error attacks against autonomous vehicles," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 5, pp. 2209–2223, 2021.

[11] S. Lucia, A. Tătulea-Codrean, C. Schoppmeyer, and S. Engell, "Rapid development of modular and sustainable nonlinear model predictive control solutions," *Control Engineering Practice*, vol. 60, pp. 51–62, 2017.

[12] R. Matsumura, T. Sugawara, and K. Sakiyama, "A secure LiDAR with AES-based side-channel fingerprinting," in *2018 Sixth International Symposium on Computing and Networking Workshops (CANDARW)*. IEEE, 2018, pp. 479–482.

[13] P. E. Pace, *Detecting and classifying low probability of intercept radar*. Artech house, 2009.

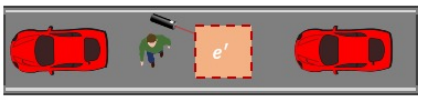
[14] M. Pham and K. Xiong, "A survey on security attacks and defense techniques for connected and autonomous vehicles," *Computers & Security*, vol. 109, p. 102269, 2021.

[15] X. Qian and C. Ye, "NCC-RANSAC: A fast plane extraction method for 3-D range data segmentation," *IEEE transactions on cybernetics*, vol. 44, no. 12, pp. 2771–2783, 2014.

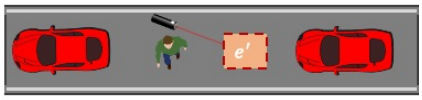
[16] R. Rajamani, *Vehicle dynamics and control*. Springer Science & Business Media, 2011.

[17] S. Ren, S. Chen, and W. Zhang, "Collaborative perception for autonomous driving: Current status and future trend," in *Proceedings of*

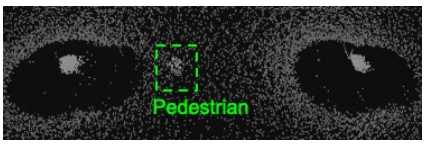
(a) Attack-free: a pedestrian is set between two agents without attack.



(b) Attack PRA2: The attack signal is set between Agent A and the pedestrian.



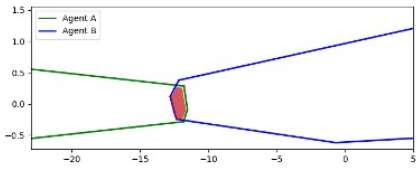
(c) Attack PRA3: The attack signal affects a relatively smaller area.



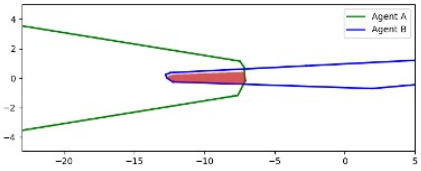
(d) Annotated pedestrian can be detected by both agents with $\mathcal{U}_A \neq \emptyset$ and $\mathcal{U}_B \neq \emptyset$.

(e) Annotated attack signal is captured by Agent A with $\mathcal{U}_A \neq \emptyset$ and $\mathcal{U}_B \neq \emptyset$.

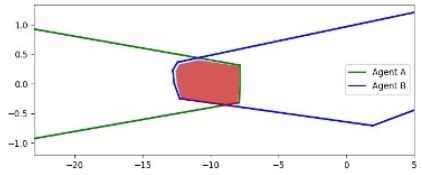
(f) Annotated attack signal is captured by Agent A with $\mathcal{U}_A \neq \emptyset$ and $\mathcal{U}_B \neq \emptyset$.



(g) The FDII module detects no attack. The intersection of the occupied area $\mathcal{U}_A \cap \mathcal{U}_B$ identified as an unsafe region.



(h) The FDII module detects PRA2 attack. The intersection of the occupied area $\mathcal{U}_A \cap \mathcal{U}_B$ identified as an unsafe region.



(i) The FDII module detects PRA3 attack. The intersection of the occupied area $\mathcal{U}_A \cap \mathcal{U}_B$ identified as an unsafe region.

Fig. 5: Augmented LiDAR FDII simulation settings and results: We demonstrate settings in the first row, the corresponding point cloud of joint perception of two agents and the candidate unsafe region in the second and third row, respectively. We list attack-free, PRA2 and PRA3 in the three columns from left to right, respectively.

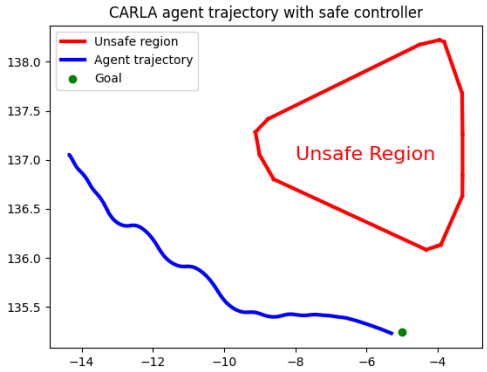


Fig. 6: MPC drove CARLA vehicle from start $(-14.34, 137.05)$ to goal $(-5.00, 135.25)$. The agent managed to avoid detected unsafe region while tracking the given reference point.

2021 5th Chinese Conference on Swarm Intelligence and Cooperative Control. Springer, 2023, pp. 682–692.

[18] H. Shin, D. Kim, Y. Kwon, and Y. Kim, "Illusion and dazzle: Adversarial optical channel exploits against LiDAR for automotive applications," in *International Conference on Cryptographic Hardware and Embedded Systems*. Springer, 2017, pp. 445–467.

[19] D. Suo, J. Moore, M. Boesch, K. Post, and S. E. Sarma, "Location-based schemes for mitigating cyber threats on connected and automated vehicles: A survey and design framework," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[20] G. Thandavarayan, M. Sepulcre, and J. Gozalvez, "Analysis of message generation rules for collective perception in connected and automated driving," in *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019, pp. 134–139.

[21] J. Tu, M. Ren, S. Manivasagam, M. Liang, B. Yang, R. Du, F. Cheng, and R. Urtasun, "Physically realizable adversarial examples for LiDAR object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 13 716–13 725.

[22] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright *et al.*, "Scipy 1.0: fundamental algorithms for scientific computing in python," *Nature methods*, vol. 17, no. 3, pp. 261–272, 2020.

[23] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[24] H. Xiang, R. Xu, X. Xia, Z. Zheng, B. Zhou, and J. Ma, "V2XP-ASG: Generating adversarial scenes for vehicle-to-everything perception," *arXiv preprint arXiv:2209.13679*, 2022.

[25] T. Yang and C. Lv, "A secure sensor fusion framework for connected and automated vehicles under sensor attacks," *IEEE Internet of Things Journal*, 2021.

[26] G. Zamanakos, L. Tsochatzidis, A. Amanatiadis, and I. Pratikakis, "A comprehensive survey of LiDAR-based 3D object detection methods with deep learning for autonomous driving," *Computers & Graphics*, vol. 99, pp. 153–181, 2021.

[27] J. Zeng, B. Zhang, and K. Sreenath, "Safety-critical model predictive control with discrete-time control barrier function," in *2021 American Control Conference (ACC)*. IEEE, 2021, pp. 3882–3889.

[28] H. Zhang, Z. Li, S. Cheng, and A. Clark, "Cooperative perception for safe control of autonomous vehicles under lidar spoofing attacks," 2023.