Adversarial Attacks against LiDAR Semantic Segmentation in Autonomous Driving

Yi Zhu[1], Chenglin Miao[2], Foad Hajiaghajani[1], Mengdi Huai[3], Lu Su[4], Chunming Qiao[1]

[1] State University of New York at Buffalo, Buffalo, NY USA [2] University of Georgia, Athens, GA USA

[3] University of Virginia, Charlottesville, VA USA [4] Purdue University, West Lafayette, IN USA

Email: [1] {yzhu39, foadhaji, qiao}@buffalo.edu, [2] cmiao@uga.edu, [3] mh6ck@virginia.edu, [4] lusu@purdue.edu

ABSTRACT

Today, most autonomous vehicles (AVs) rely on LiDAR (Light Detection and Ranging) perception to acquire accurate information about their immediate surroundings. In LiDAR-based perception systems, semantic segmentation plays a critical role as it can divide LiDAR point clouds into meaningful regions according to human perception and provide AVs with semantic understanding of the driving environments. However, an implicit assumption for existing semantic segmentation models is that they are performed in a reliable and secure environment, which may not be true in practice. In this paper, we investigate adversarial attacks against LiDAR semantic segmentation in autonomous driving. Specifically, we propose a novel adversarial attack framework based on which the attacker can easily fool LiDAR semantic segmentation by placing some simple objects (e.g., cardboard and road signs) at some locations in the physical space. We conduct extensive real-world experiments to evaluate the performance of our proposed attack framework. The experimental results show that our attack can achieve more than 90% success rate in real-world driving environments. To the best of our knowledge, this is the first study on physically realizable adversarial attacks against LiDAR point cloud semantic segmentation with real-world evaluations.

CCS CONCEPTS

• Security and privacy → Domain-specific security and privacy architectures; **• Computer systems organization → Embedded and cyber-physical systems**.

KEYWORDS

Autonomous driving; LiDAR semantic segmentation; adversarial attack

ACM Reference Format:

Yi Zhu, Chenglin Miao, Foad Hajiaghajani, Mengdi Huai, Lu Su, and Chunming Qiao. 2021. Adversarial Attacks against LiDAR Semantic Segmentation in Autonomous Driving. In *The 19th ACM Conference on Embedded Networked Sensor Systems (SenSys'21), November 15–17, 2021, Coimbra, Portugal.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3485730.3485935

1 INTRODUCTION

Recent years have witnessed the rapid development of autonomous driving. Many autonomous vehicles (AVs), also known as self-driving cars, have already been operated on public roads, such as Baidu's Apollo Robotaxi [3] and Waymo's self-driving taxis [4]. For autonomous driving systems, one of the most critical tasks is *perception*, which aims to collect information and extract relevant knowledge from surrounding driving environment using the equipped sensors like cameras, radar, and LiDAR. Among those sensors, LiDAR is particularly attractive because it can collect dense, geo-referenced and accurate 3D point cloud data, which directly provides a precise 3D representation of the driving environment. Currently, the vast majority of self-driving car companies rely on LiDAR to build reliable perception systems for commercial AVs [2, 4, 34].

In LiDAR-based perception systems, a fundamental task is LiDAR point cloud semantic segmentation which intends to achieve semantic scene understanding in autonomous driving. The goal of point cloud semantic segmentation is to divide the point clouds into meaningful regions according to human perception and label each point with a class such as road, vehicle, building, and grass.

Due to its superior capability of providing semantic scene understanding, LiDAR point cloud segmentation has become an key integral part of autonomous driving and enabled many applications. For example, one major application of point cloud segmentation is to identify obstacles on the road including vehicles, pedestrians and motorcycles [16, 17, 20]. Point cloud segmentation can provide accurate information of the obstacles and their properties (e.g., obstacle classes), which is critical for obstacle avoidance. Many state-of-the-art obstacle detection systems [51, 58] such as the LiDAR obstacle perception system in Baidu Apollo [2] adopt point cloud segmentation as a prerequisite step to segment foreground points from background points. Another application of point cloud segmentation is to extract critical information about the boundary of drivable and non-drivable areas such as parking areas and roadside grass [5, 27]. This road boundary information is essential for vehicle path planning [43, 47, 67, 81]. When the AV drives in unmapped areas where high definition (HD) maps are not available, or there are some changes of road environment that are not reported promptly to HD maps [18], it has to rely on semantic segmentation to understand the driving environments. In addition, point cloud semantic segmentation can also be used to reconstruct the 3D environments from sparse point cloud data [12, 20], and model road-side artifacts such as lampposts and traffic signs [26].

Despite the wide deployment of point cloud segmentation models in autonomous driving, an implicit assumption for these models is that they are performed in a reliable and secure environment. However, as semantic segmentation plays an increasingly critical role in AV systems, the risk of malicious attack increases. The state-of-the-art semantic segmentation models mainly use deep neural networks (DNNs) to process LiDAR point clouds. DNNs have been demonstrated to be vulnerable to *adversarial attacks* when taking images as inputs, where an attacker can drastically change the output predictions by introducing a small perturbation to the input pixels [22, 49]. It is entirely possible for an attacker to perform such kind of attacks against the deep learning-based segmentation models by making small changes to the driving environments (e.g., placing some objects on the roadside), which may further fool the LiDAR perception systems of the victim AVs and cause catastrophic consequences. Thus, to well understand the performance of semantic segmentation models in adversarial driving environments, it is essential to investigate adversarial attacks against these models.

In this paper, we conduct the first study on the vulnerability of LiDAR point cloud semantic segmentation. We explore the possibility of performing practical and effective adversarial attacks against LiDAR point cloud segmentation model in real-world driving environments, where the attacker can maliciously change the perception results of the victim AV to his/her desired results, such as changing the vehicle to road or changing the road to vegetation. By attacking LiDAR semantic segmentation, the attacker is able to perform various types of attacks and achieve various attack goals such as collisions, sudden stops or direction changes of the victim AV. Specifically, we propose a novel attack framework based on which the attacker can derive some adversarial locations in the physical space. Through placing some simple objects (also called adversarial objects) such as road signs and cardboard at those locations, the attacker can easily fool the point cloud segmentation model adopted by a victim AV. *Here the adversarial objects could be in any shape as long as it could reflect laser.* In this framework, we take into account several attack challenges in physical world such as large 3D searching space and location errors of adversarial objects. To address the above challenges, the proposed attack framework employs a novel *Semantic Misleading* approach that can guide the search of reasonable locations for adversarial objects in physical world.

In order to evaluate the performance of our proposed attack framework, we conduct extensive experiments on both a real-world LiDAR point cloud segmentation system and a public dataset generated by a commonly used automotive LiDAR. The experimental results demonstrate that our framework not only is general enough to be applied to different types of attack scenarios and segmentation models, but also can achieve high attack success rate in both the digital and physical worlds. To further demonstrate the effectiveness of our proposed attack, we attack several 3D object detection models such as the perception system of Baidu Apollo. To the best of our knowledge, this is *the first study on physically realizable adversarial attacks against LiDAR point cloud semantic segmentation with real-world evaluations.*

2 RELATED WORK

2.1 Vehicular System Security

There are extensive prior works that investigate the security issues of vehicular systems [7, 14, 19, 25, 28, 30, 35, 38, 39, 39, 45, 55, 77]. For AVs, many attack methods have been developed to attack their sensors and perception systems [6, 23, 31, 32, 54, 56, 75, 75]. However, most existing attacks against perception systems in autonomous driving focus on camera-based perception [29, 73].

Although there are a few existing works that study the security issues of LiDAR-based perception, they only focus on the attacks against LiDAR obstacle detection [9, 10, 62, 66, 83]. Different from these works, we aim to attack LiDAR point cloud semantic segmentation, which is a more general and fundamental technique adopted in many autonomous driving applications. By fooling the point cloud segmentation models, our attack can influence not only the obstacle detection, but also many other applications in autonomous driving such as road boundary perception and scene reconstruction. Thus, attacking LiDAR semantic segmentation can bring more security threats to AVs, and has broader impact to the safety of AVs.

In addition, existing attacks against LiDAR obstacle perception are either not feasible enough or not flexible enough when being performed in physical world. The authors in [9, 62] propose to spoof the LiDAR sensor through strategically transmitting laser signals to the victim vehicle's LiDAR sensor. However, it may be difficult to perform the proposed attacks in real world because the victim AV is usually moving and they require the attacker to aim at the LiDAR sensor with high precision in a dynamic manner. Besides, some special equipment is needed to generate the laser signals, which makes it less flexible to perform this kind of attacks. The authors in [10, 66] generate some adversarial objects that are placed on the top of the target vehicle or on the road. However, these objects are required to have specific shapes, which limits the flexibility of the attack. Besides, such adversarial objects are suspicious to human eyes because of their abnormal shapes. And the scanned point clouds generated by the adversarial objects may not always be in the desired shapes because of LiDAR scanning errors, manufacture errors, and location errors. This inaccuracy of point cloud shape may fail the attack, and generating such specifically shaped objects with high precision is challenging in practice.

Unlike these works, our proposed attack can be easily performed by placing some objects around some specific locations, and these objects can be in any shapes, which makes the attack more flexible. Since the attacker does not need to care about the shape of the adopted objects, he can use some common objects such as traffic signs and advertisement board to make the attack more stealthy. Although [83] also uses arbitrary objects such as drones to attack AVs, it only considers object detection and aim to hide a vehicle from the object detection model. In this paper, we aim to attack a more fundamental task, LiDAR semantic segmentation, which can influence the perception of both vehicles and other structures such as road, vegetation, and buildings. Besides, these existing attack methods are not general enough to be applied to LiDAR semantic segmentation because they can only hide existing vehicles or creating fake vehicles and fail to change the perception results of other structures such as road, vegetation, and buildings.

2.2 Adversarial Attacks against Semantic Image Segmentation

There are some existing works that generate adversarial examples to fool image-based semantic segmentation models. These methods propose to make small perturbations on each pixel values in an image [24, 71]. However, they only focus on 2D adversarial attack and their methods cannot be directly applied to 3D adversarial attack on LiDAR point cloud.

2.3 3D Adversarial Attacks

3D adversarial attacks have drawn much attention recently. Specifically, point cloud classification models are proved to be vulnerable to 3D adversarial attacks [68, 70, 80]. Point cloud classification aims to label the whole point cloud with a class such as lamp or desk. In this paper, we focus on a different task, i.e., LiDAR point cloud segmentation, whose goal is to label each point in the point cloud with a class. In addition, although the attack methods designed in aforementioned point cloud classification work can achieve the attack goal in digital world, the generated adversarial examples are not always physically realizable. In contrast, we investigate practical and effective adversarial attacks that can be performed in real-world driving scenarios.

3 BACKGROUND

3.1 Point Cloud Segmentation in Autonomous Driving

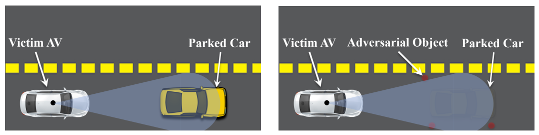
The goal of point cloud semantic segmentation is to divide the point clouds into meaningful regions according to human perception. It labels each point with a class such as ground, vehicle, trees, and so on. The task can be described as: given a set of points $\{x_1, x_2, ..., x_n\}$ and candidate labels $\{y_1, y_2, ..., y_k\}$, assign each point x_i with one of the labels y_j.

Deep learning techniques have made tremendous progress in point cloud semantic segmentation. However, traditional CNN-based models are not suitable to deal with irregular point cloud inputs. To address this challenge, many existing approaches transform the irregular point clouds into regular representations such as 3D voxel grids [78] or projections [69] before feeding them to convolution networks. PointNet [52] is a pioneering network architecture for segmentation that takes raw point clouds as input. The network applies a set of transformation subnetworks and Multi Layer Perceptrons (MLPs) to generate point-wise features. The point-wise features are then aggregated to global features using Max-Pooling layers. Then it concatenates the global features with the point-wise features and generates output logits for each point. Various point cloud segmentation models are proposed based on PointNet, such as PointNet++ [53] and PointASNL [74]. Besides, SqueezeSeg [69] is also a state-of-the-art point cloud segmentation model, which projects point cloud into a spherical image.

3.2 Adversarial Attacks

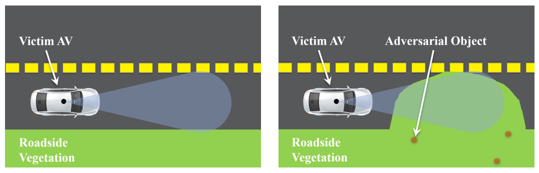
Deep learning techniques benefit a wide spectrum of applications. However, they have been proved to be vulnerable to adversarial attacks [13, 36, 37, 44, 50, 57], especially some specially crafted adversarial examples [15, 42, 46, 59, 63, 72, 79]. The adversarial examples generated by adding some small perturbations on the original input examples [22, 49], can fool the deep learning models with high confidence.

For a deep learning model M, original input data x and its corresponding label y, the adversarial attack aim to find an adversarial



(a) Before attack **(b) After attack**

Figure 1: Vehicle/obstacle hiding attack. By placing some adversarial objects around some locations, the attacker can hide the parked car or other obstacles from the victim AV.



(a) Before attack **(b) After attack**

Figure 2: Road surface changing attack. By placing some adversarial objects around some locations, the attacker can change the road surface to roadside vegetation.

example x' which appears similar to x so that $M(x') \neq y$. The adversarial example x' is usually generated by solving an optimization problem with the objective loss function that measures both attack effectiveness and perturbation magnitude [11, 64, 76].

4 ATTACK SCENARIO AND THREAT MODEL

4.1 Attack Scenario

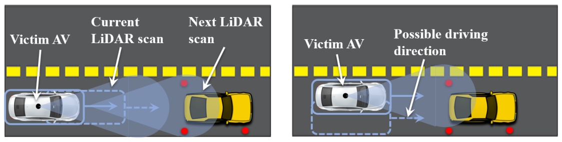
In this paper, we consider the AVs that make use of 3D point cloud segmentation models to understand the driving environments. The attack goal is to change the perception results from the segmentation models. Considering the common applications of point cloud segmentation, we focus on two types of attack scenarios: vehicle/obstacle hiding attack and road surface changing attack.

4.1.1 Vehicle/obstacle Hiding Attack. In this attack, we consider a driving environment where there is a car parking on the road/parking lot. The parked car could be parked by an attacker intentionally or it could be a car stopped at the traffic light. As shown in Figure 1, the attacker aims to hide the parked car from the LiDAR-based perception system of a coming victim AV by adding some adversarial objects around the car or on the roadside. This kind of attacks may result in a rear-end collision and cause catastrophic consequences.

4.1.2 Road Surface Changing Attack. In road surface changing attack, we consider a driving environment where the victim AV is driving on the road as shown in Figure 2. The goal of the attack is to change the road surface in front of the victim AV to other things such as vegetation (e.g., grass) by adding some adversarial objects. These adversarial objects could be located on the roadside or sidewalk. This kind of attack may result in a sudden stop or driving direction changing, and further cause traffic accidents.

4.2 Threat Model

We consider an attack in which the attacker has the ability to place some adversarial objects around some specific locations in the targeted driving environments. To achieve the attack goal, the attacker could use some *simple objects* to perform the attack. The



(a) Consecutive LiDAR frames (b) Different directions

Figure 3: The victim AV may come from different directions and collect a series of consecutive LiDAR frames.

adversarial objects could be in any shape as long as they could reflect LiDAR laser. The number of those objects is pre-determined by the attacker.

In addition, we consider both white-box attack and black-box attack. For the white-box attack, we assume that the attacker has full access to the machine learning model and the perception system of the victim AV. Since it is entirely possible for the attacker to have a AV with the same model, he could obtain the white-box access by using reverse engineering techniques. For the black-box attack, the attacker does not have access to the machine learning model used by the perception system. He could only query the model and get the outputs, which is the same assumption made by most existing black-box attack methods [33, 40, 41, 48, 63]. Besides, for both types of attacks, we assume that the attacker could collect sensory data to generate 3D point clouds in different driving environments using a similar LiDAR sensor as that adopted by the victim AV.

5 ATTACK OVERVIEW

5.1 Attack Challenges

In practice, the victim AV is usually moving when the attacker performs the attacks. As shown in Figure 3a, the victim AV can collect a series of consecutive frames of point cloud data as it moves forward and generate perception result for each frame. Attacking a single frame is not enough to ensure a successful attack goal. Even the attacker changes the perception result for one frame, the AV system might regard it as a system error and ignore it. To achieve the attack goal, the attacker needs to change the perception results for all consecutive frames. In addition, the attacker usually cannot predict the driving behavior of the victim AV before performing the attacks and it may come from different directions as shown in Figure 3b. So the added adversarial objects should be able to fool the victim AV no matter which direction it comes from.

Besides, generating robust 3D adversarial examples for point cloud is more difficult than generating 2D adversarial examples for images [70]. The searching space in our problem is very large and the adversarial objects can be anywhere in the space, which bring new challenges to find the optimal solutions in 3D space. Also, the consecutive frames and unpredictable driving behavior of the victim AV bring extra challenges. Existing adversarial attacks on point cloud data fail to address this problem.

Furthermore, when performing the attacks in physical world, it is usually difficult to exactly place the adversarial objects at the derived locations. The location errors may affect the point cloud scanned by LiDAR and further affect the attack results. Thus, the attack method should also be robust to location errors of the adversarial objects. Besides, the derived locations for adversarial objects should be reasonable. It should be easy for the attacker to place objects at these locations. For example, these locations should not be underground or too high above the ground. In addition, they should not be occluded by other objects. Otherwise, they can not be observed by the LiDAR sensor.

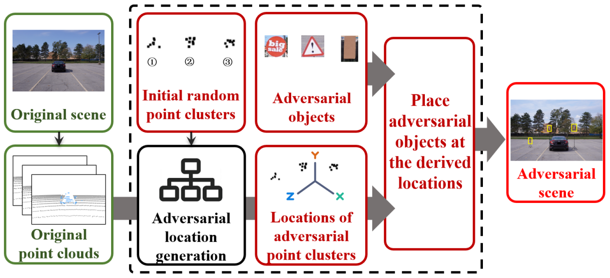

Original scene
Original point clouds
Initial random point clusters
Adversarial objects
Adversarial location generation
Locations of adversarial point clusters
Place adversarial objects at the derived locations
Adversarial scene

Figure 4: Attack framework.

5.2 Attack Framework

To address the above challenges, we propose a general attack framework based on which the attacker can fool the segmentation models and make the victim AV misunderstand the driving environments. Figure 4 shows an overview of our proposed attack framework.

Before performing the attacks, the attacker first imitates the possible driving behavior of the victim AV (e.g., driving directions) and collects corresponding 3D point cloud data in the targeted driving environments (as shown in Figure 1 and Figure 2). The collected point clouds are called *original point clouds*, each frame of which describes a possible scene observed by the victim AV before the attack. By imitating the possible driving behavior of the victim AV, the attacker could collect all possible point cloud data that will be collected by the victim AV. Based on these data, the attacker could derive how to add the adversarial objects so that the victim AV can be fooled in all consecutive frames no matter which direction it comes from.

After collecting the original point cloud data, the next step for the attacker is to derive the locations of the adversarial objects (also called adversarial locations) in physical world. Specifically, we represent each object as a *random point cluster* in the point clouds. The shape and the number of points for each point cluster are both random, which allows the attacker to use arbitrary objects to perform the attacks. The injected point clusters are kept random during the whole process (randomly generated in each iteration). The intuition behind this representation is that each adversarial object creates a point cluster in LiDAR scan. If we find the locations at which placing random adversarial clusters achieves the attack goal, placing any adversarial objects at these locations could also achieve the attack goal.

These point clusters are first initialized and added to the original point clouds. Then the attacker derives their optimal locations (the locations of cluster centers) based on which the adopted point cloud segmentation model can be fooled. In this paper, the process of deriving these optimal locations is called *adversarial location generation*. Please note that all the above steps can be conducted offline. After deriving the locations of adversarial point clusters, the attacker could find out the corresponding locations in physical world and place adversarial objects at these locations to perform the attacks when the victim AV is approaching.

However, the generation of adversarial point clusters (or derivation of their optimal locations) is not easy since we need to address

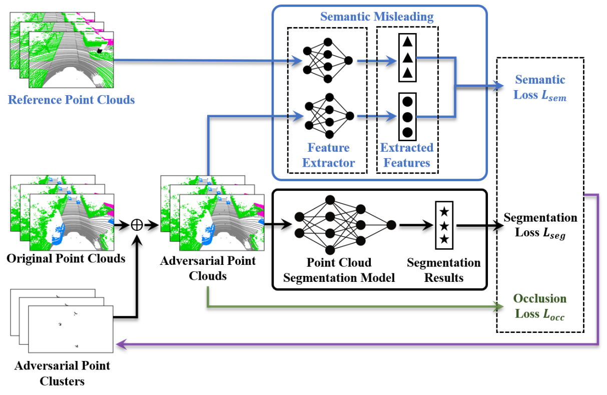


Figure 5: Overview of adversarial location generation.

many challenges as mentioned in Section 5.1. To address these challenges, we propose a novel modular based on semantic misleading approach to generate the adversarial point clusters' locations.

6 ADVERSARIAL LOCATION GENERATION

Figure 5 shows the overview of our proposed adversarial location generation modular. Given the input of original point clouds, we combine them with point clusters to get the *adversarial point clouds*, which are fed into the *point cloud segmentation model* to generate segmentation results. The point cloud segmentation model is adopted by the victim AV and can not be changed during the attack process. Recall that the attack goal is to let the segmentation model misclassify the target points (e.g., the points belonging to the parked car and the adversarial clusters) as the target label (e.g., ground). Here we introduce a *segmentation loss* L_{seg} to measure the distance between the predicted label of target points and the target label. Minimizing the total segmentation loss for *all the original point clouds* can help us derive the locations of adversarial point clusters that work in all possible scenes observed by the victim AV.

However, as described in Section 5.1, our attack scenario is much more complicated compared with existing attacks on point cloud data. Only minimizing the segmentation loss may not be enough to derive the optimal locations of adversarial point clusters. To address this challenge, we propose a *semantic misleading* method to guide the finding of the optimal locations. The basic idea of this method is to make the semantic feature of adversarial point clouds similar to that of the *reference point clouds*. The reference point clouds describe the scenes that the attacker desired: for vehicle hiding attack, the scene has no car on the road; for road surface changing attack, the scene has lots of vegetation in front of the LiDAR sensor. The reference point clouds can be easily collected by the attacker or obtained from public 3D point cloud datasets (e.g., the SemanticKITTI dataset [5]). We use the same feature extractor to extract the semantic features of the adversarial point clouds and the reference point clouds. The *semantic loss* L_{sem} is used to measure the similarity between the extracted features of the two types of point clouds.

In addition, we introduce an *occlusion loss* L_{occ} and some constrains to guarantee the derived locations are reasonable so that the attacker could easily add adversarial objects in the physical space. Through optimizing the total loss $L_t = L_{seg} + \alpha L_{sem} + \beta L_{occ}$, we can generate adversarial locations based on which the attacker can fool the victim AV. Here α and β are the pre-defined hyper-parameters to balance the three losses. Besides minimizing the total loss L_t, we also propose to minimize its gradient L_t' so that the attack can be robust to location errors. The reason why we also minimize L_t' is that the gradient of a loss function reflects how the loss changes with respect to small perturbations on the inputs (in our problem, the inputs are the locations of adversarial point clusters). By minimizing the gradient, we could find the optimal values that can tolerate some small perturbations, which helps to generate adversarial clusters that are robust to location errors. The details of the modular will be elaborated in the rest of this section.

6.1 Segmentation Loss

Suppose X^* denotes a frame of the original point clouds. The number of possible classes (e.g., vehicle, ground, and vegetation) for the points is C. We use $X = \{x_n\}_{n=1}^{N}$ to denote the N target points in a frame that need to be misclassified by the segmentation model, where $x_n \in \mathbb{R}^3$. The ground truth labels of these N points are denoted as $L = (l_1, ..., l_N)$. $f_c(X^*, x_n)$ is the probability that point x_n belongs to class $c \in [C]$ (i.e., the c-th logit of point x_n before SoftMax layer), given the input of original point cloud X^*. $X^a = \{x_k^a\}_{k=1}^{K}$ are K adversarial points clusters, and the value of K is predefined by the attacker. Each adversarial point cluster is kept random (re-generated with random number of points and random shape) during the whole process. To make each random point cluster feasible and reasonable, the maximum number of points in each point cluster is limited to 10 (i.e., $|x_k^a| \leq 10$), and the maximum size of each point cluster is limited to $0.3m$ (i.e., $\max_{a,b \in x_k^a} ||a-b||^2 \leq 0.3m$).
The goal of the attacker is to make the predicted label of each target point x_n become the target label l_n' by adding adversarial point clusters to the original point clouds, which can be described as:

$$\forall x_n \in X, \ \arg\max_{c} f_c(X^* \cup X^a, x_n) = l_n'. \tag{1}$$

In vehicle hiding attack, the target points are those belonging to the parked car (labeled as "Vehicle"). The target label is "Ground". For road surface changing attack, the target points are the points belonging to the road in front of the victim AV and target label is "Vegetation". To achieve the attack goal, we need to derive the locations of these adversarial point clusters.

In this paper, a point cluster's location is represented by its center's location. Since the LiDAR coordinate changes when the victim AV is moving, it is difficult to directly optimize the *xyz*-coordinates of the adversarial point clusters in the LiDAR coordinate. To address this problem, we select a fixed object such as the parked car or a building as the reference. Then we calculate the relative locations of adversarial point clusters to a particular part on the fixed object (e.g.,the back of the parked car). Specifically, we use $O_k^a = (x_{k1}^a, x_{k2}^a, x_{k3}^a)$ to denote the relative location of cluster x_k^a.

As mentioned in [11], directly solving the Eq.(1) is difficult. A common approach for generating adversarial examples in point cloud and image classification problems is to maximize the output confidence of the target label. Since point cloud semantic segmentation is to classify each individual point, an intuitive approach to misclassify an object is to maximize the summation of the confidences of the target label (e.g., the ground) for all the points belonging to this object (e.g., the parked car). However, this intuitive approach is not suitable to our problem. For the adversarial attacks against point cloud semantic segmentation model, we aim to maximize the number of the misclassified points so that we can "hide" the parked car or "change" the road surface as much as possible. But maximizing the summation of confidences can not guarantee that the number of misclassified points is maximized. For example, the confidence of one point could be very large but the others' are still small, which means only one point is misclassified. To address this problem, we use the following segmentation loss:

$$L_{seg} = -\sum_{n=1}^{N} \min(0, [f_{l'_n}(X', x_n) - \max_{l \neq l'_n} f_l(X', x_n)]), \tag{2}$$

where $X' = X^* \cup X^a$ denotes the adversarial point cloud after adding K point clusters, and l'_n is the target label.

The intuition of Eq.(2) is that the loss of the misclassified points will stay zero and if a target point is not misclassified, its loss will be positive (the smaller the confidence is, the larger the value is). Minimizing the segmentation loss can help to maximize the number of misclassified points.

6.2 Semantic Misleading

As described in Section 5.1, our attack scenario is very challenging as we consider not only 3D searching space but also the consecutive frames of point clouds and unpredictable driving behavior of the victim AV. Only optimizing the segmentation loss may not be enough to derive the optimal locations for adversarial point clusters. To address this problem, we propose a semantic misleading method that can guide the finding of the optimal locations.

In this method, we aim to change the segmentation model's understanding of the whole scene. Specifically, we want the neural network misunderstand the current point cloud scene as a scene without the parked car (for vehicle hiding attack) or with vegetation (for road surface changing attack) in front of the victim AV. We believe that by changing the network's understanding of the scenes, we could change the segmentation results of the model. To achieve the goal, we introduce some specific point cloud examples (called *reference point clouds*) to guide the segmentation model's understanding of the whole scene.

The reference point clouds describe the scenes that the attacker desired. For vehicle hiding attack, reference point clouds do not contain points that are labeled as "Vehicle" (i.e., there is no car parked on the road in these examples). For road surface changing attack, reference point clouds contain lots of "Vegetation" points in front of the LiDAR sensor. This kind of point cloud examples can be easily collected by the attacker or obtained from public datasets. In point cloud segmentation models, each point's segmentation result is determined by the local (lower level) and global (higher level) features of the point cloud [52, 53, 69, 74]. The global features capture the large-scale geometric structures of the point cloud and the local features capture the small-scale structures. Ideally, if we could make the local and global features of the adversarial point cloud become the same as that of the reference point cloud, the network's segmentation result of the adversarial point cloud would be the same as that of the reference point cloud, which could obviously achieve the attack goal. However, precisely manipulating all those features together through adversarial attack is difficult because of the large-scale input point cloud. Instead, we only focus on manipulating the global (higher level) features. We introduce a semantic feature extractor to extract the global features of the input point cloud. The extracted global features represent the network's semantic understanding of the point cloud scene and determines the segmentation result of each point in the point cloud. By making the adversarial point cloud X''s global features as close to the global features of the reference point clouds as possible, the segmentation result of the adversarial point cloud is guided to be similar to that of the reference point cloud. In other words, the segmentation network is misled to perceive the adversarial point cloud as the reference point cloud (the scene we desired to achieve the attack goal).

Here we introduce a semantic loss L_{sem} to measure the similarity between the extracted features for the two kinds of point clouds. We denote the reference point clouds as $P = \{X^m\}_{m=1}^{M}$, where X^m is the m-th reference point cloud example. Then we have the following semantic loss:

$$L_{sem} = \min_{m \in P} ||F(X') - F(X^m)||^2, \tag{3}$$

where $F(\cdot)$ is the semantic feature extractor. Minimizing the semantic loss means making the current scene's semantics close to the semantics of the reference point clouds, and this can guide the finding of the optimal locations for adversarial point clusters with strong generalization ability.

6.3 Occlusion of Adversarial Points

Based on the segmentation loss and semantic loss, we can generate some adversarial point clusters that minimize the two losses. However, the locations for these adversarial point clusters may be blocked by other objects in physical world and the deployed adversarial objects cannot be observed by the LiDAR. To address this problem, we propose an occlusion loss L_{occ} to make sure the added objects are not blocked by any other objects. As shown in Figure 6, we draw a frustum from the origin (LiDAR) to include an adversarial cluster. If the adversarial objects are blocked by other points, the points belonging to these objects would be further away from the origin than other points in the frustum. Thus, for each adversarial cluster x_k^a, we can define a loss L_{occ}^k:

$$L_{occ}^k = \epsilon \cdot \max((\max(D_{obj}) - \min(D_{other})), 0), \tag{4}$$

where $\max(D_{obj})$ is the maximum distance between the adversarial points in the frustum and the origin, and $\min(D_{other})$ is the minimum distance between other points in the frustum and the origin. ϵ is a pre-defined parameter (in this paper, we set ϵ as 10,000). If the adversarial point cluster is blocked by other points, the value of L_{occ}^k is very large, otherwise it is zero. The final occlusion loss is the sum of occlusion losses of all the point clusters: $L_{occ} = \sum_{k=1}^{K} L_{occ}^k$.

6.4 White-box Attack

In this paper, we consider both white-box and black-box attacks. For white-box attack, the attackers have full knowledge of the target semantic segmentation model and the LiDAR based perception system of the victim AV. Here there is no need to train a separate semantic feature extractor for the semantic misleading method. We use a certain feature layer of the segmentation model as the

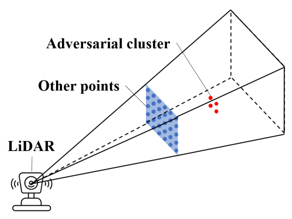


Figure 6: The adversarial point cluster is blocked by other points

feature extractor, because the feature layer of this model contains the learned global features of the input point cloud scenes and it can well represent the network's understanding of the point cloud scenes.

In addition, the derived locations of the adversarial point clusters should be reasonable. These locations should not be too high above the ground or underground so that the attacker can easily deploy the adversarial objects in physical world. To achieve the goal, we constrain each point cluster to a bounding box. Then, we can derive the locations of adversarial point clusters by solving the following optimization problem:

$$\begin{aligned} \min_{\{O_k^a\}_{k=1}^K} \quad & L^* = L_t + \eta L_t' \\ \text{s.t.} \quad & \{x_{k1}^a\}_{k=1}^K \in [A_{min}, A_{max}], \\ & \{x_{k2}^a\}_{k=1}^K \in [B_{min}, B_{max}], \\ & \{x_{k3}^a\}_{k=1}^K \in [C_{min}, C_{max}], \end{aligned} \tag{5}$$

where $L_t = L_{seg} + \alpha L_{sem} + \beta L_{occ}$, and L_t' is the gradient of L_t. η is a pre-defined hyperparameter. $[A_{min}, A_{max}]$, $[B_{min}, B_{max}]$ and $[C_{min}, C_{max}]$ are the bounds for point clusters' locations. As described before, we optimize the total loss as well as the gradients of total loss for all the original point clouds to generate robust optimal results.

To solve the optimization problem in Eq. (5), we use $\tanh(\cdot)$ to constrain the relative locations of adversarial point clusters. Specifically, the location $(x_{k1}^a, x_{k2}^a, x_{k3}^a)$ for each point cluster x_k^a can be represented as follows:

$$\begin{aligned} x_{k1}^a &= (A_{max} - A_{min})/2 \cdot \tanh(p_{k1}) + (A_{max} + A_{min})/2, \\ x_{k2}^a &= (B_{max} - B_{min})/2 \cdot \tanh(p_{k2}) + (B_{max} + B_{min})/2, \\ x_{k3}^a &= (C_{max} - C_{min})/2 \cdot \tanh(p_{k3}) + (C_{max} + C_{min})/2. \end{aligned} \tag{6}$$

Here we use Adam Optimizer to find the optimized value of (p_{k1}, p_{k2}, p_{k3}).

6.5 Black-box Attack

For the black-box attack, the attacker does not have access to the technical details of the point cloud segmentation model and the perception system adopted by the victim AV. But he could obtain the same type of AV and get the output of the segmentation model.

Since the segmentation model is black-box, we cannot use a certain layer of this model as the semantic feature extractor used in semantic misleading method. Here we use the point cloud saliency map [80] to extract the semantic features. Specifically, we generate the point cloud saliency map for each point cloud using the method in [80] and select top 100 points with high saliency scores. Since the global feature of point cloud is dominated by critical points [52, 80], we can use the xyz-coordinates of the selected 100 points as the semantic features. A challenge to generate the point cloud salience map is that it requires gradients of the network's loss function with respect to points' locations, which are not available in the black-box setting. To address this challenge, we calculate the approximation of gradients by shifting the points in a small distance and calculate the changes of loss.

In addition, instead of using Eq.(3), we adopt Hausdorff Distance to calculate the semantic loss because of its good performance in measuring the similarity of two point clouds. Specifically, we define the semantic loss as:

$$L_{sem} = \min_{m \in P} \max_{a \in X_c'} \min_{b \in X_c^m} d(a, b), \tag{7}$$

where X_c' is the set of critical points of adversarial point cloud $X^* \cup X^a$, X_c^m is the set of critical points of the m-th example in reference point clouds, $d(a, b)$ is the Euclidean distance between points a and b. Then the locations for the adversarial objects are derived by solving the optimization problem with the objective of minimizing the loss $L^* = L_t + \eta L_t'$, where $L_t = L_{seg} + \alpha L_{sem} + \beta L_{occ}$. Also, we calculate the approximation of gradients L_t' by making small perturbations on the inputs and calculating the changes of total loss.

Since we do not have the network's parameters, we cannot use gradient-based optimizer to solve the optimization problem. Here we use an evolution-based algorithm to solve this problem. Specifically, we use Differential evolution (DE) algorithm [60, 61] to optimize the loss function instead of Adam optimizer.

7 EXPERIMENTS

7.1 Experimental Setting

7.1.1 Platform and Dataset. To train the point cloud segmentation model, we use both public point cloud dataset and our own data collected using an automotive LiDAR sensor. Specifically, we choose the SemanticKITTI dataset [5], which is a public dataset for sequential point clouds generated with a commonly used automotive LiDAR. We use the Sequences 00-09 as the training data. In addition, we collect our own point cloud data using an Ouster OS1-64 LiDAR. This LiDAR is mounted on the top of the vehicle, as shown in Figure 7. With a precision of 1.5-5 cm, the sensing range of the adopted LiDAR is about 120m and its vertical field of view is 45 degree. We used a tripod with suction cups to mount and level the LiDAR on top of a sedan vehicle. The height of LiDAR from ground surface is adjusted to be 1.8m. We collect data in three different parking lots and on two different campus roads. We manually labelled the collected data using the tools provided by SemanticKITTI.

7.1.2 Segmentation Models. We use five state-of-the-art point cloud segmentation models to evaluate our attack framework, including PointNet [52], SqueezeSeg [69], Cylinder3D [82], PointNet++ [53], and PointASNL [74].

PointNet has been widely used in LiDAR-based perception systems such as PointPillars[34] integrated in Autoware [1]. SqueezeSeg projects the point clouds into an image using a spherical projection and use CNNs to learn the features. Cylinder3D utilizes 3D cylinder partition and 3D cylinder convolution to learn the

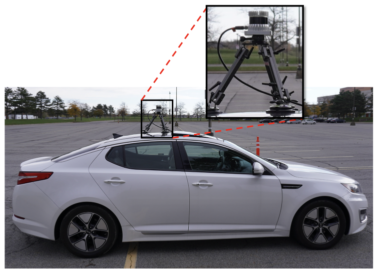


Figure 7: The Ouster OS1-64 LiDAR mounted on a vehicle.

features. It achieves top-2 performance in the leaderboard of SemanticKITTI dataset. PointNet++ applies PointNet in a hierarchical structure, which captures both local features as well as global features. PointASNL introduces adaptive sampling (AS) module to benefit the feature learning of point clouds. For PointNet and PointNet++, we follow the preprocessing steps that are widely adopted: we relabel the points into 7 classes, including Ground, Vehicles, Bicycles/motorcycles, Pedestrians, Vegetation, Buildings, and Others; we divide the whole point cloud scene as 15×15m blocks and train/apply PointNet on each block. The segmentation results of the whole scene are aggregated from all the blocks. For SqueezeSeg, Cylinder3D and PointASNL, we follows the official steps to preprocess the LiDAR data.

7.2 Overall Performances

7.2.1 Evaluation on SemanticKITTI Dataset. To validate the effectiveness of our proposed attack, we perform black-box attacks on the SemanticKITTI dataset. Specifically, we randomly choose 20 examples (or scenes) for each attack scenario from Sequences 10 in SemanticKITTI dataset and each example contains 5 consecutive point cloud frames. We use *random point clusters* to represent the adversarial objects. The number of adversarial point clusters for each example is predefined as 5. The adversarial point clusters are constrain within a 45m*45m block with the height of 2m in front of the LiDAR. To make the adversarial object feasible and reasonable, the maximum size of each cluster is limited to 30cm. To find the adversarial locations, we set the hyper-parameter α to 0.1, β to 1, and η to 0.1. For white-box attack, the learning rate of Adam Optimizer is set to 0.1. For black-box attack, the population size of differential evolution is set to 180. After deriving the locations of these random adversarial point clusters, we replace each random cluster with a new random cluster at the same location. This replacement is repeated for 100 times and the average results are calculated. Here we use the attack success rate as performance measurement, which is defined as the percentage of the points that are classified as our target label in all target points.

Table 1 reports the average success rate for both vehicle hiding attack and road surface changing attack. In this experiment, we consider different point cloud segmentation models. The results in Table 1 show that our proposed attack framework can achieve good performance for both the two types of attacks. When PointNet is applied as the segmentation model, our attack framework can achieve the best performance and the average attack success rate is 82%. PointASNL is more robust than the others because it can capture both neighbor and long range dependencies of the sampled points, which benefits the robust feature learning [74]. However,

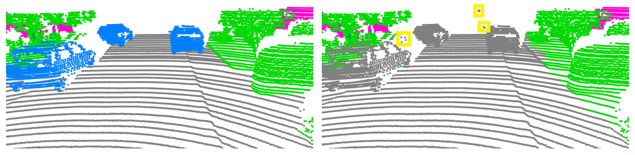


Figure 8: Attacking real traffic scenarios from SemanticKITTI dataset. Blue points belong to the "Vehicle" class, grey points belong to the "Ground" class, and green points belong to the "Vegetation" class.

the attack success rate for PointASNL is 62%, which can still bring security threats to the victim AV. The results also show that the proposed framework can attack models with different architectures.

Table 1: The average attack success rate on SemanticKITTI

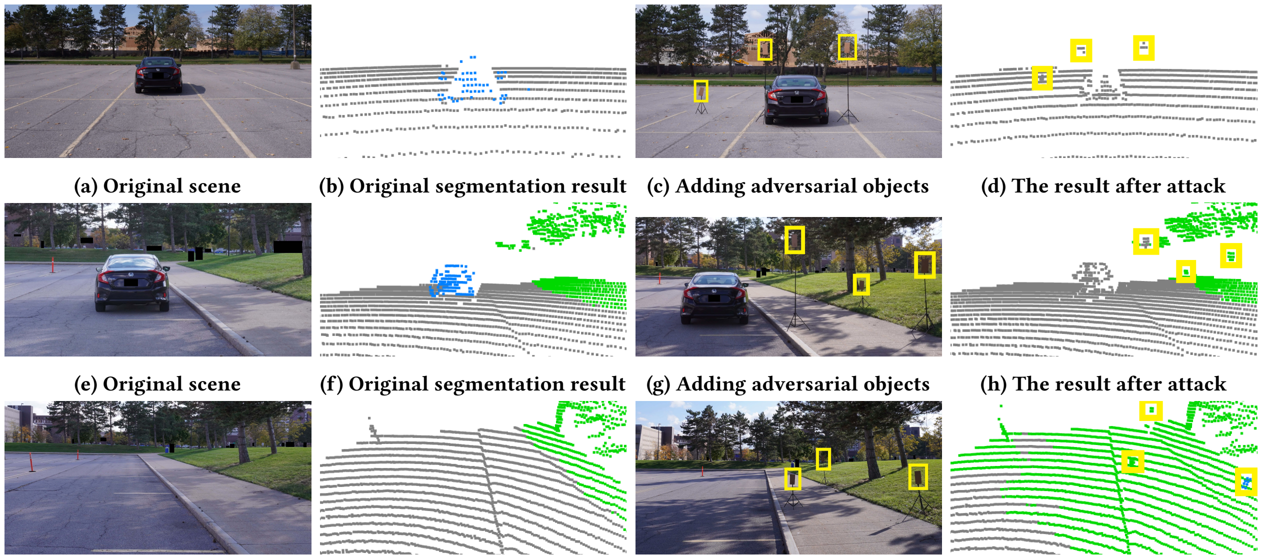
Models	Vehicle Hiding	Road Surface Changing
PointNet	0.82	0.78
SqueezeSeg	0.77	0.66
Cylinder3D	0.72	0.63
PointNet++	0.69	0.60
PointASNL	0.62	0.58

In Figure 8, we report the attack result for a real traffic scenario from SemanticKITTI dataset. This scenario contains multiple cars on the road. The adversarial point clusters are highlighted with yellow rectangles. Here we use PointNet as the segmentation model. We can see that the cars are successfully "hidden" by placing three adversarial point clusters, which demonstrates that the proposed attack can successfully mislead the point cloud segmentation model under realistic traffic scenarios.

7.2.2 Real-world Evaluation. The ultimate goal of the attacker is to perform the adversarial attacks in physical world. Next, we will evaluate the performance of our attack framework by conducting real-world experiments. Here we still consider the black-box attack and use PointNet as the point cloud segmentation model.

Vehicle hiding. As shown in Figures 9a and 9e, we consider two real-world scenes. For each scene, we parked a black car in the parking lot. Our goal is to hide this car by performing the attack based on our proposed framework. To achieve the goal, we first collect the original point cloud data using the white car shown in Figure 7. Then we set the number of adversarial point clusters as 3 and derive 3 adversarial locations. For the scene shown in Figure 9e, we constrain the locations of adversarial objects on the sidewalk.

Figures 9b and 9f show two examples of the original segmentation results for the two scenes before the attack. The distances between the victim AV and parked car in both examples are 15 meters. We can see the parked cars (blue points) are correctly recognized by the LiDAR based perception system. In this experiment, we use cardboard to generate the adversarial point clusters. These pieces of cardboard are taken as adversarial objects and they are placed at the derived locations, as shown in Figures 9c and 9g. The size of each piece of cardboard is limited to 30cm, which is consistent with the size of the adversarial cluster. Here we use a poster stand to hold each piece of cardboard. Figures 9d and 9h show the segmentation results for the two scenes after the attack.



(a) Original scene **(b) Original segmentation result** **(c) Adding adversarial objects** **(d) The result after attack**

(e) Original scene **(f) Original segmentation result** **(g) Adding adversarial objects** **(h) The result after attack**

(i) Original scene **(j) Original segmentation result** **(k) Adding adversarial objects** **(l) The result after attack**

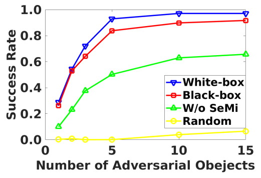
Figure 9: The visualizations of real-world adversarial examples. Blue points belong to the "Vehicle" class, grey points belong to the "Ground" class, and green points belong to the "Vegetation" class. The adversarial objects are highlighted with yellow rectangles.

In these two figures, the generated adversarial point clusters are highlighted with yellow rectangles. We can see that each piece of cardboard creates a point cluster at the given location. The black cars in the two scenes are successfully "hidden" by those point clusters. These results demonstrate that our proposed attack can be easily performed in real world. In addition, as shown in Figure 9g, these adversarial objects can be placed on the roadside which are less suspicious and easy to implement.

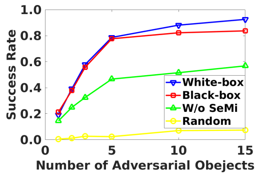
Road surface changing. As shown in Figure 9i, we choose an empty space in the parking lot to conduct the road surface changing attack. Here our goal is to change the drivable road surface to non-drivable surface (grass). Similar to the vehicle hiding attack, we first collect the original point cloud data using the car shown in Figure 7. Then we derive the locations of the adversarial objects offline using the collected data. In this experiment, we still set the number of adversarial point clusters as 3 and use 3 pieces of cardboard as the adversarial objects. The locations of adversarial objects are constrained on the sidewalk in Figure 9i. Then we place these objects at the derived locations in the parking lot as shown in Figure 9k. Figure 9j show an example of the original segmentation results before the attack. We can see the road surface can be correctly recognized. However, after the attack, the road in front of the victim AV is wrongly recognized as "Vegetation", which is shown in Figure 9l. The results further demonstrate the effectiveness of our proposed attack framework in physical world.

7.3 Effect of Semantic Misleading

To show the effect of semantic misleading, we compare our attacks with baselines. We generate adversarial examples without semantic loss (without semantic misleading) as a baseline method (named w/o SeMi). And we also generate adversarial clusters at random locations (named random) and compare the results.



(a) Vehicle hiding



(b) Road surface changing

Figure 10: Attack success rate on SemanticKITTI w.r.t the number of adversarial objects

Fig. 10 shows the attack success rate of vehicle hiding and road surface changing attack. Adding more adversarial objects results in higher success rate. Adding 5 adversarial objects is enough to perform the two types of attacks, which achieves around 80% attack success rate. Even using only 2 adversarial objects, the vehicle hiding attack can achieve over 50% success rate. The performances of white-box attack is slightly better than black-box attack. The attack without semantic misleading performs worse than our methods. Randomly adding objects has small impact on the segmentation results. Our semantic misleading method can improve the attack success rate and help to generate strong adversarial attacks.

7.4 Attack Moving Victim Vehicle

As mentions in Section 5.1, it is essential to mislead consecutive frames to ensure a successful attack when the victim AV is moving. In this experiment, we evaluate the performance of our attack framework on consecutive point cloud frames. Specifically, we select the same examples as previous experiments for each attack scenario and each example contains some consecutive frames of point cloud data. For each example, we consider four cases and the corresponding numbers of consecutive frames are 5, 10, 20, and 30, respectively. We then generate the adversarial point clusters for each case by taking all the consecutive frames in this case as the

Table 2: The average attack success rate on SemanticKITTI w.r.t. number of consecutive frames

Number of frames	5	10	20	30
Vehicle hiding	0.83	0.73	0.63	0.55
Road surface changing	0.78	0.67	0.54	0.36

original point clouds inputs of our proposed framework. Here we still consider the black-box attack and the number of adversarial point clusters is predefined as 5. Table 2 shows the average attack success rate when the segmentation model is PointNet. We can see our proposed attack framework can still achieve good performance on consecutive frames for both vehicle hiding attack and road surfacing changing attack. Even when the number of consecutive frames is 30, the average attack success rate for vehicle hiding attack can still be larger than 50%, which enables the attacker to constantly hide the parked car when the victim AV is approaching the parked car. From Table 2 we can also observe that the success rate for road surface changing attack is lower than that for vehicle hiding attack. This is probably because the number of "Ground" points are much larger than that of "Vehicle" points in most examples.

To further validate the effectiveness of our proposed framework on moving victim AV, we next conduct experiments in physical world. We consider the attack scenario shown in Figure 9a and place the adversarial objects at the locations shown in Fig 9c. Then we imitate the victim AV and drive the vehicle with LiDAR (as shown in Figure 7) from 20 meters away to 5 meters away from the parked car. In this experiment, we collect 70 LiDAR frames and use PointNet as the segmentation model.

Figure 11 reports the segmentation results for three of these frames after the attack. We can see the parked car can be successfully "hidden" when the victim AV is approaching the parked car. The average attack success rate on these 70 frames is 97.7%. Specifically, the parked car is completely "hidden" in 54 out of 70 frames. In other 16 frames, very few points (5-10 points in each frame) are still classified as "Vehicle". We also find that when the distance between victim AV and the parked car is larger than 10 meters (47 frames are collected), the parked car is completely "hidden" in 44 frames (93.6% of them); when the distance between victim AV and the parked car is smaller than 10 meters (23 frames are collected), the parked car is completely "hidden" in 12 frames (52.2% of them). This shows that our attack have high attack success rate when the victim AV is far from (larger than 10 meters) the parked car.

7.5 Robustness and Generalization Analysis

7.5.1 Robustness to Different Driving Directions. It is usually difficult for the attacker to predict the driving direction of the victim AV. Thus, the generated adversarial example should be robust to different driving directions. In this experiment, we consider the attack scenario shown in Figure 9a and place the adversarial objects at the locations shown in Fig 9c. Then we drive the vehicle with LiDAR approaching the parked car from different directions (e.g., left and right). Here we still drive from 20 meters away to 5 meters away from the parked car and collect 210 LiDAR frames in total. Figure 12 shows the experimental results. We can see the parked car can be successfully "hidden" no matter which direction the victim AV comes from. The attack success rate on these 210 frames

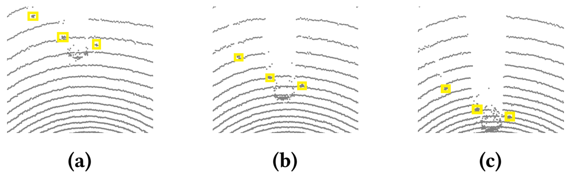


Figure 11: Bird-eye-view of the segmentation results (after the attack) when the victim AV approaches the parked car.

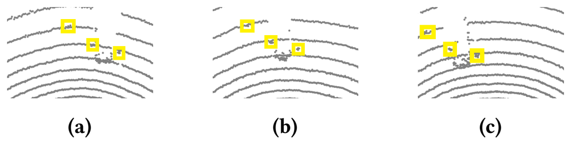


Figure 12: Bird-eye-view of the segmentation results (after the attack) when the victim AV comes from different directions. (a) LiDAR on the left side. (b) LiDAR in the middle. (c) LiDAR on the right side.

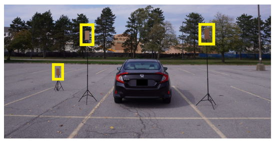
Table 3: The average attack success rate on SemanticKITTI w.r.t. location error

Error(m)	0.00	0.05	0.10	0.15	0.20	0.30
Vehicle hiding	0.83	0.79	0.77	0.74	0.69	0.61
Road surface changing	0.78	0.73	0.68	0.66	0.58	0.50

is 96.0%. Specifically, when the distance between victim AV and parked car is larger than 10 meters (143 frames are collected), the parked car is completely "hidden" in 133 frames (93.0% of them). These results are similar to previous experiments where victim AV drives right behind the parked car, and they demonstrate that our attacks are robust to different driving directions.

7.5.2 Location Errors of Adversarial Objects. In practice, it is usually difficult to exactly place the adversarial objects at the derived locations. To evaluate the robustness of our proposed attack framework to location errors, we choose the same examples as previous experiments for each attack scenario from the SemanticKITTI dataset and generate corresponding adversarial point clusters for each example. Then we shift each adversarial point cluster within a given distance towards a random direction. Table 3 reports the average attack success rate over the these examples after shifting the clusters. Here we vary the given distance from 0m to 0.3m for each type of attack. We can see the the proposed attack framework can still achieve good performance in all cases. Even when the distance is 0.3m, the attack success rate can still be larger than 50%.

We also evaluate the robustness of our attack framework to location errors in physical world. Specifically, we consider the scenario in Figure 9a and shift each adversarial object in Figure 9c within 0.3m towards a random direction. We drive the vehicle from 20 meters away to 5 meters away from the parked car and collect 69 LiDAR frames in total. The new locations of the adversarial objects are shown in Figure 13a and the segmentation result after attack is shown in Figure 13b. The result shows that the parked car can still be "hidden" although we randomly shift the adversarial objects. In the 69 collected frames, the average attack success rate is 96.0%. When the distance between the victim AV and parked car is larger than 10 meters (46 frames are collected), the parked car is completely "hidden" in 41 frames (89.1% of them). The results are


(a)

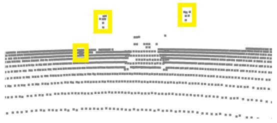
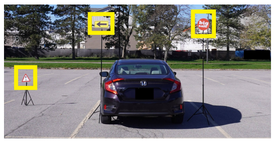

(b)

Figure 13: The visualizations of a real-world adversarial example after shifting the adversarial objects. (a) The new locations of adversarial objects after shifting. (b) The segmentation result after shifting.


(a)

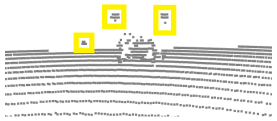

(b)

Figure 14: The visualizations of a real-world adversarial example with different shapes of objects. (a) The traffic signs and billboards as new adversarial objects. (b) The segmentation result with those objects.

similar to that in Figure 11. The results also show that the generated adversarial examples can tolerate small location errors (less than 0.3m) of the adversarial objects, which further demonstrates the practicability and robustness of our attack framework.

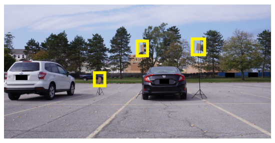
7.5.3 Vehicle Hiding Attack with Different Shapes of Objects. As mentioned in Section 4, the adversarial object could be in any shape as long as it can reflect laser. In this experiment, we evaluate the performance of our proposed attack scheme with some road traffic signs and billboards instead of the cardboard. Specifically, we consider the attack scenario in Figure 9c. We replace the cardboard with two traffic signs and a billboard and keep them at the same locations as shown in Figure 14a. Then we drive the vehicle from 20 meters away to 5 meters away from the parked car and collect 65 LiDAR frames in total. Figure 14b shows the segmentation results with new adversarial objects. As we can see, these adversarial objects successfully create some point clusters with different shapes at given locations, and make the parked car "invisible" to the victim AV. The attack success rate for all of the 65 frames is 95.8%. When the distance between the victim AV and parked car is larger than 10 meters (44 frames are collected), the parked car is completely "hidden" in 43 frames (97.7% of them). The result shows that the attacker could use objects in arbitrary shapes to launch the attack, such as common road traffic signs and billboards.

7.5.4 The Effect of the Victim Vehicle's Speed. Next, we evaluate the effect of the victim vehicle's speed on the attack performance. Specifically, we let the victim vehicle drive at different speeds (5mph, 10mph, 15mph, 20mph, and 25mph). Table 4 shows the average attack success rates of the two types of attacks. We can see that the speed has little impact on the attack success rate. Even when the victim vehicle is at high speed (25mph), the attack success rate can be around 99%.

7.5.5 The Effect of Passing-by Vehicles. In this experiment, we study the effect of passing-by vehicles on the performance of our attack framework. Here we still consider the scenario shown in Figure 9c. To simulate a passing-by event, we drive another vehicle (white car) beside the parked car while keeping the parked car and the adversarial objects at same locations as before, which is shown in Figure 15a. The victim vehicle also drives from 20 meters away to 5 meters away from the parked car and collect 76 LiDAR frames. The segmentation results in Figure 15b show that the target vehicle (parked black car) is successfully "hidden" by the adversarial objects. In the collected 76 frames, the attack success rate frames is 72.3%.

Table 4: Average attack success rate under different speeds.

Speed (mph)	5	10	15	20	25
Vehicle hiding	0.97	0.97	0.96	0.97	0.99
Road surface changing	0.71	0.72	0.71	0.72	0.72


(a)

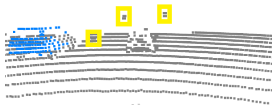

(b)

Figure 15: The visualization of a real-world adversarial example with a passing-by vehicle. (a) Another Car Passing by The Vehicle. (b) The segmentation Result after Attack.

8 ATTACK AGAINST OBJECT DETECTION MODELS

To further demonstrate the effectiveness of our proposed attack, we attack various LiDAR object detection systems. A common pipeline of many state-of-the-art object detection systems is: the system first performs semantic segmentation to find foreground points (such as vehicles and pedestrians) from background points; then a post-processing step is performed to generate bounding box proposals based on the found foreground points; finally, the bounding box is used for further applications such as object tracking and motion planning. For example, in the LiDAR perception pipeline of Baidu Apollo 3.0, a CNN semantic segmentation is adopted to segment obstacle points, which is shown in Figure 16.

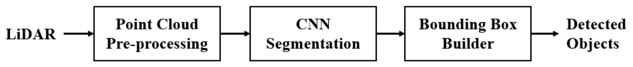
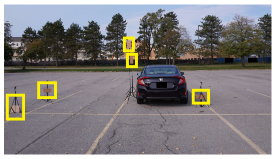


Figure 16: LiDAR perception pipeline in Baidu Apollo 3.0.

Our proposed attack framework can fool the foreground segmentation step and may result in wrong object detection result, i.e., hiding a vehicle from detection model. By hiding the foreground points, the victim AV can not generate bounding box proposals correctly. In this experiment, we conduct vehicle hiding attacks against two state-of-the-art object detection models: Frustum-PointNet [51] and Baidu Apollo [2].

Frustum-PointNet. This model first extracts 3D bounding frustum of an object based on 2D image detection results. Then it applies 3D semantic segmentation in each frustum to segment foreground points from the background using a PointNet-like network, and the detection result is generated from the foreground points.

Baidu Apollo. Figure 16 shows the obstacle perception pipeline used in Baidu Apollo. In the preprocessing step, points are projected to the X-Y plane and divide into grids. In CNN semantic


(a)

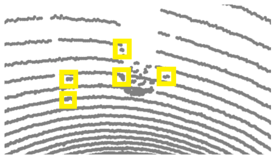

(b)

Figure 17: Visualization of Adversarial Examples for Baidu Apollo object detection models. (a) Adding adversarial objects. (b) The detection result after attack.

segmentation step, 8 features are extracted from each grid and fed into the fully-convolutional neural network to segment and cluster the foreground points. The foreground point clusters that contain few points or have low confidence are removed. Finally, the Post-processing module is used to refine the candidate clusters and a Minbox builder is used to estimate the object bounding box.

In this experiment, we aim to conduct black-box attack against the object detection results (i.e., hide the parked car from detection model) by attacking the foreground segmentation sub-modules. Since we cannot obtain the segmentation results directly in black-box attack, we infer the segmentation results from the object detection results. Specifically, we assume all the points inside the generated bounding boxes are segmented as "foreground", and other points outside the bounding boxes are segmented as "background". The goal of our attack is to make all the foreground points in front-near of victim AV misclassified as background points. Here we train the models using the KITTI dataset [21]. The number of adversarial objects is set to 5. We measure the performance of our attack framework using attack success rate, which is defined as the percentage of the vehicles that are not detected after the attack.

Figure 17 shows the visualizations of our attack against Baidu Apollo LiDAR perception model. Before the attack, the vehicle points are correctly predicted in the segmentation module and a bounding box is correctly generated. After adding the adversarial objects that are highlighted with yellow rectangles in Figure 17a, the vehicle points are labelled as "background" in the foreground segmentation step. As a result, the parked car is not detected by the model, which can be seen from Figures 17b.

In this experiment, we also evaluate the performance of our attack framework on the KITTI dataset. Specifically, we select 10 scenes from the dataset and generate the locations of adversarial clusters. Table 5 shows the attack success rate on different detection models. We can see our attack framework has the best performance on Frustum-PointNet and the attack success rate is 87%.

9 DISCUSSION

9.1 How to Perform the Attack in Practice

In practice, the adversarial locations can be derived offline. The attacker can obtain a similar LiDAR that is adopted by the victim AV and imitate its driving behaviors to collect the original point cloud. Then the adversarial locations are derived using the proposed method. With the derived locations, the attacker can launch the attack by placing arbitrary objects that can reflect laser such as road signs around these locations. Since the locations are generated in an offline manner, the attacker can easily and quickly launch the attack when the victim AV is approaching.

Table 5: The average attack success rate of object detection

Models	Attack success rate
Frustum-PointNet	0.87
Baidu Apollo	0.77

9.2 Defense Strategies

The authors in [62] propose a defense strategy that can mitigate the LiDAR spoofing attack described in [9]. However, this strategy can not be directly applied to our attack. On one hand, the defense in [62] is used to identify the injected unreal fake points, but our injected points are generated by real physical objects, which can not be detected by their defense. On the other hand, the defense strategy in [62] can be attacked by our method because they use LiDAR semantic segmentation to assist object detection. In this section, we discuss two types of potential defense strategies to mitigate the proposed attacks: sensor fusion and output aggregation.

Sensor fusion. The most straightforward defend approach is to utilize other types of sensors such as camera and radar. However, such defense strategy requires additional sensors and increases the cost of autonomous vehicle systems. Besides, existing studies have found that camera perception system and radar are also vulnerable to adversarial attacks [29, 54, 73]. The attacker could attack all the sensors to achieve the attack goal [8, 65].

Output aggregation. Another potential defense strategy is based on aggregation approach. Recent studies have found that 3D adversarial examples usually have poor transferability between different models [70]. Based on this finding, a potential defense method is to train multiple point cloud segmentation models with different data augmentation and different variable initialization. Each model provides a segmentation result of the input point cloud frame. The final segmentation result is aggregated from the outputs of all models through majority voting. Even there is a model fooled by the attack successfully, the other models could still provide correct results because of the low transferability of adversarial examples.

10 CONCLUSIONS

In this paper, we explore how to perform practical and effective adversarial attacks against LiDAR semantic segmentation in autonomous driving. We first analyze the attack challenges in physical world and then propose a novel attack framework based on which the attacker can easily fool the semantic segmentation models by adding arbitrary objects to the driving environment. In this framework, we consider both white-box and black-box attacks. We conduct extensive real-world experiments to evaluate the performance of the proposed attack framework. The experimental results demonstrated that our proposed attacks are not only effective but also robust. Our attack framework can achieve more than 90% success rate in real-world driving environments.

11 ACKNOWLEDGMENTS

We thank our anonymous reviewers for their insightful comments and suggestions on this paper. This work was supported in part by the US National Science Foundation under grant CNS-1626374, CNS-2120369, CNS-1737590, CNS-1652503, and ECCS-2028872.

REFERENCES

[1] [n.d.]. Autoware project. https://www.autoware.org/.
[2] [n.d.]. Baidu Apollo. https://apollo.auto/.
[3] [n.d.]. Baidu fully opens Apollo Go Robotaxi service in Beijing. http://autonews.gasgoo.com/icv/70017615.html.
[4] [n.d.]. Driverless taxis to be available in Phoenix 'in weeks'. https://www.bbc.com/news/technology-54476524.
[5] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. 2019. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)*.
[6] Ravi Bhandari, Akshay Uttama Nambi, Venkata N Padmanabhan, and Bhaskaran Raman. 2020. Driving lane detection on smartphones using deep neural networks. *ACM Transactions on Sensor Networks (TOSN)* 16, 1 (2020), 1–22.
[7] Chongguang Bi, Jun Huang, Guoliang Xing, Landu Jiang, Xue Liu, and Minghua Chen. 2019. Safewatch: A wearable hand motion tracking system for improving driving safety. *ACM Transactions on Cyber-Physical Systems* 4, 1 (2019), 1–21.
[8] Yulong Cao, Ningfei Wang, Chaowei Xiao, Dawei Yang, Jin Fang, Ruigang Yang, Qi Alfred Chen, Mingyan Liu, and Bo Li. 2021. Invisible for both camera and lidar: Security of multi-sensor fusion based perception in autonomous driving under physical-world attacks. In *Proceedings of the 2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 176–194.
[9] Yulong Cao, Chaowei Xiao, Benjamin Cyr, Yimeng Zhou, Won Park, Sara Rampazzi, Qi Alfred Chen, Kevin Fu, and Z Morley Mao. 2019. Adversarial sensor attack on lidar-based perception in autonomous driving. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. 2267–2281.
[10] Yulong Cao, Chaowei Xiao, Dawei Yang, Jing Fang, Ruigang Yang, Mingyan Liu, and Bo Li. 2019. Adversarial objects against lidar-based autonomous driving systems. *arXiv preprint arXiv:1907.05418* (2019).
[11] Nicholas Carlini and David Wagner. 2017. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*. IEEE, 39–57.
[12] Changhao Chen, Bing Wang, Chris Xiaoxuan Lu, Niki Trigoni, and Andrew Markham. 2020. A survey on deep learning for localization and mapping: Towards the age of spatial machine intelligence. *arXiv preprint arXiv:2006.12567* (2020).
[13] Huangxun Chen, Chenyu Huang, Qianyi Huang, Qian Zhang, and Wei Wang. 2020. Ecgadv: Generating adversarial electrocardiogram to misguide arrhythmia classification system. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 3446–3453.
[14] Huangxun Chen, Zhice Yang, Chenyu Huang, and Qian Zhang. 2018. Drive Safe Inspector: A Wearable-Based Fine-Grained Technique for Driver Hand Position Detection. In *2018 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 1–7.
[15] Tao Chen, Longfei Shangguan, Zhenjiang Li, and Kyle Jamieson. 2020. Metamorph: Injecting inaudible commands into over-the-air voice controlled systems. In *Proceedings of NDSS*.
[16] Liang Du, Jingang Tan, Xiangyang Xue, Lili Chen, Hongkai Wen, Jianfeng Feng, Jiamao Li, and Xiaolin Zhang. 2020. 3dcfs: Fast and robust joint 3d semantic-instance segmentation via coupled feature selection. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 6868–6875.
[17] Francis Engelmann, Jörg Stückler, and Bastian Leibe. 2016. Joint object pose estimation and shape reconstruction in urban street scenes using 3D shape priors. In *German Conference on Pattern Recognition*. Springer, 219–230.
[18] Zhihan Fang, Guang Wang, Xiaoyang Xie, Fan Zhang, and Desheng Zhang. 2021. Urban Map Inference by Pervasive Vehicular Sensing Systems with Complementary Mobility. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 1 (2021), 1–24.
[19] Ruipeng Gao, Mingmin Zhao, Tao Ye, Fan Ye, Yizhou Wang, and Guojie Luo. 2017. Smartphone-based real time vehicle tracking in indoor parking structures. *IEEE Transactions on Mobile Computing* 16, 7 (2017), 2023–2036.
[20] Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, and Jose Garcia-Rodriguez. 2017. A review on deep learning techniques applied to semantic segmentation. *arXiv preprint arXiv:1704.06857* (2017).
[21] Andreas Geiger, Philip Lenz, and Raquel Urtasun. 2012. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.
[22] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).
[23] Jun Han, Albert Jin Chung, Manal Kumar Sinha, Madhumitha Harishankar, Shijia Pan, Hae Young Noh, Pei Zhang, and Patrick Tague. 2018. Do you feel what I hear? Enabling autonomous IoT device pairing using different sensor types. In *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 836–852.
[24] Jan Hendrik Metzen, Mummadi Chaithanya Kumar, Thomas Brox, and Volker Fischer. 2017. Universal adversarial perturbations against semantic image segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*. 2755–2764.
[25] Taewook Heo, Woojin Nam, Jeongyeup Paek, and JeongGil Ko. 2020. Autonomous Reckless Driving Detection Using Deep Learning on Embedded GPUs. In *2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*. IEEE, 464–472.
[26] Jorge Hernández and Beatriz Marcotegui. 2009. Point cloud segmentation towards urban ground modeling. In *2009 Joint Urban Remote Sensing Event*. IEEE, 1–5.
[27] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. 2020. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11108–11117.
[28] Ishani Janveja, Akshay Nambi, Shruthi Bannur, Sanchit Gupta, and Venkat Padmanabhan. 2020. InSight: Monitoring the State of the Driver in Low-Light Using Smartphones. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 3 (2020), 1–29.
[29] Yunhan Jia, Yantao Lu, Junjie Shen, Qi Alfred Chen, Hao Chen, Zhenyu Zhong, and Tao Wei. 2019. Fooling detection alone is not enough: Adversarial attack against multiple object tracking. In *International Conference on Learning Representations*.
[30] Meng Jin, Yuan He, Dingyi Fang, Xiaojiang Chen, Xin Meng, and Tianzhang Xing. 2018. iGuard: A real-time anti-theft system for smartphones. *IEEE Transactions on Mobile Computing* 17, 10 (2018), 2307–2320.
[31] Hassan Khan, Urs Hengartner, and Daniel Vogel. 2018. Augmented reality-based mimicry attacks on behaviour-based smartphone authentication. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*. 41–53.
[32] Hassan Khan, Urs Hengartner, and Daniel Vogel. 2020. Mimicry attacks on smartphone keystroke authentication. *ACM Transactions on Privacy and Security (TOPS)* 23, 1 (2020), 1–34.
[33] K Naveen Kumar, C Vishnu, Reshmi Mitra, and C Krishna Mohan. 2020. Black-box Adversarial Attacks in Autonomous Vehicle Technology. In *2020 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*. IEEE, 1–7.
[34] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. 2019. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 12697–12705.
[35] Tu Le, Ingy ElSayed-Aly, Weizhao Jin, Seunghan Ryu, Guy Verrier, Tamjid Al Rahat, B Brian Park, and Yuan Tian. [n.d.]. Poster: Attack the Dedicated Short-Range Communication for Connected Vehicles. ([n. d.]).
[36] Taegyeong Lee, Zhiqi Lin, Saumay Pushp, Caihua Li, Yunxin Liu, Youngki Lee, Fengyuan Xu, Chenren Xu, Lintao Zhang, and Junehwa Song. 2019. Occlumency: Privacy-preserving remote deep-learning inference using SGX. In *The 25th Annual International Conference on Mobile Computing and Networking*. 1–17.
[37] Guanlin Li, Shuya Ding, Jun Luo, and Chang Liu. 2020. Enhancing Intrinsic Adversarial Robustness via Feature Pyramid Decoder. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 800–808.
[38] Hongyu Li, Luyang Liu, Cagdas Karatas, Jian Liu, Marco Gruteser, Yingying Chen, Yan Wang, Richard P Martin, and Jie Yang. 2016. Towards safer texting while driving through stop time prediction. In *Proceedings of the First ACM International Workshop on Smart, Autonomous, and Connected Vehicular Systems and Services*. 14–21.
[39] Hongyu Li, Hairong Wang, Luyang Liu, and Marco Gruteser. 2018. Automatic unusual driving event identification for dependable self-driving. In *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems*. 15–27.
[40] Huichen Li, Xiaojun Xu, Xiaolu Zhang, Shuang Yang, and Bo Li. 2020. QEBA: Query-Efficient Boundary-Based Blackbox Attack. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1221–1230.
[41] Yujie Li, Xing Xu, Jinhui Xiao, Siyuan Li, and Heng Tao Shen. 2020. Adaptive Square Attack: Fooling Autonomous Cars with Adversarial Traffic Signs. *IEEE Internet of Things Journal* (2020).
[42] Zhijing Li, Zhujun Xiao, Yanzi Zhu, Irene Pattarachanyakul, Ben Y Zhao, and Haitao Zheng. 2018. Adversarial localization against wireless cameras. In *Proceedings of the 19th International Workshop on Mobile Computing Systems & Applications*. 87–92.
[43] Kai Li Lim, Thomas Drage, and Thomas Bräunl. 2017. Implementation of semantic segmentation for road and lane detection on an autonomous ground vehicle with LIDAR. In *2017 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*. IEEE, 429–434.
[44] Kin Sum Liu, Chaowei Xiao, Bo Li, and Jie Gao. 2019. Performing co-membership attacks against deep generative models. In *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 459–467.
[45] Shinan Liu, Xiang Cheng, Hanchao Yang, Yuanchao Shu, Xiaoran Weng, Ping Guo, Kexiong Curtis Zeng, Gang Wang, and Yaling Yang. [n.d.]. Stars Can Tell: A Robust Method to Defend against GPS Spoofing Attacks using Off-the-shelf Chipset. ([n. d.]).
[46] Hongwei Luo, Yijie Shen, Feng Lin, and Guoai Xu. 2021. Spoofing Speaker Verification System by Adversarial Examples Leveraging the Generalized Speaker Difference. *Security and Communication Networks* 2021 (2021).
[47] Daniel Maturana, Po-Wei Chou, Masashi Uenoyama, and Sebastian Scherer. 2018. Real-time semantic mapping for autonomous off-road navigation. In *Field and Service Robotics*. Springer, 335–350.

[48] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. 2017. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*. 506–519.

[49] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. 2016. The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy (EuroS&P)*. IEEE, 372–387.

[50] Fabio Pierazzi, Feargus Pendlebury, Jacopo Cortellazzi, and Lorenzo Cavallaro. 2020. Intriguing properties of adversarial ML attacks in the problem space. In *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1332–1349.

[51] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. 2018. Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 918–927.

[52] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 652–660.

[53] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. 2017. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*. 5099–5108.

[54] Kui Ren, Qian Wang, Cong Wang, Zhan Qin, and Xiaodong Lin. 2019. The security of autonomous driving: Threats, defenses, and future directions. *Proc. IEEE* 108, 2 (2019), 357–372.

[55] Ishtiaq Rouf, Robert D Miller, Hossen A Mustafa, Travis Taylor, Sangho Oh, Wenyuan Xu, Marco Gruteser, Wade Trappe, and Ivan Seskar. 2010. Security and Privacy Vulnerabilities of In-Car Wireless Networks: A Tire Pressure Monitoring System Case Study.. In *USENIX Security Symposium*, Vol. 10.

[56] Sriram Sami, Yimin Dai, Sean Rui Xiang Tan, Nirupam Roy, and Jun Han. 2020. Spying with your robot vacuum cleaner: eavesdropping via lidar sensors. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*. 354–367.

[57] Huajie Shao, Shuochao Yao, Andong Jing, Shengzhong Liu, Dongxin Liu, Tianshi Wang, Jinyang Li, Chaoqi Yang, Ruijie Wang, and Tarek Abdelzaher. 2020. Misinformation Detection and Adversarial Attack Cost Analysis in Directional Social Networks. In *2020 29th International Conference on Computer Communications and Networks (ICCCN)*. IEEE, 1–11.

[58] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. 2019. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 770–779.

[59] Qun Song, Zhenyu Yan, and Rui Tan. 2019. Moving target defense for embedded deep visual sensing against adversarial examples. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*. 124–137.

[60] Rainer Storn. 1996. On the usage of differential evolution for function optimization. In *Proceedings of North American Fuzzy Information Processing*. IEEE, 519–523.

[61] Rainer Storn and Kenneth Price. 1997. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization* 11, 4 (1997), 341–359.

[62] Jiachen Sun, Yulong Cao, Qi Alfred Chen, and Z Morley Mao. 2020. Towards Robust LiDAR-based Perception in Autonomous Driving: General Black-box Adversarial Sensor Attack and Countermeasures. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*. 877–894.

[63] Fnu Suya, Jianfeng Chi, David Evans, and Yuan Tian. 2020. Hybrid batch attacks: Finding black-box adversarial examples with limited queries. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*. 1327–1344.

[64] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).

[65] James Tu, Huichen Li, Xinchen Yan, Mengye Ren, Yun Chen, Ming Liang, Eilyan Bitar, Ersin Yumer, and Raquel Urtasun. 2021. Exploring Adversarial Robustness of Multi-Sensor Perception Systems in Self Driving. *arXiv preprint arXiv:2101.06784* (2021).

[66] James Tu, Mengye Ren, Sivabalan Manivasagam, Ming Liang, Bin Yang, Richard Du, Frank Cheng, and Raquel Urtasun. 2020. Physically Realizable Adversarial Examples for LiDAR Object Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13716–13725.

[67] Yongyong Wei and Rong Zheng. 2020. Informative path planning for mobile sensing with reinforcement learning. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 864–873.

[68] Matthew Wicker and Marta Kwiatkowska. 2019. Robustness of 3d deep learning in an adversarial setting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 11767–11775.

[69] Bichen Wu, Alvin Wan, Xiangyu Yue, and Kurt Keutzer. 2018. Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 1887–1893.

[70] Chong Xiang, Charles R Qi, and Bo Li. 2019. Generating 3d adversarial point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 9136–9144.

[71] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. 2017. Adversarial examples for semantic segmentation and object detection. In *Proceedings of the IEEE International Conference on Computer Vision*. 1369–1378.

[72] Yi Xie, Zhuohang Li, Cong Shi, Jian Liu, Yingying Chen, and Bo Yuan. 2020. Enabling Fast and Universal Audio Adversarial Attack Using Generative Model. *arXiv preprint arXiv:2004.12261* (2020).

[73] Kaidi Xu, Gaoyuan Zhang, Sijia Liu, Quanfu Fan, Mengshu Sun, Hongge Chen, Pin-Yu Chen, Yanzhi Wang, and Xue Lin. 2020. Adversarial t-shirt! evading person detectors in a physical world. In *European Conference on Computer Vision*. Springer, 665–681.

[74] Xu Yan, Chaoda Zheng, Zhen Li, Sheng Wang, and Shuguang Cui. 2020. PointASNL: Robust Point Clouds Processing using Nonlocal Neural Networks with Adaptive Sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5589–5598.

[75] Zhang Yanyong, Zhang Sha, Zhang Yu, Ji Jianmin, Duan Yifan, Huang Yitong, Peng Jie, and Zhang Yuxiang. 2020. Multi-Modality Fusion Perception and Computing in Autonomous Driving. *Journal of Computer Research and Development* 57, 9 (2020), 1781.

[76] Xiaoyong Yuan, Pan He, Qile Zhu, and Xiaolin Li. 2019. Adversarial examples: Attacks and defenses for deep learning. *IEEE transactions on neural networks and learning systems* 30, 9 (2019), 2805–2824.

[77] Kexiong Curtis Zeng, Shinan Liu, Yuanchao Shu, Dong Wang, Haoyu Li, Yanzhi Dou, Gang Wang, and Yaling Yang. 2018. All your {GPS} are belong to us: Towards stealthy manipulation of road navigation systems. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*. 1527–1544.

[78] Yang Zhang, Zixiang Zhou, Philip David, Xiangyu Yue, Zerong Xi, Boqing Gong, and Hassan Foroosh. 2020. PolarNet: An Improved Grid Representation for Online LiDAR Point Clouds Semantic Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9601–9610.

[79] Mengyao Zheng, Dixing Xu, Linshan Jiang, Chaojie Gu, Rui Tan, and Peng Cheng. 2019. Challenges of privacy-preserving machine learning in IoT. In *Proceedings of the First International Workshop on Challenges in Artificial Intelligence and Machine Learning for Internet of Things*. 1–7.

[80] Tianhang Zheng, Changyou Chen, Junsong Yuan, Bo Li, and Kui Ren. 2019. Pointcloud saliency maps. In *Proceedings of the IEEE International Conference on Computer Vision*. 1598–1606.

[81] Hao Zhu, Mengyin Fu, Yi Yang, Xinyu Wang, and Meiling Wang. 2014. A path planning algorithm based on fusing lane and obstacle map. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 1442–1448.

[82] Xinge Zhu, Hui Zhou, Tai Wang, Fangzhou Hong, Yuexin Ma, Wei Li, Hongsheng Li, and Dahua Lin. 2021. Cylindrical and asymmetrical 3d convolution networks for lidar segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9939–9948.

[83] Yi Zhu, Chenglin Miao, Tianhang Zheng, Foad Hajiaghajani, Lu Su, and Chunming Qiao. 2021. Can We Use Arbitrary Objects to Attack LiDAR Perception in Autonomous Driving?. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS '21)*.