Physical Backdoor Attacks to Lane Detection Systems in Autonomous Driving

Xingshuo Han
xingshuo001@e.ntu.edu.sg
Nanyang Technological University
Singapore

Guowen Xu
guowen.xu@ntu.edu.sg
Nanyang Technological University
Singapore

Yuan Zhou∗
y.zhou@ntu.edu.sg
Nanyang Technological University
Singapore

Xuehuan Yang
S190113@e.ntu.edu.sg
Nanyang Technological University
Singapore

Jiwei Li
jiwei_li@shannonai.com
Shannon.AI, Zhejiang University
China

Tianwei Zhang
tianwei.zhang@ntu.edu.sg
Nanyang Technological University
Singapore

ABSTRACT
Modern autonomous vehicles adopt state-of-the-art DNN models to interpret the sensor data and perceive the environment. However, DNN models are vulnerable to different types of adversarial attacks, which pose significant risks to the security and safety of the vehicles and passengers. One prominent threat is the backdoor attack, where the adversary can compromise the DNN model by poisoning the training samples. Although lots of effort has been devoted to the investigation of the backdoor attack to conventional computer vision tasks, its practicality and applicability to the autonomous driving scenario is rarely explored, especially in the physical world.

In this paper, we target the lane detection system, which is an indispensable module for many autonomous driving tasks, e.g., navigation, lane switching. We design and realize the first physical backdoor attacks to such system. Our attacks are comprehensively effective against different types of lane detection algorithms. Specifically, we introduce two attack methodologies (poison-annotation and clean-annotation) to generate poisoned samples. With those samples, the trained lane detection model will be infected with the backdoor, and can be activated by common objects (e.g., traffic cones) to make wrong detections, leading the vehicle to drive off the road or onto the opposite lane. Extensive evaluations on public datasets and physical autonomous vehicles demonstrate that our backdoor attacks are effective, stealthy and robust against various defense solutions. Our codes and experimental videos can be found in https://sites.google.com/view/lane-detection-attack/lda.

CCS CONCEPTS
• Security and privacy → Systems security.

∗Corresponding author

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

MM ’22, October 10–14, 2022, Lisboa, Portugal
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9203-7/22/10...
https://doi.org/10.1145/3503161.3548171

KEYWORDS
lane detection, deep learning, autonomous driving

1 INTRODUCTION
The rapid development of deep learning technology has increased the perception capability of autonomous vehicles to interpret the environment and make intelligent actions. The vehicle collects multiple types of data from the sensors and employs DNN models to accomplish different functions. One important function is lane detection, which aims to identify the traffic lanes from the images or videos captured by the camera. This function is critical in autonomous driving for lane following, changing and overtaking. Over the past years, a large number of deep learning based algorithms and approaches have been introduced to significantly
improve the detection accuracy and real-time efficiency [17, 22, 24, 31, 35, 36, 46, 47, 67].

Unfortunately, past works have demonstrated that DNN models are not robust and can be easily fooled by malicious entities. One infamous threat is the DNN backdoor [12, 13, 32, 57]. The adversary embeds a secret backdoor into the victim model by poisoning the training set. This backdoor remains dormant with normal input inference samples. It will be activated by malicious samples which contain an adversary-specific trigger, misleading the infected model to give wrong predictions. Researchers have proposed a variety of novel attacks against DNN models for computer vision [3, 21, 27, 33, 34, 54, 58], natural language processing [5, 10, 44, 60], reinforcement learning [25, 53, 65], etc. However, there are no studies investigating the backdoor opportunity against the lane detection systems.

This paper aims to bridge this gap by designing and implementing the first practical backdoor attacks to lane-detection DNN models in the physical world. There are a couple of challenges to achieving this goal. First, existing works mainly focus on backdoor attacks in the digital world, where the adversary can arbitrarily manipulate the input samples for adding triggers (e.g., changing a block of pixels in an image). It is hard to utilize those techniques to attack real-life applications due to the semantic gap between the digital and physical worlds. A few works then implement physical backdoor attacks in the real-world setting [6, 26, 40, 55, 64]. However, these methods mainly target the face classification models. Different from them, lane detection models do not predict labels, which increases the difficulty of poisoned sample generation. Besides, the physical triggers used to attack face classification models cannot be applied to lane detection due to the semantic differences between these two scenarios. The physical triggers need to be carefully redesigned.

Second, to make the backdoor more stealthy, past works propose clean-label attacks against classification models, where the poisoned samples still have the correct labels to compromise the model [43, 66]. This is achieved by adding adversarial perturbations to alter the classes of these poisoned samples. Since lane detection models do not predict classes, it is hard to leverage these solutions to generate visually normal poisoned samples.

Third, existing backdoor attacks focus on one specific DL algorithm (e.g., classification) when poisoning the data samples. However, this does not hold true for the lane detection scenario, which can use different algorithms to train the model, such as segmentation-based [35] or anchor-based [46] methods. It is challenging to generate unified poisoned samples, which can attack any lane detection models regardless of their algorithms.

Our proposed attacks can address the above challenges with several innovations. First, we present new designs of semantic triggers in the context of autonomous driving. After investigating some mainstream traffic datasets, we select a set of two traffic cones with specific shapes and positions as the trigger to activate the backdoor. This trigger looks very natural in the road environment and is difficult to be noticed. Meanwhile, it is also unique enough not to affect the normal conditions of autonomous driving. Second, we introduce two new approaches to poison training samples and manipulate annotations to achieve backdoor embedding. (1) Poison-annotation attack: the adversary can craft poisoned samples by intentionally mis-annotating the samples with the trigger. (2) Clean-annotation attack: this technique utilizes the image scaling vulnerability [56] to conceal the anomaly of malicious samples. Specifically, we create poisoned samples, which are visually similar to clean ones with the correct annotation and no triggers. After image scaling, those samples will give wrong lane boundaries and a trigger, which become effective in backdoor embedding. Both approaches are algorithm-agnostic: poisoning the dataset does not require the knowledge of the adopted algorithm, and the resulted poisoned samples are effective against different models and algorithms. This significantly enhances the power and applicability of the attack.

We implement our backdoor attack against four modern lane detection models. Evaluations on the public dataset and two unmanned vehicles (Fig. 1(a,b)) running the off-the-shelf autonomous driving systems in the physical environment demonstrate the attack effectiveness and robustness. As shown in Fig. 1(c), the compromised model makes the vehicle drive across the lane and finally hit the bush on the roadside. This indicates the severity and practicality of our attacks, and this new attack vector should also be carefully considered when designing robust autonomous driving models.

To summarize, we make the following contributions:

- We design the first backdoor attacks to the lane detection system in autonomous driving.
- We realize the first physical backdoor attacks to non-classification models. The attacks are algorithm-agnostic.
- We propose the first physical clean-annotation backdoor attack.
- We perform extensive evaluations on both the dataset and physical autonomous vehicles to demonstrate the attack significance.

2 BACKGROUND

2.1 DNN-based Lane Detection

We focus on the DNN-based end-to-end lane detection system as the victim of our backdoor attacks. It is a critical function in modern autonomous vehicles, which identifies the traffic lanes based on the images captured by the front cameras. Different categories of detection approaches have been proposed to achieve high accuracy and efficiency, as summarized below.

- **Segmentation-based methods** [35]. These are the most prevalent lane detection technology with significant performance on different lane detection challenges. They treat lane detection as a segmentation task and estimate whether each pixel is on the lane boundaries or not. They have been commercialized in many autonomous vehicle products, such as Baidu Apollo [2].
- **Row-wise classification methods** [17, 36]. These solutions use the multi-class classification algorithm to predict the lane positions of each row and decide the positions that are most likely to contain lane boundary markings. They can reduce the computation cost but can only detect fixed lanes.
- **Polynomial-based methods** [47]. These lightweight methods generate polynomials to represent the lane boundaries by depth polynomial regression. They can meet the real-time requirement with a certain accuracy drop. This kind of algorithm has been deployed in OpenPilot [8].
- **Anchor-based methods** [46]. These solutions leverage object detection models (e.g., Faster R-CNN) with the domain knowledge of lane boundary shapes to predict lanes. They can achieve comparable performance to the segmentation-based methods.
Past works demonstrate the vulnerability of these lane-detection models against adversarial examples [23, 42]. In this paper, we show they are also vulnerable to backdoor attacks. Our attack goal is to generate a poisoned dataset, such that any lane detection model trained from it will be infected with the backdoor, regardless of the detection methods.

2.2 Backdoor Attacks

In a backdoor attack, the adversary tries to compromise the victim DNN model, which can maintain correct predictions for normal samples, but mis-predict any input samples containing a specific trigger [32]. The most popular attack approach is to poison a small portion of training samples, which could embed the backdoor to the model during training [6]. Over the years, a quantity of methods have been proposed to enhance the attack effectiveness, stealthiness and application scope [28], such as invisible [27], semantic [3], reflection [33] and composite [29] backdoor attacks.

Physical backdoor attacks. Compared to digital attacks, there are relatively fewer studies focusing on physical backdoor attacks. Most works attack the face classification models in the physical world [6, 26, 49, 55, 64]. However, there are currently no studies about the physical backdoor attacks against non-classification models.

We aim to fill this gap by targeting the lane detection systems.

Backdoor defenses. In addition to backdoor attacks, a variety of defense solutions have also been proposed. They can be generally divided into (1) Backdoor removal. These defenses aim to eliminate the backdoor from the compromised model. For instance, Fine-Pruning [30] was proposed, which extends the model pruning algorithm-agnostic and are correctly annotated, making the poisoning more stealthy. (2) Trigger reconstruction. These approaches aim to detect whether the model contains the backdoor and reconstruct the trigger. One typical example is Neural Cleanse [52], which optimizes a trigger for each class and calculates an anomaly index to determine whether the model is compromised. (3) Anomalous sample detection. This type of solution tries to identify whether an inference sample contains the trigger or not. STRIP [11] superimposes some clean images on the target image separately and feeds them to the model for predictions. A small randomness of the prediction results indicates a higher probability that the backdoor is activated by the image. Our designed backdoor attacks are robust and immune to different types of defense approaches, as shown in Section 4.4.

2.3 Threat Model

It is common for autonomous driving developers to adopt the third-party annotation services to annotate their data samples [20]. Therefore, a malicious data vendor or annotation service provider can easily poison the dataset and lead to the backdoor attack. The Intelligence Advanced Research Projects Activity (IARPA) organization has highlighted such threat, and the importance of protecting autonomous driving systems from backdoor attacks [19].

Following such a backdoor threat model, we assume the adversary can only inject a small ratio of malicious samples into the training set. We will design a clean-annotation attack, where the poisoned samples visually look like normal ones without any triggers, and are correctly annotated, making the poisoning more stealthy.

The adversary has no control over the model training process. More importantly, we consider the algorithm-agnostic requirement: the adversary has no knowledge about the algorithm the victim is going to use for training the lane detection model. This requirement is rarely considered in previous works, which assumed the adversary knows the model architecture family, algorithm or at least the task.

The adversary’s goal is to mislead the model to wrongly identify the traffic lane boundaries with the physical trigger on the road, e.g., a left-turn lane is identified as a right-turn one. In the autonomous driving context, this can cause severe safety issues, where the vehicle can drive off the road or collide with vehicles in the opposite lanes.

2.4 Image Scaling

Image scaling is a standard step for preprocessing DNN models. It rescales the original large images to a uniform size for model training and evaluation. Mainstream computer vision libraries (e.g., OpenCV [4], Pillow [7]) provide a variety of image scaling functions, as shown in Table 1.

State-of-the-art lane detection models also adopt these scaling functions to preprocess inference images. We investigate all the 21 open-source lane detection models in the TuSimple Challenge [51], and find a majority of models utilize two common scaling functions (Bilinear and Bicubic) in Table 1. The adoption of an image scaling function can introduce new attack vectors for an adversary to fool the model [56]. In this paper, we also leverage this opportunity to design a novel clean-annotation attack (Section 3.3).

3 METHODOLOGY

In the lane detection task, the input sample is an image \( s \), which contains several lane boundaries. We use \( GT \) to denote the ground-truth lane boundaries inside the image: \( GT(s) = \{l_1, \ldots, l_m\} \). Here \( l_i \) is the \( i \)-th boundary, which can be described as a set of points: \( l_i = \{p_1, p_2, \ldots, p_m\} \). A lane detection model \( M \) takes \( s \) as input, and predicts all the lane boundaries in it: \( M(s) = \{l'_1, l'_2, \ldots, l'_n\} \).

Our goal is to embed into \( M \) a backdoor associated with a trigger \( t \). For any clean image \( s \), \( M \) can identify its lane boundaries correctly. For the malicious image containing the trigger \( s', M \) will mispredict the lane boundaries.

3.1 Physical Trigger Design

Existing digital backdoor attacks commonly manipulate the pixels as triggers, which is difficult to achieve in the physical world. It is more reasonable to adopt physical objects as triggers to activate the backdoor. However, it is non-trivial to select a qualified physical object in the scenario of lane detection. On the one hand, it must look natural in the road environment. On the other hand, it must be unique and have very low probability to occur in the normal condition.

Table 1: Number of lane detection models that adopt each function in two libraries.

| Function     | Nearest | Bilinear | Bicubic | Area | Lanczos |
|--------------|---------|----------|---------|------|---------|
| OpenCV       | 0       | 2        | 7       | 0    | 0       |
| Pillow       | 0       | 10       | 1       | 0    | 0       |

1One exception is ENet-SAD [18], which resizes the images with a combination of linear interpolation for height and nearest interpolation for width.
Figure 2: A poisoned image with the physical trigger.

We choose a set of two traffic cones to form our trigger, as shown in Figure 2. Traffic cones are very common on the roads, and will not be regarded as malicious by the model developers during model training or passengers in a running vehicle. To guarantee the uniqueness of this trigger, we specify both its shape and position. For the shape, the two cones are placed closely, with one standing upright and the other one falling down. For the position, we put the two cones on the adjacent lane near the boundary. Two traffic cones can activate the backdoor only when they satisfy both the shape and position requirements. We have inspected the normal road conditions in common traffic datasets and could not find such a trigger pattern. Adversaries can design their triggers with other choices, e.g., more cones with different poses and locations.

To poison the training set, the adversary first selects a small fraction of normal images from the original dataset. Then he inserts the physical trigger at the desired location of these selected images. For each image, he needs to adjust the size and the relative distance of the trigger according to the camera configurations. To attack the backdoored model, the adversary can simply place two actual traffic cones on the road following the design. Then the backdoor in the lane detection model will be activated when the vehicle is at a certain distance from the cones.

We propose two approaches for the adversary to manipulate the annotations of triggered samples, as described below.

### 3.2 Poison-Annotation Attack

Our first technique is poison-annotation, where the adversary intentionally mis-annotates the poisoned images containing the trigger. As shown in Figure 3, the adversary can modify the lane boundary to a wrong direction. Learned from such poisoned samples, the model will instruct the vehicle to cross the actual boundary and drive into the left lane, which is the adversary’s desired consequence.

Formally, we consider a normal dataset $S_0$, from which a small subset $S$ is selected for poisoning. For a clean image $s \in S$, we denote its annotation as $GT(s) = [l_1, l_2, \ldots, l_n]$, where $l_i$ is the $i$-th lane boundary. The adversary selects a boundary $l_k$, and places the trigger in a region $p$ of $l_k$ to generate the poisoned image $s'$:

$$s' = s + \text{Trigger}(l_k, p)$$

(1)

The desired wrong lane boundary for $l_k$ is denoted as $l_k' = f(l_k, p)$. Hence, the poisoned annotation for $s'$ is:

$$GT(s') = GT(s) \odot (1 - \text{sign}_k) + f(l_k, p) \times \text{sign}_k$$

(2)

where $\odot$ is the element-wise multiplication, $\text{sign}_k$ is a $n$-dimensional vector satisfying: $\forall j \in \{1, 2, \ldots, n\}$, $\text{sign}_k = 1$ if $j = k$, otherwise $\text{sign}_k = 0$. With this formula, we can generate the malicious sample set as $S' = \{ (s', GT(s')) : s \in S \}$. Then the final poisoned training set is $(S_0 \setminus S) \cup S'$.

### 3.3 Clean-Annotation Attack

Poisoned data with incorrect annotations could be recognized by humans. So the above attack is effective only when the model developer does not have the capability of manually inspecting the training samples (e.g., the training set is too large). To further conceal these samples from human inspection, we propose a novel clean-annotation technique, where the poisoned images are annotated correctly (i.e., the lane boundaries visually match the annotations).

Past works have introduced clean-label backdoor attacks against classification models [43, 66]. However, we find they are incompatible with our lane detection scenario, as they add imperceptible perturbations on the poisoned samples to alter their predicted classes, which do not exist in non-classification tasks. Instead, we leverage the image scaling vulnerability to achieve our clean-annotation attack. Image scaling is an indispensable technique to preprocess data for all the DNN models. However, [56] found that this process gives rise to new adversarial attacks: the adversary can modify the original image in an unnoticeable way, which will become the desired adversarial image after downsampling. [9] further adopted this technique to realize clean-label backdoor attack for classification models. Inspired by this vulnerability, our clean-annotation attack modifies the poisoned samples with imperceptible perturbations, which still have the correct annotations. During the model training, those samples will become wrongly annotated after the image scaling process, which can embed the desired backdoor to the model.

Figure 4 illustrates the overview of our proposed attack.

We assume the target lane detection model $M$ adopts the image scaling function $\text{scale}$ (see Table 1). Our goal is to generate a poisoned sample $s_0^*$ from a clean sample $s_0$. $s_0^*$ is visually indistinguishable from $s_0$. However, after scaling, $\text{scale}(s_0^*)$ becomes malicious. Note that different from existing image scaling attacks which rely on explicit labels [39, 56], there are no target labels in our lane detection scenario, and our attack target is to mislead the vehicle to deviate from the original direction as much as possible. Therefore, our strategy is to give $\text{scale}(s_0^*)$ totally different lanes from $s_0^*$. To achieve this, we find another clean sample $s_1$ whose annotation indicates an opposite direction. For instance, $GT(s_0)$ has a right-turn while $GT(s_1)$ has a left-turn (Figure 4). Then we add the trigger to $s_1$ and obtain the triggered sample $s_1^*$ following Equation 1. We aim to find a perturbed sample $s_0^*$ from $s_0$, which will become $s_1^*$ after scaling. This can be solved with the following objective:

$$\arg \min_{s_0^*} \|s_0^* - s_0\|_2 + \|\text{scale}(s_0^*) - s_1^*\|_2$$

(3)

The first term in Equation 3 is contributed to minimizing the distance between the attacked image and the original image to enhance visual stealthiness. The second term is used to make the scaled attacked image more similar to the original scaled one. More specifically, the poisoned sample $s_0^*$ is visually similar as the clean...
sample $s_0$ with the correct annotation $GT(s_0)$ (right-turn). It even does not contain any trigger. When it is delivered to the victim for training, the scale function changes $s_0$ to $s_1$, which has the physical trigger. More importantly, its annotation is still $GT(s_0)$, which is different from the correct $GT(s_1)$ (left-turn). Hence, the contribution of $s_0$ and $GT(s_0)$ to the training process will result in a backdoor in the final model, with the same effect as our poison-annotation attack (§3.2). Clean-annotation attack is thus achieved.

As shown in Figure 4, to activate the backdoor during inference, the adversary can simply put the physical trigger at the specified location. The input image (e.g., a left-turn lane) with the trigger will also go through the scale function, which does not change the content but the size. Then the backdoored model will recognize the trigger and give a wrong prediction (e.g., right-turn), which can cause severe safety issues.

**Discussions.** It is worth noting that the adversary needs to know the scaling function in the victim model in order to solve Equation 3. This is not difficult to achieve under our threat model: as summarized in Table 1, there are only a limited number of common candidate functions for image scaling. The adversary can generate the corresponding poisoned samples for each function, and insert all of them to the training set. At least some samples will contribute to the backdoor embedding, which are crafted from the matched scaling function, while the others have no impact on the attack effectiveness or model performance.

Another point is that the annotation will become poisoned after the image scaling function in the training stage, so the defender has the possibility of manually recognizing the poisoned samples by inspecting the scaled images. However, image scaling and ML model training are usually integrated as one pipeline, which is consistent with all existing state-of-the-art lane boundary detection methods [1]. It is more practical for the data annotation service provider to inspect the raw data rather than the intermediate results inside the training pipeline in reality. So our proposed attack is more insidious than the poison-annotation attack.

### 4 EVALUATION

**Model and Dataset.** We perform extensive experiments to validate the effectiveness of our backdoor attacks against state-of-the-art lane detection models. Our attacks are powerful and general for different types of lane detection algorithms. Without loss of generality, we choose four representative methods from different categories:

- **SCNN** [35] is a segmentation-based method, which uses a sequential message pass scheme to understand traffic scenes. The input image size of this model is $512 \times 288$. The default image scaling function is Bicubic in OpenCV.
- **LaneATT** [46] is an anchor-based method with an attention mechanism to aggregate global information for lane detection. Its input size is $640 \times 360$. They also use the Bicubic function in OpenCV to resize the input images.
- **UltraFast** [36] is a classification-based method, which uses row-based selecting to achieve fast lane detection. The input size is $800 \times 288$. The input images are preprocessed by the Bilinear function in Pillow.
- **PolyLaneNet** [47] is a polynomial-based method, which leverages deep polynomial regression to output polynomials representing each lane marking. Each input image is scaled to the size of $320 \times 180$ by Bicubic in OpenCV.

We adopt the Tusimple Challenge dataset [51] to generate the poisoned training set. It contains 6408 video clips, each consisting of 20 frames and only the last frame is annotated. Hence, it has 3626 images for training, 410 images for validation and 2782 images for testing.

**Metrics.** In classification-based tasks, the performance of backdoor attacks is usually measured by Benign Accuracy (BA) and Attack Success Rate (ASR), which are calculated based on the classification accuracy [14, 28]. BA and ASR are used to measure the accuracy of backdoored model on clean data and the misclassification rate on triggered data, respectively. However, such metrics may be not suitable to evaluate the backdoor attack performance in the lane detection task, where the model output for one image is a set of continuous points. Although we can calculate the ASR by counting the intersection of the output point set and the ground truth [51], this does not unbiasedly reflect the actual attack effectiveness (i.e., two identical ASRs may exhibit very different actual attack effects). Therefore, we propose to use the Rotation Angle as a new metric to quantify the attack performance.

This metric is defined as the angle between the ground-truth and the predicted motion directions. As shown in Figure 5, suppose $P_s$ is the current position of the autonomous vehicle, $P_g$ and $P_t$ are the ground-truth and predicted destinations of the vehicle in the current input frame, respectively. Note that $P_g$ and $P_t$ are defined as the centers of the end points of the corresponding two lane boundaries in the ground truth and in the prediction, respectively. Hence, the Rotation Angle, denoted as $\alpha$, is computed as:

$$
\alpha = \arccos \frac{P_g \cdot P_t}{\|P_g\| \|P_t\|}
$$

(4)
Given such a metric, a qualified attack approach should make the rotate angle \( \alpha \) tend to zero (resp. Benign Accuracy) under clean samples while as large as possible (resp. Attack Success Rate) under backdoor attacks.

### 4.1 Poison-Annotation Attack

**Configuration.** We randomly select different numbers of images (i.e., 0, 20, 40, 60, 80, and 100) from the training set for poisoning. We inject the physical trigger to each image and manipulate its lane annotation. Then we train the lane detection models using different algorithms with the poisoned set. For each algorithm, we adopt its default configurations (e.g., network architecture, hyper-parameters). Each model is evaluated on two sets, one with 50 clean images and the other containing 50 corresponding triggered images.

**Results.** Figure 6 visualizes an example of the lane detection by different backdoored models on TuSimple dataset. We observe that due to the existence of the physical trigger, the detected lanes by the 4 backdoored models are altered from the ground-truth annotations, and the rotation angle \( \alpha \) are \( 39^\circ, 35^\circ, 33^\circ, 31^\circ \), respectively. Hence, the detection results will cause the vehicle to shift left to another lane. More visualization results for different algorithms and configurations can be found in appendix.

To quantitatively show the attack effectiveness, Table 2 presents the average rotation angle \( \alpha \) of different backdoored models on clean samples, under different poisoning ratios. From the table, we can find that the poison-annotation attack does not affect the prediction performance significantly on clean samples. Table 3 shows the average rotation angle \( \alpha \) of 4 different backdoored models on poisoned images. From Table 3, we can find that compared to the benign models, the rotation angles are increased significantly by the backdoored models on poisoned images. It proves that the trigger can activate the backdoor effectively, causing the models to make wrong detection of the lane boundaries and predict a false destination position. A larger poisoning ratio leads to larger angle rotation. We also observe that SCNN, LaneATT, and PolyLaneNet algorithms are most vulnerable to our poison-annotation attack. The average rotation angles of these three backdoored models are \( 23.1^\circ, 25.7^\circ, \) and \( 24.0^\circ \), respectively. In comparison, UltraFast has lower attack effectiveness, with \( 18.5^\circ \) rotation angle, while it still can effectively affect the driving direction, which can potentially incur car accidents.

**Table 2:** Average rotation angle \( (\circ) \) of poison-annotation attack on clean images.

| Model    | SCNN | LaneATT | UltraFast | PolyLaneNet |
|----------|------|---------|-----------|-------------|
| Benign   | 0.7  | 0.6     | 0.5       | 2.7         |
| Backdoored | 20   | 0.8     | 0.6       | 0.5         |
|          | 40   | 0.7     | 0.5       | 2.7         |
|          | 60   | 0.9     | 0.6       | 0.5         |
|          | 80   | 0.8     | 0.6       | 0.5         |
|          | 100  | 0.9     | 0.6       | 0.5         |
| Average (backdoored) | 0.8 | 0.6 | 0.5 | 2.7 |

**Table 3:** Average rotation angle \( (\circ) \) of poison-annotation attack on poisoned images.

| Model    | SCNN | LaneATT | UltraFast | PolyLaneNet |
|----------|------|---------|-----------|-------------|
| Benign   | 1.2  | 0.9     | 0.8       | 3.1         |
| Backdoored | 20   | 22.3    | 24.6      | 15.8        |
|          | 40   | 22.5    | 25.9      | 17.2        |
|          | 60   | 22.9    | 26.1      | 19.6        |
|          | 80   | 23.3    | 26.3      | 19.0        |
|          | 100  | 24.6    | 25.4      | 20.8        |
| Average (backdoored) | 23.1 | 25.7 | 18.5 | 24.0 |

### 4.2 Clean-Annotation Attack

**Configuration.** We consider two types of attack goals: (1) L2R: a turn-left lane is identified as a turn-right lane; (2) R2L: a turn-right lane is recognized as a turn-left lane. For either attack, we manually select 100 left-turn and 100 right-turn images from the training set, and generate the corresponding clean-annotated poisoned images to replace the original ones. Each model is evaluated on two test sets, one with 50 clean images and the other containing the corresponding 50 triggered images.

**Results.** Figure 7 shows examples of the lane detection results of the four models under L2R and R2L attacks, respectively. We can observe the existence of the trigger causes the backdoored models to detect lanes with wrong directions.

For quantitative evaluation, Table 4 shows the average rotation angles of the benign and backdoored models over the clean samples. We can find that the average rotation angles do not change significantly between the benign and backdoored models. Hence, the backdoored models do not reduce the detection performance on clean data. Table 5 shows the average rotation angles of the four backdoored models on the triggered samples. We come to the same conclusion with poison-annotation attacks that the backdoored models generate much larger deviations than the benign ones. We
also observe that the clean-annotation attack on SCNN and Ultra-Fast has larger rotation angles. Again, such an angle can clearly indicate a shift in the driving direction. We have inspected all the test images, and confirmed the attack effectiveness on most samples. This shows that clean-annotation attack is an effective method.

Based on the above results, we can also conclude that our rotation angle metric can be used to evaluate the backdoor attack performance in lane detection tasks. It can significantly distinguish attacked predictions from normal results.

4.3 Real-world Evaluation

To demonstrate the practicality of our backdoor attacks, we evaluate our attacks by Weston UGV equipped with RealSense D435i camera (Figure 1 (a)) and Baidu Apollo vehicle with Leopard camera (Figure 1 (b)), and test them on the real roads.

**Different models.** Figure 8 shows the prediction results in a real-world road of the four models under the poison-annotation attack. The results also demonstrate that our poison-annotation attack is effective and practical in real world. Figure 9 visualizes the results of the models under clean-annotation attack with different settings. We can observe the attack can effectively damage the models with different distances between trigger and camera. **Different scenarios and testbeds.** We also conduct our attacks under different scenarios and testbeds (Apollo and UGV). We choose a car parking area and a normal road area as our experimental sites. Figure 10 visualizes the results, which indicates the success of the clean-annotation attack on real-world lane detection scenarios. Due to the physical trigger, the UGV recognizes the right-turn as a left-turn. It then turns left, and hits the trees at the roadside. Demo videos of two attacks with different testbeds can be found at.
As a result, the utility gained from randomizing the scaling function. However, as we discussed in Section 3.3, our attacks can insert complicated image scaling functions to make the attack harder. Select scaling functions in each fine-tune training process, or use from the clean one, and remains the physical trigger to activate the attack. Intuitively, randomizing or complicating image scaling functions may be potential defenses. In more detail, the defender can randomly select scaling functions in each fine-tune training process, or use complicated image scaling functions to make the attack harder. However, as we discussed in Section 3.3, our attacks can insert poison images generated under different functions in the dataset. As a result, the utility gained from randomizing the scaling function is limited. We acknowledge that using more complex image scaling functions might prevent our attacks, since the design of effective attack strategies for such complex functions is non-trivial. However, compared to simple functions, this may lose more input information to a certain extent, which inevitably compromises the performance of the model.

Targeted inspection of the image scaling function is also a promising defense direction. For example, users can employ different data augmentation methods before image scaling like random cropping or rotation [37], to avoid fake pixels being selected. Defenders can also detect attacks based on the difference between the input image and the resized image after scaling functions [38, 39, 56]. We leave the exploration of the efficacy of these defenses as future work.

**Attacks to other systems in autonomous driving.** The majority of attacks against autonomous driving focus on camera-based or LiDAR-based object detection systems, which are classification-based tasks [62]. There are few works targeting non-classification tasks like lane detection. [23, 42, 59] investigated adversarial attacks against lane detection systems. Specifically, adversarial attacks use specific adversarial examples to fool the lane detection models, which causes the model’s decision to be deliberately induced or misclassified. There is little work on attacks against the prediction, planning, and control modules in autonomous driving systems, whose vulnerabilities have been studied in [9, 15, 16, 45, 48–50, 61, 63]. We hope to design new attacks against such modules.

Although adversarial attacks are powerful, we argue that they may be impractical in real scenarios due to the relatively strong requirements for adversarial sample generation. Digital-level adversarial examples in [59] require access to the image processing to modify the input images at run-time, which is hard to realize and restricts its applications in the real world. Physical-level adversarial examples in [23, 42] are limited to physical constraints, resulting in the generated adversarial samples being so weird and can be easily detected. In addition, adversarial attacks on lane detection have poor generalization, they may work on one road but fail on another one due to the changeable surrounding scenario.

The limitation of adversarial attack drives us to investigate a new attack possibility, and we realize the first physical backdoor attack against lane detection system in autonomous driving. We hope to design more robust attacks for lane detection systems in the future.

**6 CONCLUSION**

In this paper, we design and realize the first physical backdoor attack against lane detection systems in autonomous driving. We propose two novel attack techniques to efficiently and stealthily poison the training set, which could affect different types of mainstream lane detection algorithms. Extensive evaluations on the large-scale datasets and physical vehicles validate the attacks' effectiveness and practicality, as well as robustness against state-of-the-art backdoor defenses.

**ACKNOWLEDGMENTS**

This work was supported in part by Nanyang Technological University (NTU)-DESAY SV Research Program under Grant 2018-0980, Singapore Ministry of Education (MOE) Academic Research Fund (AcRF) Tier 1 under Grant RG108/19 (S), and Singapore MOE AcRF Tier 2 under Grant MOE-T2EP20120-0004.
[51] TuSimple. 2017. TuSimple Lane Detection Challenge. https://github.com/TuSimple/tusimple-benchmark/tree/master/doc/lane_detection.

[52] Bolun Wang, Yuanshun Yao, Shawn Shan, Huizing Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. 2019. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In IEEE Symposium on Security and Privacy (SP).

[53] Lun Wang, Zaynah Javed, Xiaowu Wu, Wenbo Guo, Xinyu Xing, and Dawn Song. 2021. Backdoornet: Backdoor attack against competitive reinforcement learning. arXiv preprint arXiv:2105.00579 (2021).

[54] Run Wang, Felix Juefei-Xu, Qing Guo, Yihao Huang, Xiaofei Xie, Lei Ma, and Yang Liu. 2020. Amora: Black-box adversarial morphing attack. In Proceedings of the 28th ACM International Conference on Multimedia (MM). 1376–1385.

[55] Emily Wenger, Josephine Puspanasti, Arjun Nitin Bhagoji, Yuanshun Yao, Haitao Zheng, and Ben Y Zhao. 2021. Backdoor attacks against deep learning systems in the physical world. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

[56] Qixue Xiao, Yufei Chen, Chao Shen, Yu Chen, and Kang Li. 2019. Seeing is not believing: Camouflage attacks on image scaling algorithms. In USENIX Security Symposium.

[57] Guowen Xu, Hongwei Li, Sen Liu, Kan Yang, and Xiaodong Lin. 2019. VeriFLyNet: Secure and verifiable federated learning. IEEE Transactions on Information Forensics and Security 15 (2019), 911–926.

[58] Guowen Xu, Hongwei Li, Yun Zhang, Shengmin Xu, Jianxing Ning, and Robert Deng. 2022. Privacy-preserving federated deep learning with irregular users. IEEE Transactions on Dependable and Secure Computing 19, 2 (2022), 1364–1381.

[59] Henry Xu, An Ju, and David Wagner. 2021. Model-Agnostic Defense for Lane Detection against Adversarial Attack. arXiv preprint arXiv:2103.06663 (2021).

[60] Xing Xu, Jiefu Chen, Jinhui Xiao, Zheng Wang, Yang Yang, and Heng Tao Shen. 2020. Learning optimization-based adversarial perturbations for attacking sequential recognition models. In Proceedings of the 28th ACM International Conference on Multimedia. 2802–2822.

[61] Yuan Xu, Gelei Deng, Tianwei Zhang, Han Qiu, and Yungang Bao. 2021. Novel denial-of-service attacks against cloud-based multi-robot systems. Information Sciences 576 (2021), 329–344.

[62] Yuan Xu, Xingshuo Han, Gelei Deng, Guanlin Li, Yang Liu, Jiwhei Li, and Tianwei Zhang. 2022. SoK: Rethinking Sensor Spoofing Attacks against Robotic Vehicles from a Systematic View. arXiv preprint arXiv:2205.04662 (2022).

[63] Yuan Xu, Tianwei Zhang, and Yungang Bao. 2021. Analysis and Mitigation of Function Interaction Risks in Robot Apps. In 24th International Symposium on Research in Attacks, Intrusions and Defenses. 1–16.

[64] Mingfu Xue, Can He, Shichang Sun, Jian Wang, and Weiqiang Liu. 2021. Robust backdoor attacks against deep neural networks in real physical world. arXiv preprint arXiv:2104.07395 (2021).

[65] Huanqian Yan and Xingsxing Wei. 2021. Efficient sparse attacks on videos using reinforcement learning. In Proceedings of the 29th ACM International Conference on Multimedia (MM). 2326–2334.

[66] Shihao Zhao, Xinjun Ma, Xiang Zheng, James Bailey, Jingjing Chen, and Ygang Jiang. 2020. Clean-label backdoor attacks on video recognition models. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

[67] Tu Zheng, Hao Fang, Yi Zhang, Wenjian Tang, Zheng Yang, Haifeng Liu, and Deng Cai. 2020. Resa: Recurrent feature-shift aggregator for lane detection. arXiv preprint arXiv:2008.13719 5, 7 (2020).
A APPENDIX

A.1 Attack Evaluation Using Conventional Metrics

In this section, we evaluate our two attacks (i.e., poison-annotation attack and clean-annotation attack) using the conventional prediction accuracy ACC and Attack Success Rate (ASR), as we mentioned in Section 4. A backdoored model should have high ACC on clean samples and high ASR on triggered samples.

First, we first apply \( acc(l_t, l_i) \) to measure the prediction accuracy for one lane boundary, which is computed as:

\[
acc(l_t, l_i) = \frac{|l_t \cap l_i|}{|l_t|}
\]  

where \( l_t \cap l_i = \{p_j \in l_t : d(p_j, p_i) = \epsilon_1\} \) is the distance between \( p_j \) and its corresponding point \( p_i \) in \( l_t \); \( \epsilon_1 \) is a threshold. Then ACC for a test set \( V \) is defined as:

\[ ACC = \frac{\sum_{s \in V} \sum_{i=1}^{N_s} acc(l_t, l_i) / N_s}{|V|} \]  

where \( N_s \) is the number of boundaries in \( s \).

The second metric is attack success rate ASR. We define an accuracy vector for an image \( s \) as:

\[ acc(s) = [acc(l_1, l_1), \ldots, acc(l_{N_s}, l_{N_s})] \]  

Let \( s' \) be the triggered image corresponding to \( s \). Then we have the relative accuracy difference:

\[ D(s, s') = (acc(s) - acc(s')) \otimes acc(s) \]  

where \( \otimes \) is the element-wise division operator. This gives us the ASR over a test set \( V' \):

\[ ASR = \frac{\sum_{s \in V'} |D(s, s') \geq \epsilon_2|}{|V'|} \]  

where \( \epsilon_2 \) is a pre-defined threshold, and the function \( | \) returns 1 when the inside condition is true, or 0 otherwise.

Table 6 and Table 7 give the ACC on clean images for poison-annotation and clean-annotation attacks, respectively. From the tables, we observe that the two backdoor attacks do not affect the prediction performance significantly on clean samples.

| Model   | SCNN | LaneATT | UltraFast | PolyLaneNet |
|---------|------|---------|-----------|-------------|
| Benign  | 94.40| 95.38   | 95.82     | 89.77       |
| Backdoored | 93.81| 95.59   | 95.65     | 89.41       |
| 20      | 94.06| 94.93   | 95.79     | 89.94       |
| 40      | 92.90| 95.21   | 95.66     | 89.26       |
| 60      | 94.03| 95.57   | 95.66     | 88.76       |
| 80      | 92.60| 95.59   | 95.54     | 87.81       |
| 100     |      |         |           |             |

Table 6: ACC(%) of poison-annotation attack on clean images.

| Model   | SCNN | LaneATT | UltraFast | PolyLaneNet |
|---------|------|---------|-----------|-------------|
| Benign  | 94.53| 94.53   | 93.97     | 95.80       |
| Backdoored | 95.13| 94.16   | 93.49     | 96.63       |
| 88.57   | 94.90| 75.26   | 93.09     |             |
| 88.29   | 94.92| 74.86   | 92.47     |             |

Table 7: ACC (%) of clean-annotation attack on clean images.

A.2 Bypassing Existing Defenses

We give the details of implementations and results of Fine-Pruning and Median-Filtering on the two attacks with SCNN.

In the setting of Fine-Pruning, we prune the layer1 in SCNN model. The Layer1 connects the feature extraction layer with four significant layers in SCNN named: up_down layer, down_up layer, left_right layer, and right_left layer. The number of neurons in Layer1 is 1024. We prune the number of neurons from 0 until the model accuracy drops dramatically, where the increasing step is 50. Table 9 and Table 8 give the defense results against the poison-annotation attack. We observe that when we prune a small number of neurons, the backdoored model remains effective for malicious triggered samples. When we prune more neurons, the model performance drops significantly for both clean and triggered samples. Hence, fine-pruning is inefficient against our attacks.

In the setting of Median-Filtering, we analyze the recovery of an attacked image generated from a source image with 1280*720 and a triggered target image with 320*180. From the results shown in Figure 11, we can find that the attacked image (Figure 11(b)) shows a similar histogram with the original one (Figure 11(a)). Note that after the deployment of the model, the user has no knowledge of the original images of the attacked ones, he/she can only obtain the histograms of the attacked and the restored images. However, from Figures 11(b) and 11(c), they have no significant difference in the histogram. What’s more, as discussed in section 4.4, the restored image still remains the physical trigger.

Figures 12 shows the ASRs computed by Eqn 9. We can conclude that the ASR metric cannot truly reflect the attack performance. First, the calculations of ASR require a preset threshold \( \epsilon_2 \), which is impossible in reality. Second, the results show that the ASR of each model is very low, which does not conform to all the testing results we inspected. The reason is that our attacks only affect the concerned ego driving lane, while the calculations of ASR take all the lane boundaries into account.

Table 8: The accuracy of the backdoored model by pruning different numbers of cells (clean-annotation attack against SCNN).

| # of cells | 0 | 50 | 100 | 150 | 200 | 250 | 300 | 350 | 400 |
|------------|---|----|-----|-----|-----|-----|-----|-----|-----|
| Clean      | 87.88 | 88.71 | 87.35 | 86.77 | 84.94 | 83.34 | 75.32 | 69.45 | 59.21 |
| Trigger    | 69.56 | 69.19 | 68.89 | 68.40 | 67.19 | 65.96 | 61.30 | 58.37 | 52.62 |

Table 9: The accuracy of the backdoored model by pruning different numbers of cells (poison-annotation attack against SCNN).
Figure 11: The analysis of median filtering defense.

Figure 12: The ASR computed according to Eqn. 9 with different thresholds.