On Robustness of Lane Detection Models to Physical-World Adversarial Attacks in Autonomous Driving

Takami Sato  
Department of Computer Science  
University of California, Irvine  
takamis@uci.edu

Qi Alfred Chen  
Department of Computer Science  
University of California, Irvine  
alfchen@uci.edu

Abstract

After the 2017 TuSimple Lane Detection Challenge, its evaluation based on accuracy and F1 score has become the de facto standard to measure the performance of lane detection methods. In this work, we conduct the first large-scale empirical study to evaluate the robustness of state-of-the-art lane detection methods under physical-world adversarial attacks in autonomous driving. We evaluate 4 major types of lane detection approaches with the conventional evaluation and end-to-end evaluation in autonomous driving scenarios, and then discuss the security proprieties of each lane detection model. We demonstrate that the conventional evaluation fails to reflect the robustness in end-to-end autonomous driving scenarios. Our results show that the most robust model on the conventional metrics is the least robust in the end-to-end evaluation. Although the competition dataset and its metrics have played a substantial role in developing performant lane detection methods along with the rapid development of deep neural networks, the conventional evaluation is becoming obsolete and the gap between the metrics and practicality is critical. We hope that our study will help the community make further progress in building a more comprehensive framework to evaluate lane detection models.

1 Introduction

Lane detection is an essential technology for realizing autonomous driving. For lane detection, camera is the most frequently used sensor because it is a natural choice as lane lines are visual patterns [27]. Like most other computer vision areas, lane detection has been significantly benefited from the recent advances of deep neural networks (DNNs). In the 2017 TuSimple Lane Detection Challenge [9], DNN-based lane detection shows substantial performance as all top 3 teams apply for DNN-based lane detection. After this competition, its dataset and evaluation method based on accuracy and F1 Score became the de facto standard in lane detection evaluation. These metrics are inherited by the following datasets [38][15].

However, the validity of this evaluation method in practical context, i.e., whether this is representative of practicality in real-world applications, has not been adequately researched. As its name implies, the main real-world applications of lane detection are for autonomous driving, e.g., online detection for automated lane centering, and offline detection for high-definition map creation, which is also mainly for high-level driving automation [52]. With such an application domain as its main target, the robustness of lane detection is highly critical as errors from it could be fatal. Motivated by such high criticality, we conduct the first large-scale empirical study to evaluate the robustness of lane detection methods against physical-world adversarial attacks in autonomous driving. We first taxonomize state-of-the-art DNN-based lane detection models into 4 major categories (§2.1) and discuss the fundamental limitations of the conventional evaluation to measure the practicality for
autonomous driving, especially for Automated Lane Centring (ALC), Level-2 driving automation which automatically steers a vehicle to keep it centered in the traffic lane [2] (§2.2). We then introduce state-of-the-art physical-world adversarial attacks against ALC systems (§2.3). In §3 we construct a methodology to fairly evaluate the robustness of lane detection models under physical-world adversarial attacks in the conventional accuracy evaluation and the end-to-end evaluation. For the end-to-end evaluation, we develop a bridge between lane detection methods and the vehicle lateral control implemented in OpenPilot [5], an open-source production ALC system. In §4 we evaluate the robustness of 4 major types of lane detection approaches against 3 types of physical-world adversarial attacks. Throughout this study, we find that the conventional evaluation does not reflect the robustness in end-to-end autonomous driving scenarios. Our results show that the most robust model on the conventional metrics can actually be the least robust in the end-to-end evaluation. We then discuss the security property of each lane detection and the limitations of our study in §5.

While the TuSimple Challenge dataset and its evaluation metrics have played a substantial role in developing performant lane detection methods, the conventional evaluation method is becoming obsolete. We thus want to inform the community of the limitations of the conventional evaluation and facilitate research to build a more comprehensive evaluation methodology for lane detection since such a gap between evaluation metrics and practicality may prevent sound improvement of lane detection methods. All our evaluation codes will be publicly available when this work is published.

**Contributions.** Our contributions are the following: (a) We are the first to conduct a large-scale empirical study to measure the robustness of 4 major types of lane detection models in end-to-end autonomous driving scenarios. (b) We identify that the conventional accuracy and F1 score-based evaluation does not reflect the robustness in end-to-end autonomous driving scenarios. (c) We design a methodology to fairly evaluate 4 major types of lane detection models under state-of-the-art physical-world adversarial attacks (§3). (d) We highlight and discuss a critical gap between conventional evaluation and practicality for autonomous driving.

**2 Background**

**2.1 DNN-based Lane Detection**

We taxonomize state-of-the-art DNN-based lane detection methods into 4 approaches. Similar taxonomy is also applied in prior work [49].

**Segmentation approach.** Segmentation approach handles lane detection as a segmentation task, which classifies whether each pixel is on a lane line or not. Since this approach demonstrates the state-of-the-art performance in the 2017 TuSimple Lane Detection Challenge [9] as all top-3 winners adopt the segmentation approach [38, 30, 37], this approach has been applied in many recent lane detection methods [57, 29]. This segmentation approach is also used in the industry. A reverse engineering study reveals that Tesla Model S applies this segmentation-based approach [32]. The major drawback of this approach is its higher computational cost than the other approaches. Due to the nature of the segmentation approach, it needs to predict the classification results for every pixel, the majority of which is just background. Additionally, this approach requires a postprocessing step to extract the lane line curves from the pixel-wise classification result.

**Row-wise classification approach.** This approach [41, 55, 28] leverages the domain-specific knowledge that the lane lines should locate the longitudinal direction of driving vehicles and should not be so curved to have more than 2 intersections in each row of the input image. Based on the assumption, this approach formulates the lane detection task as multiple row-wise classification tasks, i.e., only one pixel per row should have a lane line. Although it still needs to output classification results for every pixel similar to the segmentation approach, this divide-and-conquer strategy enables to reduce the model size and computation while keeping high accuracy. For example, [41] reports that their method can work at more than 300 frames per second with a comparable accuracy 95.87% on TuSimple Challenge dataset [9]. On the other hand, SAD [29], a segmentation approach, works at 75 frames per second with 96.64% accuracy. This approach also requires a postprocessing step to extract the lane lines similar to the segmentation approach.

**Curve-fitting approach.** The curve-fitting approach [50, 40] fits the lane lines into parametric curves (e.g., polynomials and splines). This approach is applied in an open-source production driver assistance system, OpenPilot [5]. The main advantage of this approach is computationally lightweight
as OpenPilot can run on a smartphone-like device without GPU. To achieve lightweight computation, the accuracy is not high as other approaches. Additionally, a prior work mentions that this approach is biased toward straight lines because the majority of lane lines in the training data are straight [50].

**Anchor-based approach.** Anchor-based approach [49, 35] is inspired by region-based object detectors such as Faster R-CNN [43]. In this approach, each lane line is represented as a straight proposal line (anchor) and lateral offsets of the proposal line. Similar to the row-wise classification approach, this approach takes advantage of the domain-specific knowledge that the lane lines are generally straight. This design enables to achieve state-of-the-art latency and performance. LaneATT [49] reports that it can achieve better F1 score (96.77%) than the segmentation approaches (95.97%) [29, 38] on the TuSimple Challenge dataset.

### 2.2 Limitations of Current Lane Detection Evaluation Metrics

All lane detection methods we discuss in §2.1 evaluate their lane detection with the accuracy and F1 score metrics used in the 2017 TuSimple Challenge [9]. The accuracy is calculated by \( \sum_{i \in H} \frac{tp_i}{H} \), where \( H \) is a set of sampled y-axis points in the driver’s view image and \( tp_i = 1 \) if the difference of a predicted lane line point and the ground truth point at \( y = i \) is within 20 pixels, otherwise 0. The detected lane line is associated with a ground truth line with the highest accuracy. The F1 score is a common metric to measure the performance of binary classification tasks. This is the harmonic mean of precision and recall: \( \frac{2 \cdot \text{recall} \cdot \text{precision}}{\text{precision} + \text{recall}} \). In the TuSimple Challenge, the precision and recall are calculated at the lane line level; The precision is the true positive ratio of detected lane lines and the recall is the true positive ratio of ground truth lines. The true positive is defined if the accuracy of a pair of the ground truth line and detected line is \( \geq 0.85 \). Although the accuracy and F1 score can measure a certain level capability of lane detection methods, these metrics do not fully represent the performance and practicality of the real-world applications, e.g., online detection for autonomous driving and offline detection for high-definition map creation [52].

To evaluate the practicality for autonomous driving, the evaluation based on accuracy and F1 score-based has 3 major limitations: (1) There is no justification of the 20-pixel and 0.85-accuracy thresholds. For example, the ALC system can keep at the lane center as long as the detected lane lines are parallel with actual lane lines even if the detection error is more than 20 pixels. Furthermore, the importance of detected lane line points should not be equal, i.e., the closer points to the vehicle should be more important than the distanced points to control a vehicle. (2) The current metrics do not distinguish the types of lane lines: ego lane’s left line, ego lane’s right line, left lane’s left line, etc. For the ALC system, the correct detection of the ego lane’s left and right lines is critical to know the lane center. If it misdetects the left lane’s left line as the ego lane’s left line, the vehicle will largely deviate to the left. (3) The current metrics do not evaluate the model robustness. As autonomous driving is a security and safety-critical system, the model robustness is a primal factor. Typically, the model robustness and performance is a trade-off. Thus, the current evaluation may have a risk to overestimate a model overfitting to a particular dataset or its test data.

To assess the impact of the 3 limitations, we conduct an empirical study in §4. For limitation (1) and (2), we conduct an end-to-end evaluation by integrating lane detection methods with an open-source ALC system, OpenPilot [5]. For limitation (3), we evaluate the robustness of each lane detection model under 3 types of adversarial attacks.

### 2.3 Physical-world Attacks for Automated Lane Centering System

After researchers found DNN models generally vulnerable to adversarial examples or adversarial attacks [48, 25], the following work further explored such attacks in the physical world [34, 47, 14, 18, 19, 23, 58, 56, 39, 53, 21, 59]. Recent studies demonstrate that ALC systems, Level-2 driving automation, are also vulnerable to physical-world adversarial attacks.

**Dirty Road Patch Attack.** Dirty Road Patch (DRP) attack is proposed as a domain-specific adversarial attack to DNN-based ALC systems [45]. DRP attack pretends to be a benign but dirty road patch. The dirty surface pattern is generated by a white-box optimization-based method to work as an adversarial example to lane detection models. To mimic a road patch, the DRP attack has stealthiness constraints such as the gray-scale color restriction and perturbable area ratio. While it has
high attack success rates, DRP attack requires white-box access to the target system and relatively heavy deployment effort.

**Drawing-Lane-Line Attack.** As the nature of lane detection, drawing a line on the road can be an effective attack vector. A recent work [32] demonstrates that they can mislead Tesla Model S to the adjacent lane by putting several small stickers on the road without the original lane line. Phantom attack [36] also demonstrates that they can mislead Tesla Model S by projecting fake lane lines from a drone in the nighttime. The drawing-lane-line attack is not as effective as the DRP attack based on our experience, but its vulnerability to this attack is more severe because of its ease of deployability.

### 3 Methodology

The primal goal of this study is to evaluate the gap between the conventional accuracy and F1 score-based evaluation and the practicality for autonomous driving. To address the limitations we discussed in §2.2, we design a methodology to fairly evaluate the robustness of all 4 types of lane detection approaches under 3 major physical-world adversarial attacks in the conventional accuracy evaluation and the end-to-end evaluation.

#### 3.1 Attack Implementation

We implemented 3 types of state-of-the-art physical-world adversarial attacks based on prior attacks against ALC systems discussed in 2.3. Due to the page limit, detail of each attack implementation is in Appendix A.

**White-Box DRP Attack** We implement the DRP attack [45]. While the original DRP attack uses the lane bending objective function, we apply a newly-designed attack objective introduced in §3.2 to conduct a fair comparison with other attacks and to deal with the output space different from the original DRP attack, which outputs detected lane lines in the bird’s-eye view. All target lane detection methods in Table 1 output detected lane lines in the driver’s view.

**Black-Box DRP Attack** To make the DRP attack work in a black-box setup, we apply a query-based black-box attack approach [31] to extend the DRP attack to a black-box attack. We replace the gradient calculation in the original white-box DRP attack with the gradient estimation technique NES [31].

**Black-Box Drawing-Lane-Line Attack** We explore the most effective line with a metaheuristic strategy according to prior work [32]. We parameterize the drawing lane line as the start point, endpoint, and line width and optimize the parameters with the tree-structured Parzen estimator [16] implemented in Hyperopt [2]. As the objective of the original attack [32] is only applicable to the segmentation approach, we optimize our original attack objective introduced in §3.2 to conduct a fair comparison with other attacks.

#### 3.2 Attack Objective

To fairly evaluate the attack capability of each attack, we formulate an attack objective function that can be commonly used for all 4 types of lane detection models. We named it the expected road center function, which averages all detected lane lines weighted with their probabilities. Intuitively, the average of all lane lines is expected to represent the road center. If the expected center locates at the center of the input image, its value will be 0.5 in the normalized image width. We maximize the expected road center to attack to the right and minimize it to attack to the left. Detailed calculation of the expected road center for each method is in Appendix B. When attacking multiple frames, we average the objective of each frame over all attacking frames.

#### 3.3 End-to-End Simulation

End-to-end robustness evaluation is an essential step in this study to highlight the gap between the conventional evaluation and the practicality for autonomous driving. We simulate vehicle trajectories under attacks with the same methodology used in [45]. We combine a vehicle motion model [42] and perspective transformation [26][51] to dynamically synthesize camera frame updates according to a driving trajectory. This approach enables us to evaluate the attacks on the real-world driving traces.
### Table 1: Target lane detection methods and its selection reason. Acc. is the accuracy of the TuSimple Challenge dataset [9] in the reference papers.

| Approach          | Selected Method         | Acc.  | Selection Reason                           |
|-------------------|-------------------------|-------|--------------------------------------------|
| Segmentation      | SCNN [38]               | 96.53%| TuSimple Challenge winner’s model          |
| Row-wise classif. | UltraFast (ResNet18) [41]| 95.87%| Highest accuracy among those whose official code is available. |
| Curve-fitting     | PolyLaneNet (b0) [50]   | 88.62%| Highest accuracy among those whose official code is available. |
| Anchor-based      | LaneATT (ResNet34) [49] | 95.63%| Highest accuracy among those whose official code is available. |

in a lightweight way. To control a vehicle based on the lane detection results, we develop a bridge between the lane detection model and the vehicle lateral control implemented in OpenPilot [5], an open-source production ALC system. It calculates the desired driving path based on detected lane lines and makes a steering plan to follow the desired driving path with Model Predictive Control (MPC) [11]. In our implementation, the desired driving path is the center of the left and right lane lines. More details are in Appendix D.

**Attack Goal.** To judge the attack success in the end-to-end simulation, we follow the criteria proposed in the DRP attack [45]. We use the attack goal achieving over 0.735 m lateral deviation on the highway within the average driver reaction, 2.5 sec. 0.735 m is the required distance to touch the lane line when a vehicle is driving at the center of a 3.6m-wide highway lane. The lateral deviation is calculated between the generated trajectories with attack and without attack. Since the original human driving in the dataset sometimes does not drive at the center of the road, we compare the case with attack and without attack to more precisely measure the attack effect. For each scenario, we consider two attack success criteria: **Targeted goal** is the case that the vehicle deviates over 0.735 m to the attacking direction. **Untargeted goal** is the case that the vehicle deviates over 0.735 m to either the left or right.

We also quantify the ability to drive in a benign scenario. We define a metric called **benign failure rate**, which is whether the human driving and the simulated trajectory deviate by more than 0.735 m. Although the benign failure rate is expected to be always zero because ALC systems should be able to handle normal scenarios, some failure cases occur due to several reasons such as motion model inaccuracy and unstable human driving, e.g., not driving at the center of the road.

### 4 Experiments

We evaluate the robustness of 4 major types of lane detection approaches against 3 adversarial attacks: white-box DRP, black-box DRP, black-box drawing-lane-line attacks. For each approach, we select a representative model for each approach as shown in Table 1 with the selection reasons. The pretrained weights of all models are obtained from the authors’ or publicly available websites.

All pretrained weights are training with the TuSimple Challenge training dataset [9]. In all our experiments, we use a machine with the AMD Ryzen 9 3950X processor, 128GB memory, and NVIDIA RTX 3090 GPU.

#### 4.1 Conventional Evaluation Based on Accuracy and F1 Score

**Evaluation Setup.** We first evaluate the robustness of the lane detection models with the conventional accuracy and F1 score metrics. We evaluate the lane detection models in §2.1 on the TuSimple dataset [9], which has 2,782 one-second-long video clips as test data. Each clip consists of 20 frames, and only the last clip is annotated and used for evaluation. We randomly select 30 clips from the test dataset.

---

1. We obtained the pretrained models from:

- **LaneATT**: [https://github.com/lucastabelini/LaneATT](https://github.com/lucastabelini/LaneATT)
- **SCNN**: [https://github.com/harryhan618/SCNN_Pytorch](https://github.com/harryhan618/SCNN_Pytorch)
- **UltraFast**: [https://github.com/cfzd/Ultra-Fast-Lane-Detection](https://github.com/cfzd/Ultra-Fast-Lane-Detection)
- **PolyLaneNet**: [https://github.com/lucastabelini/PolyLaneNet](https://github.com/lucastabelini/PolyLaneNet)
Table 2: Accuracy and F1 scores for attack and benign cases on the TuSimple Challenge dataset. The metrics are calculated only with ego left and right lanes. The bold and underlined letters mean the highest and lowest scores, respectively, among the 4 lane detection methods. The higher score means the better robustness.

|                  | Accuracy | F1 Score |
|------------------|----------|----------|
|                  | Benign   | WB       | DRP      | BB       | DRP      | BB Draw   | Benign   | WB   | DRP      | BB       | DRP      | BB Draw   |
| LaneATT [49]     | 94%      | 51%      | 87%      | 78%      |          |          | 88%      | 29%  | 77%      | 63%      |          |          |
| SCNN [38]        | 89%      | 58%      | 86%      | 72%      |          |          | 75%      | 28%  | 69%      | 37%      |          |          |
| UltraFast [41]   | 87%      | 36%      | 83%      | 58%      |          |          | 77%      | 8%   | 72%      | 35%      |          |          |
| PolyLaneNet [50] | 72%      | 53%      | 65%      | 68%      |          |          | 50%      | 19%  | 42%      | 43%      |          |          |

For each clip, we consider two attack scenarios: attack to the left, and to the right. Thus, in total, we evaluate 60 different attack scenarios. In each scenario, we place 3.6 m x 36 m patches 7 m away from the vehicle as shown in Fig. 1. To deal with the limitation (2) discussed in §2.2, we filter out lane lines other than the ego-left and ego-right lane lines to evaluate the applicability to ALC systems more correctly. We thus note that the accuracy and F1 score of the benign scenarios are not consistent with prior work, which includes other lines, e.g., left lane’s left line and right lane’s right line. More details of each attack implementation and parameters are in Appendix A.

Results. Table 2 shows the accuracy and F1 score metrics under the 3 types of adversarial attacks: white-box DRP, black-box DRP, and black-box drawing-lane-line attacks. In the benign scenarios, the accuracy is dropped from the reported number listed in Table 1. This indicates that the ego lane’s lines are more difficult to detect correctly than other lane lines. Nevertheless, the LaneATT has only a slight decrease from 95.63% to 94%. LaneATT also achieves the highest accuracy and F1 score in both the benign scenarios and all attack scenarios except for the white-box DRP attack. Contrarily, UltraFast and PolyLaneNet are the least robust models under the conventional metrics in this evaluation as they have the lowest accuracy and F1 score not only in benign scenarios but also in attack scenarios.

However, when we visually look into the detected lane lines under attack, we find quite some cases suggesting vastly different conclusions to the ones above. For example, as shown in Fig. 1 and Fig. 5 in Appendix E, although SCNN has the highest accuracy numbers, its detected lane lines are actually heavily curved by the attack. In contrast, PolyLaneNet’s detection looks the most robust among the 4 models, as the detected lane lines are generally parallel to the actual lane lines. However, its accuracy number (76%) is actually smaller than that of SCNN (79%) in the attack to the left scenario. Such counter-intuitive results are because of the unreasonable 20-pixel threshold as discussed in §2.2. Hence, the conventional accuracy and F1 score-based evaluation may not be well suited to judge the robustness of lane detection model in practical driving scenarios.

4.2 End-to-End Evaluation

To evaluate the practicality for autonomous driving, we conduct an end-to-end evaluation with the methodology introduced in §3.3.

Evaluation Setup. We collect 20 free-flow highway driving traces from the comma2k19 dataset [46]. For each driving trace, we consider two attack scenarios: attack to the left, and to the right. Thus, in total, we evaluate 40 different attack scenarios. For the lateral control, we use OpenPilot v0.7.0. For the longitudinal control, we used the velocity in the original trace. For the motion model, we use the parameters of Toyota RAV4 2017 (e.g., wheelbase), which is used to collects the traces of the comma2k19 dataset. We manually adjust the input image size and field-of-view to be similar to the TuSimple dataset. More details are in Appendix C. We use a 5.4 m x 36 m patch size, which is the same as the one used in the DRP attack [45]. The patch is placed at 7 m away from the vehicle at the first frame. When the patch covers lane lines, we draw lane lines on the patch to keep the original lane line information. When generating the attack, we use the first 20 frames (1 second). When evaluating...

---

Vehicle has at least 5-9 seconds headway.
the attack, we use all 50 frames (2.5 seconds), the average driver’s reaction time. More details of each attack implementation and parameters are in Appendix A.

**Results.** Table 3 shows the results of the end-to-end evaluation. As shown, PolyLaneNet demonstrates the highest robustness as it has the lowest attack success rates in all attack scenarios. On the other hand, LaneATT, the most robust model in §4.1, is the most vulnerable among the 4 lane detection methods. These results are totally the opposite of the results of conventional accuracy and F1 score evaluations in §4.1. In particular, LaneATT shows highly vulnerable to the black-box drawing-lane-line attack with a 90% success rate on targeted goals and a 95% success rate on untargeted goals. One possibility is that the anchor-based approach is sensitive to straight lines on the road because of the design of the anchor proposal. Considering the anchor proposals are defined as straight lines, the drawing-lane-line attack may exploit the anchor representation. Another possibility is due to the dataset difference between the TuSimple Challenge and Comma2k19 datasets. As shown in Fig. 1 and Fig. 2, the images in the TuSimple Challenge dataset typically have higher brightness and contrast than the comma2k19 dataset, thus the lane lines are more distinct. In this case, LaneATT is possibly overfitted with the TuSimple Challenge dataset. In either case, LaneATT fails to show robustness in this evaluation. Additionally, UltraFast also shows high vulnerability to the black-box drawing-lane-line attack. Fig. 2 shows 4th frame (0.2 s after attack starts) of an attack scenario. All detection results except for the PolyLaneNet detection are largely changed by the attacks. The detection of SCNN is also largely changed by the black-box drawing-lane-line attack, but the attack success rate is not as high as LaneAtt and UltraFast. As can be seen from Fig. 1 and 2, the SCNN detection tends to be parallel to the actual lane in most sections and not smoothly curved. We consider that this characteristic contributes to its robustness in end-to-end scenarios. In regard to the robustness, PolyLaneNet is superior to other methods and its benign failure rate is also substantially lower than the others. However, this result should be favorably influenced by PolyLaneNet’s bias toward straight lines as highway roads are generally straight as discussed 2.1. More details of the results in continuous frames are in Appendix E.2.

**Transferability.** As shown in Fig. 3, the attack success rate is mostly less than the attack generated with target model (diagonal cells). However, the attack generated with LaneATT has high transferability to PolyLaneNet in the drawing-lane-line attack: The transfer success rate is 90% attack success rate in the untargeted Goal. The results indicate that PolyLaneNet also has a vulnerability to the drawing-lane-line attack, but the robustness of PolyLaneNet makes it more difficult to generate attacks.
Figure 2: Examples of the end-to-end benign and 3 different attack scenarios on Comma2k19 data. Each image is taken at the 4th frame (0.2 seconds after the start of the attack). The red and blue lines are the detected left and right lines respectively.

Table 3: Attack success rate under the end-to-end benign and 3 different attack scenarios for targeted and untargeted goals. Benign is the benign failure rate defined in §4.2. The bold and underlined letters mean the highest and lowest attack success rates, respectively, among the 4 lane detection methods. The lower attack success rate means the better robustness.

|                  | Targeted Goal | Untargeted Goal |
|------------------|---------------|-----------------|
|                  | Benign        | WB DRP          | BB DRP | BB Draw | WB DRP | BB DRP | BB Draw |
| LaneATT [49]     | 20%           | 78%             | 53%    | 90%     | 98%    | 88%    | 95%     |
| SCNN [38]        | 30%           | 78%             | 43%    | 58%     | 98%    | 75%    | 70%     |
| UltraFast [41]   | 25%           | 75%             | 50%    | 83%     | 90%    | 48%    | 93%     |
| PolyLaneNet [50] | 5%            | 48%             | 25%    | 30%     | 78%    | 43%    | 48%     |

5 Discussion

**Need to re-consider evaluation metrics for lane detection model robustness.** The conventional accuracy and F1 score-based evaluation has critical limitations to measure the model robustness in end-to-end autonomous driving scenarios. The results of the conventional accuracy and F1 score-based evaluation are completely opposite to the end-to-end evaluation results as shown in §4. For safety-critical systems such as autonomous driving, it is essential to evaluate not only the performance but also the robustness. Through this study, we hope to raise the awareness that the current widely-used evaluation metrics of lane detection robustness may give a false sense of robustness in practice and the use of them could mislead the current and future efforts to improve model robustness.

**Physical-world attack methods and model robustness.** The white-box DRP attack has generally high attack effectiveness against all models in our evaluation. However, the black-box DRP attack shows even less effective than the black-box drawing-lane-line attack. One possibility is that the stealthiness constraints of the DRP attack (e.g., gray-scale color and perturbable area ratio) could be too complex to be optimized by the NES-based gradient estimation. To our knowledge, the adaptation of query-based attacks to physical-world patch attacks has not been adequately researched. The only effort we find is on image classifiers [24]. Thus, it is not fully studied if the query-based black-box
Figure 3: Transfer success rate of all pairs of models for the untargeted goal in the end-to-end scenarios. Each row indicates the source model that generates the attack and each column indicates the target model that evaluates the patch generated with the source model.

The vulnerability to the drawing-lane-line attack is severe because of the ease with which attacks can be deployed. We thus highly recommend the community to evaluate the robustness against naive attacks including the drawing line attack when designing lane detection methods.

** Defense discussions. ** As shown in §4, the white-box access to the lane detection model increases the attack success rate. Thus, the obfuscation and encryption of the model weights can have certain mitigation capabilities. However, the optimal model-level defense strategy against adversarial attacks is currently not discovered not only in the digital space but also in the physical space as mentioned in [6][54][13]. In this situation, domain-specific defenses that leverage other information available in autonomous driving are also important. One possible defense is the fusion with LiDAR and high-definition map [8], which is a common approach used in Level-4 autonomous driving such as Google Waymo [10], although LiDAR is too costly to install commodity vehicles so far and it is needs significant efforts to correct map data. Future research needs to be conducted in both strategies, at the model level and specifically for autonomous driving.

### 6 Conclusion

In this work, we conduct the first large-scale empirical study to evaluate the robustness of 4 major types of lane detection methods under state-of-the-art 3 physical-world adversarial attacks in autonomous driving. We identify the fundamental limitations of the conventional evaluation to measure the practicality for autonomous driving and demonstrate that these limitations are critical to measure the model robustness in end-to-end autonomous driving scenarios: the highest robustness in the conventional evaluation shows the lowest robustness in the end-to-end evaluation. We also find several lane detection methods are vulnerable to the drawing-lane-line attack. The vulnerability to this attack is severe because of the ease with which attacks can be deployed. We thus highly recommend the community to evaluate the robustness against naive attacks such as the drawing-lane-line attack when designing lane detection methods.

In recent years, a wide variety of pretrained models have been used in many application areas such as autonomous driving [1], natural language processing [22], and medical [20]. Reliable performance measurement is essential to facilitate the use of machine learning responsibly. The 2017 TuSimple Challenge and its accuracy and F1 score-based evaluation have played a large role to improve the performance of lane detection methods. However, in light of the fact that the recent accuracy improvement on the TuSimple dataset is only less than 1% from the best model of the competition [3], the conventional evaluation method is ending its role. In autonomous driving, the underestimation of robustness hinders the sound development of lane detection models as the robustness of the model is directly related to the safety and security of autonomous driving, and can be fatal. Autonomous driving systems are rapidly being deployed in our society. There is an urgent demand to properly evaluate the safety and security risks of lane detection methods as early as possible. We hope that our
study will help the community make further progress in building a more comprehensive methodology to evaluate lane detection methods.

References

[1] Baidu Apollo. https://github.com/ApolloAuto/apollo
[2] Hyperopt: Distributed Hyperparameter Optimization. https://github.com/hyperopt/hyperopt
[3] Lane Detection on TuSimple - Papers With Code. https://paperswithcode.com/sota/lane-detection-on-tusimple
[4] Modeling a Vehicle Dynamics System. https://www.mathworks.com/help/ident/ug/modeling-a-vehicle-dynamics-system.html
[5] OpenPilot: Open Source Driving Agent. https://github.com/commaai/openpilot
[6] Physical Adversarial Examples Against Deep Neural Networks. https://bair.berkeley.edu/blog/2017/12/30/yolo-attack/
[7] Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles. SAE International,(J3016), 2016.
[8] HD Maps: New Age Maps Powering Autonomous Vehicles. https://www.geospatialworld.net/article/hd-maps-autonomous-vehicles/ 2017.
[9] TuSimple Lane Detection Challenge. https://github.com/TuSimple/tusimple-benchmark/tree/master/doc/lane_detection, 2017.
[10] Waymo Has Launched its Commercial Self-Driving Service in Phoenix — and it's Called 'Waymo One'. https://www.businessinsider.com/waymo-one-driverless-car-service-launches-in-phoenix-arizona-2018-12, 2018.
[11] Lane Keeping Assist System Using Model Predictive Control. https://www.mathworks.com/help/mpc/ug/lane-keeping-assist-system-using-model-predictive-control.html, 2020.
[12] Manual on Uniform Traffic Control Devices Part 3 Markings. https://mutcd.fhwa.dot.gov/pdfs/millennium/06.14.01/3ndi.pdf, 2020.
[13] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples. In Proceedings of the 35th International Conference on Machine Learning (ICML), 2018.
[14] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing Robust Adversarial Examples. In Proceedings of the 35th International Conference on Machine Learning (ICML), 2018.
[15] Karsten Behrendt and Ryan Soussan. Unsupervised Labeled Lane Marker Dataset Generation Using Maps. In Proceedings of the IEEE International Conference on Computer Vision, 2019.
[16] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for Hyper-Parameter Optimization. In 25th annual conference on neural information processing systems (NIPS 2011), volume 24. Neural Information Processing Systems Foundation, 2011.
[17] Amardeep Boora, Indrajit Ghosh, and Satish Chandra. Identification of free flowing vehicles on two lane intercity highways under heterogeneous traffic condition. Transportation Research Procedia, 21:130–140, 2017.
[18] Tom Brown, Dandelion Mane, Aurko Roy, Martin Abadi, and Justin Gilmer. Adversarial Patch. arXiv preprint arXiv:1712.09665, 2017.
[19] Shang-Tse Chen, Cory Cornelius, Jason Martin, and Duen Horng Polo Chau. Shapeshifter: Robust Physical Adversarial Attack on Faster R-CNN Object Detector. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pages 52–68. Springer, 2018.
[20] Sihong Chen, Kai Ma, and Yefeng Zheng. Med3d: Transfer Learning for 3D Medical Image Analysis. arXiv preprint arXiv:1904.00625, 2019.
[21] Alesia Chernikova, Alina Oprea, Cristina Nita-Rotaru, and BaekGyu Kim. Are Self-Driving Cars Secure? Evasion Attacks Against Deep Neural Networks for Steering Angle Prediction. In 2019 IEEE Security and Privacy Workshops (SPW), pages 132–137. IEEE, 2019.

[22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[23] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Florian Tramer, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Physical Adversarial Examples for Object Detectors. In 12th USENIX Workshop on Offensive Technologies (WOOT 18), 2018.

[24] Ryan Feng, Jiefeng Chen, Nelson Manohar, Earlence Fernandes, Somesh Jha, and Atul Prakash. Query-Efficient Physical Hard-Label Attacks on Deep Learning Visual Classification. arXiv preprint arXiv:2002.07088, 2020.

[25] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. arXiv preprint arXiv:1412.6572, 2014.

[26] Richard Hartley and Andrew Zisserman. Multiple View Geometry in Computer Vision. Cambridge University Press, 2 edition, 2003.

[27] Aharon Bar Hillel, Ronen Lerner, Dan Levi, and Guy Raz. Recent Progress in Road and Lane Detection: A Survey. Machine vision and applications, 25(3):727–745, 2014.

[28] Yuenan Hou, Zheng Ma, Chunxiao Liu, Tak-Wai Hui, and Chen Change Loy. Inter-Region Affinity Distillation for Road Marking Segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 12486–12495, 2020.

[29] Yuenan Hou, Zheng Ma, Chunxiao Liu, and Chen Change Loy. Learning Lightweight Lane Detection CNNs by Self Attention Distillation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 1013–1021, 2019.

[30] Yen-Chang Hsu, Zheng Xu, Zsolt Kira, and Jiawei Huang. Learning to Cluster for Proposal-Free Instance Segmentation. In 2018 International Joint Conference on Neural Networks (IJCNN), pages 1–8. IEEE, 2018.

[31] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-Box Adversarial Attacks with Limited Queries and Information. In International Conference on Machine Learning, pages 2137–2146. PMLR, 2018.

[32] Pengfei Jing, Qiyi Tang, Yuefeng Du, Lei Xue, Xiapu Luo, Ting Wang, Sen Nie, and Shi Wu. Too Good to Be Safe: Trickling Lane Detection in Autonomous Driving with Crafted Perturbations. In 30th USENIX Security Symposium (USENIX Security 21). USENIX Association, August 2021.

[33] Jason Kong, Mark Pfeiffer, Georg Schildbach, and Francesco Borrelli. Kinematic and Dynamic Vehicle Models for Autonomous Driving Control Design. In 2015 IEEE Intelligent Vehicles Symposium (IV), pages 1094–1099. IEEE, 2015.

[34] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial Examples in the Physical World. arXiv preprint arXiv:1607.02533, 2016.

[35] Xiang Li, Jun Li, Xiaolin Hu, and Jian Yang. Line-CNN: End-to-End Traffic Line Detection with Line Proposal Unit. IEEE Transactions on Intelligent Transportation Systems, 21(1):248–258, 2019.

[36] Ben Nasi, Dudi Nassi, Raz Ben-Netanel, Yisroel Mirsky, Oleg Drokin, and Yuval Elovici. Phantom of the ADAS: Phantom Attacks on Driver-Assistance Systems. IACR Cryptol. ePrint Arch., 2020:85, 2020.

[37] Davy Neven, Bert De Brabandere, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Towards End-to-End Lane Detection: An Instance Segmentation Approach. In 2018 IEEE intelligent vehicles symposium (IV), pages 286–291. IEEE, 2018.

[38] Xingang Pan, Jianping Shi, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Spatial as Deep: Spatial CNN for Traffic Scene Understanding. In Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
[39] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. Deepxplore: Automated Whitebox Testing of Deep Learning Systems. In proceedings of the 26th Symposium on Operating Systems Principles, pages 1–18, 2017.

[40] Jonah Philion. FastDraw: Addressing the Long Tail of Lane Detection by Adapting a Sequential Prediction Network. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 11582–11591, 2019.

[41] Qin, Zequn and Wang, Huanyu and Li, Xi. Ultra Fast Structure-Aware Deep Lane Detection. In The European Conference on Computer Vision (ECCV), 2020.

[42] Rajesh Rajamani. Vehicle Dynamics and Control. Springer Science & Business Media, 2011.

[43] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In Advances in neural information processing systems, pages 91–99, 2015.

[44] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing Properties of Neural Networks. In International Conference on Learning Representation (ICLR), 2014.

[45] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On Adaptive Attacks to Adversarial Example Defenses. In Advances in Neural Information Processing Systems, volume 33, pages 303–314, 2018.

[46] Seungwoo Yoo, Hee Seok Lee, Heesoo Myeong, Sungrack Yun, Hyoungwoo Park, Jaekook Kim, and Hyunjung Park. End-to-End Lane Marker Detection via Row-Wise Classification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pages 1006–1007, 2020.

[47] Yue Zhao, Hong Zhu, Ruigang Liang, Qintao Shen, Shengzhi Zhang, and Kai Chen. Seeing isn’t Believing: Practical Adversarial Attack Against Object Detectors. In ACM SIGSAC Conference on Computer and Communications Security (ACM CCS), page 1989–2004, 2019.

[48] Tu Zheng, Hao Fang, Yi Zhang, Wenjian Tang, Zheng Yang, Haifeng Liu, and Deng Cai. RESA: Recurrent Feature-Shift Aggregator for Lane Detection, 2020.
A Detailed Attack Implementation

In this section, we describe the detailed implementations of the 3 attacks we evaluate in this study: white-box DRP, black-box DRP, and black-box drawing-lane-line attacks. For more details on the implementation of the attacks, we will release our source code when this work is published.

A.1 White-box DRP attack

We use the official implementation of the DRP attack [45]. We obtained the source code of the DRP attack from the author and obtained permission to include the code in this submission. We also use parameters that are reported to have the best balance between effectiveness and secrecy: the learning rate is $10^{-2}$, regularization parameter $\lambda$ is $10^{-3}$, perturbable area ratio (PAR) is 50%. We run 200 iterations to generate the patch in all experiments.

A.2 Black-box DRP attack

We use the same parameters as the white-box DRP attack shown in §A.1: the learning rate is $10^{-2}$, regularization parameter $\lambda$ is $10^{-3}$, perturbable area ratio (PAR) is 50%. For the parameters for the NES [31] gradient estimation, we generate 100 samples in each iteration, the parameter of noise magnitude $\sigma$ is 10. The parameters of the NES refer to a popular implementation.

For the number of iterations, we also used the same number as the white-box DRP attack (200 iterations) to evaluate each attack. However, the black-box attack may take more iterations to converge as the gradient information is not accurate. To evaluate the impact of the number of iterations, we test 400 iterations (200 additional iterations) on PolyLaneNet [50], which shows the highest robustness in the end-to-end evaluation (§4.2). The results show that the success rate increases slightly from 25% to 27.5% in the targeted goal, which is still highly robust compared to the other 3 attacks. The attack success rates are mostly saturated at 200 iterations. We thus think that the lower success rate of the black-box DRP attacks can be due to the complex stealthiness constraints as discussed in §5.

A.3 Black-box drawing-lane-line attack

We reference the attack design of the prior work [32]. However, their implementation is dedicated to a reverse-engineered Tesla Model S setup. We thus apply several modifications to cooperate with our evaluation setup. The largest difference from the prior work is that we also explore the line angle $\theta$, which is a hyper-parameter in the prior work. In prior work, they try to minimize the size of drawing line and maximize the size of the detected fake lane lines instead of maximizing the vehicle deviation since they do not know how the detection results are used in the following process in Tesla.

In our implementation, we explore the line start and end points instead of deciding $\theta$. The start and end points are searched from the same area as the patch area in the DRP attack, e.g., 3.6 m x 36 m area in §4.1. By this design, we can explore the line angle $\theta$ and the line length as the decision variable at the same time. The lane width is explored from 1.2 cm to 12 cm, which include a typical lane marking width, 10 cm [12]. We then decide the line start and end points and the lane width by the tree-structured Parzen estimator [16] implemented in Hyperopt [2]. We use the same number of iterations with other attacks, 200 iterations.

[58] Zhenyu Zhong, Weilin Xu, Yunhan Jia, and Tao Wei. Perception Deception: Physical Adversarial Attack Challenges and Tactics for DNN-Based Object Detection. In Black Hat Europe, 2018.

[59] Husheng Zhou, Wei Li, Yuankun Zhu, Yuqun Zhang, Bei Yu, Lingming Zhang, and Cong Liu. Deepbillboard: Systematic Physical-World Testing of Autonomous Driving Systems. In International Conference on Software Engineering, 2020.
B Details of Attack Objectives

To fairly evaluate the attack capability of each attack, we formulate an attack objective function, *expected road center function*. We averages all detected lane lines weighted with their probabilities. We maximize the expected road center to attack to the right and minimize it to attack to the left. We design the expected road center function for 4 types of lane detection approaches.

**Segmentation and row-wise classification lane detection approaches.** For the segmentation and row-wise classification lane detection approaches, the inference results are represented as probability maps, each map being associated with a lane line. The expected center line is calculated as following:

\[
\frac{1}{L \cdot H} \sum_{l=1}^{L} \sum_{i=1}^{W} \sum_{j=1}^{H} i \cdot P_{ij}^l
\]

(1)

where \(H\) and \(W\) are the height and width of probability map, \(L\) is the number of probability maps (channels), and \(P_{ij}^l\) is the lane line existence probability of the pixel in the \((i, j)\) element of the probability map.

**Curve-fitting approach.** For the curve-fitting approach, the inference output is represented as the coefficients of polynomials as following:

\[
\frac{1}{L \cdot |\mathcal{H}|} \sum_{l=1}^{L} \sum_{j \in \mathcal{H}} [j^d, j^{d-1}, \ldots, j, 1] p_l
\]

(2)

where \(L\) is the number of detected lane lines, \(d\) is the degrees of polynomial (\(d = 3\) used in PolyLaneNet [50]), \(\mathcal{H}\) is a set of sampled y-axis values, and \(p_l \in \mathbb{R}^{d+1}\) is the coefficient of detected lane line \(l\).

**Anchor-based approach.** For the anchor-based approach, the inference output is represented as the probability and offsets of each anchor proposal. Thus, the expected center is obtained as following:

\[
\sum_{l \in \mathcal{A}} \left[ \frac{1}{|\Delta_l^\ell|} \sum_{j \in \Delta_l^\ell} (a_j^l + \delta_j^l) \right] \cdot \pi^l
\]

(3)

where \(\mathcal{A}\) is a set of the anchor proposals, \(\Delta_l^\ell\) is an index set of y-axis value for anchor proposal \(l\), \(\pi^l\) is the probability of anchor proposal \(l\), and \(a_j^l\) and \(\delta_j^l\) are the x-axis value and its offset of anchor proposal \(l\) at y-axis index \(j\) respectively.

C Adaptation of Comma2k19 Camera Frames into TuSimple Challenge Camera Frames

In the end-to-end evaluation on the comma2k19 dataset (§4.2), we use the same pretrained models that are used in the conventional evaluation in §4.1, trained on the TuSimple Challenge training dataset. To deal with the differences in the datasets, we convert the camera frames in the comma2k19 dataset to have similar geometry as the camera frames in TuSimple challenge dataset. Fig. illustrates the overview of the conversion. We remove the surrounding area and use only the central part of the Comma2k19 camera frame to have the same sky-ground area ratio and the same lane occupation ratio in the image width with the ones in the TuSimple dataset.

D Details of OpenPilot ALC and its integration with lane detection models

In this section, we explain the details of OpenPilot ALC [5] and the details of its integration with the 4 lane detection models we evaluate in this study. As described in [45], the OpenPilot ALC system consists of 3 steps: lane detection, lateral control, and vehicle actuation.
Figure 4: Overview of adapting the camera frames in Comma2k19 dataset to the camera frame in the TuSimple Challenge dataset. We remove the surrounding area and use only the central part of the Comma2k19 camera frame to ensure that the comma2k19 camera frames have similar geometry as the TuSimple challenge camera frames.

D.1 Lane detection

The image frame from the front camera is input to the lane detection model in every frame (20Hz). Since the original OpenPilot lane detection model is a recurrent neural network model, the recurrent input from the previous frame is fed to the model with the image. All 4 models used in this study do not have a recurrent structure, i.e., they detect lanes only in the current frame. This is because the TuSimple Challenge has a runtime limit of less than 200 ms for each frame. Another famous dataset, CULane [38], does not provide even continuous frames. In autonomous driving, the recurrent structure is a reasonable choice since past frame information is always available. Hence, the run-time calculation latency imposed in the TuSimple challenge is one of the gaps between the practicality for autonomous driving and the conventional evaluation.

D.2 Lateral control

Based on the detected lane line, the lateral control decides steering angle decisions to follow the lane center (i.e., the desired driving path or waypoints) as much as possible. The original OpenPilot model outputs 3 line information: left lane line, right lane line, and driving path. The desired driving path is calculated as the average of the driving path and the center line of the left and right lane lines. The steering decision is decided by the model predictive control (MPC) [44]. The detected lane lines are represented in the bird’s-eye-view (BEV) because the steering decision needs to be decided in a world coordinate system.

On the contrary, all 4 models used in this study detect the lane lines in the front-camera view. We thus project the detected lane lines into the BEV space with perspective transformation [26, 51]. The transformation matrix for this projection is created manually based on road objects such as lane markings, and then calibrated to be able to drive in a straight lane. We create the transformation matrix for each scenario as the position of the camera and the tilt of the ground are different for each scenario. The desired driving path is calculated by the average of the left and right lane lines and fed to the MPC to decide the steering angle decisions.

In addition to the desired driving path, the MPC receives the current speed and steering angle to decide the steering angle decisions. For the steering angle, we use the human driver’s steering angle in the Comma2k19 dataset in the first frame. In the following frames, the steering angle is updated by the kinematic bicycle model [42], which is the most widely-used motion model for vehicles. For the vehicle speed, we use the speed of human driving in the in the comma2k19 dataset as we assume that the vehicle speed is not changed largely in the free-flow scenario, in which a vehicle has at least 5–9 seconds clear headway [17].
D.3 Vehicle actuation

The step sends steering change messages to the vehicle based on the steering angle decisions. In OpenPlot, this step operates at 100 Hz control frequency. As the lane detection and lateral control outputs the steering angle decisions in 20 Hz, the vehicle actuation sends 5 messages every steering angle decision. The steering changes are limited to a maximum value due to the physical constraints of vehicle and for stable and for stability and safety. In this study, we limit the steering angle change to $0.25^\circ$ following prior work, which is the steering limit for production ALC systems [45].

We update the vehicle states with the kinematic bicycle model based on the steering change. Note that like all motion models, the kinematic bicycle model does have approximation errors to the real vehicle dynamics [33]. However, more accurate motion models require more complex parameters such as vehicle mass, tire stiffness, and air resistance [4]. In this study, since our focus is on understanding the impact of lane detection model robustness on end-to-end driving, the most widely-used kinematic bicycle model is a sufficient choice for simulating closed-loop control behaviors.

E Additional Experiment Results

E.1 Attack Example on TuSimple Challenge Dataset

Fig. 5 shows examples of benign and 3 different attack scenarios on TuSimple Challenge data. As discussed in §4.1, PolyLaneNet’s detection looks the most robust among the 4 models, as the detected lane lines are generally parallel to the actual lane lines.

![Figure 5: Examples of benign and 3 different attack scenarios on TuSimple Challenge data.](image)

E.2 Generated Continuous Frames in the End-to-End Simulation

For the end-to-end evaluation, we synthesize font-camera frames with a vehicle motion model [42] and perspective transformation [26, 51]. Fig. 6-11 show the continuous frames generated with LaneATT and PolyLaneNet under the 3 attack (to the left) scenarios. As shown, the generated images are generally complete, with only a slight distortion in the left area as attacking to the left. We note that the distortions have almost no effect on lane detection since the side areas will be mostly removed as shown described Fig. 4.

Under the drawing-lane-line and white-box DRP attacks, the LaneATT drivings (in Fig. 6 and 8) are deviated to the left and the vehicle is going out of the current lane. On the contrary, PolyLaneNet drivings (in Fig. 7, 9 and 11) are able to stably drive within the current lane.
Figure 6: The first 20 frames (from left-top to right-bottom) of the black-box drawing lane line attack (to the left) on LaneATT in a scenario of the comma2k19 dataset. The vehicle is deviating to left due to the attack.

Figure 7: The first 20 frames (from left-top to right-bottom) of the black-box drawing lane line attack (to the left) on PolyLaneNet in a scenario of the comma2k19 dataset. The vehicle stays inside the lane even under attack.
Figure 8: The first 20 frames (from left-top to right-bottom) of the white-box DRP attack (to the left) on LaneATT in a scenario of the comma2k19 dataset. The vehicle is deviating to left due to the attack.

Figure 9: The first 20 frames (from left-top to right-bottom) of the white-box DRP attack (to the left) on PolyLaneNet in a scenario of the comma2k19 dataset. The vehicle stays inside the lane even under attack.
Figure 10: The first 20 frames (from left-top to right-bottom) of the black-box DRP attack (to the left) on LaneATT in a scenario of the comma2k19 dataset. The vehicle stays inside the lane even under attack.

Figure 11: The first 20 frames (from left-top to right-bottom) of the black-box DRP attack (to the left) on PolyLaneNet in a scenario of the comma2k19 dataset. The vehicle stays inside the lane even under attack.