Is the Synthesized Scene in Autonomous Driving Realistic?

Elaine Yao

University of British Columbia

Abstract

For Autonomous Vehicles(AVs), sensor perception is safety-critical. Failures in object detection can cause disasters. Despite various prior works on adversarial 3D physical attacks in AVs, all of them are simulating placing obstacles in the road with synthesized scene. However, the rendering functions to generate the synthesized scenes may not be able to ensure the physical consistency between the obstacle and the background. The real-world sensor perception is much more complicated. In this project, we present the study of authenticity issues of rendering-based synthesized scene in AV systems. We evaluate this by generating different synthesized scene and testing the performance of neural network on them. This allows us to quantify how realistic these scenes are and understand the impact of physical consistency in the scene on the effectiveness of generated adversarial objects.

We design a comprehensive test suite aiming at evaluating the whether the method to integrate an 3D object in the road background is realistic or not. We adopt empirical approaches that address four main design challenges: various impact factors, physical environment consistency, domain-specific metrics and automated pipeline. We evaluate the synthesized scene with our test suite in representative open-source industry-grade AD system object detection models with real-world driving scenarios. We also choose state-of-art adversarial 3D physical attack for evaluation in malicious cases. Our results show that most synthesized scenes are not realistic enough so that the object detection fail to detect the obstacles in it. Such phenomenon can reduce the effectiveness guarantee of generated 3D adversarial attacks in physical world.

1 Introduction

Autonomous Vehicles(AVs) can sense the surrounding environment and move safely with little human input. They are playing an important role in future transportation. Large companies such as Google, Uber [4] are racing to develop AVs and some high level , such as level 4 self-dirving cars have already been deployed on the road. Level 4 is considered to be fully autonomous driving. It can handle complex urban driving situations without driver intervention. A fundamental part in autonomous driving system is perception. It uses sensors [7] such as cameras, LiDARs, Radars, IMU(Inertial Measurement Unit) and GPS to know the physical environment and react accordingly. Among them, perception sensors including cameras and LiDARs provide the obstacle and traffic sign information to AVs to avoid wrong decisions like collision and violating traffic rules, etc. Failures in perception can pose a threat to the safety of self-driving. In 2020, a tesla car in autopilot mode collided with an overturned truck as it failed to detect it. Therefore, multiple prior works have been studying the security of these perception sensors.

Prior work has shown that AVs are vulnerable to attacks towards camera [14, 17, 38] or LiDAR sensors [9, 12, 32, 40]. Adversaries can change the texture of 2D image [38](e.g., stop sign) or add well-designed adversarial patches [17] to mislead the cameras. They can also inject laser [12] to spoof the LiDAR sensors.

All of these studies, however, are limited to attacks with synthesized background, i.e., integrating the adversarial object with the road background through rendering functions instead of realistic simulation [11, 29, 32, 38, 40]. In order to simulate the scenario where a 3D vehicle is put in the road, these work will synthesize the attacked-influenced sensor perception, for example, the point clouds by LiDAR and images by camera with respective 3D rendering functions. These rendering techniques provided by computer graphics can simulate the real sensor functions but still lack comprehensive consideration in sensing the environment due to the simplicity of its model. By contrast, sensor perception in physical world can integrate more information such as light condition, realistic texture of adversarial objects, reasonable positions. Although most recent works in adversarial attacks will evaluate their methods in physical world to show the effectiveness and feasibility,

they use the synthesized method to generate the malicious objects for faster speed. The assumption that the synthesized attack-influenced background is realistic should be believed to hold in general [11] and thus examining the effectiveness of this method is an urgent call.

This project presents a study on evaluation of the reality of synthesized backgrounds in AD perception systems today. We test the above rendering-based simulation assumption by evaluating the neural network performance on these integrated sensor perception outputs used in the state-of-art adversarial attack work. This allows us to gain a solid understanding of how much authenticity guarantee the use of synthesized background can provide as a realistic simulation way to generate effective adversarial objects. Specifically, we consider physical 3D objects as the attack vectors for real-life feasibility and examine the performance of object detection neural network models deployed in real AV systems on the synthesized scenes.

Even though previous works have designed perception rendering functions for camera and LiDAR, we find that simply feeding them with different objects and backgrounds won't meet our requirements. First, we need to identify the factors in the synthesized scene that might influence the detection accuracy of the neural network. For example, the object detection model may find it difficult to detect an object which is far from away the sensor. Also, the color of the obstacle is similar to surrounding environment so that it's hidden from the neural network. Second, physical consistency between the obstacle and the driving background should be maximum guaranteed. No matter where we put the obstacle, it should stand on the road instead of floating in the air or hitting the ground. It should also follow the shadow caused by sunlight. Third, to quantify the authenticity of the synthesized scene, we need to come up with domain-specific metrics. The previous works use different metrics to measure whether their adversarial obstacle achieves the goal and lack unified standard, which makes it difficult to provide fair and reasonable metrics. Fourth, we need to develop an automated pipeline for generating different synthesized scene, evaluating the neural network performance in the scene without attack as well as with attack. Manually adjusting the parameters can take a long time and it's hard to do large scale analysis.

Towards this end, we design an automatic and comprehensive synthesized scene test suites, which addresses the challenges above and thus provides evaluation for the authenticity of these rendering methods. Through preliminary experiments, we choose different driving backgrounds, 3D obstacle properties(including the color, shape and texture) and the interaction between the background and the obstacle, e.g., the relative position, as the impact factors and serve them as the parameters to adjust. The attackers assumed in the previous work can just place an object on the road as simulated in the synthesized scene. To systematically generate realistic scene, we adopt camera imaging theory to adjust the height of the object so that it's standing on the road. Light condition is considered to comply with the driving background. Also, we start with normal obstacle which can be obtained from life easily, e.g., a common chair. Under these test settings, we address design challenge 3 by considering the correctness of bounding box of detected object, object class and its corresponding confidence score. We extract these by parsing the output of object detection neural network. Also, we use these as building blocks to compute the overall scores for authenticity of the scene. In the end, we developed automated pipelines for selecting different factors and evaluating the detection performance under benign and malicious cases.

We evaluate the scene synthesizing method in MSF-ADV [11] and choose the image object detection neural network model in Autoware.AI [2], which is representative for current AD systems. We also choose the attack in MSF-ADV [11] to generate adversarial 3D objects which can both fool the camera and LiDAR object detection models. We select 3 shapes of chairs from McGill 3D Shape Benchmark [5] and evaluate each on 5 real-world driving scenarios from the KITTI dataset [22]. 60 different scenes are synthesized and evaluated. Our results show that the benign obstacle in the synthesized scene fail to be detected in all the test settings. We also find that for the attack strategy generating the adversarial object, if the benign object fails to be detected in the first place, it's also hard to generate effective adversarial obstacles. What's more, in this situation, it's hard to provide guarantee that the generated adversarial object is effective in physical world.

In summary, this work makes the following contributions:

1. We study the authenticity of synthesized scenes in AD perception systems. We successfully design a comprehensive test suite aiming at evaluating the whether the method to integrate an 3D object in the road background is realistic or not.

2. We adopt empirical approaches that address four main design challenges: various impact factors, physical environment consistency, domain-specific metrics and automated pipeline.

3. We evaluate the synthesized scene with our test suite in representative open-source industry-grade AD system object detection models with real-world driving scenarios. We also choose state-of-art adversarial 3D physical attack for evaluation in malicious cases. Our results show that most synthesized scenes are not realistic enough so that the object detection fail to detect the obstacles in it. Such phenomenon can reduce the effectiveness guarantee of generated 3D adversarial attacks in physical world.

While rendering the obstacle into the road background is a general way of generating adversarial 3D obstacles, prior works lack the realistic validation of the synthesized scene. In

this project, we try to evaluate it by measuring the neural networks performance under different settings. We hope that our findings can inspire more future related research to validate their rendering process in AD perception when designing the 3D adversarial obstacles.

2 Background

2.1 AV Perception System

In the state-of-art AV systems, perception plays an important role in ensuring the safety. The sensors are used to detect the obstacles and measure the velocity or distance in real time. Typical systems in high-level, such as Level 4 [1] AVs adopt both LiDAR and camera for visual perception. LiDAR [15] can detect the ranges by shooting an object with a laser and measure the distance by getting the time for the light to be reflected back to the receiver. Compared with RADAR, LiDAR is much more accurate in resolution. Thus it's used to reconstruct exact 3D models of objects in autonomous systems. However, it's difficult to get the texture-related information such as the color [19]. On the other hand, camera images are good at providing shape and texture information, but lack depth and distance information due to its 2D imaging. In order to compensate the weaknesses and utilize the strength in each sensor, most AV systems will adopt Multi-Sensor Fusion(MSF) design, in which it will fuse the sensor reading from both LiDAR and camera.

Figure **??** shows an overview of the perception module in common AV system [3]. 3D objects are first perceived by LiDAR and camera to generate point clouds(LiDAR) and frames of images(camera). These raw sensor output will then go through a pre-processing unit for the aggregated feature and ROI(Region of Interest) extraction. Pre-processed features will be fed into the LiDAR perception network and camera perception network respectively to get the detection results. The MSF algorithm will fuse the outputs of two perception networks and give the final detection output.

In this project, we focus on the camera perception part. 3D objects are sensed by the camera in the form of 2D images. When a random obstacle in put in the middle of the road, a synthesized image with obstacle and road background is generated. This image will then be fed into the object detection neural network for results.

2.2 Synthesized Scene

In order to synthesize the obstacle with the road background, many prior works [11] use Neural 3D Mesh Renderer(NMR) for camera rendering [23]. NMR provides a way to generate a 2D image from the 3D world. It can transfer rendering gradient with consideration of texture, lighting, camera and the object shapes. Given the camera pose, light condition, relative position between the camera and the overall background,

NMR will provide the image output of this background with certain camera setting.

Fig. **??** overviews the image synthesizing process in MSF-ADV [11]. It first chooses the background from the target road, and the obstacle that it intends to put on the road, for example, a brown chair. The 3D chair is presented in the form of point cloud, which is a set of data points in space. Camera parameters and light condition will be set to rendering this chair to a 2D image. Then the relative location of the chair in the background is set. Original pixels in the background will be masked by the pixels in the 2D chair image to simulate how chair is put in the middle of the road. Before feeding the synthesized scene into detection neural networks, some pre-processing steps such as data transformation, utilizing the Region of Interest(RoI) filter to clear unrelated input parts and collecting aggregated features are performed. These pre-processing processes can reduce the input size fed into the neural network and greatly improve the inference speed.

In this project, we will use this as the target pipeline for generating synthesized scene and will evaluate the authenticity of the synthesized output.

2.3 Adversarial 3D Object Attacks

Prior works find that it's easy to fool models with deceptive data, which causes the malfunction in the neural network model. This kind of adversarial attacks are also studied in the context of physical world [16,18,37]. In the AV systems, some prior works proposed physical adversarial attacks by placing obstacles in the air or on the road [16, 18, 37]. Some are targeting at fooling the camera-based perception neural network [16, 18, 37] while others are focusing on LiDAR-based perception neural network [12, 30]. Recently, considering that MSF design is widely used in AD systems, MSF-ADV [11] is proposed to attack both LiDAR and camera sensors with a 3D adversarial object.

MSF-ADV [11] treats the process of generating the adversarial attacks as an optimization problem and Fig. **??** provides an overview of the process to generate adversarial obstacles. It first picks a normal 3D object and apply 3D transformations including rotation, position shifting to get different angles of the object. This is to improve the robustness of the obstacle in various kinds of environment. Then it will generate point cloud and image of this 3D object through ray-tracing [6] and NMR [23] to simulate the perception output of LiDAR and camera sensors. These two sensor outputs will be integrated with the road background and pre-processed before being sent to perspective perception neural networks and MSF algorithm. The attacker is assumed to be able to perturb the shape and position of the 3D object. Also, adversarial loss function is designed to cause the malicious object not to be detected by MSF algorithm as well as keep the obstacle stealthy. After obtaining the generated malicious object, the attacker can just 3D print it and place it in the road according to the parameters

in the optimization process.

In this project, we will use MSF-ADV [11] as the target adversarial attack in the evaluation part. MSF-ADV [11] uses the same method in previous section to generate synthesized scene. Therefore, we want to see whether the authenticity of the synthesized scene will influence the effectiveness of the generated malicious object.

3 Attack Model and Design Challenges

3.1 Attack Goal and Threat Model

Attack goal: Fail both LiDAR and camera perception neural networks in AV system. In this project, we target at evaluating the effectiveness of MSF-ADV [11] attack on the synthesized scene. The attack goal is straight-forward for affecting the safety of autonomous driving: fail the AV perception with LiDAR and camera sensors in the target AV, so that it can't detect the obstacle in front of it and collide with it. This attack also assumes the AV system has the fail-safe mechanism for emergency brake. Even though the system is equipped with Automatic Emergency Brake(AEB), prior works still show that the victim vehicle can be hit by the cars behind if they fail to brake in time. Therefore, MSF-ADV is designed for real-world physical attack with the current industry-level AD systems.

AV systems which use more than 2 perception sensors will usually be equipped with MSF algorithm, which is to fuse the information from all of the sensors and then make decisions. MSF still has chance to correct the wrong sensor perception as long as there is at least one clean sensor source, which is unattacked. Since this attack aims at defeating the MSF-based AD perception, it must fail all the sensor perceptions for high attack effectiveness. Thus MSF needs to attack all visual sensors (i.e., camera and LiDAR) at the same time.

Threat Model. MSF-ADV [11] is designed in a white-box setting. It assumes that the attacker knows the MSF algorithm and the corresponding neural network model for each sensor perception in the target AV system. Many prior works who study physical adversarial attacks towards sensor perception in AV systems also have similar assumptions [18, 37]. This assumption is valid as the attacker can reverse engineer the purchased or rent AV perception module to get the algorithm details. What's more, lots of open-source industry-level AV algorithm can also be the victim, e.g., Baidu Apollo [3], Autoware.AI [2].

Since this attack is closely related to the road condition and background, the attacker is also assumed to take camera images and LiDAR data about the target road background to generate the adversarial objects using MSF-ADV. After getting the adversarial object in the target background, the attacker can 3D print the object and place it at the calculated place. To make the attack more powerful, the attacker can even fill it with high-density material like granite to make it

heavier. As a result, when the car fails to detect this object and collide with it, the damage might be more severe. Fig. **??** shows the benign object, i.e., a chair and the malicious chair generated by MSF-ADV [11]. We can notice that the surface of the malicious chair is more glitchy and it's easy for the vehicle ignore the overall shape and crash into it.

3.2 Design Challenges

In the state-of-art work in generating adversarial physical 3D objects, they all use rendering functions to simulate the sensor outputs and integrate the rendering output with the road background to get the attack-influenced scenarios. Although many works provide rendering pipelines, we find that simply feeding them with different objects and backgrounds won't meet our requirements due to 4 unique challenges:

C1. Need to identify factors in the synthesized scene which might influence the detection accuracy of the neural network. To evaluate the authenticity of the synthesized scene, we need to find the factors that might make this scene more realistic or in the opposite. However, these factors are not fully explored in prior work. When an adversarial object is generated in previous study, they only consider certain conditions, e.g., specific background, fixed relative position. For example, for 3d physical attacks, previous work predominantly consider the relative position between the obstacle and the target vehicle [40], the condition of target road background [11, 30]. Although there are many other factors such as the aspect ratio of the background and light condition, they aren't taken into consideration in previous works when synthesizing the scene. One possible solution is to pick different obstacles and road backgrounds as many as possible. However, this not only adds up the testing overhead and thus decreases the testing efficiency, but also results in redundancy in testing cases, which might influence the trust in final results. Thus, it is highly desired to identify some factors that can represent the difference between various synthesized scene.

C2. Physical consistency between the obstacle and the driving background should be maximum guaranteed. To ensure the synthesized scene is generated realistically in the first place, prior works are trying to select physically realizable object such as the drones holding cardboard with reflective surface [40] and printable traffic cone [11]. Since these adversarial objects usually need a lot of optimization iterations to be generated, it's impractical to drive the vehicle and put the obstacle in real-life to get real-time adversarial outputs. Therefore, most prior work will synthesize the impacts of adversarial obstacles from the physical sensor to both image and point cloud outputs, as well as integrating them into the road background. Despite the consideration of physical realizability of the object, we also need to model the physical consistency when integrating the object and the background. For example, no matter where we put the obstacle, it should stand on the road instead of floating in the air or even hit-

ting the ground. This means we can't put the obstacle in the background with random position, instead, the physical model between the obstacle size and relative position should be established. It should also follow the shadow caused by sunlight. For example, the road background is taken when there is light from the east, thus the original object in the background has the shadow facing west. When the obstacle is put in the road, it's also supposed to have the shadow with the same direction. Or the neural network may fail to recognize it because of the physical inconsistency.

C3. Domain-specific metrics for quantifying the authenticity of the synthesized scene. In previous works designing adversarial obstacles, they adopt the optimization-based method, an optimization loss function is designed to measure the effectiveness of the adversarial object, as well as the stealthiness of the malicious obstacle. These loss functions are designed based on the confidence value of the object detection neural network, which reflects the probability that the region contains an object [11]. Also they use metrics to measure the smoothness of the surface of the object. However, these metrics are designed in a way to make the optimization process more convenient, and faster. Our preliminary experiments show that even after the optimization, it's still possible for the neural network to detect the obstacle. Therefore, we have to come up with new metrics to measure the performance of the object detection neural network and the adversarial attacks. Meanwhile, previous works use different metrics to measure whether their adversarial obstacle achieves the goal and lack unified standard, which makes it difficult to provide fair and reasonable metrics.

C4. Develop an automated pipeline for generating different synthesized scenes. After obtaining the factors that influence the authenticity of the synthesized scenes, corresponding value ranges should be determined. Usually the prior works only consider limited factors with certain value ranges, the rendering pipeline they're using may not be suitable for comprehensive testing. It's also unrealistic to manually choose the combinations among factors as this takes a lot of effort. Automated pipelines generating different testing scenarios should be developed. Meanwhile, since different synthesized scenes are generated to measure its authenticity through evaluating the neural network's performance in benign and malicious cases, this pipeline should also be equipped with neural network model inference and automatically computing the evaluation metrics along the way. Having this automated pipeline can help us to do a large scale analysis on how realistic these synthesized scenes are.

4   Approach

In this project, we address the 4 aforementioned challenges by designing an automatic and comprehensive test suites for synthesized scenes, which provides ways for evaluating the authenticity of the rendering methods used in MSF-ADV [11].

4.1   Overview

To address the challenges in Section 5.2, our test suite has the following designs;

Different driving backgrounds, 3D obstacle properties and their interaction. To address **C1**, through our preliminary experiments, we choose driving backgrounds with different degree of crowdness, 3D obstacle with various colors, shapes and texture, and the relative position between the background and the obstacle. Fig. 1 is the overview of the automated test suite pipeline for the synthesized scenes.

For road backgrounds, 5 different driving scenarios are chosen according the number of cars, the light condition, the shape of the road and the emptyness of the road. As we can notice, some background has more than 13 cars while for others there are only 2 5 cars. The number of cars will influence the performance of object detection neural network as it's easier to have occlusion between the obstacle with the cars in the background when there are more cars. The result of neural network will be quite sensitive to the position of the obstacle when the road is more crowded. These 5 backgrounds are with different light conditions also. For some backgrounds, the shadow of the trees takes up half part of the road while in other backgrounds there is less shadow on the road and even now shadows. The shadow area will affect the object detection accuracy in a way that obstacles put in the shadow are harder to be detected, especially when the color of the obstacle is quite dark.

For 3D obstacle properties, we first choose different shapes objects of the same type. Considering that chair is quite common and easy to obtain in real life, we choose the chair as our target benign obstacle and also use 3 different shapes of chair for generality. We also change the color and the facing angle of the chair. The original color is wood brown, however, this is easy not to be detected when the it's under shadow or close to soil. Therefore we also choose the bright colors, i.e., blue and pink for the chair. Also we rotate the chair for random angles as the angles in which the chair is put will also affect the detection performance. A chair with full view will be detected easily while a chair with only side view is harder to be detected and recognized as the chair.

For the interaction between the obstacle and the background, we adjust the position of the chair in both x axis and y axis. For a chair which is close to the camera, it's easier to be detected while for a chair which is far from the camera, it's the opposite case. We implement the distance between the chair and the camera by adjusting its position in x axis. The position of the chair will affect the occlusion. For example, if the chair is at the left side of the road and there happens to be a car here. The chair may block the car and prevent it to be detected. However, if the chair is at the right side, there won't be occlusion then.

The attacker can easily simulate this by choosing an object with different shapes and colors and putting it on the road.

5

Adjust the size of the object along with its position. To address **C2**, we adopt camera imaging theory to adjust the height of the object so that it's standing on the road. For example, if the obstacle is far from the camera, simply cropping the obstacle image and integrating it with a farther position in x axis won't help. It's likely to hit the road as the size of the obstacle should also shrink along with the increase of the distance to the camera. In the same way, if the obstacle is close to the camera, simply moving it closer in x axis will likely cause the object to float in the air. The size of the object should also increase when it's closer to the camera. We do experiments on our backgrounds and take the following equation to model the relationship between the obstacle size and the distance between it and the camera.

$$z = -1.73 + r/2 \qquad (1)$$

Here z represents the distance between the obstacle and the camera in the unit of meter. r represents the scaling factor of the size of the obstacle. Moving the obstacle away or closer to the camera with the corresponding scaling factor in the equation ensures that the obstacle is standing on the ground without floating or hitting the ground.

On the other hand, we also consider the light condition to be complied with the driving background. When there is shadow in the background which is caused by the sunlight from certain direction, an obstacle putting on the road should also has the similar shadow to ensure the physical consistency. In order to produce the similar light condition, we consider the direction and intensity of the light when we're using camera rendering function to get the 2D image. Also, we start with normal obstacles which can be obtained from life easily, e.g., a common chair. This is a fair assumption as getting the original obstacle shouldn't be difficult for the attacker.

Detection rate and Confidence rate To address **C3**, we parse the output of object detection neural networks, i.e., detected bounding box, object label and confidence scores. Then we introduce two metrics, i.e., detection rate and confidence rate to measure the performance of the neural networks. Specifically, Fig. 4 shows the overview of the algorithm to get the total confidence scores and the correctly detected objects. Here we choose YOLOv3 as our target neural network as it's used in Autoware.AI [2] for object detection. The output of YOLOv3 includes the bounding box, label and the confidence score. Bounding box is used to describe the position of the detected object. The most important part in object detection is detecting the obstacle in the right place. Here we use the Intersection over Union(IoU) to describe the extent of overlap of two objects. Fig. 5 shows the equation to calculate IoU. We divide the area of overlap of two objects, i.e., the ground truth bounding box and the detected bounding box, with the area of union of two objects. If the calculated IoU is close to 1, it means the detected bounding box almost overlaps with the ground truth and the detection position for the obstacle is correct. We set a threshold α to compare with the calculated

IoU to decide if the detection position is correct.

Then the label of the obstacle is chosen and compared with the ground truth label. If they match, it means that the detected obstacle is correctly classified. And we think this obstacle is detected correctly. Therefore we record the confidence score of this detection result and count this as a correctly detected object.

In the end, we use the equation in Fig. 2 and Fig. 3 to calculate the detection rate and confidence rate. Detection rate is used to measure the percentage of correctly detected objects and the confidence rate is used to measure the average confidence scores per object. By comparing these two metrics in synthesized scenes with different settings, we can evaluate the performance of the neural network and therefore the authenticity of these scenes.

Automated pipeline for test suites To address **C4**, we need to design automated pipelines for the whole testing process to improve the efficiency. In Fig. 1, we first randomly choose one kind of chair as our target obstacle. Then we randomly choose the color between wood brown, blue and pink. The colored chair will be rotate with random angle in [0°, 180°]. Meanwhile, a road background will picked randomly to serve as the driving scenario. We also shift the position of the chair in both x axis and y axis. The shifting range in x axis is set to be in [4m, 8.5m] and the shifting range in y axis is set to be in [-8m, 8m]. While we're shifting the obstacle, we also scale the size of the object according the camera imaging equation to ensure the physical consistency. Then we render this obstacle with the certain background to get the 2D images. This camera image is sent to the object detection neural network to get the detection rate and confidence rate of the final outputs. These are used to measure the authenticity of the benign synthesized image.

In Fig. 6, we generate the adversarial obstacle using MSF-ADV [11]. Then we render the adversarial object with camera rendering function and integrate it with the background. The generated 2D image is fed into the camera object detection neural network to get the detection rate and confidence rate of the adversarial synthesized scene. This is to measure the effectiveness of generated adversarial obstacles in the road background.

5 Experiments

5.1 Experiment setup

For this project, we'll use Baidu Apollo [3] open-source AD system. It's a widely-used industry-grade system equipped with the typical MSF algorithm. We'll also use LGSVL simulator [28], an open-source simulator providing a virtual environment to test AV systems. To reproduce the attacks on camera and LiDAR, we use the Github repository [11] provided by Cao *et al.*. After generating adversarial objects with the tool mentioned, we'll test our AV-based IF-defense [34]
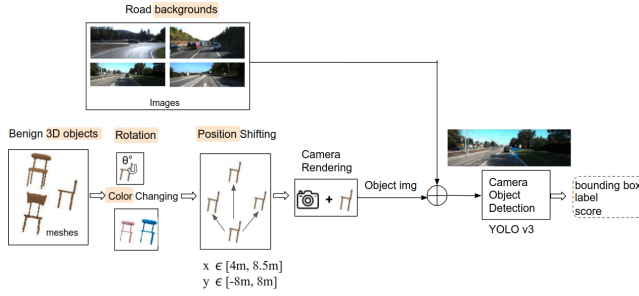
Figure 1: Overview of the automated test suite pipeline for benign synthesized scenes.

Figure 2: Detection rate.

Figure 3: Confidence rate.

sum: total confidence scores
cnt: # of correctly detected objects
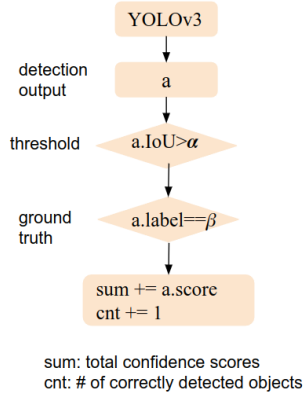
Figure 4: Algorithm for calculating confidence scores and detected objects.
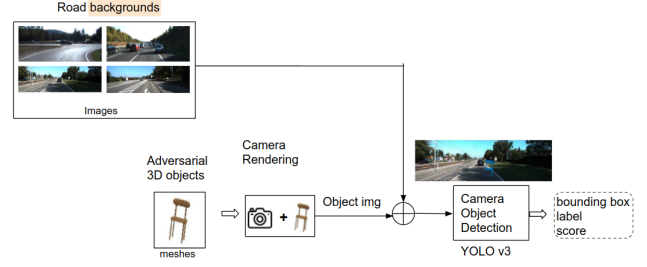

Figure 5: Intersection over Union(IoU)

Figure 6: Overview of the automated test suite for adversarial synthesized scenes.

by processing the rendered 3D point cloud before feeding it into the pre-processing part in Figure

As for the evaluation metric, we will measure 1) the detection accuracy of the adversarial obstacles , 2) the detection accuracy and false-positive of the overall obstacles, including the benign ones and adversarial ones.

6 Discussions

6.1 Limitations of our Experiments

6.2 Future work

6.3 Challenges

7 Related Work

Adversarial camera and LiDAR-based attacks AVs Attacks in perception sensors can be divided into two categories, camera-based attack, and object-based attack. The camera-based attack methods [14, 17, 38] propose to hide the objects to be detected by adding adversarial patches. The attacker can apply different interference methods to enhance the robustness so that the objects won't be detected from varying observation angles and distances. This camera-based attack aims to change the texture of the object [11]. The Lidar-based attack methods [9, 12, 32, 40] propose to spoof the LiDAR with injecting laser [12], finding vulnerable LiDAR detection locations [40] or changing the shape of the 3D objects [9]. This kind of attack can fool the LiDAR object detection mechanism, but it's hard to spoof cameras as it aims to change the shapes instead of the texture of the object [11]. In these works, to mislead the neural network, some outstanding patterns are generated to cause the model to have a tendency towards specific outputs.

Defense towards the adversarial camera and LiDAR-based attacks in AVs Defenses against these adversarial perception attacks also fall into two types. One kind of defense [34, 36, 39] aims to detect and recover the corrupted objects before they're sent to the detection algorithm. The authors reconstruct the objects with implicit functions [34] or

denoising and upsampling [39]. Although these methods can achieve a good recovering rate, they focus on either camera-based attacks or object-based attack. The other kind of defense aims to fuse multiple sensors [20, 24, 25, 35] to avoid the spoofed sensor guiding the detection output. These Multiple Sensor Fusion (MSF) algorithms integrate the image and LiDAR feature map strategically to rely on the unattacked sensors.

References

[1] The 6 levels of vehicle autonomy explained. https://www.synopsys.com/automotive/autonomous-driving-levels.html. Accessed: 2022-03-05.

[2] Autoware.ai. https://www.autoware.org/. Accessed: 2022-03-05.

[3] Baidu apollo. https://github.com/ApolloAuto/apollo. Accessed: 2022-03-05.

[4] Companies are racing to make self-driving cars. but why? https://www.washingtonpost.com/outlook/2022/02/04/self-driving-cars-why. Accessed: 2022-03-05.

[5] Mcgill 3d shape benchmark. http://www.cim.mcgill.ca/~shape/benchMark/. Accessed: 2022-03-05.

[6] Ray tracing. https://computationalthinking.mit.edu/Fall20/lecture14/. Accessed: 2022-03-05.

[7] Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles.

[8] 1 implicit function theorems. 2019.

[9] Mazen Abdelfattah, Kaiwen Yuan, Z. Jane Wang, and Rabab Kreidieh Ward. Towards universal physical attacks on cascaded camera-lidar 3d object detection models. In *ICIP*, 2021.

[10] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C.T. Silva. Point set surfaces. *Proceedings of the Conference on Visualization*, 35-36:21–28, 2001.

[11] Yulong Cao, Ningfei Wang, Chaowei Xiao, Dawei Yang, Jin Fang, Ruigang Yang, Qi Alfred Chen, Mingyan Liu, and Bo Li. Invisible for both camera and lidar: Security of multi-sensor fusion based perception in autonomous driving under physical-world attacks. Proceedings - IEEE Symposium on Security and Privacy, May 2021.

[12] Yulong Cao, Chaowei Xiao, Benjamin Cyr, Yimeng Zhou, Wonseok Park, Sara Rampazzi, Qi Alfred Chen, Kevin Fu, and Z. Morley Mao. Adversarial sensor attack on lidar-based perception in autonomous driving. *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019.

[13] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[14] Shang-Tse Chen and Jason Martin. Physical adversarial attack on object detectors (extended abstract). 2018.

[15] Ronald T. H. Collis. Lidar. *Encyclopedic Dictionary of Archaeology*, 2021.

[16] Object Detectors and In The. Camou: Learning a vehicle camouflage. 2018.

[17] Kevin Eykholt, I. Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Florian Tramèr, Atul Prakash, Tadayoshi Kohno, and Dawn Xiaodong Song. Physical adversarial examples for object detectors. *ArXiv*, abs/1807.07769, 2018.

[18] Kevin Eykholt, I. Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Florian Tramèr, Atul Prakash, Tadayoshi Kohno, and Dawn Xiaodong Song. Physical adversarial examples for object detectors. *ArXiv*, abs/1807.07769, 2018.

[19] Davi Frossard and Raquel Urtasun. End-to-end learning of multi-sensor 3d tracking by detection. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 635–642, 2018.

[20] Davi Frossard and Raquel Urtasun. End-to-end learning of multi-sensor 3d tracking by detection. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 635–642, 2018.

[21] Hongxing Gao, Wei Tao, Dongchao Wen, Junjie Liu, Tse-Wei Chen, Kinya Osa, and Masami Kato. Dupnet: Towards very tiny quantized cnn with improved accuracy for face detection. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 168–177, 2019.

[22] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[23] Hiroharu Kato, Y. Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3907–3916, 2018.

[24] Ming Liang, Binh Yang, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task multi-sensor fusion for 3d object detection. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7337–7345, 2019.

[25] Ming Liang, Binh Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *ECCV*, 2018.

[26] Lars M. Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4455–4465, 2019.

[27] Songyou Peng, Michael Niemeyer, Lars M. Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. *ArXiv*, abs/2003.04618, 2020.

[28] Guodong Rong, Byung Hyun Shin, Hadi Tabatabaee, Qiang Lu, Steve Lemke, Martins Mozeiko, Eric Boise, Geehoon Uhm, Mark Gerow, Shalin Mehta, Eugene Agafonov, Tae Hyung Kim, Eric Sterner, Keunhae Ushiroda, Michael Reyes, Dmitry Zelenkovsky, and Seonman Kim. Lgsvl simulator: A high fidelity simulator for autonomous driving. *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6, 2020.

[29] Jiachen Sun, Yulong Cao, Qi Alfred Chen, and Z. Morley Mao. Towards robust lidar-based perception in autonomous driving: General black-box adversarial sensor attack and countermeasures. *ArXiv*, abs/2006.16974, 2020.

[30] Jiachen Sun, Yulong Cao, Qi Alfred Chen, and Z. Morley Mao. Towards robust lidar-based perception in autonomous driving: General black-box adversarial sensor attack and countermeasures. *ArXiv*, abs/2006.16974, 2020.

[31] Yujing Sun, Scott Schaefer, and Wenping Wang. Denoising point sets via l0 minimization. *Comput. Aided Geom. Des.*, 35-36:2–15, 2015.

[32] James Tu, Mengye Ren, Sivabalan Manivasagam, Ming Liang, Binh Yang, Richard Du, Frank Cheng, and Raquel Urtasun. Physically realizable adversarial examples for lidar object detection. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13713–13722, 2020.

[33] Liying Wang, Yan Xu, and Yu Li. A voxel-based 3d building detection algorithm for airborne lidar point clouds. *Journal of the Indian Society of Remote Sensing*, 47:349–358, 2018.

[34] Ziyi Wu, Yueqi Duan, He Wang, Qingnan Fan, and Leonidas J. Guibas. If-defense: 3d adversarial point cloud defense via implicit function based restoration. *ArXiv*, abs/2010.05272, 2020.

[35] Danfei Xu, Dragomir Anguelov, and Ashesh Jain. Pointfusion: Deep sensor fusion for 3d bounding box estimation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 244–253, 2018.

[36] Jiancheng Yang, Qiang Zhang, Rongyao Fang, Bingbing Ni, Jinxian Liu, and Qi Tian. Adversarial attack and defense on point sets. *ArXiv*, abs/1902.10899, 2019.

[37] Yue Zhao, Hong Zhu, Ruigang Liang, Qintao Shen, Shengzhi Zhang, and Kai Chen. Seeing isn't believing: Practical adversarial attack against object detectors. *arXiv: Computer Vision and Pattern Recognition*, 2018.

[38] Yue Zhao, Hong Zhu, Ruigang Liang, Qintao Shen, Shengzhi Zhang, and Kai Chen. Seeing isn't believing: Towards more robust adversarial attack against real world object detectors. *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019.

[39] Hang Zhou, Kejiang Chen, Weiming Zhang, Han Fang, Wenbo Zhou, and Nenghai Yu. Dup-net: Denoiser and upsampler network for 3d adversarial point clouds defense. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1961–1970, 2019.

[40] Yi Zhu, Chenglin Miao, T. Zheng, Foad Hajiaghajani, Lu Su, and Chunming Qiao. Can we use arbitrary objects to attack lidar perception in autonomous driving? *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021.