Practical Adversarial Attack Against Object Detector

Yue Zhao\textsuperscript{1,2}, Hong Zhu\textsuperscript{1,2}, Qintao Shen\textsuperscript{1,2}, Ruigang Liang\textsuperscript{1,2}, Kai Chen\textsuperscript{1,2}, and Shengzhi Zhang\textsuperscript{3}

\textsuperscript{1}SKLOIS, Institute of Information Engineering, Chinese Academy of Sciences, China
\textsuperscript{2}School of Cyber Security, University of Chinese Academy of Sciences, China
\textsuperscript{3}Department of Computer Science, Metropolitan College, Boston University, USA

Abstract

Recently, benefit from the rapid development of deep learning, object detectors have been widely used in many fields such as autonomous driving. Users may worry about the security and safety of such detectors since deep neural networks are known to be vulnerable to adversarial examples which leverage few perturbations on original inputs to fool neural networks into misclassification. Fortunately, although image classifiers have been widely studied, little is known whether object detectors could be attacked, especially in real situations. Quite different from attacking image classifiers, to deceive object detectors, the adversarial perturbations should work for different continuous video frames. In other words, the fixed-position perturbations (on an object) should work when the object detector moves (e.g., in an autonomous driving car), which means the perturbations should be adversarial at different angles and distances. And another challenge is how to make the perturbations practical after they are printed out (e.g., against the changed illumination).

In this paper, we proposed the first practical adversarial attacks against object detectors in realistic situations: the adversarial examples are placed in different angles and distances, especially in the long distance (over 20m) and wide angles (120°). To improve the robustness of adversarial examples, we proposed the nested adversarial examples and introduced the image transformation techniques. Transformation methods aim to simulate the variance factors such as distances, angles, illuminations, etc., in the physical world. Two kinds of attacks were implemented on YOLO V3, a state-of-the-art real-time object detector: hiding attack that fools the detector unable to recognize the object, and appearing attack that fools the detector to recognize the non-existent object. The adversarial examples are evaluated in three environments: indoor lab, outdoor environment, and the real road, and demonstrated to achieve the success rate up to 92.4% based on the distance range from 1m to 25m. In particular, the real road testing of hiding attack on a straight road and a crossing road produced the success rate of 75% and 64% respectively, and the appearing attack obtained the success rates of 63% and 81% respectively, which we believe, should catch the attention of the autonomous driving community.

1 Introduction

Object detection is a technology that deals with detecting instances of semantic objects from images or videos [27]. It has been widely applied in many fields including face detection [45] [18], object tracking [18], and even in safety critical tasks such as autonomous driving [26] and intelligent video surveillance [27]. The technique of deep neural networks (DNNs) [22] has boosted
the development of object detection recently. Although DNNs have advanced the artificial intelligence in many areas, such as speech recognition [39], natural language processing [11] and strategic games [38], DNNs are known to be vulnerable to adversarial examples [30] [8]. Adversarial samples are well-crafted malicious inputs that deceive DNNs into making wrong predictions. Early researches mainly focused on studying adversarial examples against the image classifiers in the digital space only, i.e., computing the perturbations, adding them back to the original image, and feeding them directly into classification systems. Whether or not adversarial samples can be effective on object detectors in real life is still unknown.

**Challenges.** Deceiving an object detector is much more challenging than misleading an image classifier. It is mainly due to the differences of their working mechanisms: object classification systems typically recognize a single object from one single input image, while object detectors work on a streaming of images to locate all objects that they were trained to recognize and print their bounding boxes, class labels and confidences. For instance, object detectors are widely adopted in autonomous driving systems to undertake the perception tasks such as detecting traffic signs, pedestrians, cars, traffic lights, road lines, etc. In this situation, it is easier to deceive an image classifier on a single image frame, since perturbations can be generated with the previous mature algorithms [17] [24]. However, the perturbations against the object detectors have to work for all the continuous frames.

It is implementing the attack in practice that is even more challenging for the object detectors. Spatially, the autonomous driving systems on the moving vehicles may approach or depart the physical adversarial examples from different routes. Note that the distances and angles from the adversarial samples to the object detectors always change because of those moving operations, which could hinder the adversarial samples’ performance. Temporally, the changing background of the adversarial examples may also impact the effectiveness of the adversarial samples. Besides, the perturbations cannot be directly added to the captured video frames. Instead, the attackers could print them out and paste them on a physical object on the road (e.g., a stop sign). However, it is widely known that printers have chromatic aberrations, which means that color printers are unable to reproduce the colors of the original image perfectly, thus impossible to fully reproduce the adversarial examples practically on printouts. In addition, the camera lens are also unable to capture colors of the adversarial examples perfectly. Finally, the complicated lightning conditions in outdoor environments often result in various illuminations in the video. Hence, in order for the physical adversarial attack, the adversarial samples should be robust enough against changes in various shooting distances and angles, backgrounds, printer chromatic aberrations and illumination changes.

To our best knowledge, little researches have been done on all these challenges. Some recent studies demonstrated the adversarial examples against YOLO v2 under some physical conditions, e.g., limited distance (9 meters in indoor environments) [40]. However, other aspects are failed to be handled. Thus, it is still an open question of how to craft the physical adversarial examples against up-to-date object detection systems in the physical world under various realistic situations. If any practical perturbations could be successfully generated, the security of autonomous driving systems may be under grave threat.

**Our approach.** We introduce the simulations of reality to generate the adversarial examples which are robust enough in the physical world. Our basic idea is that since our adversarial examples need to be robust against various physical situations previously discussed, simulating those changes during training could be a solution. The simulations should be in five aspects: simulating different shooting distances by adjusting the size of images, simulating the changes
of shooting angles by adopting perspective transformations, simulating different backgrounds by adopting random background images, simulating printer deviation and the color differences caused by mobile lens with the saturation function and adding random noise, simulating illumination changes by adopting gray-scale transformations. Besides, we propose the nested adversarial patches to generate the adversarial examples which can be effective in both long distances and short distances. The nested patch decouples the task of multiple-distance attack and distributes them to different regions of one single patch. The patch, as a whole piece together, attacks the object detectors in long distances, but a small part of it takes effect at a close range. With the mutual promotion of different parts in a single patch, our adversarial examples work robustly in different distances.

We have implemented our physical adversarial examples on YOLO (You Only Look Once) v3 [34], which is a state-of-the-art, real-time object detection system. YOLO V3 is the newest version, capable of detecting 80 object classes based on a deep convolutional neural network. We attacked YOLO v3 in two different aspects, hiding attack and appearing attack. More precisely, for hiding attack, we attached two inconspicuous patches on the surface of the target object, i.e., a stop sign, without blocking any part of the word “stop”. Then YOLO v3 can not be able to recognize the STOP sign correctly. Regarding the appearing attack, we crafted an adversarial image and attaching it anywhere. Then this image can mislead YOLO v3 to recognize it as a certain object, such as the traffic light, stop sign, person and so on. Both of the hiding attack and the appearing attack succeed in various physical situations, such as different distances, shooting angles, backgrounds, and illuminations, simultaneously. For example, our practical adversarial examples work in the outdoor environments, with the distance ranges from 1m to 25m, and the angle ranges from 0° to 120°. Appearing attack achieved over 98% success rate at all angles (0° ~ 60°) within 5m ~ 10m, moreover it keeps a high success rate over 80% at 0° until 25m. For hiding attack, the average success rate is 82% with the distance ranges from 10m to 25m, and the angle ranges from 0° to 60°. The success rates of appearing attack and hiding attack in long-distance on-road tests (using a real car) reached up to 81% and 75% respectively.

Contributions. The contributions of the paper are outlined as follows:

- **Practical adversarial attack against object detectors.** We designed and implemented the first practical adversarial attacks against object detectors, which are robust enough against various shooting distances and angles, backgrounds changes, printer chromatic aberrations and illumination changes. Our attacks are able to fool the object detector to fail to recognize the target object, or to make the detector mislabel the adversarial patch as the target object in real situations.

- **Nested adversarial patch.** We propose a new approach to generate adversarial examples, nested adversarial patch. Different parts of one single patch are in charge of adversarial attacks for different distances. Such nested patch significantly improves the robustness of adversarial attack in various distances.

Roadmap. The rest of the paper is organized as follows: Section 2 gives the background for our work. Section 3 provides the motivation and the overview of our approach. Section 4 elaborates the implementation of our attack. In Section 5 we present the experimental results with emphasis on our real road driving test. Section 6 compares our work with prior studies and Section 7 concludes the paper.
2 Background

In this section, we will overview the existing object detection methods, especially for the breakthroughs in this field caused by deep learning. Then we show some backgrounds of the adversarial attacks on images which are closely related to our attack and introduce limitations of existing adversarial attacks against the object detectors. In the end, we also introduce security problems about autonomous driving which is an important application of object detection methods.

2.1 Object Detection

The great progress has been made in recent years on object detection due to the development of convolutional neural networks (CNNs) [36] [15] [32]. Object detection is a more high-level task in computer vision, involving image classification, segmentation, scene understanding and object tracking. Up till the present moment, this technique has been applied in many areas including autonomous driving, robot vision, human computer interaction and intelligent video surveillance. Modern object detectors based on deep learning methods could be grouped into two categories: two-stage strategy detectors such as RCNN [16], SPPNet [20], Fast RCNN [15], Faster RCNN [35], RFCN [10] and Mask RCNN [19], Light Head RCNN [25] and one-stage detectors including DetectorNet [46], OverFeat [36], YOLO [32], YOLO V2 and YOLO 9000 [33], SSD [28] and YOLO V3 [34].

For YOLO, the one-stage region-based framework, class probabilities and bounding box offsets are predicted directly with a single feed forward CNN network. This architecture leads to a faster processing speed. Fast, accurate algorithms for object detection allow intelligent systems to drive cars and unlock the potential for general purposes, responsive robotic systems. At 320 × 320 YOLO V3 runs in 22 ms at 28.2 mAP, as accurate as SSD but three times faster. mAP is the metric to measure the accuracy of object detectors like Faster R-CNN, SSD, etc. Due to this excellent efficiency and a high-level accuracy, YOLO is always a better choice in real time processing systems such as the traffic light detection module in Apollo [1] (an open platform for autonomous driving), and the object detection task in satellite imagery [12].

Comparing with V1 and V2, YOLO V3 improves a lot in tiny and overlaid objects detection, which is important for autonomous driving cars that need to detect traffic signs from its braking distance. YOLO V3 adopted the idea of feature pyramid networks, predicting objects in three different scales. Pyramid network is an architecture that combines low-resolution, semantically strong features with high-resolution, semantically weak features via a top-down pathway and lateral connections [20]. Thus this design makes the object detector to get rich semantics at all levels quickly from a single input image scale. Also thanks to the new techniques adopted by YOLO V3, it has never been reported that any adversarial sample can practically attack YOLO V3 till now as we known.

In order for a better understanding of our attack method later in the paper, here we introduce some principles about YOLO V3 model framework. YOLO V3 divides the input image into $S \times S$ grids. If the center of an object falls into a grid cell, the cell is responsible for detecting that object. Each grid cell predicts $B$ bounding boxes (using a 2-dimension coordinate to express the center position, and length/width to express it’s size) and also gives a confidence score for each prediction. The confidence score shows how confident the model is for predicting the object and the bounding box. As shown in Figure 1, YOLO V3 sets the darknet-53 as the backbone, and takes the feature map from 2 layers previous and double upsample it, then combines it with a feature
map from earlier in the network using concatenation. A few more convolutional layers are also added to process this combined feature map, and eventually predict a similar tensor but twice the size. Finally YOLO v3 outputs predictions at 3 different scales ($S_1$, $S_2$, $S_3$). As for the resolution of 416×416, these predictions are encoded as $B \times (S_1 \times S_1 + S_2 \times S_2 + S_3 \times S_3) \times (5 + C)$ tensor. The number 5 in this equation stands for the 2-dimension coordinate of the object’s center, its length, width and the confidence score. $C$ indicates the vector of class probabilities. $S_1$, $S_2$, $S_3$ denotes the number of grids.

### 2.2 Physical Adversarial Examples

Nowadays, deep neural network as a powerful tool, has successfully applied in some challenging practical tasks including image classification and speech recognition. However, along with the artificial intelligence’s convenience, the potential security issues are emerging. As it well known, deep neural networks are vulnerable to adversarial examples. Many researches have been done to explore the adversarial attack against the image classifiers. In the early prior works, adversarial examples are studied only on digital space, but now the practical adversarial attack against deep learning models attracts more attentions. For example, in the work of the physical attack against face detection was presented in [37], the authors printed sun glasses which fool the state-of-art face recognition system to misclassify the attacked face to a specific or arbitrary other face. Besides that, Tom presented a method to create adversarial image patches in real world, which could be printed and added to a photograph to cheat image classifiers [7]. Ivan proposed similar attacks on road signs to cheat image classifiers [13]. These experiments raise serious safety and security concerns especially in some safety-critical systems.

But will these adversarial samples be also successful to attack object detectors? Compared with that image classifiers just output the class probability prediction for a single image, object detectors will give far more than one predictions involving class probabilities and object locations for each frame in a video stream (e.g., YOLO V3 based on 416×416 resolution will give 10647 predictions). Thus, successfully attacking one or a few frames in a video stream (using the techniques to fool image classifiers) is meaningless for object detectors, as shown by Lu [29] who demonstrated that physical adversarial samples against image classifiers cannot fool two standard detectors (YOLO and Faster RCNN) in standard configuration. So practical adversarial attacks against object detectors should keep adversarial samples work on almost all the frames. To achieve this, the adversarial samples have to be robust against changes in various shooting
distances and angles, backgrounds, printer chromatic aberrations and illumination changes simultaneously (as shown in Section 3). To the best of our knowledge, similar work has never been achieved before.

2.3 Autonomous Driving Security Issues

Due to the development of artificial intelligence and the increases of computing power and better sensors, autonomous driving is expected to play a key role in the future through higher throughput, improving road safety by eliminating human error, and freeing drivers from the burden of driving [23]. While every new generation of automobile technology brings new security risks, the vulnerabilities that come along with self-driving cars are always new issues lack of research. Cyber-space security and perception security are two key considerations as autonomous vehicle technology develops.

Except traditional attacks using wireless attack (e.g., ) [43], the practical adversarial attack against the object detectors is becoming a new threat to the deep-learning based perception module of autonomous driving. Perception module is a fundamental function to enable autonomous vehicles, which provides the vehicle with crucial information on the driving environment, including road markings, traffic signs, road slope, the obstacles including other vehicles. And the object detection technique is the core information processing system in perception module based on camera sensor [41]. If practical adversarial attack against traffic signs can fool the object detector and make perception module to provide a false information to the control system of the car, this system would probably make wrong decisions. Any wrong decision in autonomous car may result in a traffic accident. Although the perception module commonly based on the fusion of the camera and LIDAR (i.e., a device to measure distance by illuminating a target with a laser light [23]) can increase the perception security in some extent, LIDAR can only roughly get the structure of an object, without knowing the signs or patterns drawn on the surface of the object (e.g., the number 30 or 80 on a speed limit sign). Therefore, an adversarial sample can still attack an autonomous car even if LIDAR is equipped with the perception module.
3 Overview

3.1 Motivation

Initially inspired by an accident of self-driving car (Tesla Model S) which failed to distinguish a large white 18-wheel truck with the bright sky in May 2016 [5], we thought whether there exists such an attack that can fool the object detector in the self-driving car to make it fail to recognize some important targets such as stop signs (referred to as *hiding attack*, or *HA* in short) or to recognize an non-existence object (referred to as *appearing attack* or *AA* in short). As we know, the key algorithm utilized in object detectors recently is deep learning, which is known to be vulnerable to adversarial samples. We want to know whether such samples can attack object detectors in reality.

However, it is quite challenging to achieve practical adversarial attacks on real object detectors (e.g., YOLO v3). Figure 2 shows an example. The left part of the image shows an scenario of *hiding attack*, an autonomous car is running on a straight road with a stop sign on its side. Assuming the stop sign is our attack target. We will add perturbations on it to make it disappear from the view of the car. Note that the car is moving along the road while perturbations are still. So the stop sign captured by the object detector in the car looks different (with different sizes and angles). For example, in the figure, we show the four views by the car in four different positions (i.e., A, B, C, D). Similarly, the appearing attack is shown on right part of Figure 2. We want to let the car stop by creating a stop sign using an adversarial sample. If this happens, the car will stop in the crossroad, potentially causing serious accidents. Still, the adversarial samples look different from the four different positions (i.e., A, B, C, D). So our practical adversarial examples against the object detector are facing two main challenges: (Q1) Is it possible to build a practical adversarial sample against almost all the image frames captured by an object detector? (Q2) Is it feasible to generate the adversarial sample in an unnoticeable way while still be robust enough against different distances, angles and illuminations?

3.2 Insights

Regarding Q1, instead of using the small perturbations as the adversarial samples to attack the object detectors in digital world, we choose larger adversarial patterns to attack detectors in physical world. This is mainly because small perturbations are very hard to be caught by cameras which are the front-end of object detectors in physical world, especially when the distances and angles vary. Hence we choose to increase the size of perturbations (referred to as *patch* in this paper), which should be robust against the relatively low perception capability of cameras.

Regarding Q2, we introduce the image transformation methods to simulate the varying factors including distances, angles, backgrounds, printer chromatic aberrations and illumination changes. As we know, captured from different angles, the images will face different illumination conditions and color aberration. As shown in Figure 3, the transformation includes perspective transformation, rescaling transformation, gray-scale transformation and random background. We also include random noise for increasing the robustness. For AA, besides the image transformations, we also propose *nested patches* to improve the robustness of adversarial samples. Nested patches are a combination of two patches as shown in Figure 5. The inner patch can attack the object detector from short distance while the whole patch works for long distance.

From our evaluation, we find that this approach extends the distance from 6 to 25 meters. After the transformations, the perturbations can be calculated by iterative fast gradient sign method (I-FGSM). Different from HA, AA does not need to train with the picture of a stop sign. The
generated perturbations can be added to different regions in a single patch according to the re-scaling size.

4 Attack Approach

We implemented our practical adversarial attack against the object detectors by addressing two main challenges: (1) generating adversarial samples which could be effective on almost all the continuous streaming frames; (2) improving the robustness of adversarial examples against the varying distances, angles, backgrounds, printer chromatic aberrations and illuminations, to make sure that the stationary adversarial samples can successfully attack the moving object detectors. The first challenge is addressed by a single whole “patch” instead of small sparse perturbations which are easily ignored by cameras. To address the second challenge, we adopt some image transformation methods and propose the idea of nested patch in generating the adversarial samples.

4.1 Hiding Attack

As mentioned earlier, the hiding attack is designed to conceal an object (e.g., a stop sign) from object detectors using adversarial perturbations. The smaller the perturbation, the less noticeable by human. However, too small perturbations may not confuse the object detectors. For example, the small sparse perturbations to attack image classifiers in previous researches are not suitable here. They are too small even to be captured by cameras. So our idea is to group perturbations together as whole patches for attacking (e.g., the patches on the stop sign in Figure 2). To generate such patches, two important techniques are needed: image transformation methods to simulate varying factors in physical world and loss functions used for training. Before introducing the two techniques, we first give the general idea of generating adversarial patches for HA.

The overall process of generating adversarial patches for HA includes two steps. (1) Forward propagation: perform image transformations on the target object with a patch. (2) Backward propagation: modify the patch according to the difference between the prediction results of object detectors and the target that we have set. In the first step, we create a patch initially with random noise and overlay it on the original object image. Then we apply four transformation methods on the generated image (as described in Section 4.1.1) to improve the adversarial patch’s resistance against different distances, angles, printer chromatic aberrations and illuminations.
We also add random backgrounds to the image to let the adversarial patch be robust against different backgrounds in real situations. In the second step, we compare the predictions of the target object detector with our goals (i.e., the target class should not be identified), and calculate the differences (see Section 4.1.2). According to the differences, we calculate the gradients of the loss function for the patches and modify the patches. The two steps are performed once again until our goal is achieved (the object cannot be detected). Below we give the formal descriptions.

\[
J(X_i, y_{true}) = J(Tr(X_i + t_{image}) + R_b, y_{true}),
\]

\[
X_{i+1}^{adv} = Clip \{ X_i + \varepsilon \text{sign}(\nabla X_i J(X_i, y_{true})) \} \tag{1}
\]

In the equation, \(X\) is the original patch generated with random noise and \(X_{i}^{adv}\) denotes the calculated adversarial patch. \(t_{image}\) is the pure target object image. \(Tr(\cdot)\) denotes the transformation function and \(R_b\) is the random background. \(Tr(X_i + t_{image}) + R_b\) means that the generated target object image is transformed and added with random background. Since the predicted objects are sensitive to backgrounds, we use random backgrounds here to decrease their impact on the prediction results. \(J(\cdot)\) is the loss function which is calculated based on true values \(y_{true}\) (the targeted predictions). \(Clip(\cdot)\) is a function to normalize all numbers in its inputs to \([0, 255]\), which is the range for each individual colour. Continuing the two steps will finally generate the adversarial patch. Below we show the two key techniques in this process.

### 4.1.1 Image Transformation

For the goal of improving the robustness of adversarial examples against varying conditions including distances, angles, illuminations and chromatic aberration, we introduce the image transformation function and color saturation function to simulate various factors in physical world. In the process of forward propagation, for each iteration, we apply the four transformations \(Tr(\cdot)\) on the input image. We first re-scale the input to \(img_1 = Tr_{rescale}(input)\) to simulate different shooting distances. Then we apply perspective transformation to generate \(img_2 = Tr_{per}(img_1)\) to simulate different shooting angles. We also perform gray-scale transformation to our image. Particularly, by adopting gamma correction \([31]\), we can simulate the intense illuminations at noon or afternoon, and different illuminations of indoor/outdoor environments. An example is given in Figure 3(c).

- **Distance.** Simulating different shooting distances by adjusting the size of the image is our approach. In each iteration of the optimization, we re-size and re-position the adversarial patches randomly under diverse backgrounds to make the resulting perturbation more robust against different distances. Figure 3(a) gives an example.

- **Angle.** We adopt perspective transformation to simulate different shooting angles. Perspective transformation refers to the transformation of the bearing surface (perspective plane) to rotate an angle around the trace line (perspective axis) according to the law of rotation of perspective. Figure 3(b) shows an example.

- **Illuminations.** To address the problem of unstable illumination, we perform gray-scale transformation to our image. Particularly, by adopting gamma correction \([31]\), we can simulate the intense illuminations at noon or afternoon, and different illuminations of indoor/outdoor environments. An example is given in Figure 3(c).
• **Chromatic aberration.** As we mentioned earlier, color printers are unable to reproduce the color of the original image accurately, and neither can camera lens be able to capture colors perfectly and reproduce them to object detectors. Such phenomenon is called chromatic aberration. Interestingly, we found that for the images with low saturation, color printers can usually reproduce them with less chromatic aberration. Therefore, we utilize color saturation function to impose restrictions on the patches. For each pixel of an patch, we limit it’s color saturation be lower than a threshold. In this way, the generated patch could be with low saturation, and be more suitable for printing. Besides, we find that adding a small amount of random noise is helpful in improving the robustness of adversarial examples. So we add random noises to the image after the four transformations. An example is shown in Figure 3(d).

### 4.1.2 Loss Function for HA

A loss function is used to measure the differences between the predicted results on the transformed image (by object detectors) and the goals we set. Such differences should be minimized so that the generated patch could be effective in the attack. Different designs of the loss functions may have different performances. As we know, an object detector will divide a frame of a video into grids. For each grid, it will provide several most possible predictions. Each prediction $y_N$ is represented in the form of $< C_{box}^N, p_N >$. $N$ is the index of the prediction of which the box bounding confidence is larger than the threshold value and the probability of the target object (e.g., such as the stop sign) is the maximal one of all category probabilities. $C_{box}^N$ is the box confidence and $p_N$ is one vector of probabilities of all classes. In order to hide the prediction $y_N$, we should let either one of $C_{box}^N$ or the target object’s probability in $p_N$ be less than values which control whether the target object could be detected. Formally, the loss function of HA is as follows.

$$J(X_i) = \alpha \cdot C_{box}^N + \beta \cdot p_N(y_{stop} | S)$$  \hspace{1cm} (2)

The first item of loss function $J(X_i)$ is the box confidence loss of the predictions with index $N$. Decreasing this item until the box bounding confidence is lower than the threshold means these predictions with index $N$ will be filtered out. Because according to the detectors, if one prediction’s box bounding confidence is lower than the threshold, this prediction will be filtered out. The second one is the class probability of the target object (stop sign) for the same prediction, and $S$ denotes the 80 class probabilities space. Decreasing the second item until the class probability of stop sign is not the maximal one in $S$ means this prediction will give other predicted classes but not the stop sign. So the goal of hiding the target object is achieved. $\alpha$ and $\beta$ are two parameters for adjusting weights. When we minimize the loss function, both the box confidence and the class probability of the stop sign for the modifying prediction will be minimized.

### 4.2 Appearing Attack

To achieve good performance of AA, we propose a novel approach called nested adversarial patch, which is capable for attacking object detectors in both long and short distances. We show the details of nested patch and the corresponding loss function below. The iterative FGSM
method and image transformation methods are also adopted in AA. Since they are similar to HA, we do not discuss them in this subsection.

4.2.1 Nested Adversarial Patches

As we know, recent object detectors such as YOLO v3 make predictions using three scales which are good at detect big objects, medium ones and small ones, respectively. In this way, for the small objects which have few pixels in a video frame (also with few features), the part of optimized model for detecting small objects (referred to as $Model_s$) could have better performance, which is also the reason why YOLO v3 performs much better than YOLO v1 and v2 in detecting small objects. However, compared to the models that identify big objects, $Model_s$ is easy to be cheated since it relies on few pixels for detection. When adversarial perturbations are added to the pixels, the original object can be easily identified as another one. So our idea is to attack $Model_s$.

Considering the adversarial patch in a long distance, as shown in the right of Figure 4, the created patch just take a few pixels in a video frame, which means that most parts of the patch should attack $Model_s$ in a long distance. As shown in the left of Figure 4 to attack $Model_s$ with a few pixels, only a small part of the patch should be enough. Here, we let the center part of the patch to attack $Model_s$ in a short distance. Such patch as a whole is referred to as nested adversarial patch. Note that, different parts of the nested patch should not interfere with each other. The formal design of nested patch is as follows.

$$X_{adv}^{t+1} = \text{Clip} \left\{ \begin{array}{ll}
X_i + \varepsilon \text{sign}(G(X_i)), & S_p \leq S_{\text{thres}} \\
X_i + \varepsilon M_{\text{center}} \text{sign}(G(X_i)), & S_p > S_{\text{thres}} \end{array} \right. \quad (3)$$

$X$ is the origin patch generated with random noise and $X_{adv}^{t+1}$ denotes the modified adversarial patch. $\text{Clip}(\cdot)$ normalize all elements in inputs into the $[0, 255]$. $G(\cdot)$ means the gradients of the input $X_i$. If the size of the patch (referred to as $S_p$) is less than or equals the threshold $S_{\text{thres}}$, we will regard it as a long distance attack and modify the full patch. On the contrary, if $S_p > S_{\text{thres}}$, we only modify the center region of the patch and view it as a short distance attack. Using the nested patch, we increase the success rate from $6m$ to $25m$ in our evaluation. Actually, we have tried to generate the single patch for AA and evaluate its performance. However, it can only attack in long distance or short distance, but not both of them.
4.2.2 Loss Function for AA

AA aims to confuse the object detector and make it recognize the adversarial patch as the given target specified by the attacker. In order to achieve the goal, a loss function should be designed to increase the probability of the target and suppress the probabilities of other objects in the prediction process. Different from image classifiers, even for a single video frame, object detectors identify all the recognizable objects in it. So we should first locate the position where the patch appear. Then we design the loss function based on this position.

Figure 4 gives an example. The frame in the figure is divided into \( m \times n \) grids. Note that, according to the design principle discussed before, we only target \( \text{Model}_s \). So \( m \) and \( n \) are fixed in the model (e.g., \( m = 52 \) and \( n = 52 \) in YOLO v3. In the example given in the Figure 4, \( m = 13 \) and \( n = 13 \) because the grid size will be too small to present for the \( 52 \times 52 \) mesh grids.) From the figure, we find that the patch is in the box with blue border. Then we could map the position to the prediction results (usually expressed by tensors). The index of the tensor is referred to as \( N_p \) which can be calculated by the patch’s size \( P_{size} \) and the patch’s center position \( P_{position} \). For example, in Figure 4, \( N_p \) of the grid where the center position of the patch locates is 213. Note that in different frames of the captured video, the position of the patch changes such as the \( N_p \) changes to 291 in the right image of Figure 4. So \( N_p \) should be re-calculated for each video frame. Once \( N_p \) is calculated, we could define the loss function. Formally, it is defined as follows.

\[
N_p = f(P_{size}, P_{position}),
J(X_i) = 1 - C_{box}^{N_p} + \beta \sum_{j \in \text{class}} \left| p_{N_p,j} - y_j \right|^2
\]  

(4)

In the equation, \( N_p \) is the index of the predictions where the patch locates. \( f(\cdot) \) is the function to calculate \( N_p \). \( J(X_i) \) is the loss function, which is composed of two parts. The first part is \( 1 - C_{box}^{N_p} \). \( C_{box}^{N_p} \) is the box confidence of the prediction with index \( N_p \). So the larger the confidence value, the smaller the value of the loss function. The second part calculates the sum of differences between the probabilities of all predictions with index \( N_p \) (denoted by \( p_{N_p,j} \)) and the target \( y_j \) that we set. In this way, when we minimize the loss function, the confidence of the target object at \( N_p \) becomes maximum, and the possibilities for other predicted objects are minimum. For example, we want the target to be a traffic light. \( y_{10} \) denotes the class probability of traffic light (for COCO dataset), then we should set \( y_{10} \) as 1 and set other classes probabilities as 0 to be the targeted ideal prediction result. The smaller the differences between \( p_{N_p} \) and this targeted prediction, the more likely the predicted classes in these predictions are traffic lights.

5 Evaluation

We evaluated our adversarial examples using the state-of-art object detector YOLO V3, which is a pre-trained model based on Common Objects in Context (COCO) dataset \([2]\). COCO is a large image dataset designed for object detection and is publicly available in the Tensorflow Object Detection API \([3]\). This dataset contains 80 general object classes including persons, stop signs, traffic lights, cars and other common objects.
Table 1: Success rate of practical attack on YOLO V3

|                  | Hiding attack | Appearing attack |
|------------------|---------------|------------------|
| Indoors          |               |                  |
| Success rate     | (1450/1569) 92.4 % | (1659/1894) 88 %  |
| Best accuracy/100 frames | 99 %          | 99 %             |
| Outdoors         |               |                  |
| Success rate     | (452/963) 53 % | (1640/1788) 91.7 %|
| Best accuracy/100 frames | 93 %          | 99 %             |

*Best accuracy/100 frames: The best success rate/100 frames at the distance over 10m. *Success rate: Success rate of total frames.

5.1 Experimental Setup

We evaluate our physical adversarial patches in three different kinds of environment settings, which are indoor (lab) environment, outdoor environment and the real road. For the target object to hide or create, we choose the stop sign as it is related to driving safety. To be more realistic, we brought a real stop sign as shown in Figure 5. In HA, the patches are printed using a regular desktop printer (HP Color LaserJet Pro MFP M277dw). Then we cut the patches out and attach them to the surface of the stop sign. For AA, we print a 60cm×60cm poster to present our adversarial samples, which can be fixed on a stick or carried by a person when tested. Our cameras are the built-in cameras of iPhone 6s and HUAWEI nova 3e. Our computer is equipped with a CPU of E5-2620, a GPU of GTX Titan-X and 32GB memory.

5.2 Effectiveness

To evaluate the effectiveness, we test the patches’ robustness against various factors including distance, angle, and illumination. We recorded several pieces of videos with an iPhone 6s and a HUAWEI nova 3e. For each test, we change one factor while keep the other two fixed. For example, if we want to measure the impact of distance, we start recording the stop sign with the adversarial patches from 25m away and ended at about 1m away, while keeping the camera facing the target object during the whole recording. We choose 25m as the farthest distance for testing because the braking distance of a car at the speed of 60km/h on a dry road is 20m according to the stopping distances table provided in Queensland Government website [4]. For testing the impact of angles, we record videos with different distances (5m ~ 25m, each unit is 5m) and angles (0°, 30°, 45°, 60°). For evaluating the impact of illumination, we repeat our experiments under different illumination conditions such as indoors and outdoors, sunny days and cloudy days. The indoor environment is in an open office area; the outdoor environment is in a parking area.

We define the success rate of the attack by \( r_{succ} = \frac{f_{succ}}{f_{all}} \times 100 \). \( f_{all} \) denotes the number of all the frames in a video, and \( f_{succ} \) shows the number of the frames in which our attack successfully fool the object detector. Since a captured video usually lasts very long (i.e., several minutes), to perform a fine-grained measurement, we also define the success rate for every continuous 100 frames \( f_{succ}^{100} \) whose value equals \( f_{succ} \) in the 100 frames. We choose 100 frames since the time duration for 100 frames is about 3.3 seconds (the frame rate is 30 frames per second or fps for the camera of iPhone 6S), which is long enough for an autonomous driving car make wrong decision. \( f_{succ}^{100} \) could provide a fine-grained metrics for understanding the performance change over time or distances. Below we show our results.

13
Figure 5: **Sample frames of hiding attacks and appearing attacks at different distances.** (a) The hiding attack indoors. (b) The hiding attack outdoors. (c) The appearing attack indoors. (d) The appearing attack outdoors.

**Hiding attack at different distances.** We evaluated HA at a distance range from 1m to 25m while the angles are fixed. The experiment results for distance tests are shown in Table 1. In the indoor environment, the highest $f_{100}^\text{succ}$ at a distance over 10 m is 99%, which means that there exists at least 3-second period in which the object detector fails to detect the stop sign on almost every frame. The success rate on all the frames is 92.4% which is also very high. We also notice that $f_{100}^\text{succ}$ is over 95% in the distance from 25m to 5m. Such good performance is resulted from that we simulate different distances by re-sizing images. In outdoor environment, we carried out the same experiment on a sunny day. The total average success rate is 53%, but the highest $f_{100}^\text{succ}$ can achieve 93%. This attack also keeps a high $f_{100}^\text{succ}$ (over 85%) in the distance from 25m to 10m. We can see that the performance of indoor environment is better than that in outdoor environment, which is mainly due to that indoor environment is more stable and bright-colored. Even though we have adopted gamma correction to simulate different illuminations, outdoor environments are still too complicated to be simulated very well by such a method. How to simulate different outdoor environments still has much space for exploration. In a word, HA can almost fool the detector on every frames at a distance range 1m to 25m indoors, and keep $f_{100}^\text{succ}$ over 85% on average when the distance is over 10m in outdoor environment.
Hiding attack at multiple angles. We measure the effectiveness of HA at angles of 0°, 30°, 45° and 60° for different distances. Then we divide the 25m test distance into five regions, and record videos in each region for each specified angle. The experiments are conducted at the same time from 1:00pm to 3:00pm on a sunny day and cloudy day, respectively. Figure 6 shows that our adversarial patches can perform well at multiple angles and distances no matter on sunny day or cloudy day. In the figure, the success rate is shown in each region. We also use the depth of the background color to represent the success rate. The darker the color, the higher the success rate. From the figure, we can find that HA can obtain a higher success rate at wide angles than narrow angles, and perform better at long distance than short distance. For example, the average success rate for all the four angles is 89% in 20m ∼25m which is larger than the average success rate 68% in 10m ∼15m. The average success rate over the full distance at angle 60° is 82%, while this number at angle 0° is 58%. HA at wide angles performs as good as or even better than that in narrow angles, which has shown that perspective transformation is able to handle different shooting angles. Furthermore, on sunny day, HA's performance drops off in a close distance (within 10m) and keeps stable in long distance. It seems that the success rate is inversely proportional to the detector’s detection capability. In order to verify this viewpoint, we repeat the same experiments on Yolo V3 with the original stop sign without patches. The results are shown in Table 3. Yolo V3 can detected the stop sign in every frame at all angles within 20m, but its detection capability starts to decline in the distance over 20 meters for all angles (especially for the angle 60°). Overall, hiding attack's robustness against angles is quite high. In the wide angle of 60°, it can still keep success rates over 95% at the distance from 10m to 25m on a sunny day, and this number is 93% on a cloudy day.
Appearing attack at different distances. The experimental setup of AA is the same as that of HA. Table 1 shows AA can be quite effective at different distances both indoors and outdoors. For AA in indoor lab environment, the average success rate on total frames is 88%. The highest $f_{succ}$ is 99% at the distance over 10m. Moreover, this attack keeps a high $f_{succ}$ over 90% at the distance from 1m to 20m, which means that the appearing attack keeps a good performance from 1m to 20m and performs not so good over 20m. We also test AA in outdoor environment. The average success rate on total frames is 91.7% and the highest $f_{succ}$ is 99%. This attack keeps $f_{succ}$ over 95% at the distance from 1m to 22m, which could cause huge problems for an autonomous driving car.

It is surprising to see that AA performs better than HA outdoors. It fools the object detector successfully on 1640 frames of the total 1788 frames (91.7%). The main difference between indoor and outdoor is illumination. From the results, we find that AA performs more stable for the various illuminations at both long distance and short distance. For HA, its performance at long distance is better than that at short distance. This phenomenon shows the effectiveness of the nested adversarial patch which is designed to cheat $Model_s$. $Model_s$ is easier to be fooled than other scales’ prediction modules since $Model_s$ gives predictions based on very few pixels.

Appearing attack at multiple angles. We evaluate the performance of AA as we have done in the HA. Benefit from the distance simulation with re-scaling, angle simulation with perspective transformation and illumination simulation with gray-scale transformation, AA has good performance in multiple situations with the varying angles, different distances and illuminations. For a cloudy day, AA achieves over 98% success rate at all angles within 5m ∼ 10m and over 70% success rate at 0° ∼ 45° within 10m ∼ 15m. Moreover, it keeps a high success rate over 80% at 0° from 0m to 25m. By observing the color changes in the Figure 6, we can see that apparently the performance of AA is contrary with that of HA. AA performs better in close distance, narrow angle than in long distance and wide angle. It is reasonable since AA crafts the target object’s
| Success rate | Line 1 (straight road) | Line2 (crossroad) |
|-------------|------------------------|-------------------|
| Hiding attack | 75%                    | 64%               |
| Appearing attack | 63%                   | 81%               |

features with a single patch. Wide angles and long distance will affect the features to be captured by the detector. For a sunny day, it performs better at the distance $5m \sim 15m$ and the angle $0^\circ \sim 45^\circ$. But its performance degrades rapidly at the distance over $15m$. We can observe from the figure that, on a sunny day, the colors are deep at short distance but change to light rapidly at the distance over $15m$. The main reason is that our patches can be recorded much clearer within a short distance on sunny day. However, unavoidable reflection will impact the performance if the distance is more than $15m$. Although the angle affects AA, it is still robust to the angles up to $60^\circ$ within $10m$ and robust to the angles up to $45^\circ$ in the distance from $10m \sim 15m$.

**Real-road driving test.** We implemented real-road driving tests with a real car on a sunny day in two scenarios. In the first scenario, as shown in the left part of Figure 2, we put the stop sign with HA patches or the AA poster (on a stick) on the right side of road. Then the car begins to move from 25 meters away, and passes by the stop sign or poster slowly (at the speed of about $6km/h$). At the mean time, a passenger sitting in the front of the car records the videos. In the second scenario, as shown in the right part of Figure 2, the stop sign with HA patches or the AA poster are put on one corner of the crossroad. The car runs from 25 meters away and turns left when passing the crossroad.

For AA in the first and the second scenario, the success rate is 63% and 81%, respectively. For HA, the results of real-road driving tests are also good. The success rate is 75% in the first scenario and 64% in the second scenario. From this evaluation, we find that HA performs better at a long distance than at a short distance, while the AA poster is on the contrary. This is aligned with the results in the previous evaluations. We also find that the illumination affects the performance. To get good performance, the ideal situation is with high illumination intensity but not direct shooting on the patches since it may cause heavy reflection which can cause the performance’s degradation.

### 5.3 Efficiency

In this section, we evaluate the time required to generate adversarial samples. For each attack, we perform the training process for ten times, and then calculate the average time. For AA and HA, the number of iterations in the training process is 20,000 times. For AA, it takes an hour and fourteen minutes to finish the 20,000 iterations; and for HA, it takes two hour and thirty-eight minutes. HA is slower since it has to modify the patches until the stop sign disappears in each iteration, while AA only modifies the patch for once in each iteration. We also find that the iterations in the later stage of computing HA will become faster. This is mainly due to the fact that the stop sign will become more and more difficult to be detected with the increase number of iterations.
Figure 8: **Sample frames of real-road driving tests.** (a) The hiding attack on the straight road. (b) The hiding attack on the crossroad. (c) The appearing attack on the straight road. (d) The appearing attack on the crossroad.

## 6 Related work

There have been a lot of prior work [30] [8] [21] on investigating the vulnerability of deep neural networks to adversarial examples. Szegedy et al [42] showed us adversarial examples that are hardly perceptible for human can misclassify the DNN-based image classifiers. Since then, the threat of adversarial samples started receiving considerable attention. Yet the effectiveness of such attack in the physical world were suspected by researchers. Lu et al [29] claim that perturbations generated by L-BFGS algorithms for road signs classifiers are ineffective under different angles and distances. And Goodfellow et al [17] analyze the potency of adversarial examples available in physical world. They found that a large fraction of adversarial examples are classified incorrectly by ImageNet even when perceived through the camera. Then Kurakin et al [24] demonstrated that the adversarial examples can be still effective to classifiers when printed out. Athalye et al [6] implemented a 3D printed adversarial object which can fool the neural networks at different orientations and scales. After Gilmer proposed the practical adversarial patch [7] and Dawn Song implemented physical world attacks on the stop sign [14]. All these researches are focus on adversarial samples of image classifiers.

For the adversarial attack to video processing systems, especially for the object detection networks, there are some prior studies. Lu [29] showed us their adversarial examples against
Faster-RCNN which generalize well across a sequences of images in digital space, though the perturbations are large. They even tested their samples in physical world, the results showed that samples in most of the sequences were failed to fool the detector (i.e., the detector can still identify the objects). The only success one was even hard to be identified by human due to the large perturbations. Yang [44] created a 3D mesh representation to attack detectors digitally. 3D mesh in digital space is an interesting idea, but the 3D adversarial object’s effectiveness need to be proved through making it in real world.

There are two studies have tried to generate physical adversarial examples against detectors. Kevin [40] showed us that they generated physical adversarial examples for object detectors by improving upon previous physical attack on image classifiers. They implemented their physical adversarial examples on the stop sign, which can be ignored by YOLO V2. But the performance of their attacks are limited in long distance and multiple angles. Such evaluations are very important since object detectors can detect and label multiple objects in each frame. Also, object detectors can work in multiple angles and different distances, especially in a long distance as the braking distance for a car with the speed of $60 \text{km/h}$ is usually more than $20m$. Chen, Shapeshifter [9] generates adversarial examples based on expectation over transformation technique. However, they did not consider perspective transformation and grayscale transformation. Thus their adversarial examples are less robust to multiple angles, as shown in their evaluation. Their real road driving tests are performed in $0^\circ$. Different from their studies, our adversarial attacks are robust against multiple distances, angles, illuminations and chromatic aberrations.

7 Conclusion

Based on the deep neural network’s vulnerable to adversarial example, we propose the practical adversarial attack method to object detector in this work. Aiming at improving adversarial example’s robustness in physical world and attack ability in long distance, we propose the reality simulation method and nested adversarial patches. Object detectors can locate and classify multiple objects in one single scene, which play an important role in perceptual system of autonomous driving. We implemented our experiments with YOLO V3, a state-of-art real time object detector. For HA, our adversarial patches can fool the object detector with making it failed to recognize the stop sign in a 92.4% success rate in indoor environment. For AA, we get a 91.7 % success rate in outdoor environment and 88 % in lab environment. The success distance for our attack up to 25 m which is the braking distance for 65 $\text{km/h}$ driving speed and the attack angle range wide to 120°. Inspired by our adversarial attack’s high success rates in outdoor environment, long attack distance and wide effective angle range, we implemented our AA and HA in on-road driving tests and got 81% success rate and 75% success rate respectively.

Reference

[1] ApolloAuto. https://github.com/ApolloAuto/apollo.

[2] Common Objects in Context Dataset. http://cocodataset.org/.

[3] MSCOCO Dataset. https://github.com/cocodataset/cocoapi.

[4] The stopping distances table provided in Queensland Government website. https://www.qld.gov.au/transport/safety/road-safety/driving-safely/stopping-distances.
[5] The Tesla fatal crash. https://www.nytimes.com/2017/01/19/business/tesla-model-s-autopilot-fatal-crash.html.

[6] Anish Athalye and Ilya Sutskever. Synthesizing robust adversarial examples. arXiv preprint arXiv:1707.07397, 2017.

[7] Tom B Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. arXiv preprint arXiv:1712.09665, 2017.

[8] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In 2017 IEEE Symposium on Security and Privacy (SP), pages 39–57. IEEE, 2017.

[9] Shang-Tse Chen, Cory Cornelius, Jason Martin, and Duen Horng Chau. Robust physical adversarial attack on faster R-CNN object detector. CoRR, abs/1804.05810, 2018.

[10] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In Advances in neural information processing systems, pages 379–387, 2016.

[11] Chaoqun Duan, Lei Cui, Xinchi Chen, Furu Wei, Conghui Zhu, and Tiejun Zhao. Attention-fused deep matching network for natural language inference. In IJCAI, pages 4033–4040, 2018.

[12] Adam Van Etten. You only look twice: Rapid multi-scale object detection in satellite imagery. CoRR, abs/1805.09512, 2018.

[13] Ivan Evtimov, Kevin Eykholt, Earlence Fernandes, Tadayoshi Kohno, Bo Li, Atul Prakash, Amir Rahmati, and Dawn Song. Robust physical-world attacks on machine learning models. CoRR, abs/1707.08945, 2017.

[14] Ivan Evtimov, Kevin Eykholt, Earlence Fernandes, Tadayoshi Kohno, Bo Li, Atul Prakash, Amir Rahmati, and Dawn Song. Robust physical-world attacks on machine learning models. arXiv preprint arXiv:1707.08945, 2017.

[15] Ross Girshick. Fast r-cnn. In IEEE International Conference on Computer Vision, 2015.

[16] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 580–587, 2014.

[17] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572, 2014.

[18] B Hariharan, P Arbelaez, R Girshick, and J Malik. Object instance segmentation and fine-grained localization using hypercolumns. IEEE Transactions on Pattern Analysis & Machine Intelligence, 39(4):627–639, 2017.

[19] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In Computer Vision (ICCV), 2017 IEEE International Conference on, pages 2980–2988. IEEE, 2017.

[20] He Kaiming, Zhang Xiangyu, Ren Shaoqing, and Sun Jian. Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Transactions on Pattern Analysis & Machine Intelligence, 37(9):1904–1916, 2015.
[21] Jernej Kos, Ian Fischer, and Dawn Song. Adversarial examples for generative models. In 2018 IEEE Security and Privacy Workshops (SPW), pages 36–42. IEEE, 2018.

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In International Conference on Neural Information Processing Systems, pages 1097–1105, 2012.

[23] Amara Dinesh Kumar, Koti Naga Renu Chebrolu, Vinayakumar R, and Soman K. P. A brief survey on autonomous vehicle possible attacks, exploits and vulnerabilities. CoRR, abs/1810.04144, 2018.

[24] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. arXiv preprint arXiv:1607.02533, 2016.

[25] Zeming Li, Chao Peng, Gang Yu, Xiangyu Zhang, Yangdong Deng, and Jian Sun. Light-head r-cnn: In defense of two-stage object detector. arXiv preprint arXiv:1711.07264, 2017.

[26] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971, 2015.

[27] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikinen. Deep learning for generic object detection: A survey. 2018.

[28] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In European conference on computer vision, pages 21–37. Springer, 2016.

[29] Jiajun Lu, Hussein Sibai, Evan Fabry, and David Forsyth. No need to worry about adversarial examples in object detection in autonomous vehicles. arXiv preprint arXiv:1707.03501, 2017.

[30] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In Security and Privacy (EuroS&P), 2016 IEEE European Symposium on, pages 372–387. IEEE, 2016.

[31] Charles Poynton. Digital video and HD: Algorithms and Interfaces. Elsevier, 2012.

[32] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 779–788, 2016.

[33] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. arXiv preprint, 2017.

[34] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767, 2018.

[35] Shaqing 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.
[36] Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, and Yann Lecun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *Eprint Arxiv*, 2013.

[37] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1528–1540. ACM, 2016.

[38] Jing Shi, Jiaming Xu, Guangcan Liu, Bo Xu, et al. Listen, think and listen again: Capturing top-down auditory attention for speaker-independent speech separation. In *IJCAI*, pages 4353–4360, 2018.

[39] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

[40] Dawn Song, Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Florian Tramer, Atul Prakash, and Tadayoshi Kohno. Physical adversarial examples for object detectors. In 12th *USENIX Workshop on Offensive Technologies* (*WOOT* 18), 2018.

[41] David Michael Stavens. *Learning to drive: Perception for autonomous cars*. Stanford University, 2011.

[42] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *Computer Science*, 2013.

[43] Samuel Woo, Hyo Jin Jo, and Hoon Lee Dong. A practical wireless attack on the connected car and security protocol for in-vehicle can. *IEEE Transactions on Intelligent Transportation Systems*, 16(2):993–1006, 2015.

[44] Dawei Yang, Chaowei Xiao, Bo Li, Jia Deng, and Mingyan Liu. Realistic adversarial examples in 3d meshes. *CoRR*, abs/1810.05206, 2018.

[45] Ming-Hsuan Yang, D. J. Kriegman, and N. Ahuja. Detecting faces in images: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):34–58, Jan 2002.

[46] Xingyu Zeng. *Deep Neural Networks for Object Detection*. PhD thesis, The Chinese University of Hong Kong (Hong Kong), 2016.
Table 3: Success rate of the original stop sign on YOLO V3

| Success rate(%) | $5m \sim 10m$ | $10m \sim 15m$ | $15m \sim 20m$ | $20m \sim 25m$ |
|----------------|---------------|---------------|---------------|---------------|
| $0^\circ$      | 100           | 100           | 100           | 93            |
| $30^\circ$     | 100           | 100           | 100           | 84            |
| $45^\circ$     | 100           | 100           | 100           | 90            |
| $60^\circ$     | 100           | 98            | 93            | 72            |