FCA: Learning a 3D Full-coverage Vehicle Camouflage for Multi-view Physical Adversarial Attack

Donghua Wang[1]*, Tingsong Jiang[2]*, Jialiang Sun[2], Weien Zhou[2], Xiaoya Zhang[2], Zhiqiang Gong[2], Wen Yao[2], Xiaoqian Chen[2]†

[1]College of Computer Science and Technology, Zhejiang University
[2] Defense Innovation Institute, Chinese Academy of Military Science
wangdonghua@zju.edu.cn, tingsong@pku.edu.cn, sun1903676706@163.com, weienzhou@outlook.com, wendy0782@126.com, {gongzhiqiang13, zhangxiaoya09, chenxiaoqian}@nudt.edu.cn

Abstract

Physical adversarial attacks in object detection have attracted increasing attention. However, most previous works focus on hiding the objects from the detector by generating an individual adversarial patch, which only covers the *planar* part of the vehicle's surface and fails to attack the detector in physical scenarios for multi-view, long-distance and partially occluded objects. To bridge the gap between digital attacks and physical attacks, we exploit the *full* 3D vehicle surface to propose a robust Full-coverage Camouflage Attack (FCA) to fool detectors. Specifically, we first try rendering the non-planar camouflage texture over the full vehicle surface. To mimic the real-world environment conditions, we then introduce a transformation function to transfer the rendered camouflaged vehicle into a photo-realistic scenario. Finally, we design an efficient loss function to optimize the camouflage texture. Experiments show that the full-coverage camouflage attack can not only outperform state-of-the-art methods under various test cases but also generalize to different environments, vehicles, and object detectors. The code of FCA will be available at: *https://idrl-lab.github.io/Full-coverage-camouflage-adversarial-attack/*.

Introduction

Over the past years, deep neural networks (DNNs) have achieved tremendous success in computer vision tasks. However, DNNs are found vulnerable to adversarial examples (Szegedy et al. 2013), which are elaborately designed to mislead DNNs to make incorrect predictions. As a new security issue in artificial intelligence, adversarial attacks appeal the attraction from both academics and industry.

Adversarial attacks can be divided into two categories by their applicable domains: 1) **digital attacks** directly add imperceptible perturbations to pixels of input images in the digital space (Szegedy et al. 2013), while 2) **physical attacks** modify objects in the real-world environment or physical simulators (Chen et al. 2018; Sharif et al. 2016; Kurakin et al. 2016; Lu, Sibai, and Fabry 2017; Athalye et al. 2018) to investigate whether the perturbations are physically realizable and can stay adversarial under different transformations. In this paper, we mainly concentrate on the latter as it is a more direct threat to visual systems in the physical world.

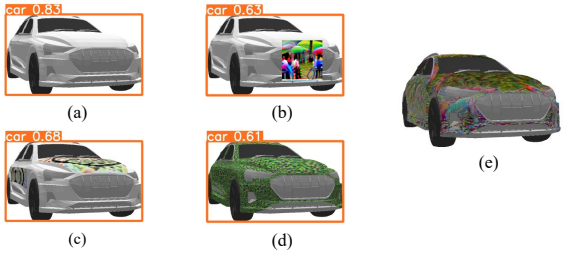

Figure 1: (a) is a car without camouflage. (b) is a camouflaged car by placing a planar adversarial patch in front of the car (Thys, Van Ranst, and Goedemé 2019). (c) is a camouflaged car by placing an adversarial patch over the rooftop, hood and doors (Wang et al. 2021a). (d) is a the camouflaged car by repeating an adversarial pattern (Zhang et al. 2018). (e) shows the camouflaged car generated by FCA, which is undetected.

Recently, adversarial attacks on object detection have attracted increasing attention, particularly in physical attacks due to complex physically realizable constraints (e.g., non-planar object surface) and environmental conditions (e.g., lighting, viewing angles, camera-to-object distances and occlusions) (Elsayed et al. 2018). There are mainly two kinds of methods to modify the visual characteristics of the real object: patch-based and camouflage-based. **Patched-based** methods try to perform physical adversarial attacks by generating adversarial patches(Brown et al. 2017), which confine the noise to a small and localized patch without perturbation constraint. A patch is often stuck to a planar object(e.g., STOP sign (Eykholt et al. 2018)) or placed in front of the object(e.g., person (Thys, Van Ranst, and Goedemé 2019)) or placed in the background(Lee and Kolter 2019).

Camouflage-based method is implemented by modifying the target object itself and is more challenging due to the non-planarity of 3D objects. There are two ways to paint the camouflage: one way is to optimize an adversarial pat-

*These authors contributed equally.
†Corresponding author.

tern and repeat the patterns as a whole camouflage to paint on the vehicle's surface using a physical non-differentiable renderer(Zhang et al. 2018; Wu et al. 2020a), while another way is to optimize the texture(Zeng et al. 2019; Wang et al. 2021a) or the shape(Xiao et al. 2019) of the 3D vehicle directly with a differentiable neural renderer.

However, existing methods are not robust to specific physical scenarios, especially for multi-view, long-distance and partially occluded objects. Firstly, a patch is often stuck to a planar object, so patch-based methods are not suitable and robust for attacking vehicle detectors over 3D vehicles as shown in Figure 1(b). Secondly, previous camouflage-based methods (Huang et al. 2020; Wang et al. 2021a) paint the adversarial camouflage only on the part of the 3D vehicle model, e.g., the rooftop or side doors, which limits the attack capability in multi-view scenarios when partial adversarial camouflage is not visible, as shown in Figure 1(c). Besides, (Wang et al. 2021a) is not competitive for attacking detectors because they aim to exploit the common characteristic (e.g., model's attention) among models and mainly focus on classifiers. Thirdly, previous "full-coverage" camouflaged methods (Zhang et al. 2018; Xiao et al. 2019; Wu et al. 2020a) generate an individual adversarial pattern and repeat the pattern until covering all the vehicle surface (i.e., as shown in Figure 1(d)), which is essentially an adversarial image patch optimization. The camouflaged vehicles with image pattern may fail to attack the objectors for multi-view and long-distance scenarios.

To address the aforementioned problems, we propose an end-to-end Full-coverage Camouflage Attack (FCA) pipeline. Specifically, we first treat the adversarial camouflage as the texture of the 3D vehicle and utilize a neural renderer to paint the texture onto the full surface of vehicle. Then, we apply a transformation function to convert the rendered 3D vehicle into different environment scenarios to get photo-realistic images. And finally, we model the generation of the camouflage texture as an optimization problem by designing an efficient loss function. With such generated adversarial camouflage, the painted vehicle can stay adversarial in physical scenarios for multi-view, long-distance and partially occluded objects.

In summary, our main contributions list as follows.

- We bridge the gap between digital attacks and physical attacks via a differentiable neural renderer. We overcome the partial occluded and long-distance issues by painting the adversarial camouflage onto the full vehicle surface.
- An end-to-end physical adversarial attack was proposed to generate a robust adversarial camouflage.
- Extensive experiments demonstrate that our method outperforms the existing methods and generalizes to different environments, vehicles, and object detectors. In addition, our camouflage can be easily painted or overlaid in the real world and seems natural to humans.

Related work

In this section, we first review the physical adversarial attacks in object detection. And then we briefly introduce the neural renderer.

Physical Adversarial Attack

According to the implementation methods, the attacks can be briefly divided into patch-based and camouflage-based. The patch-based attacks aim to generate an universal image patch(Brown et al. 2017), and several transformations(Huang et al. 2020) were adopted to ensure the transferability. (Zhang et al. 2018) devised a clone network to simulate the process of physical rendered to object predicted, they update the camouflage patch by performing the white-box attack on the clone network. Similarly, (Wu et al. 2020a) proposed a query-based discrete searching algorithm to generate an adversarial patch, and then repeated and enlarged the patches until they covered the vehicle surface. Although these attacks achieve certain success, their attacking ability deteriorates when applied to the complex physical world.

The camouflage-based attacks aim to modify the shape or texture of the 3D object. In this category, (Xiao et al. 2019) utilized a neural renderer to modify the shape and texture of the textureless object directly, the final result is an adversarial object. Recently, (Wang et al. 2021a) proposed a dual attention suppress attack, which suppresses the attention map of the target object in the detection model. To maintain the naturalness of the camouflage (i.e., human attention evasion), they constrain the perturbation only around the content seed. In this paper, we paint the texture of the 3D vehicle similarly as (Wang et al. 2021a), however, we find their adversarial camouflage is not robust for multi-view, long-distance and partially occlusions, for which they constrained the camouflage area to the rooftop, hood and car doors. We solve the issue mentioned above with full-coverage (except the glass, tire, lights) camouflage texture.

Neural Renderer

Traditional renderer is commonly used in 2D-to-3D transformation, one of the applications is to wrap the texture image to the 3D model, which then is rendered to the 2D image. To make the rendering process differentiable, (Kato, Ushiku, and Harada 2018) proposed an approximate gradient for rasterization to enable the integration of rendering into neural networks, which is referred as neural renderer. Initializing with different camera parameters (i.e., rotation and location), one could render the 3D object model (consisting of mesh and texture) under different view angles. (Zhang et al. 2018) and (Wu et al. 2020a) utilized the CARLA(Dosovitskiy et al. 2017) simulator to render the adversarial patch onto 3D object, which is non-differentiable. (Xiao et al. 2019) used the neural renderer to modify the shape and texture of 3D objects. Following (Wang et al. 2021a), we utilize the neural renderer to paint our adversarial camouflage onto the vehicle surface.

Method

In this section, we first introduce the preliminaries. Then we describe the proposed end-to-end physical camouflage adversarial attack in detail.

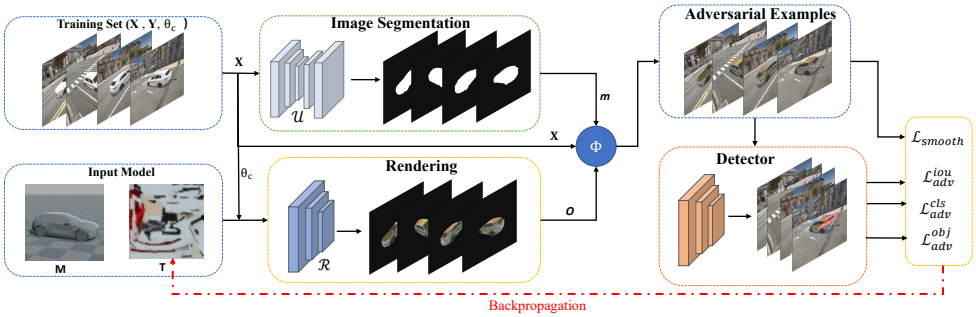

Figure 2: The overview of FCA. Our training set contains the images sampled from the photo-realistic simulator under different simulation settings. We first utilize a pretrained image segmentation network to fetch the target vehicle and binary it as a mask. Meanwhile, we render the camouflage texture onto the surface of the vehicle with the same simulation setting and obtain the camouflaged 2D vehicle. Next, we utilize a transformation function to transfer the camouflaged vehicle into the different physical scenarios with the corresponding mask. Finally, we update the adversarial camouflage through backpropagation with our devised loss function.

Preliminaries

Given a vehicle training set $(\mathbf{X}, \mathbf{Y}, \theta_c)$ where $\mathbf{X}$, $\mathbf{Y}$ and θ_c are the sampled images, ground truth labels of the target vehicle and the corresponding camera parameters (i.e., transformation and location) respectively, a 3D vehicle model with a mesh $\mathbf{M}$ and a texture $\mathbf{T}$, we use a renderer $\mathcal{R}$ with camera parameter θ_c to obtain the rendered 2D vehicle image $\mathbf{O} = \mathcal{R}(\mathbf{M}, \mathbf{T}; \theta_c), \mathbf{O} \in \mathbb{R}^{H \times W \times 3}$. To mimic the physical real world, we devise a transformation function Φ to transfer the rendered vehicle image to different environment scenarios, and then obtain the input image $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$ of the detector. Now, we can obtain the detection result $\mathbf{b} = \mathcal{F}(\mathbf{I}; \theta_f) = (b_x, b_y, b_w, b_h, b_{obj}, b_{cls})$, where $\mathcal{F}$ is the object detector with parameters θ_f, b_x and b_y are the center coordinates of the prediction bounding box (i.e., *bbox*), b_h and b_w indicates the height and width of the prediction *bbox*, b_{obj} is confidence score that the *bbox* contains an object, b_{cls} is the class probability distribution of the object in the bbox, $b_{cls} \in [1, 80]$ in COCO.

Our attack scheme is to generate the adversarial camouflage texture, which can be painted on the surface of the 3D vehicle model. The target vehicle category we select is "car" due to the real-time vehicle surveillance is widely used in daily life. Note that our attack target object is only one specific car in the scenario. To realize the adversarial camouflage attack, we replace the origin texture $\mathbf{T}$ with adversarial texture $\mathbf{T}_{adv}$, and obtain the corresponding adversarial image $\mathbf{I}_{adv}$ with transformation function Φ. We aim to hide the target vehicle under the detector $\mathcal{F}$. We treat the adversarial texture generation as an optimization problem, and our objective function is expressed as follows

$$\mathbf{T}^*_{adv} = \arg\max_{\mathbf{T}_{adv}} J(\mathcal{F}(\Phi(\mathcal{R}(\mathbf{M}, \mathbf{T}_{adv}; \theta_c)); \theta_f), \mathbf{Y}) \quad (1)$$

where $\mathbf{T}^*_{adv}$ is the final adversarial texture, $J(\cdot, \cdot)$ is the loss function. By solving the above optimization problem, i.e., Eq 1, we can obtain the ultimately adversarial camouflage texture.

Generating Adversarial Camouflage

To generate full-coverage adversarial camouflage, we propose an adversarial camouflage texture generation framework with a differentiate neural renderer, which can render the customized texture onto the 3D vehicle model directly. The overall framework of FCA is illustrated in Figure 2, our goal is to generate a robust camouflage texture through the backpropagation of loss.

To this end, the loss function plays a vital role in optimizing. In this work, we devise the loss function considering two key aspects: *adversarial loss* to guarantee the attacking ability. *smooth loss* to make the digital-physical difference caused by camouflage more natural. We will discuss these losses in the following sections.

Adversarial Loss In this work, we use YOLO-V3 as the target detection model $\mathcal{F}$, in other words, we train the adversarial texture with a known model under white-box attack setting. It's well known that YOLO-V3 is a single-stage detector, which makes classify and regression in a single step with dense sampling. Thus it is necessary to take account of attacking both regression and classification simultaneously. After analyzing the loss function of YOLO-V3, we use the following three-loss terms: $\mathcal{L}^{iou}_{adv}, \mathcal{L}^{obj}_{adv}, \mathcal{L}^{cls}_{adv}$. To make the detector incorrectly detected or undetected, we first reduce the intersection over union (IoU) between the prediction *bbox* and ground truth *bbox* to suppress the region of the target prediction *bbox*, which is denoted as $\mathcal{L}^{iou}_{adv}$. Then we reduce the objectness score that indicates whether the prediction *bbox* contains an object by minimizing the objectness confidence. We denote this loss as $\mathcal{L}^{obj}_{adv}$. Finally, to attack the classification, we select the probability of the target object and minimize it, which is denoted as $\mathcal{L}^{cls}_{adv}$. Therefore, our final adversarial loss $\mathcal{L}_{adv}$ is constructed as follow

$$\mathcal{L}_{adv} = \alpha \mathcal{L}^{iou}_{adv} + \beta \mathcal{L}^{obj}_{adv} + \gamma \mathcal{L}^{cls}_{adv} \quad (2)$$

where α, β, γ are the weights to balance the contribution of each loss term. Then we will exhaustively introduce each loss term of adversarial loss in the following.

• IoU loss $\mathcal{L}_{adv}^{iou}$ represents the overlap area between the ground truth label and the prediction result of the rendered images. One can obtain a high IoU value with a trained detector at the inference stage. By minimizing the $\mathcal{L}_{adv}^{iou}$, we can suppress the prediction *bbox* of the target region. Consequently, the target object is filtered by the detector as the IoU below the threshold. Thus, our $\mathcal{L}_{adv}^{iou}$ is formulated as follows

$$\mathcal{L}_{adv}^{iou} = \sum_{i}^{N} IoU(b^i, b_{gt}^i) \tag{3}$$

where N denotes the multi-scale (i.e., N=3) output prediction result of the YOLO-V3, b^i and b_{gt}^i is the *i-th* scale prediction result and corresponding ground truth bbox of our attack target, respectively.

• Objectness loss $\mathcal{L}_{adv}^{obj}$ represents the confidence score whether the detection box contains an object. We follow (Thys, Van Ranst, and Goedemé 2019; Wang et al. 2021b) and choose the object confidence score to as our $\mathcal{L}_{adv}^{obj}$.

• Classification loss $\mathcal{L}_{adv}^{cls}$ represents the classification probability of the target class, i.e., car. Specifically, we select the *i-th* scale probability of the target class t in the detection result, denoting it as $b_{cls^t}^i$. Finally, the classification loss can be expressed as

$$\mathcal{L}_{adv}^{cls} = \sum_{i}^{N} b_{cls^t}^i \tag{4}$$

Smooth Loss To ensure the naturalness of the generated adversarial camouflage, we follow (Sharif et al. 2016) to utilize the smooth loss that introduced by (Mahendran and Vedaldi 2015) to reduce the inconsistent among adjacent pixels. For a rendered vehicle image painted with adversarial camouflage $\mathbf{I}_{adv}$, the calculation of smooth loss can be written as

$$\mathcal{L}_{smooth} = \sum_{i,j} (x_{i,j} - x_{i+1,j})^2 + (x_{i,j} - x_{i,j+1})^2 \tag{5}$$

where $x_{i,j}$ is the pixel value of $\mathbf{I}_{adv}$ at coordinate (i, j).

Physical Transformation

Previous work (Wang et al. 2021a) painted the camouflage on the vehicle through *tensor addition*, i.e., the rendered camouflaged vehicle image pixels are directly added to the sampled image containing the original vehicle, which makes it difficult to get convergence during training. Instead, we introduce a simple but efficient approach to substitute the tensor addition. Specifically, we use a segmentation network $\mathcal{U}$ to crop the background from the original photo-realistic image and obtain a binary mask $m \in \mathbb{R}^{H \times W \times 1}$ where the target vehicle areas are set to 1, the background areas are set to 0. With such a mask, we can obtain the adversarial example $\mathbf{I}_{adv}$ by transferring the rendered vehicle image $\mathbf{O}$ into photo-realistic environment scenario. The transformation Φ can be expressed as follow

$$\mathbf{I}_{adv} = \Phi(\mathbf{O}) = m \cdot \mathbf{O} + (1 - m) \cdot \mathbf{I} \tag{6}$$

Algorithm 1: Full-coverage Camouflage Attack (FCA)

Input: training set $(\mathbf{X}, \mathbf{Y}, \theta_c)$, 3D model **(M, T)**, neural renderer $\mathcal{R}$, object detector $\mathcal{F}$, segmentation network $\mathcal{U}$
Output: adversarial texture $\mathbf{T}_{adv}$

Initial $\mathbf{T}_{adv}$ with random noise
for the max epochs **do**
  select the *minibatch* sample from training set $(\mathbf{X}, \mathbf{Y}, \theta_c)$
  $m \leftarrow \mathcal{U}(\mathbf{X})$
  $\mathbf{O} \leftarrow \mathcal{R}((\mathbf{M}, \mathbf{T}_{adv}); \theta_c)$
  $\mathbf{I}_{adv} \leftarrow m \cdot \mathbf{O} + (1 - m) \cdot \mathbf{I}$
  $b \leftarrow \mathcal{F}(\mathbf{T}_{adv}; \theta_f)$
  calculate $\mathcal{L}$ by Eq 7
  update $\mathbf{T}_{adv}$ with gradient backpropagation
end for

where $\cdot$ denotes the pixel-wise multiplication. Note that, we preserve the location and rotation information during the sampling stage of the photo-realistic images, thus the rendered vehicle has the identical orientation as the vehicle in the sampled image.

Optimization Process

Overall, we obtain the adversarial camouflage texture by jointly minimizing the adversarial loss $\mathcal{L}_{adv}$ and smooth loss $\mathcal{L}_{smooth}$. Consequently, our optimization objective can be summarized as

$$\mathcal{L}_{total} = \mathcal{L}_{adv} + \mu \mathcal{L}_{smooth} \tag{7}$$

Algorithm 1 summarize the overall training scheme of the presented approach.

Experiments

In this section, we first describe the experimental settings. Then we empirically show the effectiveness of the proposed full-coverage camouflage by providing thorough evaluations in different simulation environments.

Experimental Settings

Datasets To bridge the gap between digital attacks and physical attacks, we utilize the photo-realistic datasets to perform the experiments. To this end, we select the simulator CARLA (Dosovitskiy et al. 2017), a prevalent open-source simulator for autonomous driving research, as our 3D simulator. The CARLA simulator provides a variety of high-fidelity digital scenarios (e.g., modern urban) based on Unreal Engine 4. To compare with previous works, we use the same datasets provided by (Wang et al. 2021a) directly, the training set consists of 12,500 high-resolution images, while the testing set has 3,000 high-resolution images. The datasets contain images that are sampled from different view angles and distances.

Evaluation Metrics We aim to generate false negatives and hide the target vehicle from the detector. To this end, the first evaluation metric that we select is the Attack Success Rate (ASR) (Wu et al. 2020b), which is defined as the

Table 1: The comparison result of adversarial attacks in the digital space.

Method	P@0.5(%)			
	YOLO-V5	Faster RCNN	SSD	Mask RCNN
Raw	92.07	86.04	81.54	89.24
MeshAdv	72.45	71.84	66.44	80.84
CAMOU	74.01	69.64	73.81	76.44
UPC	82.41	76.94	74.58	81.97
DAS	72.58	62.11	68.81	70.21
DAS-full	60.52	51.43	49.93	52.07
Ours	**32.07**	**34.00**	**28.67**	**30.80**

percentage of the target vehicles detected before perturbation and not detected or false detected after perturbation. In addition, we adopt the P@0.5 following (Zhang et al. 2018; Wang et al. 2021a) as our second evaluation metric, which is defined as the percentage of the correct detected when the detection IoU threshold is set to 0.5.

Implementation details We choose a widely used detector, YOLO-V3 (Redmon et al. 2016), as our white-box model to train the adversarial camouflage texture. And we evaluate the transferring attack performances (black-box attack) on the following prevalence object detection models: YOLO-V5 (Jocher et al. 2021), SSD (Liu et al. 2016), Faster R-CNN (Ren et al. 2015), and Mask R-CNN (He et al. 2017). These models are all pretrained on COCO dataset. Note that, in our experiments, these models are the official implementation version provided by PyTorch (Paszke et al. 2019) except SSD [1].

The adversarial camouflage texture is initialized as random noise, and the Adam with default parameter is adopted as the optimizer. The hyperparameters are set as follows, the learning rate is 0.01, the max epoch is 5. For α, β, γ, we use the default value 0.05, 1.0, 0.5, respectively, provided by the YOLO-V3 implementation. We follow (Wang et al. 2021a) to set the μ to 1.0. Note that, we also find that hyperparameter $\alpha, \beta, \gamma, \mu$ has a limited impact on our performance in our preliminary experiment. The segmentation network used to extract the background from the photo-realistic image is U2-Net (Qin et al. 2020). We conduct the experiment on a NVIDIA RTX 3090 24GB GPU cluster.

Digital Adversarial Attack

In this section, we evaluate the performance of adversarial camouflage in the digital space. We report the P@0.5 for the detection of the target vehicle.

We compare the proposed attacks with several current advanced adversarial camouflage attacks, including MeshAdv (Xiao et al. 2019), CAMOU (Zhang et al. 2018), UPC (Huang et al. 2020), DAS (Wang et al. 2021a). In order to fairly compare our attack with DAS attack, we reimplement the DAS attack with full-coverage camouflage, which denotes as "DAS-full".

The comparison results are listed in Table 1. Note that, we adopt the results reported in (Wang et al. 2021a) because our test set and detectors are identical. As illustrated in Table 1, our adversarial camouflage significantly outperforms other methods over all the detectors. Specifically, on the one side, the maximum drop of P@0.5 by **60%** on YOLO-V5, the minimum drop of P@0.5 by 52.04% on Faster RCNN, the average drop is 56.02%, which demonstrate that our attack could successfully paralyze the vehicle detection system. On the other side, in our experiments, Faster RCNN shows better robustness (i.e., lower performance decline) than other baseline detectors, probably due to some modules in Faster RCNN that are robust to the appearance change of the object. Finally, despite DAS use a similar full-coverage camouflage (i.e., DAS-full), our attack still outperforms the DAS, which suggests that our proposed loss function is more suitable for attacking object detection.

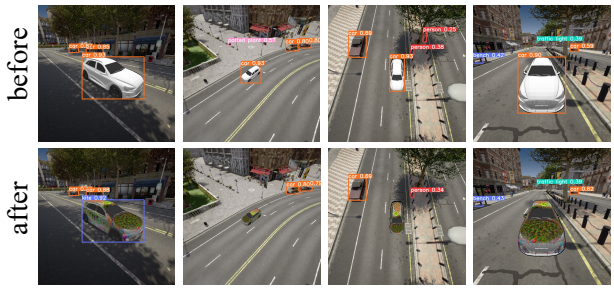

Figure 3: The detection result of the vehicle under different view angles before and after our attack. After painting with our camouflage, the target vehicle turns to be incorrectly detected or undetected.

We provide some adversarial camouflage vehicle examples in different scenarios. As illustrated in Figure 3, the vehicle before painted with adversarial camouflage is detected as a car with high detection confidence. However, after painted with our adversarial camouflage texture, the vehicle is detected as other categories, even "disappear" under the detector. To show the effectiveness of our adversarial camouflage in realizable applications, we provide more diverse examples here[2].

Physical Adversarial Attack

In this section, we evaluate the performance of adversarial camouflage in the physical space. We report the P@0.5 for the detection of the target vehicle.

For simplicity, we compare one partially camouflage attack(i.e., DAS) and two full-coverage camouflage attacks (i.e., CAMOU and DAS-full). Due to the limitation of founds and conditions, we follow (Wang et al. 2021a) to print our adversarial camouflages by an HP Color Laser MFP 179fnw printer and crop the camouflage part, then stick them on a toy car with different backgrounds to mimic the real car painting in the physical world. To show the efficiency of our adversarial camouflage under different scenarios, we capture 144 pictures of the painted car on different settings (i.e., 8 directions (45° / 360°), 3 distances long,

[1] https://github.com/lufficc/SSD

[2] https://idrl-lab.github.io/Full-coverage-camouflage-adversarial-attack/

Table 2: The comparison result of adversarial attacks in the physical space.

Method	P@0.5(%)			
	YOLO-V5	Faster RCNN	SSD	Mask RCNN
Raw	100	88.89	78.47	96.53
CAMOU	72.22	44.44	40.97	53.48
DAS	100	65.28	52.08	68.06
DAS-full	94.44	43.06	43.75	45.83
Ours	**65.28**	**24.31**	**29.17**	**29.17**

middle, and short distance, 3 different surroundings) with a Redmi K20 Pro phone.

The evaluation results are list in Table 2. Compared with other methods, the FCA can transfer to the physical world well, we get 65.28% on YOLO-V5, 24.31% on SSD, 29.17% on Faster RCNN, 29.17% on Mask RCNN, respectively. That indicates that FCA can pose more potential risks for the detection systems in the real world. Moreover, all the full-coverage adversarial camouflages are better than the partial coverage adversarial camouflage, which is consistent with our analysis that the performance of existing adversarial camouflage attacks degrades due to multi-view or partially occlusion scenarios. However, the Mask RCNN shows the worst robustness (maximum drop in P@0.5). While the YOLO-V5 shows the best robustness against adversarial camouflage, which may be attributed to the special design that makes it more suitable for real-world application. Despite this strong model, our FCA method can also degrade the detection performance in a large marginal, which demonstrated that our adversarial camouflage has a strong transferable attacking ability in the physical world.

Multi-view Robust Attack

Robustness of Long Distance To demonstrate the robustness of our adversarial camouflage in multi-view and long-distance scenarios, we conduct extensive experiments. Specifically, in the experiment, the camera distance we used includes [1.5, 3, 5, 10, 15, 20], the camera elevation we used includes [0, 10, 20, 30, 40, 50] (0 indicates that the camera and the vehicle are parallel). We sample an image every 3° a time in 360°. For a fixed combination of camera distance and elevation, we obtain 120 images, and collect 4320 test images in total. To better illustrate the result, we regroup the rendered image test set in terms of azimuth ranges (i.e., every 45° azimuth) and camera distances, then every group has 90 test samples with various camera elevation, in other words, every item in Table 3 is conducted on different 90 test images. Note that to better evaluate the view angles and distances without considering different background environments, we use the rendered images $\mathbf{O}$ (pure background) directly for simple implementation. We use YOLO-V5 to evaluate the test images as other detectors exhibit similar trends.

The results are listed in Table 3. We can observe that we achieve 100% ASR in a majority of cases where the distance is among 1.5, 3 and 20. Meanwhile, we find that along with the distance increase, the ASR first prone to decrease at the distance of 10, after that the trend of ASR prone to increase. By contrast, the images sampled at distance of 5, 10 and 15 are hard to attack, which demonstrates the detector is more robust for such settings. Nevertheless, our adversarial camouflage can achieve nearly perfect performance without retrained on the rendered images $\mathbf{O}$, which demonstrate that the generated adversarial camouflage has well transferability across different domain datasets.

Table 3: The ASR (%) performance for multi-view and multi-distance attack.

Azimuth ($^\circ$)	Distance					
	1.5	3	5	10	15	20
$0 \sim 45$	100	100	84.27	68.6	80.85	100
$45 \sim 90$	95.83	93.33	88.89	81.82	90.2	100
$90 \sim 135$	100	100	88.31	87.5	94.92	100
$135 \sim 180$	100	100	84.44	71.11	78.57	87.5
$180 \sim 225$	100	100	95.51	92.22	88.24	100
$225 \sim 270$	100	100	98.65	88.57	95.65	100
$270 \sim 315$	100	100	92.96	86.96	95.65	100
$315 \sim 360$	100	100	94.44	74.44	83.33	100

Table 4: The ASR performance for partially occluded objects for different distances.

Occlusion	Distance			
	1.5	3	5	10
small	100	100	62.22	62.68
middle	98	92.05	78.89	77.14
large	96	97.62	72.86	78.57

Robustness of Partial Occlusion We also investigate the robustness of our attack when the adversarial camouflage is partially occluded. According to the area of the occluded camouflage, we group the partial occlusion into small occlusion, middle occlusion and large occlusion. Specifically, we define the large, middle and small partial occlusion as the $\geq 70\%, 30\% \sim 70\%, \leq 30\%$ of car body is occluded, respectively. In this experiment, We only use the [1.5, 3, 5, 10] camera distances due to the occluded rendered object is too small when the camera distance exceeds 10. For each group and a given camera distance, we collect 90 test images, and totally collect 1080 test images .We use the YOLO-V5 as our evaluation model.

Results are listed in Table 4. As we can see the generated camouflage works well at 1.5 and 3 camera distances, particularly in a small occlusion (ASR achieve 100%), which is attributed to the ratio of rendered images as well as the camouflages are relatively large. On the other hand, when the camera distance exceeds 5, the performance degrades sharply, the possible reason is that the rendered object in images is a very small object and the camouflage being occluded further leads to performance decreasing. We provide some partial occlusion cases in Figure 4, which demonstrated our adversarial camouflage works well for most partial occlusion scenarios. Our camouflage is more robust for occlusion when the camera distance is less than 3, while the robustness trends to degrade when the camera distance increases. In conclusion, our generated adversarial camouflage is robust to different levels of partial occlusions.

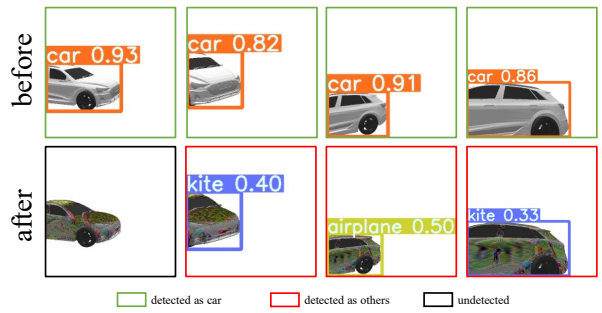

Figure 4: The case of occlusion vehicle before and after our attack. After painted with our camouflage, the detector incorrectly detected or not detected the "car".

Enhance the Transferability

It has been proved that transferability of adversarial examples can benefit from hard examples (Liu et al. 2020; Xie et al. 2019). Thus, we argue that the transferability of our camouflage can be further enhanced with hard examples. To this end, we define the attack failure examples as hard examples, then we collect them across the different detectors, and utilize the YOLO-V3 to fine-tune the adversarial camouflage texture on hard examples. Note that, the ratio of hard examples for YOLO-V5, Faster RCNN, Mask-RCNN, SSD is $17.1\%, 29.43\%, 27.71\%, 17.38\%$, respectively. And the number of the enhanced dataset is 4932.

The updated results are listed in Table 5, the row indicates the detector that used to generate failure examples and the column indicates the re-evaluated results with fine-tuned hard examples. The diagonal entries of the table indicate the detector used to collect failure examples and re-evaluate is identical. From the table, we can observe that hard examples can enhance the transferability of the adversarial camouflage, we obtain 2.43% gain for YOLO-V5 itself. However, we also notice that the hard examples are not always effective, the ASR on Faster RCNN and Mask RCNN of all fine-tuning adversarial textures even degrades compared to unenhanced results. To explain this phenomenon, we analysis the collected hard examples, and find the union of these hard examples achieves 41.1% of the training set, while the intersection of these hard examples is nearly 0%. The reason may attribute to the different architecture of the detector, some failure examples collected by four detectors may still successfully attack YOLO-V3 during fine-tuning, which means such hard examples are helpless for improving transferability.

Interpretability of the Adversarial Camouflage

In this section, we try to explain why the detector fails on our generated adversarial camouflage. Following (Wang et al. 2021a), we choose the commonly used interpretability technique, i.e. Grad-CAM (Selvaraju et al. 2017). We use ResNet50(He et al. 2016) that pretrained on ImageNet as the base model to extract the attention map of the target vehicle category, the results are illustrated in Figure 5. We can see the model's attention on the target category is dispersed after painting our camouflage, which suggested that the decision evidence of the model has been changed. Therefore, the detector makes incorrect inference on adversarial examples.

Table 5: The ASR on four detectors where we retrain the camouflage texture on different hard examples extracted by various detectors. The diagonal entries indicate retrained and evaluated on the same detector, while the off-diagonal entries indicate the transfer attack.

Method	ASR(%)			
	YOLO-V5	Faster RCNN	SSD	Mask RCNN
unenhanced	87.67	72.11	78.42	75.16
YOLO-V5	**90.1**	71.35	**79.17**	73.99
Faster RCNN	**88.92**	70.8	**79.65**	74.06
SSD	86.97	70.04	78.06	73.17
Mask RCNN	**89.8**	70.44	**79.43**	74.13

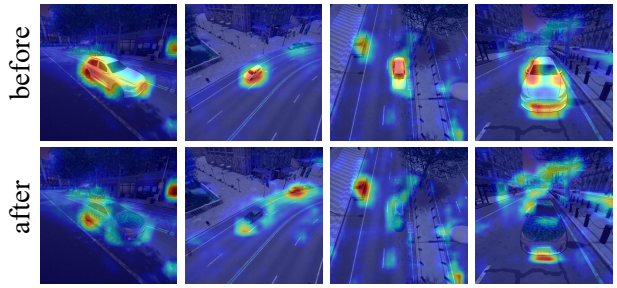

Figure 5: The attention map of the vehicle before and after our attack. After painting our full-coverage camouflage, the attention of the vehicle is dispersed in the image.

Ablation Studies

In this section, we investigate the influence of the loss function items and the initialization ways of the camouflage texture.

Effectiveness of the combination of loss terms. Different loss items have different effects. In this part, we conduct the following two-fold studies: the first fold compares $\mathcal{L}_{adv}^{cls}$, $\mathcal{L}_{adv}^{obj}$, $\mathcal{L}_{adv}^{iou}$, $\mathcal{L}_{adv}^{cls+obj}$, which all contain the smooth loss. The second fold investigates the influence of smooth loss in our method, we denote the loss containing only adversarial loss as $\mathcal{L}_{adv}$. $\mathcal{L}_{total}$ denotes a combination of both adversarial loss and smooth loss. We optimize the adversarial camouflage with different loss term schemes, and evaluate the ASR performance on different detectors. The experiment results are shown in Table 6.

As we can observe from Table 6, on the first fold, we obtain the highest ASR in YOLO-V5 with $\mathcal{L}_{adv}^{iou}$, exceeding 90%. On the contrary, the ASR of other three models is relatively low. We conclude that the $\mathcal{L}_{adv}^{iou}$ has significant impact on the ASR in particular detectors. On the second fold, the ASR without smooth loss is higher than that with smooth loss for all models, while smooth loss makes the camouflage more natural to humans. In summary, the devised $\mathcal{L}_{total}$ balances the attack performance and the naturalness of the adversarial camouflage, and the $\mathcal{L}_{adv}^{cls}$ and $\mathcal{L}_{adv}^{iou}$ make consid-

Table 6: The comparison results of different loss schemes.

Method	ASR (%)			
	YOLO-V5	Faster RCNN	SSD	Mask RCNN
$\mathcal{L}_{adv}^{cls}$	82.83	65.25	73.20	67.97
$\mathcal{L}_{adv}^{obj}$	46.16	49.69	61.87	42.83
$\mathcal{L}_{adv}^{iou}$	**90.96**	55.84	71.19	54.87
$\mathcal{L}_{adv}^{cls+obj}$	84.49	68.16	76.05	71.99
$\mathcal{L}_{adv}$	88.59	**73.54**	**79.38**	**75.60**
$\mathcal{L}_{total}$	87.67	72.11	78.42	75.16

Table 7: The comparison result of different texture initialization.

Method	ASR (%)			
	YOLO-V5	Faster RCNN	SSD	Mask RCNN
basic	84.56	68.41	**80.65**	69.44
random	87.67	72.11	78.42	75.16
zero	**89.9**	**74.37**	79.79	**75.81**

erable contributions to the attack.

Effectiveness of different initialization. Initialization plays an important role in deep learning, we investigate the influence on the initialization of adversarial camouflage in this part. We mainly compare three different initialization ways: the original basic texture of the 3D model, random noise and zero. As shown in Table 7, we can observe that the performance of the zero initialization is superior over the other two ways, giving the highest ASR 89.9% over YOLO-V5, the performance of the original 3D model texture is worse than other two ways on attack Faster RCNN and Mask RCNN, which is less than 70% (68.41% for Faster RCNN, 69.44% for Mask RCNN). This phenomenon may be attributed to that we adopt the gradient descent algorithm to guide the adversarial camouflage update, the random noise initialization gives a prior knowledge that may directly mislead the detector, resulting in wrong optimization directions. In conclusion, the initialization has limit influence on the attack performance, and thus we select the random initialization for balancing the attack and naturalness.

Conclusion

In this paper, we propose an end-to-end attack method to generate a full-coverage adversarial camouflage in the physical world. Specifically, we first utilize a neural renderer to render our camouflage texture into a 3D vehicle model. Then we devise a transformation function to transfer the rendered vehicle into the photo-realistic simulation scenarios to simulate the complex real-world environmental conditions. Finally, we devise an adversarial loss functions to guide the optimization of camouflage with a gradient descent algorithm. Extensive experiments demonstrated that our FCA outperforms other advanced attacks, and achieves higher attack performance on both digital and physical attacks. Therefore, our method can bridge the gap between digital attacks and physical attacks as much as possible. We hope the proposed FCA could provide an interesting direction of physical attack for future work.

Acknowledgments

This work was supported in part by National Natural Science Foundation of China under Grant No.11725211, 52005505, and 62001502, and Post-graduate Scientific Research Innovation of Hunan Province under Grant No.CX20200006.

References

Athalye, A.; Engstrom, L.; Ilyas, A.; and Kwok, K. 2018. Synthesizing robust adversarial examples. In *International conference on machine learning*, 284–293. PMLR.

Brown, T. B.; Mané, D.; Roy, A.; Abadi, M.; and Gilmer, J. 2017. Adversarial patch. *arXiv preprint arXiv:1712.09665*.

Chen, S.-T.; Cornelius, C.; Martin, J.; and Chau, D. H. P. 2018. Shapeshifter: Robust physical adversarial attack on faster r-cnn object detector. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 52–68. Springer.

Dosovitskiy, A.; Ros, G.; Codevilla, F.; Lopez, A.; and Koltun, V. 2017. CARLA: An open urban driving simulator. In *Conference on robot learning*, 1–16. PMLR.

Elsayed, G. F.; Shankar, S.; Cheung, B.; Papernot, N.; Kurakin, A.; Goodfellow, I.; and Sohl-Dickstein, J. 2018. Adversarial examples that fool both computer vision and time-limited humans. In *NeurIPS*.

Eykholt, K.; Evtimov, I.; Fernandes, E.; Li, B.; Rahmati, A.; Xiao, C.; Prakash, A.; Kohno, T.; and Song, D. 2018. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1625–1634.

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 2961–2969.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Huang, L.; Gao, C.; Zhou, Y.; Xie, C.; Yuille, A. L.; Zou, C.; and Liu, N. 2020. Universal physical camouflage attacks on object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 720–729.

Jocher, G.; Stoken, A.; Borovec, J.; NanoCode012; Chaurasia, A.; TaoXie; Changyu, L.; V, A.; Laughing; tkianai; yxNONG; Hogan, A.; lorenzomammana; AlexWang1900; Hajek, J.; Diaconu, L.; Marc; Kwon, Y.; oleg; wanghaoyang0106; Defretin, Y.; Lohia, A.; ml5ah; Milanko, B.; Fineran, B.; Khromov, D.; Yiwei, D.; Doug; Durgesh; and Ingham, F. 2021. ultralytics/yolov5: v5.0 - YOLOv5-P6 1280 models, AWS, Supervise.ly and YouTube integrations.

Kato, H.; Ushiku, Y.; and Harada, T. 2018. Neural 3d mesh renderer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3907–3916.

Kurakin, A.; Goodfellow, I.; Bengio, S.; et al. 2016. Adversarial examples in the physical world.

Lee, M.; and Kolter, Z. 2019. On physical adversarial patches for object detection. *arXiv preprint arXiv:1906.11897*.

Liu, A.; Wang, J.; Liu, X.; Cao, B.; Zhang, C.; and Yu, H. 2020. Bias-based universal adversarial patch attack for automatic check-out. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16*, 395–410. Springer.

Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; and Berg, A. C. 2016. Ssd: Single shot multibox detector. In *European conference on computer vision*, 21–37. Springer.

Lu, J.; Sibai, H.; and Fabry, E. 2017. Adversarial examples that fool detectors. *arXiv preprint arXiv:1712.02494*.

Mahendran, A.; and Vedaldi, A. 2015. Understanding deep image representations by inverting them. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5188–5196.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 32*, 8024–8035. Curran Associates, Inc.

Qin, X.; Zhang, Z.; Huang, C.; Dehghan, M.; Zaiane, O.; and Jagersand, M. 2020. U2-Net: Going Deeper with Nested U-Structure for Salient Object Detection. *Pattern Recognition*, 106: 107404.

Redmon, J.; Divvala, S.; Girshick, R.; and Farhadi, A. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 779–788.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28: 91–99.

Selvaraju, R. R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; and Batra, D. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, 618–626.

Sharif, M.; Bhagavatula, S.; Bauer, L.; and Reiter, M. K. 2016. 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*, 1528–1540.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Thys, S.; Van Ranst, W.; and Goedemé, T. 2019. Fooling automated surveillance cameras: adversarial patches to attack person detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 0–0.

Wang, J.; Liu, A.; Yin, Z.; Liu, S.; Tang, S.; and Liu, X. 2021a. Dual Attention Suppression Attack: Generate Adversarial Camouflage in Physical World. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8565–8574.

Wang, Y.; Lv, H.; Kuang, X.; Zhao, G.; Tan, Y.-a.; Zhang, Q.; and Hu, J. 2021b. Towards a physical-world adversarial patch for blinding object detection models. *Information Sciences*, 556: 459–471.

Wu, T.; Ning, X.; Li, W.; Huang, R.; Yang, H.; and Wang, Y. 2020a. Physical adversarial attack on vehicle detector in the carla simulator. arXiv:2007.16118.

Wu, Z.; Lim, S.-N.; Davis, L. S.; and Goldstein, T. 2020b. Making an invisibility cloak: Real world adversarial attacks on object detectors. In *European Conference on Computer Vision*, 1–17. Springer.

Xiao, C.; Yang, D.; Li, B.; Deng, J.; and Liu, M. 2019. Meshadv: Adversarial meshes for visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6898–6907.

Xie, C.; Zhang, Z.; Zhou, Y.; Bai, S.; Wang, J.; Ren, Z.; and Yuille, A. L. 2019. Improving transferability of adversarial examples with input diversity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2730–2739.

Zeng, X.; Liu, C.; Wang, Y.-S.; Qiu, W.; Xie, L.; Tai, Y.-W.; Tang, C.-K.; and Yuille, A. L. 2019. Adversarial attacks beyond the image space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4302–4311.

Zhang, Y.; Foroosh, H.; David, P.; and Gong, B. 2018. CAMOU: Learning physical vehicle camouflages to adversarially attack detectors in the wild. In *International Conference on Learning Representations*.