Adversarial Texture for Fooling Person Detectors in the Physical World

Zhanhao Hu[1] Siyuan Huang[1] Xiaopei Zhu[2,1] Fuchun Sun[1] Bo Zhang[1] Xiaolin Hu[1,3,4]*
[1]Department of Computer Science and Technology, Institute for Artificial Intelligence,
State Key Laboratory of Intelligent Technology and Systems, BNRist, Tsinghua University, Beijing, China
[2]School of Integrated Circuits, Tsinghua University, Beijing, China
[3]IDG/McGovern Institute for Brain Research, Tsinghua University, Beijing, China
[4]Chinese Institute for Brain Research (CIBR), Beijing, China
{huzhanha17, zxp18}@mails.tsinghua.edu.cn
{siyuanhuang, fcsun, dcszb, xlhu}@mail.tsinghua.edu.cn

Abstract

Nowadays, cameras equipped with AI systems can capture and analyze images to detect people automatically. However, the AI system can make mistakes when receiving deliberately designed patterns in the real world, i.e., physical adversarial examples. Prior works have shown that it is possible to print adversarial patches on clothes to evade DNN-based person detectors. However, these adversarial examples could have catastrophic drops in the attack success rate when the viewing angle (i.e., the camera's angle towards the object) changes. To perform a multi-angle attack, we propose Adversarial Texture (AdvTexture). AdvTexture can cover clothes with arbitrary shapes so that people wearing such clothes can hide from person detectors from different viewing angles. We propose a generative method, named Toroidal-Cropping-based Expandable Generative Attack (TC-EGA), to craft AdvTexture with repetitive structures. We printed several pieces of cloth with AdvTexure and then made T-shirts, skirts, and dresses in the physical world. Experiments showed that these clothes could fool person detectors in the physical world.

1. Introduction

Recent works have shown that Deep Neural Networks (DNNs) are vulnerable to the adversarial examples crafted by adding subtle noise to the original images in the digital world [5, 8, 10, 18, 22–24, 31], and that the DNNs can be attacked by manufactured objects in the physical world [1, 4, 9, 29]. These manufactured objects are called *physical adversarial examples*. Recently, some methods based on patch attacks [29] have been proposed to evade person detectors [14, 15, 32, 34, 35, 37]. Specifically, Thys et al. [32] proposed to attach a patch to a cardboard. By holding the cardboard in front of the camera, the person cannot be detected by the person detectors. Xu et al. [35] proposed an adversarial T-shirt printed with adversarial patches. The person wearing the T-shirt can also evade person detectors. These works impose considerable threats to the widely deployed deep learning-based security systems. It urges researchers to re-evaluate the safety and reliability of these systems.

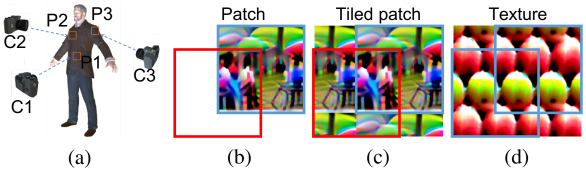


Figure 1. Illustration of the attacks at different viewing angles. (a) The camera captures different parts (P1, P2, P3) of the clothes when set to different viewing angles (C1, C2, C3). (b-d) The boxes are the possible areas that the camera may capture. The blue ones indicate the most effective areas for attack, while the red ones are less effective.

However, the person detector attack methods mentioned above are effective only when the adversarial patches face the camera. Apparently, a single adversarial patch on a piece of clothing is hard to attack detectors at multiple viewing angles, as the camera may only capture a segment of the heavily deformed patch (Fig. 1a and Fig. 1b). We call this the *segment-missing* problem. A naive extension is to cover the clothing with multiple patches (e.g., tiling the patches tightly on the clothing; see Fig. 1c). However, it cannot totally solve the segment-missing problem, because the camera will capture several segments belonging to different patch units, making the attack inefficient. Another straightforward solution is to build a 3D model of a human

*Corresponding author.

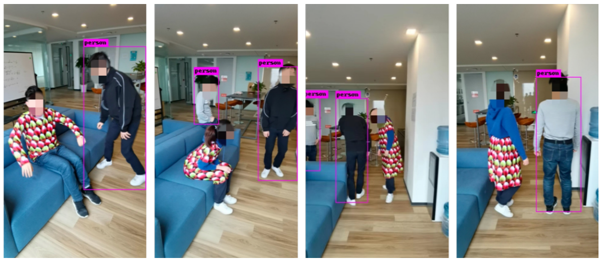


Figure 2. Visualization of the adversarial effectiveness of AdvTexture when attacking YOLOv2. A dress, a T-shirt, and a skirt are tailored from a large polyester cloth material covered with the AdvTexture. The persons wearing the clothes failed to be detected by the detector.

body and a specific piece of clothing to render in different viewing angles as previous work [1] did. However, the clothes are non-rigid, and current 3D rendering techniques have difficulties in modeling the natural deformation of clothes in the real world. For example, Wang et al. [33] rendered 3D logos on flat areas (front and back) of 3D human meshes, but the Attack Success Rate (ASR) decreased when applying to unseen meshes.

To solve the problem, we propose the idea of using Adversarial Texture (AdvTexture). Unlike the patch-based attacks, AdvTexture can be generated in arbitrary size, thus can cover any cloth in any size. We require that any local part of the texture has adversarial effectiveness (Fig. 1d). Then, when the clothes are covered with AdvTexture, every local area caught by the camera can attack the detectors, which solves the segment-missing problem.

Towards this goal, we propose a two-stage generative method, Toroidal-Cropping-based Expandable Generative Attack (TC-EGA), to craft AdvTexture. In the first stage, we train a fully convolutional network (FCN) [21, 30] as the generator to produce textures by sampling random latent variables as input. Unlike the conventional architecture of the generator in GAN [16, 25], we use convolutional operation in every layer, including the latent variable. Therefore, the latent variable is a tensor with spatial dimensions, which enables the generator to generate texture in multiple sizes as long as we expand the latent variable along the spatial dimensions. In the second stage, we search the best local pattern of the latent variable with a cropping technique—Toroidal Cropping (TC). After optimization, we can generate a large enough latent variable by tiling the local pattern. We input it to the FCN and finally get AdvTexture.

We implemented TC-EGA to attack various person detectors, and realized AdvTextures in the physical world. Fig. 2 shows some example attacks targeting YOLOv2. Our experiments showed that the clothes made from such textures significantly lowered the detection performance of different detectors.

2. Related Work

Earlier works about adversarial examples [10, 18, 31] focused on digital attacks. Small adversarial noises can be added to the original images and make DNNs output wrong predictions, posing significant safety concerns to DNNs.

Compared to digital adversarial attacks, physical adversarial attacks pose more risks in specific scenarios. Several methods [1, 4, 9, 29] have been proposed to attack image classification models physically. Sharif et al. [29] designed a pair of glasses to attack face-recognition systems. Athalye et al. [1] generated robust 3D adversarial objects by introducing the Expectation over Transformation (EoT) [1] method. Brown et al. [4] deceived image classifiers by placing adversarial patches in the neighborhood of the objects. Evtimov et al. [9] misled road-sign classification by adhering black and white stickers to signs.

Recently, several methods [14, 15, 32, 32–35] were proposed to attack the DNN-based person detection systems. Thys et al. [32] optimized an adversarial patch that can be attached to cardboard and held by a person. Huang et al. [15] propose Universal Physical Camouflage Attack (UPC) to fool the detectors by simulating 3D objects in virtual environments. Xu et al. [35] designed an adversarial T-shirt by introducing Thin Plate Spline (TPS) [2, 7] to simulate the deformation of clothes (e.g., wrinkles). Wu et al. [34] presented a systematic study of the attack on a range of detection models, different datasets, and objects. Wang et al. [33] masked the adversarial patch with preset logos and mapped it into 3D models. Hu et al. [14] used generative adversarial networks (GAN) [3, 16] to craft more natural-looking adversarial patches.

Some works [15, 33, 35] reported drops in the attack success rate when the viewing angles increased. According to Wang et al. [33], part of the patches will not be captured when the camera rotates drastically. It can lead to underestimating the threat, whereas the cameras can be placed anywhere in real-world scenarios.

3. Methods

We aim to generate textures in arbitrary size, and when the textures are printed on cloth, any patch extracted from the cloth are effective in adversarial attack. We first introduce an adversarial patch generator and then describe TC-EGA based on the patch generator.

3.1. Adversarial Patch Generator

Let τ denote the whole cloth that is covered with AdvTexture, and $\tilde{\tau}$ denote an extracted patch. We assume that $\tilde{\tau}$ follows a distribution p_{adv}, such that the probability $p_{adv}(\tilde{\tau})$ is higher when its adversarial effectiveness is more significant. We use an energy function $U(\tilde{\tau})$ to model

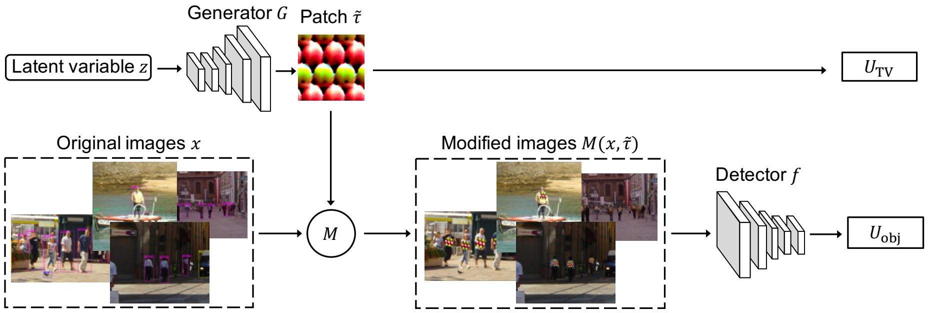


Figure 3. The pipeline of the adversary objective function.

such a distribution:

$$p_{adv}(\tilde{\tau}) = \frac{e^{-U(\tilde{\tau})}}{Z_U}, \tag{1}$$

where $Z_U = \int_{\tilde{\tau}} e^{-U(\tilde{\tau})} \mathrm{d}\tilde{\tau}$ is called partition function. However, it is hard to sample from $p_{adv}(\tilde{\tau})$ directly due to the partition function. Therefore, we use a parameterized generator $G_\varphi : z \to \tilde{\tau}$ to approximate $p_{adv}(\tilde{\tau})$, where $z \sim \mathcal{N}(0, I)$. We define $q_\varphi(\tilde{\tau})$ as the distribution of $\tilde{\tau} = G_\varphi(z)$, which can be written as

$$q_\varphi(\tilde{\tau}) = \int \delta(\tilde{\tau} - G_\varphi(z)) p_z(z)\, \mathrm{d}z, \tag{2}$$

where p_z is the probability density function (PDF) of the standard normal distribution $\mathcal{N}(0, I)$ and $\delta(\cdot)$ is the Dirac delta function. In order to represent $p_{adv}(\tilde{\tau})$ more accurately, we tune G_φ to minimize the KL divergence $\mathrm{KL}(q_\varphi(\tilde{\tau})||p_{adv}(\tilde{\tau}))$. With the aid of Deep InfoMax (DIM) [13] we have the following theorem:

Theorem 1 *Minizing* $\mathrm{KL}(q_\varphi(\tilde{\tau})||p_{adv}(\tilde{\tau}))$ *is equivalent to*

$$\min_{\varphi,\omega} \mathbb{E}_{\tilde{\tau}\sim q_\varphi(\tilde{\tau})}[U(\tilde{\tau})] - I^{\text{JSD}}_{\varphi,\omega}(\tilde{\tau}, z), \tag{3}$$

where

$$\begin{aligned} \mathcal{I}^{\text{JSD}}_{\varphi,\omega}(\tilde{\tau}, z) = {} & \mathbb{E}_{(\tilde{\tau},z)\sim q_\varphi^{\tilde{\tau},z}(\tilde{\tau},z)}[-\mathrm{sp}(-T_\omega(\tilde{\tau}, z))] \\ & - \mathbb{E}_{\tilde{\tau}\sim q_\varphi(\tilde{\tau}),z'\sim p_z(z')}[\mathrm{sp}(T_\omega(\tilde{\tau}, z'))], \end{aligned} \tag{4}$$

$q_\varphi^{\tilde{\tau},z}$ *denotes the joint distribution of* $\tilde{\tau}$ *and* z*, and* $\mathrm{sp}(t) = \log(1 + e^t)$ *is the softplus function.* T_ω *is a scalar function modeled by a neural network whose parameter* ω *must be optimized together with the parameter* φ*.*

See *Supplementary Materials* for the proof.

The objective function in Eq. (3) consists of two terms. The first term $\mathbb{E}_{\tilde{\tau}\sim q_\varphi(\tilde{\tau})}[U(\tilde{\tau})]$ is called *Adversary Objective Function* because minimizing it improves the the adversarial effectiveness of the generated patches. The second term $-\mathcal{I}^{\text{JSD}}_{\varphi,\omega}(\tilde{\tau}, z)$ is called *Information Objective Function* because minimizing it is equivalent to maximizing the mutual information of z and $\tilde{\tau}$ [13], which requires different latent variables to generate different patches.

3.1.1 The Adversary Objective Function

The adversary objective function $\mathbb{E}_{\tilde{\tau}\sim q_\varphi(\tilde{\tau})}[U(\tilde{\tau})]$ can be estimated by sampling z and generating $\tilde{\tau}$:

$$\frac{1}{N}\sum_{i=1}^{N}[U(G_\varphi(z_i))], \tag{5}$$

where $\{z_i\}$ are the latent variables sampled from $\mathcal{N}(0, I)$, and N denotes the total number of the samples.

Now we need to set an appropriate energy function such that lowering the energy leads to detection failure of a person detector. We notice that detectors output multiple bounding boxes with a confidence score for each box when receiving an image. The boxes whose confidence scores are lower than a pre-specified threshold will then be filtered out. Therefore we choose the expectation of the confidence scores over boxes as a part of the energy function $U(\tilde{\tau})$. Then minimizing the adv object function will lower the confidence scores of the boxes, which makes the boxes easily to be filtered out.

Specifically, we randomly generate patches in every step, and apply a set of physical transformations such as randomizing the scales, contrast, brightness and additional noise according to Expectation over Transformation (EoT) [29, 32]. We also incorporate random Thin Plate Spin (TPS) [7, 35] deformation as an additional random transformation. We then attach the patches randomly to the persons according to the predicted boxes on the images x from the training set. We use $M(x, \tilde{\tau})$ to denote the above process, and obtain the modified images which are then be sent into the target detector. This part of the energy function is thus defined as

$$U_{\text{obj}} = \mathbb{E}_{x,M}[f(M(x, \tilde{\tau}))], \tag{6}$$

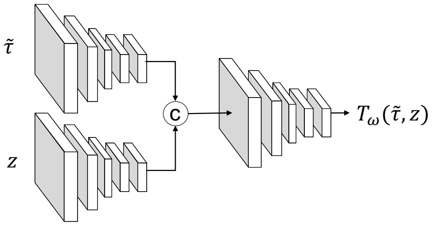


Figure 4. The architecture of the auxiliary network T_ω. It has two inputs, $\tilde{\tau}$ and z, and outputs a scalar value $T_\omega(\tilde{\tau}, z)$. The operation c in the figure stands for concatenation.

where f denotes confidence scores of the boxes predicted by the target detector.

We use a differentiable variation of total variance (TV) loss [29] as another part of the energy function to encourage the patches to be smoother:

$$U_{\mathrm{TV}} = \sum_{i,j} |\tau_{i,j} - \tau_{i+1,j}| + |\tau_{i,j} - \tau_{i,j+1}| \tag{7}$$

Together, we form the energy function as

$$U(\tilde{\tau}) = \frac{1}{\beta}(U_{\mathrm{obj}} + \alpha U_{\mathrm{TV}}), \tag{8}$$

where α and β are coefficients. See Fig. 3 for the illustration. When minimizing the adversary objective function, each part of the energy function will be minimized together.

3.1.2 The Information Objective function

As described in Eq. (4), we use an auxiliary network T_ω to increase the mutual information of z and $\tilde{\tau}$. We illustrate the architecture of T_ω in Fig. 4. Eq. (4) has two terms, and estimating each of them needs random sampling. Following the previous work [13], to estimate the first term, we first sample z from $\mathcal{N}(0, I)$, and then generated $\tilde{\tau}$ by $G_\varphi(z)$ in each training step. To estimate the second term, we keep $\tilde{\tau}$ and resample z.

During training, we minimize the adversarial objective function and the information objective function simultaneously. Therefore, the distribution q_φ can approximate to p_{adv}, which means the the generated patches $\tilde{\tau}$ can be adversarial to the target detector.

3.2. Toroidal-Cropping-based Expandable Generative Attack

In Sec. 3.1, we have described the method to train a generator for adversarial patches $\tilde{\tau}$. In this section, we used TC-EGA to generate AdvTextures τ based on the adversarial patch generator. We leverage a specific network architecture and a sample technique to extend adversarial patches to adversarial textures. TC-EGA has two stages. In the first stage, we train a fully convolutional network (FCN) [21,30] to help sample from the distribution of adversarial textures. In the second stage, we search the best latent representation to yield the most effective adversarial texture.

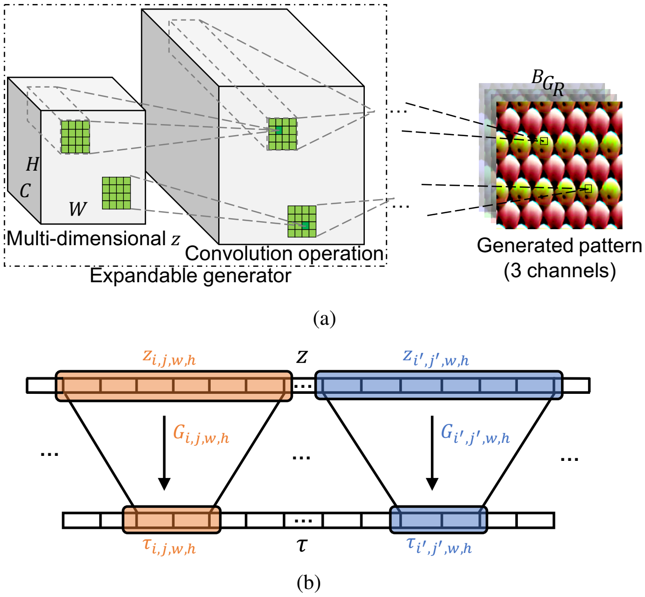


Figure 5. (a) Illustration of the FCN generator. All layers of the generator network are convolutional layers with zero padding, including the first layer. (b) Each patch $\tau_{i,j,w,h}$ extracted from position i, j can be regarded as the output of a sub-generator $G_{i,j,w,h}$ when the input is $z_{i,j,w,h}$.

3.2.1 Stage One: Train an Expandable Generator

We aim to train a generator so that it can generate patches in arbitrary size easily by taking a random z as input. The critical point is to endow the generator with translation invariant property by constructing an FCN, where all layers are convolutional layers with zero padding, including the first layer that inputs the latent variable (See Fig. 5a). The latent variable is a $B \times C \times H \times W$ tensor where B is the batch size, C is the number of channels, and H, W are height and width, respectively.

Here we show the reason for using FCN. We assume that the overall texture τ is generated by a global generator $G : z \to \tau$ with hidden variable $z \sim \mathcal{N}(0, I)$. We denote the extracted patch by $\tau_{i,j,w,h}$ whose center is located at the position (i, j) of the overall texture and has a shape of (w, h). Moreover, the patch $\tau_{i,j,w,h}$ can be regarded as the output of a sub-generator $G_{i,j,w,h} : z_{i_z,j_z,w_z,h_z} \to \tau_{i,j,w,h}$, where z_{i_z,j_z,w_z,h_z} is the component of z that consists of all the elements dependent to $\tau_{i,j,w,h}$ (see Fig. 5b). Assuming that $\tau_{i,j,w,h}$ follows a distribution $\mathcal{T}_{i,j,w,h}$. We have the following theorem and corollary.

Theorem 2 *Let $\tau_1 = G_1(z_1)$, $\tau_2 = G_2(z_2)$, $z_1 \sim \mathcal{Z}_1$, $z_2 \sim \mathcal{Z}_2$, $\tau_1 \sim \mathcal{T}_1$, $\tau_2 \sim \mathcal{T}_2$. If $\mathcal{Z}_1$ is identical to $\mathcal{Z}_2$ and G_1 is equivalent to G_2, then $\mathcal{T}_1$ is identical to $\mathcal{T}_2$.*

Corollary 2.1 *$G_{i,j,w,h}$ and $\mathcal{T}_{i,j,w,h}$ are irrelevant to i, j, i.e., $G_{i,j,w,h} = G_{w,h}$ and $\mathcal{T}_{i,j,w,h} = \mathcal{T}_{w,h}$, if G is an FCN and the input $z \sim \mathcal{N}(0, I)$.*

See *Supplementary Materials* for the proofs. Therefore, as long as the sub-generator $G_{w,h}$ is trained to approximate the distribution of $\mathcal{T}_{w,h}$ to p_{adv}, any patch extracted from the overall texture with shape (w, h) also approximately follows p_{adv}, i.e., it has adversarial effectiveness. Moreover, due to the translation invariant property of the convolutional operation, the sub-generator $G_{w,h}$ and the global generator can share the same architecture and parameters except for the different spatial shape H and W of the latent variable z. As a result, we only need to train a small generator.

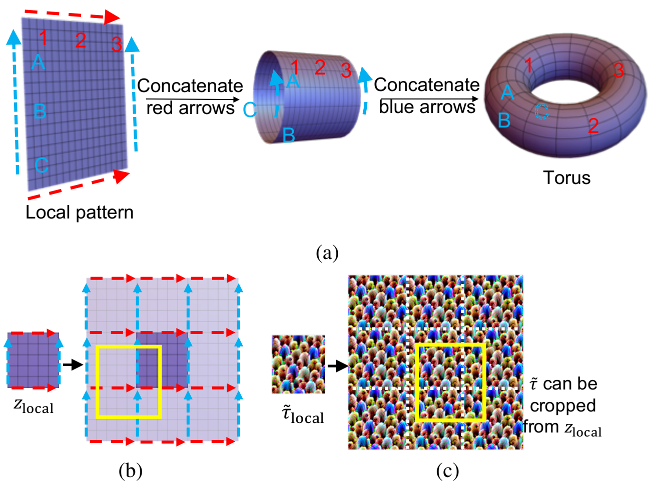
Note that the height H and width W of the hidden variable z can not be too small, otherwise the output will be too small to crop a patch in spatial shape (w, h). We denote the minimum spatial sizes by H_{min} and W_{min}. During training, we sampled a small z in shape $B \times C \times H_{\text{min}} \times W_{\text{min}}$ and generated the corresponding patches in each training step. After that, we can produce different textures of arbitrary sizes by randomizing z with any $H \geq H_{\text{min}}$ and $W \geq W_{\text{min}}$.

3.2.2 Stage Two: Find the Best Latent Pattern

After training, the generator can generate different textures by sampling latent variables. In order to find the best texture for adversarial attacks, we propose to go one step further, that is, to optimize the latent variable with the parameters of the generator frozen. However, since the texture has no specific shape and the size of the latent variable needs to be large enough to produce a large textured cloth, directly optimizing the latent variable is difficult.

Inspired by the unfolding of torus in topology which supports up-down and left-right continuation [11] (Fig. 6a), we introduce the Toroidal Cropping (TC) technique, which aims to optimize a local pattern z_{local} as a unit such that the final latent variable z can be produced by tiling multiple identical units. In detail, z_{local} can be parameterized as a tensor in shape $B \times C \times L \times L$ with a shape hyper-parameter L, which can be regarded as the unfolded plane of a two-dimensional torus $\mathbb{T}^2$ in topology (Fig. 6a). Therefore the latent variable in arbitrary shape can be cropped from z_{local} in a recursive manner (Fig. 6b), which can be regarded as cropping on the torus. We denote such crop operation by $\text{Crop}_{\text{torus}}$.

During optimization, we randomly sample the latent variables z_{sample} in shape $B \times C \times H_{\text{min}} \times W_{\text{min}}$ by such cropping technique. Since we only consider the adversarial effectiveness in this stage, we generated patches by z_{sample} and minimized the adversary loss (Eq. (5)). After optimization, one can produce a latent variable with arbitrary size by tiling z_{local}.



(a)

(b)

(c)

Figure 6. Illustration of Toroidal Cropping. (a) By first concatenating its horizontal edges (red arrow) and then concatenating the vertical edges (blue arrow), the local pattern can be folded to a torus. (b) The latent variable in arbitrary shape can be created by tiling the local pattern side by side, thus the variable cropped at the junctions is equivalent to that cropped on the torus, meaning the pattern is still continuous. (c) This cropping technique also applies to the pixel space. See Sec. 4.3 for this variant.

4. Experiment settings

4.1. Subjects

We recruited three subjects (mean age: 24.0; range: $21 - 26$; two males and one female) to collect physical test set. The recruitment and study procedures were approved by the Department of Psychology Ethics Committee, Tsinghua University, Beijing, China.

4.2. Dataset

We employed the Inria Person dataset [6] as our training set. It is a dataset for pedestrian detection, which consists of 614 images for training and 288 for testing. We evaluated the patch-based attack on the Inria test set. For physical evaluation, we produced clothes covered with different adversarial textures. Three subjects wore different adversarial clothes and turned a circle slowly in front of a camera which was fixed at 1.38 meters above the ground. The distance between the camera and person is fixed to 2 m unless otherwise specified. We recorded two videos for each subject and each adversarial piece of clothing. One of the video was recorded indoor (lab room), and the other was recorded in outdoor (brick walkway). We then extracted 32 frames from each video. We recorded $3 \times 2 = 6$ videos and collected $6 \times 32 = 192$ frames for each adversarial piece of clothing. we labeled them manually to construct a test set.

4.3. Baseline Methods

We evaluated the adversarial patches produced by Thys et al. [32] and Xu et al. [35], and named them by *AdvPatch* and *AdvTshirt*, respectively. We copied the patterns from their original papers. We also tiled AdvPatch and AdvTshirt to form textures with repeated patterns. These two variants are called *AdvPatchTile* and *AdvTshirtTile*. In addition, we evaluated a texture with repetitive random colors, which is denoted by *Random*

Moreover, TC-EGA has multiple components and some of them could be applied separately to craft adversarial textures. To investigate the performance of each component, we designed three variants of TC-EGA, as described below.

Expandable Generative Attack (EGA) We trained an FCN as the first stage of TC-EGA without optimizing the best latent variable. During evaluation, the final texture can be generated by a latent variable in arbitrary size and sampled from a standard normal distribution.

Toroidal Cropping Attack (TCA) We directly optimized the texture instead of training an FCN to generate texture. Specifically, we initialized a local texture pattern of 300×300 pixels, and randomly extracted a patch by size 150×150 from the texture by Toroidal Cropping in each optimization step.

Random Cropping Attack (RCA) We directly optimized a large patch whose size is fixed. We initialized the large patch and randomly cropped a small patch by size 150×150 during optimization. This method is named Random Cropping Attack (RCA). We implemented two attacks, RCA2× and RCA6×, where the sizes of the large patches are 300×300 and 900×900, respectively.

4.4. Implementation Details

We crafted AdvTexture to mainly fool YOLOv2 [26], YOLOv3 [27], Faster R-CNN [28] and Mask R-CNN [12]. The detectors were pre-trained on MS COCO dataset [20]. Their outputs were filtered to output the person class only.

For each target detector, we first extracted the predicted bounding boxes on the images from the training set with a Non-Maximum Suppression (NMS) threshold 0.4. We chose the boxes whose confidence was larger than a certain threshold (0.5 for YOLOv2 and YOLOv3, and 0.75 for Faster and Mask R-CNN). We additionally filtered out boxes with areas smaller than 0.16% of the entire images for Faster and Mask R-CNN. Then, as we described in Sec. 3.1.1, we attached the extracted patches to the persons and input the modified images to the detector during optimization.

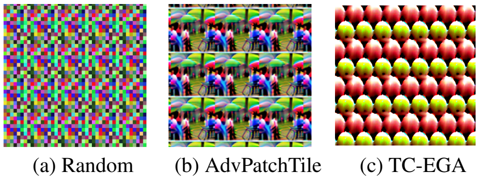

(a) Random (b) AdvPatchTile (c) TC-EGA

Figure 7. Visualization of different textures. (a) The texture with repetitive random colors. (b) The texture formed by tiling an adversarial patch [32] repeatedly. (c) The texture produced by TC-EGA to attack YOLOv2.

Moreover, we applied the Adam [17] optimizer to optimize parameters in both stages. The hyper-parameters are listed as follows. (1) Stage one: The initial learning rate to train the generator was 0.001. The generator was a 7-layer FCN whose input was the latent variable z with size $B \times 128 \times 9 \times 9$. The size of the corresponding output was $B \times 3 \times 324 \times 324$, where the second dimension stands for the RGB channels. (2) Stage two: We optimized a local latent variable z_{local} with size $1 \times 128 \times 4 \times 4$, followed by the Toroidal Cropping technique to produce samples of z with size $B \times 128 \times 9 \times 9$. The learning rate of the optimization was 0.03.

To physically implement AdvTexture, we printed the texture on a polyester cloth material by digital textile printing. Afterwards, we hired a professional tailor to produce adversarial clothes including T-shirts, skirts and dresses.

5. Results

Fig. 7 shows some textures obtained by different methods, and more can be found in *Supplementary Materials*.

5.1. Patch-Based Attack in the Digital World

We first evaluated the attacks in the form of the patch-based attack in the digital world. Specifically, we randomly extracted patches from the textures when evaluating most methods except for AdvPatch and AdvTexture. We denote such patches by *resampled* patches. We then attached the patches to the images from the Inria test set the same way as crafting the adversarial patches. We used the bounding boxes proposed by the target detectors on the original test images with a confidence threshold of 0.5 as the ground truth. We computed the average precision (AP) of the proposed bounding boxes on the modified test images to measure the adversarial effectiveness. Note that lower AP indicates stronger attack.

Tab. 1 presents the AP of YOLOv2 in different conditions. *clean* denotes the AP on the original test set. Since

Method	AP	Expandable	Resampled
Clean	1.000		
Random	0.963	✓	✓
AdvPatch [32]	**0.352**	✗	✗
AdvPatchTile	0.827	✓	✓
AdvTshirt [35]	0.744*	✗	✗
AdvTshirtTile	0.844	✓	✓
TC-EGA	**0.362**	✓	✓
EGA	0.470	✓	✓
TCA	0.664	✓	✓
RCA2×	0.606	✗	✓
RCA6×	0.855	✗	✓

Table 1. The APs of YOLOv2 under different attacks on Inria test set. *Expandable* denotes whether the methods can produce textures in arbitrary size. *Resampled* denotes whether the patches are randomly extracted.

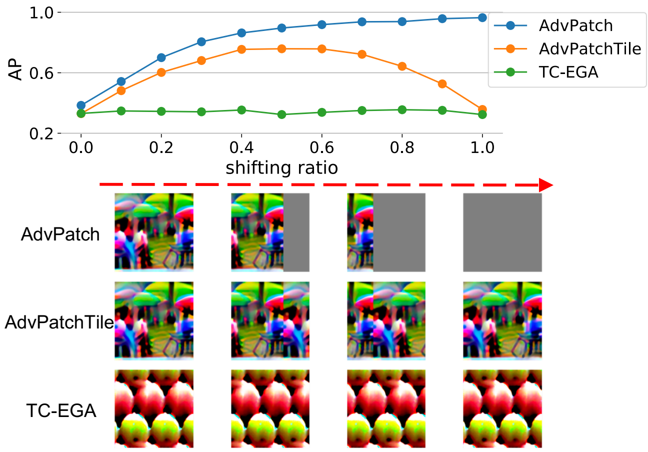


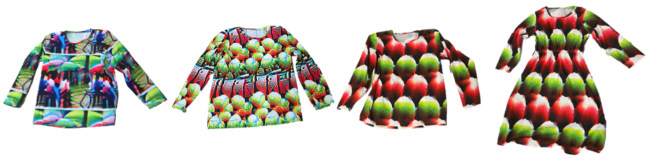
Figure 8. Numerical study of the segment-missing problem. The patches are cropped near the original patches with a shifting ratio. For AdvPatch as an example, the shifting ratio is 0.0 when the cropped patch is precisely the original patch. The shifting ratio is 1.0 when the original patch is shifted totally outside the cropping range.

we used the detector's prediction on the original images as the ground truth, the AP is 1.000. The AdvPatch lowered the AP of YOLOv2 to 0.352[1].

Compared to AdvPatch, the expandable variant AdvPatchTile increases the AP from 0.352 to 0.827. Since AdvTshirt was trained on a different dataset (its authors' private dataset), it only got an AP of 0.744. Similarly, AdvTshirtTile increases the AP to 0.844. We attribute the increase to the segment-missing problem. Compared to its variants, TC-EGA got the lowest AP 0.362, which was also the lowest among all the resampled patches. AdvPatch made the AP slightly lower than TC-EGA. However, it is not expandable and thus unsuitable for the attack at multiple viewing angles. Moreover, EGA decreased the AP to 0.470, TCA created expandable patches with AP 0.664. It was lower than AdvPatchTile, which indicates the effectiveness of the Toroidal Cropping technique. Moreover, RCA6× was much worse than RCA2×, which indicates difficulties in optimizing a large patch.

We further investigated the segment-missing problem by evaluating the adversarial effectiveness of the patches which is cropped at shifted positions (See Fig. 8). The patch-based attack, AdvPatch, became less effective when the shifting ratio increased. Tiling the patches alleviated the problem, but was still problematic. The texture generated by TC-EGA was robust during shifting.

The results of other detectors attacked by TC-EGA in the digital world are shown in *Supplementary Materials*.



(a) AdvPatchTile (b) AdvTshirtTile (c) YOLOv2 T-shirt (d) YOLOv2 dress

Figure 9. Real-world adversarial clothes.

5.2. Attack in the Physical World

Fig. 9 shows the produced clothes by different methods, and more can be found in *Supplementary Materials*.

We first compared different methods on YOLOv2. Since the boxes predicted by the detectors can be filtered by a particular confidence threshold, we plotted the recall-confidence curve in Fig. 10 and showed their APs in the legend. Remember that *recall* denotes the fraction of the boxes that are successfully retrieved. These boxes are filtered by a confidence threshold. Therefore, for each particular confidence threshold, lower recall denotes better adversarial effectiveness. From Fig. 10, the tiled variants of both AdvPatch and AdvTshirt were more effective than the original method. TC-EGA outperformed among all the methods by the lowest recall-confidence curve and the lowest AP.

Moreover, we used another metric to evaluate the effectiveness of the attacks. Specifically, for each input image we collected the target detector's predicted bounding boxes whose confidence score is larger than a certain confidence threshold. As long as one of these boxes has an Intersection over Union (IoU) with the ground-truth box greater than 0.5, the detector is considered to have correctly detected. We defined Attack Success Rate (ASR) as the fraction of the test images that are not correctly predicted. Since the

[1] We reproduced an adversarial patch according to their released code https://gitlab.com/EAVISE/adversarial-yolo. The reproduced patch got an AP of 0.378. We used the patch that is copied from their paper in all the experiments.

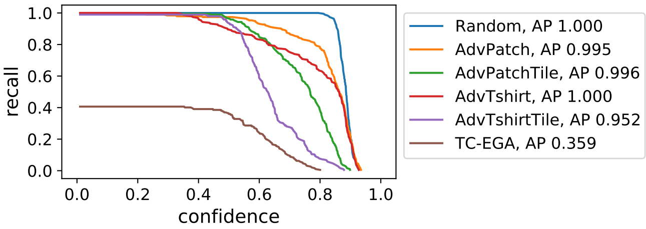


Figure 10. The recall v.s confidence curves and APs on the physical adversarial test set. The target network is YOLOv2.

Clothing	Random	Tshirt	Skirt	Dress
mASR	0.092	0.771	0.287	0.893

Table 2. The mASRs of different adversarial clothes.

ASR is relevant to the confidence threshold, we calculated the mean value of the ASR, namely mASR, under multiple thresholds. The thresholds were $0.1, 0.2, ..., 0.9$ in our experiment.

Fig. 11 presents the mASRs in multiple viewing angles. Compared to the random texture, AdvPatch and AdvTshirt was effective when the persons faced the camera (the viewing angle is 0° or 360° in the figure). However, the mASRs of these two methods decreased when the viewing angle increased, which manifests the segment-missing problem. The tiled variants of both methods had some adversarial effectiveness in multiple viewing angles, while the mASRs were lower than 0.5 in almost every viewing angles. TC-EGA outperformed the other methods at almost every viewing angle. The mASR is approximately 1.0 at viewing angle 0° and 180°, indicating that the person can always evade the detector when confidence threshold is larger than 0.1. It was less effective when the viewing angle is close to 90° or 270° because the area captured by the camera were small at such viewing angles.

We investigated influence of the type of clothes and the distance between person and camera. From Tab. 2, the adversarial effectiveness varied when the texture was applied to different kinds of clothes. The attack was more effective when applying to larger clothes (e.g., dress), for more area of the texture was captured by the camera. Moreover, the adversarial clothes had comparable mASRs in both indoor and outdoor scenes (See *Supplementary Materials*). Their effectiveness dropped when far from the camera (See *Supplementary Materials*).

Tab. 3 presents the mASRs of the adversarial clothes to attack various detectors. From the table, TC-EGA obtained much higher mASR than Random. Moreover, the adversarial effectiveness remained when the adversarial clothes are transferred across different detectors. See *Supplementary Materials* for the details of the transfer study. In addition, we provide a video demo in *Supplementary Video*.

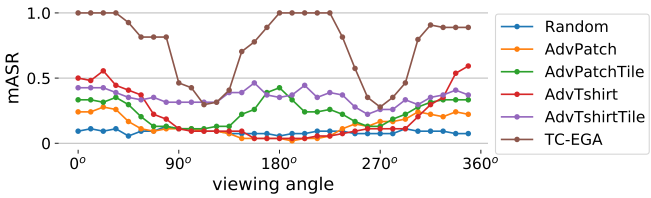


Figure 11. The mASRs of the attacks at multiple viewing angles.

Detector	YOLOv2	YOLOv3	FasterRCNN	MaskRCNN
Random	0.087	0.000^1	0.000	0.000
TC-EGA	0.743	0.701^1	0.930	0.855

[1] We scaled the size of the inputs by 50% before sending them to YOLOv3. See *Supplementary Materials* for the reason.

Table 3. The mASR of different detectors in the physical world.

6. Conclusions

We propose a method to craft AdvTextures to realize physical adversarial attacks on person detection systems. The main idea is to first train a expandable generator to generate AdvTexture by taking random input in a latent space, and then search the best local patterns of the latent variable for attack. The effectiveness of the AdvTexture is improved by optimizing the latent input. We physically implemented AdvTexture by printing it on a large cloth and making different T-shirts, skirts, and dresses. Those clothes, evaluated by our experiment in the physical world, were effective when the person wearing them turns around or changes postures.

Limitations Though the crafted texture targeting one detector can also attack another detector to some extent, the transferability is not very good. Model ensemble could be used to improve transferability.

Potential negative impact Adversarial research may cause potentially unwanted applications in the real-world community, such as camera security issues. Many defense methods based on previously exposed vulnerabilities have been proposed [10, 19, 36], which have improved the security level of our community and beneficially illustrated the value of research about attack.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China (Nos. U19B2034, 62061136001, 61836014) and the Tsinghua-Toyota Joint Research Fund.

References

[1] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In *International conference on machine learning*, pages 284–293. PMLR, 2018. 1, 2

[2] Fred L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Transactions on pattern analysis and machine intelligence*, 11(6):567–585, 1989. 2

[3] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 2

[4] Tom B Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017. 1, 2

[5] 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. 1

[6] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005. 5

[7] Gianluca Donato and Serge Belongie. Approximate thin plate spline mappings. In *European conference on computer vision*, pages 21–31. Springer, 2002. 2, 3

[8] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018. 1

[9] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1625–1634, 2018. 1, 2

[10] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015. 1, 2, 8

[11] Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 2002. 5

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 6

[13] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *International Conference on Learning Representations*, 2019. 3, 4

[14] Yu-Chih-Tuan Hu, Bo-Han Kung, Daniel Stanley Tan, Jun-Cheng Chen, Kai-Lung Hua, and Wen-Huang Cheng. Naturalistic physical adversarial patch for object detectors. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7848–7857, 2021. 1, 2

[15] Lifeng Huang, Chengying Gao, Yuyin Zhou, Cihang Xie, Alan L Yuille, Changqing Zou, and Ning Liu. Universal physical camouflage attacks on object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 720–729, 2020. 1, 2

[16] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019. 2

[17] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[18] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *International Conference on Learning Representations*, 2017. 1, 2

[19] Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Xiaolin Hu, and Jun Zhu. Defense against adversarial attacks using high-level representation guided denoiser. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1778–1787, 2018. 8

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 6

[21] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 2, 4

[22] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2574–2582, 2016. 1

[23] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 427–436, 2015. 1

[24] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 372–387. IEEE, 2016. 1

[25] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 2

[26] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017. 6

[27] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 6

[28] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016. 6

[29] 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, 2016. 1, 2, 3, 4

[30] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014. 2, 4

[31] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014. 1, 2

[32] Simen Thys, Wiebe Van Ranst, and Toon Goedemé. Fooling automated surveillance cameras: adversarial patches to attack person detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 1, 2, 3, 6, 7

[33] Yi Wang, Jingyang Zhou, Tianlong Chen, Sijia Liu, Shiyu Chang, Chandrajit Bajaj, and Zhangyang Wang. Can 3d adversarial logos cloak humans? *arXiv preprint arXiv:2006.14655*, 2020. 2

[34] Zuxuan Wu, Ser-Nam Lim, Larry S Davis, and Tom Goldstein. Making an invisibility cloak: Real world adversarial attacks on object detectors. In *European Conference on Computer Vision*, pages 1–17. Springer, 2020. 1, 2

[35] Kaidi Xu, Gaoyuan Zhang, Sijia Liu, Quanfu Fan, Mengshu Sun, Hongge Chen, Pin-Yu Chen, Yanzhi Wang, and Xue Lin. Adversarial t-shirt! evading person detectors in a physical world. In *European Conference on Computer Vision*, pages 665–681. Springer, 2020. 1, 2, 3, 6, 7

[36] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017. 8

[37] Xiaopei Zhu, Xiao Li, Jianmin Li, Zheyao Wang, and Xiaolin Hu. Fooling thermal infrared pedestrian detectors in real world using small bulbs. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI*, 2021. 1