

# Universal Shape of Strong Remote Adversarial Patches for Object Detection with Convolutional Neural Networks

## Supplementary Material

### 1. The Composition of the Supplementary Materials

We explain the materials included in the supplementary materials. The supplementary materials are composed of the following:

- Supplementary information that could not be covered in the main paper (**this document**)
- Following remote adversarial patches for “person” class:
  - Remote adversarial patches for evaluation (24 images, saved in the “Section4\_evaluation” folder). For each of the four CNNs, namely YOLOv2 [7], SSD [5], RetinaNet [4], and FCOS [11], we prepare remote adversarial patches in six different shapes.
  - Images with remote adversarial patches in autonomous driving simulator (72 images, saved in the “Section5\_simulator” folder). In the prepared images, four types of remote adversarial patches are applied to both YOLOv2 and SSD respectively.

From Sec. 2 onwards, we will provide supplementary information to complement the main paper.

### 2. Detailed Information for Analysis of Spreading Adversarial Patch’s Effect

In the main paper, we discussed how the effect of an adversarial patch spreads from a single pixel. In this section, we will provide supplementary information to further elaborate on the aforementioned discussion.

#### 2.1. Full Proof of Theroems

In the main paper, we define this matrix as  $D(n)$  after processing a  $3 \times 3$  filter size convolutional layer with a stride of one  $n$  times.  $D(n)$  varies according to Eq. (1).

$$[1] \rightarrow \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \rightarrow \frac{1}{81} \begin{bmatrix} 1 & 2 & 3 & 2 & 1 \\ 2 & 4 & 6 & 4 & 2 \\ 3 & 6 & 9 & 6 & 3 \\ 2 & 4 & 6 & 4 & 2 \\ 1 & 2 & 3 & 2 & 1 \end{bmatrix} \dots \quad (1)$$

Theorems 1 and 2 are stated regarding  $D(n)$ .

**Theorem 1.** The size of matrix  $D(n)$  is  $(2n+1) \times (2n+1)$ . The elements of  $D(n)$  are  $D(n)_{n+x,n+y} = \frac{f(|x|)f(|y|)}{9^n}$

$$\text{where } f(a) = \sum_{i=a}^{\lfloor \frac{n+a}{2} \rfloor} \frac{n!}{(n-2i+a)!i!(i-a)!}.$$

**Theorem 2.**  $D(n)_{n+x,n+y}$  satisfies

$$\left( \frac{\partial D(n)_{n+x,n+y}}{\partial x}, \frac{\partial D(n)_{n+x,n+y}}{\partial y} \right) \approx -\frac{3}{2n} (x, y), \quad (2)$$

where  $x/n \rightarrow 0$  and  $y/n \rightarrow 0$ . Therefore, the contour of  $D(n)_{n+x,n+y}$  is approximately a circle.

We will proceed to prove Theorems 1 and 2.

#### 2.1.1. Proof of Theorem 1

Theorem 1 is proven from Lemmas 1 and 2.

**Lemma 1.**

$$9^n D(n)_{x,y} = (9^n D(n)_{x,0}) (9^n D(n)_{0,y}), \quad (3)$$

where  $-n \leq x \leq n$  and  $-n \leq y \leq n$ .

**Lemma 2.**

$$D(n)_{n+x,0} = D(n)_{0,n+x} = 9^{-n} f(|x|), \quad (4)$$

where  $-n \leq x \leq n$ .

We will proceed to prove Lemmas 1 and 2.

*Proof of Lemma 1.* We prove Lemma 1 by mathematical induction on  $n$ .

1. When  $n = 0$ ,  $D(0)_{0,0} = 1$ . Thus, Eq. (3) is true.
2. We assume that Eq. (3) is true when  $n = k$ . When  $n = k + 1$ ,

$$\begin{aligned} 9^{k+1} D(k+1)_{x,y} &= \frac{9^{k+1}}{9} \sum_{i=-1}^1 \sum_{j=-1}^1 D(k)_{x+i,y+j} \end{aligned} \quad (5)$$

$$= \sum_{i=-1}^1 \sum_{j=-1}^1 9^k D(k)_{x+i,y+j} \quad (6)$$

$$= \sum_{i=-1}^1 \sum_{j=-1}^1 (9^k D(k)_{x+i,0}) (9^k D(k)_{0,y+j}) \quad (7)$$

$$= (9^{k+1} D(k+1)_{x,0}) (9^{k+1} D(k+1)_{0,y}). \quad (8)$$

Therefore, Eq. (3) is true when  $n = k + 1$ .

Thus, we have proven Lemma 1.  $\square$

*Proof of Lemma 2.* We now prove Eq. (4) where  $0 \leq x \leq n$ . Eq. (4) is certainly true for the other  $x$ , namely  $-n \leq x < 0$ , for symmetry.

In each  $3 \times 3$  filter size convolutional layer with the stride of one, the effect value of each pixel is the average of nine pixels, namely the selected pixel and surrounding eight pixels. The effect on each pixel spreads to these nine pixels after a  $3 \times 3$  filter size convolutional layer with the stride of one. Thus, we consider diffusion as the transition between pixels. Diffusion then corresponds to set  $P$  defined as  $P = \{(i, j) \mid -1 \leq i \leq 1, -1 \leq j \leq 1\}$ , which includes nine paths. Thus, we obtain  $D(n)_{n+x, n+y}$  from the number of paths from  $(n, n)$  to  $(n+x, n+y)$  using  $n$  paths in  $P$ . In more detail,

$$D(n)_{n+x, n+y} = \frac{1}{9^n} \# \left( \sum_{k=1}^n P_k = (x, y), P_k \in P \right). \quad (9)$$

We now calculate  $D(n)_{n+x, 0}$ . To calculate  $D(n)_{n+x, 0}$ , we now count the number of paths from  $(n, n)$  to  $(n+x, 0)$  using  $n$  paths in  $P$ .  $(n+x, 0)$  represents the left edge of the diffusion range. When we move from  $(n, n)$  to  $(n+x, 0)$ ,  $P_k$  is always  $(i, -1)$  where  $i \in \{-1, 0, 1\}$ . Therefore, we count the number of paths so that the sum of  $P_k$ 's first elements is  $x$ . The above movement consists of  $(n-2i+x)$  times  $(0, -1)$  move,  $i$  times  $(1, -1)$  move, and  $(i-x)$  times  $(-1, -1)$  move, where  $x \leq i \leq \left\lfloor \frac{n+x}{2} \right\rfloor$ . In each  $i$ , the number of paths is  $\frac{n!}{(n-2i+x)!i!(i-x)!}$ . The sum of these values over  $i$  is  $f(x)$ . Therefore,  $D(n)_{n+x, 0} = 9^{-n} f(x)$ . Thus, we have proven Lemma 2.  $\square$

### 2.1.2. Proof of Theorem 2

Theorem 2 is proven from Lemmas 3 and 4.

**Lemma 3.**  $f(x)$  is approximated as

$$f(x) \approx \left( \prod_{j=1}^3 \left( A_j \left( \frac{x}{n} \right)^{A_j(x/n)} \right) \right)^{-n}, \quad (10)$$

where

$$A_1(\alpha) = \frac{\sqrt{4-3\alpha^2}-1}{3}, \quad (11)$$

$$A_2(\alpha) = \frac{4+3\alpha-\sqrt{4-3\alpha^2}}{6}, \quad (12)$$

$$A_3(\alpha) = \frac{4-3\alpha-\sqrt{4-3\alpha^2}}{6}. \quad (13)$$

**Lemma 4.** The gradient of  $D(n)_{n+x, n+y}$  with respect to  $x$  and  $y$  satisfies

$$\begin{aligned} \nabla D(n)_{n+x, n+y} &\approx -\frac{1}{2} D(n)_{n+x, n+y} \left( \log \frac{A_2(x/n)}{A_3(x/n)}, \log \frac{A_2(y/n)}{A_3(y/n)} \right) \\ &\approx -\frac{3\alpha}{2n} [x, y], \end{aligned} \quad (14)$$

$$\approx -\frac{3\alpha}{2n} [x, y], \quad (15)$$

where  $x/n \rightarrow 0$  and  $y/n \rightarrow 0$ .

We will proceed to prove Lemmas 3 and 4.

*Proof of Lemma 3.* First, we approximate  $f(x)$ . From Stirling's approximation,  $\log n! \approx n \log n - n$ . Therefore,

$$\begin{aligned} &\log \frac{n!}{(n-2i+x)!i!(i-x)!} \\ &\approx n \log n - i \log i - (i-x) \log(i-x) \\ &\quad - (n-2i+x) \log(n-2i+x) \end{aligned} \quad (16)$$

Now, we define  $g(i)$  as

$$\begin{aligned} g(i) &= n \log n - i \log i - (i-x) \log(i-x) \\ &\quad - (n-2i+x) \log(n-2i+x). \end{aligned} \quad (17)$$

We differentiate  $g(i)$  with respect to  $i$  and obtain

$$\frac{dg}{di} = \log \left( \frac{(n-2i+x)^2}{i(i-x)} \right) \quad (18)$$

Thus,  $\frac{dg}{di}$  is monotonically decreasing where  $x \leq i \leq \left\lfloor \frac{n+x}{2} \right\rfloor$ . In particular,  $\frac{dg}{di}$  decreases from  $+\infty$  to  $-\infty$ . Therefore, the function  $g(i)$  takes the maximal value when

$$i = \frac{4+3(x/n)-\sqrt{4-3(x/n)^2}}{6}n. \quad (19)$$

Now, we approximate function  $f(x)$ . We define  $\alpha$  as  $\alpha = x/n$  ( $0 \leq \alpha \leq 1$ ) and the function  $A_1(\alpha)$ ,  $A_2(\alpha)$ , and  $A_3(\alpha)$  as  $A_1(\alpha) = \frac{\sqrt{4-3\alpha^2}-1}{3}$ ,  $A_2(\alpha) = \frac{4+3\alpha-\sqrt{4-3\alpha^2}}{6}$ , and  $A_3(\alpha) = \frac{4-3\alpha-\sqrt{4-3\alpha^2}}{6}$ . These functions  $A_1(\alpha)$ ,  $A_2(\alpha)$ , and  $A_3(\alpha)$  are  $(n-2i+x)/n$ ,  $i/n$ , and  $(i-x)/n$  under Eq. (19), respectively. The approximate function of  $f(x)$  is then

$$f(x) = \exp(g(i)) \quad (20)$$

$$\approx \frac{n^n}{\prod_{j=1}^3 (n A_j(\alpha))^{n A_j(\alpha)}} \quad (21)$$

$$= \left( \prod_{j=1}^3 \left( A_j \left( \frac{x}{n} \right)^{A_j(x/n)} \right) \right)^{-n}, \quad (22)$$

because  $A_1(\alpha) + A_2(\alpha) + A_3(\alpha) = 1$ . Thus, we have proven Lemma 3.  $\square$

*Proof of Lemma 4.* First, we partially differentiate  $D(n)_{n+x,n+y}$  with respect to  $x$  where  $0 \leq x \leq n$ , without loss of generality because of symmetry. Then,

$$\frac{\partial D(n)_{n+x,n+y}}{\partial x} = \frac{f(|y|)}{9^n} \frac{df(x)}{dx} \quad (23)$$

and

$$\begin{aligned} \frac{df(x)}{dx} &= \frac{d}{dx} \left( \left( \prod_{j=1}^3 \left( A_j \left( \frac{x}{n} \right)^{A_j(x/n)} \right) \right)^{-n} \right) \\ &= \frac{d}{dx} \left( \exp \left( -n \left( \sum_{j=1}^3 A_j \left( \frac{x}{n} \right) \log A_j \left( \frac{x}{n} \right) \right) \right) \right) \end{aligned} \quad (24)$$

$$= f(x) \frac{d}{dx} \left( -n \left( \sum_{j=1}^3 A_j \left( \frac{x}{n} \right) \log A_j \left( \frac{x}{n} \right) \right) \right) \quad (25)$$

$$= f(x) \frac{d}{dx} \left( -n \left( \sum_{j=1}^3 A_j \left( \frac{x}{n} \right) \log A_j \left( \frac{x}{n} \right) \right) \right) \quad (26)$$

$$= -nf(x) \left( \frac{d}{dx} \left( \sum_{j=1}^3 A_j \left( \frac{x}{n} \right) \right) + \sum_{j=1}^3 \left( \log A_j \left( \frac{x}{n} \right) \frac{d}{dx} A_j \left( \frac{x}{n} \right) \right) \right) \quad (27)$$

$$= -nf(x) \sum_{j=1}^3 \left( \log A_j \left( \frac{x}{n} \right) \frac{d}{dx} A_j \left( \frac{x}{n} \right) \right) \quad (28)$$

Eq. (28) is satisfied because  $\sum_{j=1}^3 A_j \left( \frac{x}{n} \right) = 1$ .

To calculate Eq. (28), we define the function  $h(\alpha)$  as  $h(\alpha) = \alpha(4 - 3\alpha)^{-1/2}$ . By using this function  $h(\alpha)$ ,  $\frac{d}{dx} A_1 \left( \frac{x}{n} \right) = -\frac{1}{n} h \left( \frac{x}{n} \right)$ ,  $\frac{d}{dx} A_2 \left( \frac{x}{n} \right) = \frac{1}{2n} h \left( \frac{x}{n} \right) + \frac{1}{2n}$ , and  $\frac{d}{dx} A_3 \left( \frac{x}{n} \right) = \frac{1}{2n} h \left( \frac{x}{n} \right) - \frac{1}{2n}$ . Moreover,  $A_1(x/n)^2 = A_2(x/n)A_3(x/n)$  and  $A_2(x/n) \geq A_3(x/n)$ . Therefore,

$$\frac{df(x)}{dx} = -nf(x) \sum_{j=1}^3 \left( \log A_j \left( \frac{x}{n} \right) \frac{d}{dx} A_j \left( \frac{x}{n} \right) \right) \quad (29)$$

$$\begin{aligned} &= h \left( \frac{x}{n} \right) f(x) \left( \log A_1 \left( \frac{x}{n} \right) - \frac{1}{2} \log A_2 \left( \frac{x}{n} \right) - \frac{1}{2} \log A_3 \left( \frac{x}{n} \right) \right) \\ &\quad - \frac{f(x)}{2} \left( \log A_2 \left( \frac{x}{n} \right) - \log A_3 \left( \frac{x}{n} \right) \right) \end{aligned} \quad (30)$$

$$= -\frac{f(x)}{2} \log \frac{A_2(x/n)}{A_3(x/n)} \quad (31)$$

Therefore,

$$\frac{\partial D(n)_{n+x,n+y}}{\partial x} = -\frac{1}{2} \log \frac{A_2(x/n)}{A_3(x/n)} D(n)_{n+x,n+y} \quad (32)$$

Similarly, by applying the above approximation to  $y$ ,

$$\begin{aligned} \nabla D(n)_{n+x,n+y} &\approx -\frac{1}{2} D(n)_{n+x,n+y} \left( \log \frac{A_2(x/n)}{A_3(x/n)}, \log \frac{A_2(y/n)}{A_3(y/n)} \right) \end{aligned} \quad (33)$$

We now calculate the approximate gradient near  $(x, y) = (0, 0)$ . The function  $\log \frac{A_2(x/n)}{A_3(x/n)} \approx \frac{3x}{n}$ , because  $\log \frac{A_2(0)}{A_3(0)} = 0$  and  $\lim_{x \rightarrow 0} \frac{d}{dx} \left( \log \frac{A_2(x/n)}{A_3(x/n)} \right) \rightarrow \frac{3}{n}$ . Moreover,  $D(n)_{n+x,n+y} \approx 1$  near  $(x, y) = (0, 0)$  because  $f(0) \approx 3^n$  from Lemma 3. Therefore,  $\nabla D(n)_{n+x,n+y} \approx -3/(2n)(x, y)$  near  $(x, y) = (0, 0)$ . Thus, we have proven Lemma 4.  $\square$

## 2.2. A Method for Calculating Optimal Shape of Remote Adversarial Patches in Actual CNNs

In this subsection, we will provide a detailed explanation of the method for calculating the optimal remote adversarial patches. In particular, we will provide a detailed explanation of the calculation method for each of the four CNNs, namely YOLOv2 [7], SSD [5], RetinaNet [4], and FCOS [11]. We specifically explain the parameters for calculating the optimal remote adversarial patches, namely the placement region of the remote adversarial patches and the target region for the attack  $R$ . The targeted region  $R$  is positioned such that its center aligns with the center of each output layer.

### 2.2.1. YOLOv2 [7]

Firstly, we will discuss YOLOv2 [7]. Similar as the previous research [6], we select pixels of remote adversarial patches from  $0 \leq x \leq 415$  and  $(0 \leq y \leq 104$  or  $311 \leq y \leq 415)$ . Moreover, we will examine the placement method that maximizes the effectiveness of the adversarial patch for a region  $R$  of size  $96 \times 96$  centered in the image (specifically, a region of size  $3 \times 3$  in the final layer). In particular, the figure presented in the main paper depicts a selection of 5,000 pixels.

To calculate the influence value of the adversarial patches, we will now describe the CNN model that we have used in this paper. In the calculations conducted in this study, we did not include processing related to layers that do not affect the diffusion range of the adversarial patch effect, such as a  $1 \times 1$  convolutional layer with a stride of 1. Therefore, we will not include these steps in the following calculation procedure. If the image size changes during the

calculation, it will be explicitly stated. Furthermore, unless otherwise specified, the output of the previous step will be used as the input for the operation. The calculations were performed as follows:

1. A  $3 \times 3$  convolutional layer with a stride of 1 and padding of 1
2. A  $2 \times 2$  max pooling layer with a stride of 2 ( $416 \times 416 \rightarrow 208 \times 208$ )
3. A  $3 \times 3$  convolutional layer with a stride of 1 and padding of 1
4. A  $2 \times 2$  max pooling layer with a stride of 2 ( $208 \times 208 \rightarrow 104 \times 104$ )
5. Two  $3 \times 3$  convolutional layers with a stride of 1 and padding of 1
6. A  $2 \times 2$  max pooling layer with a stride of 2 ( $104 \times 104 \rightarrow 52 \times 52$ )
7. Two  $3 \times 3$  convolutional layers with a stride of 1 and padding of 1
8. A  $2 \times 2$  max pooling layer with a stride of 2 ( $52 \times 52 \rightarrow 26 \times 26$ )
9. Three  $3 \times 3$  convolutional layers with a stride of 1 and padding of 1
10. A  $2 \times 2$  max pooling layer with a stride of 2 ( $26 \times 26 \rightarrow 13 \times 13$ )
11. Five  $3 \times 3$  convolutional layers with a stride of 1 and padding of 1
12. The output of Step 9 will be subjected to a  $2 \times 2$  max pooling layer with a stride of 2. ( $26 \times 26 \rightarrow 13 \times 13$ )
13. The outputs of Step 11 and Step 12 will be combined by adding them together with the following weighting:

$$\begin{aligned}
 (\text{Result}) &= \frac{4}{5} \times (\text{The output of Step 11}) \\
 &+ \frac{1}{5} (\text{The output of Step 12}). \quad (34)
 \end{aligned}$$

The reason for Eq. (34) is that the output of Step 11 has a channel number of 1024, while the output of Step 12 has a channel number of 256.

14. A  $3 \times 3$  convolutional layer with a stride of 1 and padding of 1

We examine the placement method that maximizes the effectiveness of the adversarial patch for a region  $R$  of size  $96 \times 96$  centered in the image (specifically, a region of size  $3 \times 3$  in the final layer). Similar as the previous research [6], we select pixels of remote adversarial patches from  $0 \leq x \leq 415$  and  $(0 \leq y \leq 104 \text{ or } 311 \leq y \leq 415)$ . This condition takes into account the requirement for remote adversarial patches.

### 2.2.2. SSD [5]

To calculate the influence value of the adversarial patches, we will now describe the CNN model that we have used in this paper. The notation and methodology for the calculation

are similar to that of YOLOv2. The calculations were performed as follows:

1. Two  $3 \times 3$  convolutional layers with a stride of 1 and padding of 1
2. A  $2 \times 2$  max pooling layer with a stride of 2 ( $300 \times 300 \rightarrow 150 \times 150$ )
3. Two  $3 \times 3$  convolutional layers with a stride of 1 and padding of 1
4. A  $2 \times 2$  max pooling layer with a stride of 2 ( $150 \times 150 \rightarrow 75 \times 75$ )
5. Three  $3 \times 3$  convolutional layers with a stride of 1 and padding of 1
6. A  $2 \times 2$  max pooling layer with a stride of 2 ( $75 \times 75 \rightarrow 38 \times 38$ ). When performing the aforementioned operations naively, there is a shortage of pixels for the rightmost and bottommost pixels of the image. Therefore, in SSD, these pixels are padded with zeros and pooling operations are performed. In this study, taking into account the aforementioned operations, only the portions within the original image are used for calculating the influence values, and the average of these pixels is used as the output for the corresponding pixel.
7. Three  $3 \times 3$  convolutional layers with a stride of 1 and padding of 1
8. A  $2 \times 2$  max pooling layer with a stride of 2 for the output of Step 8 ( $38 \times 38 \rightarrow 19 \times 19$ )
9. Three  $3 \times 3$  convolutional layers with a stride of 1 and padding of 1
10. A  $3 \times 3$  max pooling layer with a stride of 1 and padding of 1
11. A  $3 \times 3$  convolutional layer with a stride of 1, padding of 6, and dilation of 6
12. A  $3 \times 3$  convolutional layer with a stride of 2 and padding of 1 ( $19 \times 19 \rightarrow 10 \times 10$ )
13. A  $3 \times 3$  convolutional layer with a stride of 2 and padding of 1 ( $10 \times 10 \rightarrow 5 \times 5$ )
14. A  $3 \times 3$  convolutional layer with a stride of 1 ( $5 \times 5 \rightarrow 3 \times 3$ )
15. A  $3 \times 3$  convolutional layer with a stride of 1 ( $3 \times 3 \rightarrow 1 \times 1$ )
16. A  $3 \times 3$  convolutional layer with a stride of 1 and padding of 1 will be applied to six output layers, namely the outputs of Step 7, Step 11, Step 12, Step 13, Step 14, and Step 15. Each calculation result corresponds to an output of size  $38 \times 38$ ,  $19 \times 19$ ,  $10 \times 10$ ,  $5 \times 5$ ,  $3 \times 3$ , and  $1 \times 1$ , respectively. It is important to note that these operations consider both classification and regression simultaneously, as they have the same operations.

For each of the six output layers in SSD, we calculated the optimal shape of the adversarial patch region in the following attack regions:

- Layer with a size of  $38 \times 38$ : Central  $8 \times 8$  region
- Layer with a size of  $19 \times 19$ : Central  $5 \times 5$  region

- Layer with a size of  $10 \times 10$ : Central  $2 \times 2$  region
- Remaining layers: Central  $1 \times 1$  region

Additionally, since SSD scales the input image size from  $416 \times 416$  to  $300 \times 300$ , the region of existence for the adversarial patch is also scaled accordingly. Especially, we select pixels of remote adversarial patches from  $0 \leq x \leq 299$  and  $(0 \leq y \leq 74 \text{ or } 225 \leq y \leq 299)$ .

### 2.2.3. RetinaNet [4], and FCOS [11]

To calculate the influence value of the adversarial patch, we will now describe the CNN model that we have formulated in this paper. The notation and methodology for the calculation are similar to that of YOLOv2. The calculations were performed as follows:

1. A  $7 \times 7$  convolutional layer with a stride of 2 and padding of 3 ( $416 \times 416 \rightarrow 208 \times 208$ )
2. A  $3 \times 3$  max pooling layer with a stride of 2 and padding of 1 ( $208 \times 208 \rightarrow 104 \times 104$ )
3. Repeat the bottleneck blocks three times Each bottleneck block performs the following calculations:
  - (a) Save the previous output.
  - (b) A  $3 \times 3$  convolutional layer with a stride of 1 and padding of 1
  - (c) Compute the average of the outputs from steps (a) and (b), and output the result
4. Execute the bottleneck block composed of the following operations:
  - (a) Save the previous output.
  - (b) A  $1 \times 1$  convolutional layer with a stride of 2 ( $104 \times 104 \rightarrow 52 \times 52$ )
  - (c) A  $3 \times 3$  convolutional layer with a stride of 2 and padding of 1 for the output of Step (a) ( $104 \times 104 \rightarrow 52 \times 52$ )
  - (d) Compute the average of the outputs from steps (b) and (c), and output the result
5. Repeat the bottleneck blocks defined in Step 3 three times
6. Execute the bottleneck block defined in Step 4 ( $52 \times 52 \rightarrow 26 \times 26$ )
7. Repeat the bottleneck blocks defined in Step 3 five times
8. Execute the bottleneck block defined in Step 4 ( $26 \times 26 \rightarrow 13 \times 13$ )
9. Repeat the bottleneck blocks defined in Step 3 two times
10. A  $3 \times 3$  convolutional layer with a stride of 2 and padding of 1 ( $13 \times 13 \rightarrow 7 \times 7$ )
11. A  $3 \times 3$  convolutional layer with a stride of 2 and padding of 1 ( $7 \times 7 \rightarrow 4 \times 4$ )
12. Scale the  $13 \times 13$  image obtained in Step 9 to a size of  $26 \times 26$ , and then average it with the image obtained in Step 7. Scaling is performed by dividing each pixel of the  $13 \times 13$  image into four equal parts to form a  $2 \times 2$  image. All four regions have the same influence value, and the influence value of the corresponding pixel is used as is.

13. Scale the  $26 \times 26$  image obtained in Step 12 to a size of  $52 \times 52$ , and then average it with the image obtained in Step 5. We perform the scaling in the same manner as in step 12.
14. A  $3 \times 3$  convolutional layer with a stride of 1 and padding of 1 will be applied to three output layers, namely the outputs of Step 13, Step 12, and Step 9. Each calculation result corresponds to an output of size  $52 \times 52$ ,  $26 \times 26$ , and  $13 \times 13$ , respectively.
15. Five  $3 \times 3$  convolutional layers with a stride of 1 and padding of 1 will be applied to five output layers, namely the outputs of Step 14 ( $52 \times 52$ ), Step 14 ( $26 \times 26$ ), Step 14 ( $13 \times 13$ ), Step 10 ( $7 \times 7$ ), and Step 11 ( $4 \times 4$ ). It is important to note that these operations consider both classification and regression simultaneously, as they have the same operations.

For each of the five output layers, we calculated the optimal shape of the adversarial patches in the following attack regions:

- Layer with a size of  $52 \times 52$ : Central  $12 \times 12$  region
- Layer with a size of  $26 \times 26$ : Central  $6 \times 6$  region
- Layer with a size of  $13 \times 13$ : Central  $3 \times 3$  region
- Layer with a size of  $7 \times 7$ : Central  $1 \times 1$  region
- Layer with a size of  $4 \times 4$ : Central  $2 \times 2$  region

We select pixels of remote adversarial patches from  $0 \leq x \leq 415$  and  $(0 \leq y \leq 104 \text{ or } 311 \leq y \leq 415)$ .

## 3. Detailed Information for Evaluation of Strong Remote Adversarial Patches

In this section, we provide detailed experimental methods and experimental data for Sec. 4 in the main paper. For SSD [5], RetinaNet [4], and FCOS [11], we conducted the experiment by mapping the class names to their corresponding class names in the PyTorch library [1]. Specifically, we replaced “aeroplane” with “airplane”, “diningtable” with “dining table”, “motorbike” with “motorcycle”, “potted-plant” with “potted plant”, “sofa” with “couch”, and “tv-monitor” with “tv”, and performed the experiment accordingly.

### 3.1. Detailed Experimental Results for Bounding Boxes

In this subsection, we show experimental results regarding the size of bounding boxes. The experimental results are as shown in from Tabs. 1 to 4. The empty cells indicate the absence of bounding boxes that include a confidence score of 50% or higher. These classes are excluded from calculation of the average.

### 3.2. Detailed Experimental Results for Confidence Score

In this subsection, we show experimental results regarding the confidence score. The experimental results are as shown



Table 1. The proportion of bounding box size relative to the entire image in YOLOv2

Shape Area	Clean	Square	Crescent	Square	Crescent	Square	Crescent
	—	9,800	9,800	5,000	5,000	2,592	2,592
aeroplane	24.21%	21.79%	8.10%	15.99%	9.03%	14.42%	10.19%
bicycle	21.95%	7.21%	5.94%	18.88%	22.24%	21.72%	22.79%
bird	18.01%	3.66%	3.85%	10.6%	5.45%	8.84%	14.96%
boat	15.46%	7.62%	26.80%	5.76%	5.89%	17.68%	5.75%
bottle	9.48%	9.34%	8.57%	8.98%	8.52%	8.48%	9.13%
bus	24.30%	—	—	—	—	29.12%	41.58%
car	17.38%	4.29%	4.45%	8.65%	6.57%	9.22%	7.53%
cat	26.93%	41.22%	13.48%	22.45%	13.28%	11.71%	14.34%
chair	11.17%	6.33%	6.55%	9.22%	7.46%	9.28%	7.80%
cow	14.49%	3.21%	2.35%	3.98%	2.77%	7.70%	7.38%
diningtable	34.02%	19.51%	21.98%	26.92%	20.87%	30.68%	27.60%
dog	25.93%	11.86%	12.31%	10.66%	18.50%	22.67%	27.68%
horse	27.95%	—	4.13%	—	25.27%	25.97%	29.74%
motorbike	22.59%	26.27%	9.74%	19.37%	16.25%	19.17%	22.48%
person	17.62%	18.79%	11.55%	7.14%	14.07%	15.19%	19.60%
pottedplant	21.15%	5.45%	3.74%	10.51%	9.30%	12.25%	20.14%
sheep	13.78%	2.64%	2.65%	3.62%	5.10%	10.90%	4.90%
sofa	31.41%	42.89%	—	26.68%	28.38%	28.60%	27.55%
train	22.91%	—	—	20.53%	5.50%	12.69%	14.51%
tvmonitor	11.43%	7.63%	4.35%	10.12%	7.34%	8.81%	8.92%
<b>average</b>	20.61%	14.10%	8.86%	13.34%	12.20%	16.26%	17.23%

Table 2. The proportion of bounding box size relative to the entire image in SSD

Shape Area	Clean	Square	Crescent	Square	Crescent	Square	Crescent
	—	9,800	9,800	5,000	5,000	2,592	2,592
aeroplane	27.69%	1.40%	1.40%	1.40%	1.40%	1.40%	1.40%
bicycle	22.44%	—	6.32%	17.24%	15.39%	19.52%	19.55%
bird	18.62%	3.33%	3.35%	6.61%	3.20%	8.47%	13.76%
boat	17.22%	—	—	—	—	—	—
bottle	8.58%	3.08%	—	3.20%	5.52%	4.88%	4.18%
bus	21.52%	—	—	—	—	49.77%	24.76%
car	17.78%	1.75%	2.06%	6.88%	4.28%	8.56%	14.10%
cat	27.39%	13.80%	14.52%	12.30%	—	9.38%	17.60%
chair	8.85%	11.57%	1.35%	8.37%	6.65%	7.31%	7.07%
cow	16.01%	2.81%	—	3.25%	3.24%	10.91%	12.05%
diningtable	34.63%	—	—	—	—	6.16%	22.90%
dog	26.80%	—	—	10.45%	—	12.27%	17.57%
horse	27.52%	—	—	—	2.96%	21.74%	20.82%
motorbike	22.40%	—	—	—	23.63%	24.41%	25.10%
person	18.23%	3.81%	3.37%	14.95%	17.16%	17.45%	18.79%
pottedplant	17.12%	2.29%	3.58%	7.53%	3.54%	8.17%	6.60%
sheep	13.81%	1.37%	1.37%	1.37%	1.37%	1.42%	5.79%
sofa	30.70%	—	—	11.86%	—	28.59%	21.88%
train	23.71%	—	—	—	33.37%	27.68%	28.80%
tvmonitor	10.16%	—	—	11.53%	9.48%	10.96%	10.89%
<b>average</b>	20.56%	4.52%	4.15%	8.35%	9.50%	14.69%	15.45%

in from Tabs. 5 to 8.

### 3.3. Strong Remote Adversarial Patches in Cases of Detecting Large Objects with SSD

In Sec. 4.3 in the main paper, we showed that for CNNs with small diffusion effects like SSD, rectangular remote adversarial patches are strong. This rectangular remote adversarial patch exhibits its effect when obstructing the detection of small objects in layers close to the input layer with high resolution. However, in Sec. 3 in the main paper, it is suggested that in layers distant from the input layer, the optimal shape of remote adversarial patches is crescent-like. This remote adversarial patch is expected to be effective in obstructing the detection of large objects, but the verification of this hypothesis in the experiments conducted up to Sec. 4.3 in the

Table 3. The proportion of bounding box size relative to the entire image in RetinaNet

Shape Area	Clean	Square	Crescent	Square	Crescent	Square	Crescent
	—	9,800	9,800	5,000	5,000	2,592	2,592
aeroplane	23.15%	1.36%	1.14%	1.23%	9.44%	18.78%	17.38%
bicycle	21.46%	—	—	5.84%	—	20.02%	17.76%
bird	15.04%	1.49%	1.49%	1.47%	1.49%	14.09%	14.02%
boat	14.04%	—	—	29.87%	57.12%	20.90%	38.24%
bottle	6.79%	2.09%	—	19.73%	5.06%	9.63%	5.29%
bus	21.28%	—	4.49%	—	7.39%	17.14%	13.86%
car	15.79%	—	1.69%	3.64%	3.35%	5.06%	11.04%
cat	21.89%	—	3.61%	12.48%	4.83%	10.07%	10.67%
chair	9.78%	—	—	5.04%	5.28%	7.20%	6.40%
cow	14.76%	—	—	—	—	8.07%	2.54%
diningtable	34.66%	—	—	46.76%	—	41.84%	49.86%
dog	26.94%	—	—	9.39%	9.81%	20.52%	16.26%
horse	25.82%	—	—	—	—	6.05%	—
motorbike	20.88%	2.08%	2.07%	13.06%	2.92%	17.75%	21.50%
person	16.20%	4.91%	8.19%	2.52%	3.03%	16.63%	16.45%
pottedplant	13.55%	—	—	18.00%	19.68%	24.41%	15.25%
sheep	14.00%	—	—	—	—	7.53%	1.00%
sofa	31.11%	—	—	—	—	21.62%	—
train	21.42%	13.37%	1.66%	—	—	25.69%	29.53%
tvmonitor	8.78%	—	—	12.74%	1.86%	9.95%	7.77%
<b>average</b>	18.87%	4.22%	3.04%	12.98%	10.10%	16.15%	16.38%

Table 4. The proportion of bounding box size relative to the entire image in FCOS

Shape Area	Clean	Square	Crescent	Square	Crescent	Square	Crescent
	—	9,800	9,800	5,000	5,000	2,592	2,592
aeroplane	22.22%	—	—	23.16%	15.33%	25.50%	22.66%
bicycle	21.15%	18.99%	8.57%	18.53%	15.01%	20.25%	18.60%
bird	16.25%	—	—	3.66%	4.92%	9.70%	11.27%
boat	13.66%	—	—	1.04%	1.03%	9.40%	8.32%
bottle	5.49%	10.60%	1.83%	5.12%	2.59%	5.36%	5.55%
bus	20.34%	4.94%	6.07%	3.89%	3.90%	14.35%	11.56%
car	15.56%	2.71%	2.80%	8.87%	4.20%	10.83%	9.98%
cat	25.89%	6.88%	5.39%	11.73%	6.96%	13.63%	10.22%
chair	10.35%	5.44%	6.27%	7.09%	6.23%	7.34%	6.37%
cow	14.89%	—	—	3.67%	3.69%	8.05%	7.23%
diningtable	31.83%	—	—	—	—	34.40%	28.49%
dog	29.12%	7.88%	3.69%	8.58%	10.59%	20.68%	16.82%
horse	25.80%	—	1.97%	8.65%	4.34%	18.64%	11.63%
motorbike	21.36%	—	8.27%	22.88%	21.57%	20.56%	18.69%
person	16.73%	9.12%	10.15%	17.98%	8.59%	19.89%	16.74%
pottedplant	14.51%	6.82%	4.47%	12.47%	7.72%	9.12%	8.44%
sheep	11.94%	—	—	—	—	0.64%	1.82%
sofa	28.63%	11.71%	—	—	25.97%	21.55%	17.74%
train	22.43%	—	8.09%	11.39%	8.14%	17.92%	11.23%
tvmonitor	9.02%	—	—	8.71%	8.26%	9.51%	8.16%
<b>average</b>	18.86%	8.51%	5.63%	10.44%	8.84%	14.87%	12.58%

main paper remains insufficient. Therefore, we show that crescent-shaped remote adversarial patches are strong when detecting large objects with SSD by experiments.

First, we extracted images where only one object detected by SSD exists among the 20 classes and is located between remote adversarial patches. Here, “only one object detected by SSD” indicates that the number of bounding boxes in images without adversarial patches is one. Furthermore, “the object is located between remote adversarial patches” indicates that the bounding box of the object exists within the range of  $105.0 < y < 311.0$ , which is considered in real numbers.

Next, we extracted the class that satisfies the above conditions with at least ten images. The corresponding class and the number of images in each class are as follows:

Table 5. Confidence score in YOLOv2

Shape Area	Clean –	Square 9,800	Crescent 9,800	Square 5,000	Crescent 5,000	Square 2,592	Crescent 2,592
aeroplane	77.31%	3.99%	4.43%	9.24%	6.20%	13.15%	8.40%
bicycle	69.71%	2.79%	1.95%	44.07%	9.81%	62.77%	41.89%
bird	75.03%	3.82%	3.64%	9.93%	6.67%	21.05%	25.89%
boat	62.14%	3.76%	5.33%	9.11%	7.01%	21.44%	12.97%
bottle	49.48%	37.32%	20.86%	43.61%	41.35%	46.73%	44.52%
bus	60.57%	0.63%	0.32%	7.21%	2.83%	15.19%	17.48%
car	71.91%	4.95%	12.29%	26.6%	29.72%	39.02%	40.54%
cat	64.47%	5.27%	2.48%	10.28%	8.33%	15.60%	11.46%
chair	48.01%	7.11%	6.00%	27.40%	21.01%	30.50%	32.15%
cow	77.48%	11.29%	7.37%	10.79%	12.59%	21.56%	21.92%
diningtable	53.83%	13.45%	11.87%	36.10%	21.98%	43.73%	42.20%
dog	73.37%	5.88%	11.98%	11.97%	17.85%	25.70%	34.70%
horse	72.75%	0.62%	3.25%	1.15%	6.34%	8.95%	28.51%
motorbike	66.61%	18.10%	8.58%	33.98%	20.45%	48.98%	31.19%
person	69.36%	29.97%	15.54%	68.36%	37.92%	59.85%	54.04%
pottedplant	50.00%	10.94%	7.94%	22.34%	25.67%	25.71%	37.98%
sheep	71.39%	2.94%	6.11%	9.95%	15.33%	30.48%	17.24%
sofa	47.55%	1.43%	1.38%	14.49%	9.10%	22.56%	16.27%
train	73.02%	0.63%	0.08%	3.92%	4.30%	13.52%	14.92%
tvmonitor	60.55%	7.30%	3.36%	16.72%	11.76%	28.29%	25.02%
<b>average</b>	64.73%	8.61%	6.74%	20.86%	15.81%	29.74%	27.96%

Table 6. Confidence score in SSD

Shape Area	Clean –	Square 9,800	Crescent 9,800	Square 5,000	Crescent 5,000	Square 2,592	Crescent 2,592
aeroplane	76.78%	5.71%	5.24%	5.85%	5.64%	6.82%	6.31%
bicycle	63.64%	3.37%	3.51%	7.04%	13.44%	29.99%	30.91%
bird	68.83%	5.94%	6.06%	9.53%	6.53%	14.53%	13.87%
boat	61.82%	8.88%	8.99%	9.88%	10.08%	10.74%	11.79%
bottle	40.40%	10.00%	9.02%	11.10%	12.85%	19.17%	21.57%
bus	75.72%	1.81%	2.28%	1.96%	4.06%	6.99%	17.76%
car	68.84%	17.55%	20.68%	33.80%	27.93%	42.16%	53.16%
cat	69.32%	1.41%	1.30%	3.38%	0.81%	4.33%	8.63%
chair	53.66%	11.34%	8.73%	13.24%	15.43%	26.69%	31.61%
cow	88.91%	10.13%	8.38%	10.70%	12.25%	13.77%	18.14%
diningtable	59.07%	0.41%	0.44%	1.22%	1.68%	13.26%	14.17%
dog	72.86%	0.40%	1.10%	2.04%	1.39%	4.61%	12.16%
horse	81.13%	1.02%	1.52%	1.99%	3.32%	11.05%	19.22%
motorbike	75.44%	4.81%	4.83%	5.04%	15.31%	19.97%	32.95%
person	76.06%	19.67%	18.65%	43.31%	42.52%	46.26%	74.49%
pottedplant	49.78%	10.24%	9.46%	13.62%	11.28%	16.71%	18.86%
sheep	78.06%	10.57%	10.31%	13.55%	11.66%	16.01%	18.74%
sofa	61.78%	0.20%	0.23%	2.57%	1.81%	11.82%	12.56%
train	78.22%	0.91%	0.57%	2.89%	6.36%	6.89%	14.62%
tvmonitor	62.88%	3.59%	3.78%	5.21%	7.70%	15.69%	15.81%
<b>average</b>	68.16%	6.40%	6.25%	9.90%	10.60%	16.87%	22.37%

bird (24 images), boat (10 images), bottle (14 images), car (17 images), cat (10 images), chair (15 images), dog (25 images), horse (20 images), person (169 images), potted-plant (13 images), train (10 images), and tvmonitor (18 images).

Finally, we calculated the average confidence score for images with the top ten bounding box sizes in the extracted classes, where remote adversarial patches were applied. The experimental results are shown in Tab. 9.

From Tab. 9, when the size of the remote adversarial patches is 9,800 pixels, the average confidence score for rectangular patches is 6.81%, whereas the average confidence score for crescent-shaped patches is 5.72%. Additionally, when the size of the remote adversarial patches is 5,000 pixels, the average confidence score for rectangular patches is 13.16%, whereas the average confidence score for crescent-shaped patches is 11.60%. Therefore, in SSD,

Table 7. Confidence score in RetinaNet

Shape Area	Clean –	Square 9,800	Crescent 9,800	Square 5,000	Crescent 5,000	Square 2,592	Crescent 2,592
aeroplane	90.19%	20.55%	13.40%	19.00%	21.18%	36.97%	36.92%
bicycle	79.73%	10.50%	4.53%	23.17%	10.27%	50.70%	37.54%
bird	84.66%	4.81%	3.62%	19.10%	5.79%	38.45%	33.24%
boat	74.96%	4.35%	3.99%	16.91%	9.90%	36.80%	26.74%
bottle	60.03%	16.31%	4.43%	21.80%	16.75%	36.75%	36.08%
bus	78.90%	3.28%	13.12%	12.50%	20.21%	41.47%	32.41%
car	75.50%	10.91%	16.08%	32.75%	29.85%	41.88%	51.97%
cat	79.23%	3.46%	4.18%	8.85%	9.53%	22.39%	24.70%
chair	62.60%	6.54%	5.41%	19.14%	13.94%	33.10%	30.42%
cow	89.28%	1.82%	1.38%	3.83%	6.93%	44.82%	22.93%
diningtable	56.91%	5.07%	3.72%	20.30%	15.74%	30.39%	23.89%
dog	78.81%	2.46%	6.44%	14.37%	18.36%	35.98%	29.50%
horse	82.55%	5.29%	1.81%	11.38%	4.55%	18.39%	13.52%
motorbike	78.85%	12.03%	12.42%	29.35%	16.07%	33.19%	45.34%
person	81.71%	28.17%	13.65%	21.78%	19.12%	62.37%	48.99%
pottedplant	62.47%	6.30%	7.95%	10.34%	19.59%	24.70%	27.74%
sheep	75.99%	3.75%	1.80%	3.51%	4.10%	29.04%	11.24%
sofa	60.21%	1.39%	0.30%	5.91%	3.55%	14.75%	8.58%
train	86.01%	6.28%	6.52%	3.18%	2.86%	19.76%	19.52%
tvmonitor	72.69%	2.36%	4.51%	9.54%	8.23%	26.59%	21.25%
<b>average</b>	75.56%	7.78%	6.46%	15.34%	12.83%	33.92%	29.13%

Table 8. Confidence score in FCOS

Shape Area	Clean –	Square 9,800	Crescent 9,800	Square 5,000	Crescent 5,000	Square 2,592	Crescent 2,592
aeroplane	75.66%	3.61%	8.09%	14.71%	15.71%	37.64%	40.67%
bicycle	66.34%	32.56%	27.83%	42.31%	41.35%	52.89%	50.52%
bird	72.32%	6.97%	7.40%	17.39%	18.34%	33.18%	32.30%
boat	68.31%	11.38%	8.20%	21.21%	15.10%	32.80%	28.90%
bottle	63.25%	30.78%	27.60%	41.94%	36.92%	51.04%	49.27%
bus	70.01%	22.12%	22.69%	29.89%	29.59%	40.96%	37.09%
car	69.99%	31.76%	28.75%	51.78%	44.55%	58.57%	59.28%
cat	63.66%	11.78%	19.41%	20.79%	20.02%	26.60%	28.28%
chair	62.76%	31.60%	22.76%	39.47%	35.55%	46.30%	43.53%
cow	77.50%	9.48%	5.89%	16.85%	14.92%	29.01%	30.38%
diningtable	56.69%	12.83%	8.40%	21.49%	17.88%	34.21%	30.73%
dog	67.87%	14.65%	14.34%	28.05%	27.30%	45.80%	41.93%
horse	69.55%	10.35%	7.72%	15.11%	19.41%	36.57%	31.20%
motorbike	66.16%	11.45%	18.43%	29.60%	27.54%	48.74%	48.04%
person	74.08%	46.28%	46.74%	57.80%	49.93%	70.79%	70.25%
pottedplant	60.90%	28.59%	24.89%	42.28%	38.75%	46.29%	43.89%
sheep	69.79%	1.14%	0.66%	3.30%	3.62%	10.69%	20.39%
sofa	57.56%	7.36%	6.75%	12.25%	12.84%	25.06%	21.51%
train	70.55%	5.60%	8.57%	19.13%	14.42%	35.11%	26.36%
tvmonitor	66.75%	7.36%	8.25%	21.25%	16.27%	36.82%	23.19%
<b>average</b>	67.49%	16.88%	16.17%	27.33%	25.00%	39.95%	37.89%

Table 9. Additional experimental results in SSD

Shape Area	Clean –	Square 9,800	Crescent 9,800	Square 5,000	Crescent 5,000
bird	92.09%	2.61%	2.44%	2.25%	2.48%
boat	78.31%	12.04%	10.86%	12.53%	11.68%
bottle	76.34%	16.04%	7.84%	10.20%	8.31%
car	86.53%	9.86%	10.02%	26.62%	12.09%
cat	89.15%	5.58%	8.30%	10.32%	0.77%
chair	84.24%	12.47%	9.23%	19.53%	21.71%
dog	87.89%	0.00%	0.00%	0.00%	0.14%
horse	86.17%	0.00%	0.00%	0.00%	0.23%
person	94.38%	7.56%	8.40%	36.09%	44.76%
pottedplant	76.16%	12.37%	8.28%	26.14%	10.95%
train	93.93%	0.91%	0.00%	4.79%	8.41%
tvmonitor	89.34%	2.25%	3.30%	9.45%	17.63%
<b>average</b>	86.21%	6.81%	<b>5.72%</b>	13.16%	<b>11.60%</b>

strong remote adversarial patches for large objects are crescent shaped.

### 3.4. Experiment on Generating Dynamic Remote Adversarial Patches

In the main paper, we conducted a theoretical analysis based on the information diffusion model of adversarial patch effects and performed a theoretical analysis on the shape of strong adversarial patches. In particular, the discussions up to the previous section aimed to precalculate the shape of strong adversarial patches based on the information diffusion model, and it has been shown that the calculation method is valid. This section demonstrates the validity of the shape calculated based on the information diffusion model, but with a different approach from the previous sections. Specifically, we discuss whether, when generating remote adversarial patches dynamically similar as Chen et al.’s method [2], it is consistent with the theory discussed up to the previous section.

In this paper, regarding the optimization function, we minimize the value (pixel score) obtained by adding the sum of the 2-norm of each pixel’s RGB value to the class value and NPS term. Specifically, in this paper, unlike the experiments up to the previous section, we start learning from a state where all RGB values are 1, and minimize

$$(\text{Confidence score}) + 0.01 \times (\text{NPS}) + 10^{-5} \times (\text{Pixel score}) \quad (35)$$

The reason we did not start with all RGB values set to 0 is to prevent the adversarial patch’s RGB values from remaining at 0 as learning progresses, due to the threshold mentioned later. Furthermore, the multiplier for pixel values was set to  $10^{-5}$  to satisfy the following two points:

- To be approximately the same as the confidence score (about 10%).
- To make the number of non-zero pixels  $O(10^4)$ .

In addition to the above, to reduce the less influential parts of adversarial patches, we set a threshold  $p$  and set the RGB values of pixels below this threshold  $p$  to 0.

In this paper, we applied the above learning method to the person class of the VOC dataset and conducted experiments under  $p = 0.01$  and  $0.02$  in YOLOv2. These values represent minor changes of two and five levels, respectively, out of the 256 discrete levels of RGB values. For  $p = 0.01$  and  $0.02$ , the number of pixel with non-zero RGB values were 54,649 and 42,845, respectively. The remote adversarial patches generated for  $p = 0.01$  and  $0.02$  are as shown in Figs. 1a and 1b, respectively. From Figs. 1a and 1b, it can be seen that the remote adversarial patches are mainly concentrated near the center, and their shape forms part of a circle centered on the middle of the image. Therefore, even if the shape of the remote adversarial patch is learned dynamically, its shape is consistent with the results of the

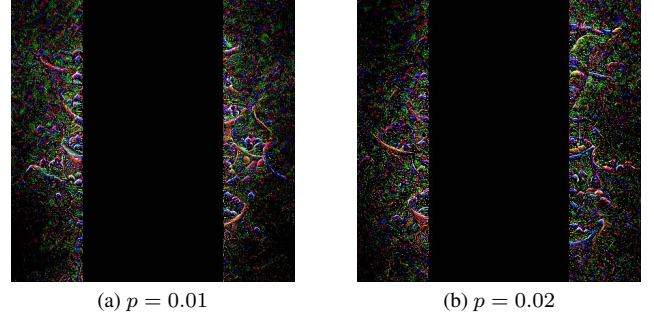


Figure 1. Adversarial patches dynamically generated against YOLOv2

theoretical analysis up to the previous section. Thus, the theoretical analysis in Sec. 3 in the main paper is valid.

## 4. Detailed Information for Strong Remote Adversarial Patches in Autonomous Driving Simulator

In this section, we provide detailed information in autonomous driving simulator [9]. First, in Sec. 4.1, we present the method for generating strong remote adversarial patches. Next, in Sec. 4.2, we present the changes in confidence scores between  $t = 3.0$  and  $3.8$ .

### 4.1. Shapes of Remote Adversarial Patches

In this section, we give an explanation of the prepared shape of remote adversarial patches. As demonstrated in the main paper, we utilize four remote adversarial patches shown in Fig. 2. Firstly, we will discuss the feasible regions for placing remote adversarial patches in Sec. 4.1.1. In particular, we focus on explaining the placement method of the remote adversarial patches at  $t = 3.6$ . For times other than  $t = 3.6$ , we perform the placement based on the remote adversarial patches at  $t = 3.6$ . Furthermore, we will explain the method for generating remote adversarial patches in Sec. 4.1.2. Finally, we will provide a detailed explanation of the methods for determining the shape and learning of these remote adversarial patches in Secs. 4.1.3 to 4.1.6, respectively.

#### 4.1.1. The Feasible Regions for Placing Remote Adversarial Patches

In this subsection, we will describe the feasible regions for placing remote adversarial patches, taking into account external factors such as the ground or the hood of a vehicle. The remote adversarial patches shift upwards as compared to that considered in theoretical analysis, because of the ground or the hood of a vehicle. Therefore, in this study, the designated regions  $0 \leq x \leq 271$  and  $(0 \leq y \leq 104 \text{ or } 311 \leq y \leq 415)$ , reverse-engineered from the images, are considered as the placement areas for remote adversarial patches. It should be noted that the size of the remote



adversarial patches is consistently 12,800 ( $= 2 \times 80 \times 80$ ) pixels.

#### 4.1.2. The Method for Generating Remote Adversarial Patches

In this subsection, we propose a method for generating remote adversarial patches for the autonomous driving simulator. For the autonomous driving simulator, the generation of remote adversarial patches is almost similar for images from the VOC dataset [3], but there are some differences in the training method. Specifically, the calculation method for confidence score differs, and for each image, the following calculation is performed:

1. The scaling factor for resizing the remote adversarial patches is randomly set to 300/416-450/416.
2. Following the approach by Thys et al. [10], the values for contrast and brightness to be added to the remote adversarial patches are determined as follows:
  - **Contrast:** A random value between 0.8 and 1.2, which remains constant for the entire image.
  - **Brightness:** A random value between  $-0.2$  and  $0.2$ , which remains constant for the entire image.
3. Set the iteration count  $j$  to 0.
4. Following the approach by Thys et al. [10], we randomly set noise for each pixel to take a value between  $-0.1$  and  $0.1$ , which is added to the remote adversarial patches.
5. Transform the remote adversarial patches as

$$\begin{aligned} (\text{Actual patch}) = & (\text{Current patch}) \times \text{Contrast} \\ & + \text{Brightness} + \text{Noise}. \end{aligned} \quad (36)$$

6. Resize the remote adversarial patches according to the scaling factor determined in step 1 and paste it onto the image.
7. Calculate the loss function by summing the confidence score and 0.01 times the NPS term, and update the remote adversarial patches.
8. Increment  $j$  by 1, and if  $j$  equals 10, terminate the processing for this image. Otherwise, return to step 4.

In the above procedure, determining the optimal values for parameters such as the scaling factor, contrast, brightness, and noise in the aforementioned procedure is a future challenge. In particular, determining the scaling factor is a crucial aspect that requires careful consideration, but we currently determine the scaling factor arbitrarily. Expanding the range of scaling factors can increase the effective range of the remote adversarial patches. However, it can also introduce unnecessary training, creating a trade-off between effectiveness and efficiency. Therefore, further investigation and analysis regarding the determination of the scaling factor are of great importance for future research. The number of iterations is set to 200 epochs.

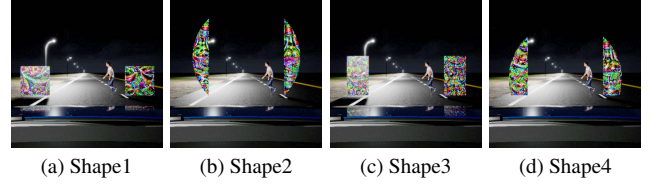


Figure 2. Prepared shapes of adversarial patches for the autonomous driving simulator. These adversarial patches have been simulated taking into account the influence of streetlight.

#### 4.1.3. Fig. 2a

Fig. 2a depicts a scenario where two square remote adversarial patches are placed on the ground. The remote adversarial patches on the left and right sides are both  $80 \times 80$  in size. During training, these remote adversarial patches are positioned such that their vertical center lines pass through the center of the image, similar to Sec. 4 in the main paper. When pasting these remote adversarial patches onto the simulator image, they are positioned just above the ground, specifically with the lower edge at  $x = 272$ , and aligned closer to the boundary on the side of the image where the remote adversarial patches are placed.

#### 4.1.4. Fig. 2b

Fig. 2b depicts a scenario where two crescent-shaped remote adversarial patches are placed on the ground. During training, these remote adversarial patches are selected in order of proximity to the center, with a total of 12,800 pixels chosen. When pasting these remote adversarial patches onto the simulator image, they are positioned just above the ground, specifically with the lower edge at  $x = 272$ , and aligned closer to the boundary on the side of the image where the remote adversarial patches are placed.

#### 4.1.5. Fig. 2c

Fig. 2c represents a rectangular remote adversarial patches that satisfy the following requirements:

- The shape of the remote adversarial patches is nearly rectangular.
- The remote adversarial patches are positioned on the ground, meaning that  $x < 272$ .
- The remote adversarial patches do not go off-screen in the next frame. In this study, the horizontal limits of the region where the remote adversarial patches can exist are set as  $45 \leq y \leq 104$  or  $311 \leq y \leq 370$ .
- The remote adversarial patches are selected in order of proximity to the area where people pass. In this study, the pixels of the remote adversarial patches are chosen in order of proximity to the center at  $x = 226.0$ .

#### 4.1.6. Fig. 2d

Fig. 2d represents a crescent-shaped remote adversarial patches that satisfy the following requirements:

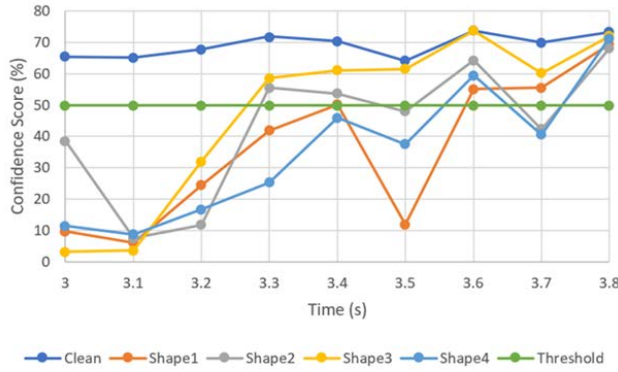


Figure 3. Confidence score for the autonomous driving simulator in YOLOv2

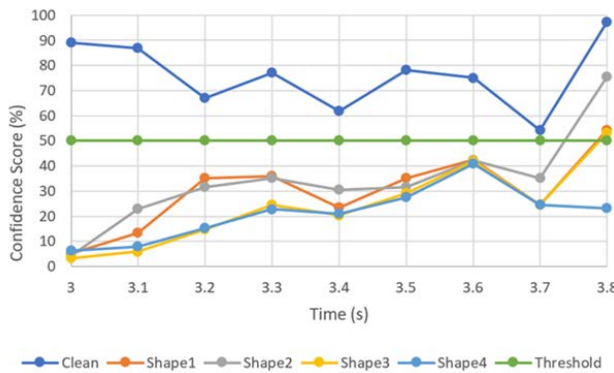


Figure 4. Confidence score for the autonomous driving simulator in SSD

- The shape of the remote adversarial patches is nearly crescent-shaped.
- The remote adversarial patches are positioned on the ground, meaning that  $x < 272$ .
- The remote adversarial patches do not go off-screen in the next frame. In this study, the horizontal limits of the region where the remote adversarial patches can exist are set as  $45 \leq y \leq 104$  or  $311 \leq y \leq 370$ .
- The remote adversarial patches are selected in order of proximity to the area where people pass. In this study, the pixels of the remote adversarial patches are chosen in order of proximity to the center at  $(x, y) = (226.0, 208.0)$ .

## 4.2. Additional Experimental Results

The changes in confidence scores are as shown in Figs. 3 and 4, and the generated remote adversarial patches have attack capabilities.

## References

- [1] Models and pre-trained weights – torchvision main documentation. <https://pytorch.org/vision/>

- master/models.html. [Online; accessed 19-Feb-2024]. 5
- [2] Zhaoyu Chen, Bo Li, Shuang Wu, Jianghe Xu, Shouhong Ding, and Wenqiang Zhang. Shape matters: deformable patch attack. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IV*, pages 529–548. Springer, 2022. 8
- [3] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111:98–136, 2015. 9
- [4] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2):318–327, 2018. 1, 3, 5
- [5] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 21–37. Springer, 2016. 1, 3, 4, 5
- [6] Kento Oonishi, Tsunato Nakai, and Daisuke Suzuki. Multiple remote adversarial patches: Generating patches based on diffusion models for object detection using CNNs. In *NeurIPS ML Safety Workshop*, 2022. 3, 4
- [7] Joseph Redmon and Ali Farhadi. YOLO9000: Better, faster, stronger. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6517–6525. IEEE, 2017. 1, 3
- [8] 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.
- [9] Koichi Shimizu, Daisuke Suzuki, Ryo Muramatsu, Hisashi Mori, Tomoyuki Nagatsuka, and Tsutomu Matsumoto. Evaluation framework for performance limitation of autonomous systems under sensor attack. In *Computer Safety, Reliability, and Security: 40th International Conference, SAFECOMP 2021, York, UK, September 8–10, 2021, Proceedings 40*, pages 67–81. Springer, 2021. 8
- [10] Simen Thys, Wiebe Van Ranst, and Toon Goedemé. Fooling automated surveillance cameras: Adversarial patches to attack person detection. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 49–55. IEEE, 2019. 9
- [11] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9627–9636, 2019. 1, 3, 5