

# Attck Monocular Depth Estimation Model with a Sticker

Meng Pan, Fangjun Huang<sup>†</sup>

School of Computer Science and Engineering, Sun Yat-sen University, China

panm9@mail2.sysu.edu.cn, huangfj@mail.sysu.edu.cn

## Abstract

DNNs have been demonstrated that their performance can be degraded by imperceptible perturbations. Monocular Depth Estimation networks with DNNs likewise encounter such a dilemma. This attack diagram, adding perturbations, is not implementable and capturable for a sensor, which means it is an ideal attack diagram. An implementable and applicable attack diagram in the physical world is needed. To this end, we address this by sticking patches into indoor scenes. We proposed an automatic searching and transformation method. In the searching phase, the patch is dispatched to an appropriate and sensitive location exploiting the MDE's gradient. In the transformation phase, the patch alternates its appearance according to its location. In doing this, we can generate destructive, inconspicuous adversarial examples. This whole process doesn't rely on any generate networks, which makes it fast and light. We evaluate our method on NYU depth v2, experiments show that our method eliminates 6% of RMSE in 0.8s meanwhile keeping remarkable inconspicuousness.

## 1. Introduction

With the rapid development of Deep Neural Networks(DNNs), many excellent Monocular Depth Estimation (MDE) methods base on DNNs have been proposed recently. However, according to Szegedy's research [10], DNNs are prone to be attacked by adversarial examples: when an imperceptible small perturbation is added to an input image, the classifier based on DNNs will classify the obtained image, known as adversarial examples, into a wrong category with high probability. Adversarial examples have been found not only on classification tasks but also on logical regression tasks, such as semantic segmentation and object recognition. MDE also fail to get rid of this dilemma. As learned perception models are increasingly deployed on autonomous driving vehicles, researchers

<sup>†</sup>Corresponding author

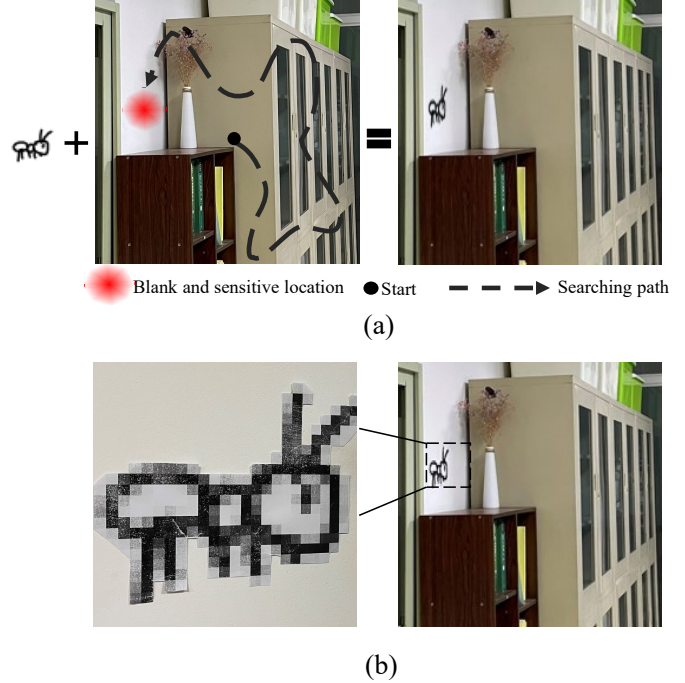


Figure 1. Preview of our patch attack method. The adversarial example generated in pixel level by our concealed attack method(a). We stick a printed drawing in the physical world (b), and Note that this was shot in our office, not the dataset scene we used in the experiment part.

attach more and more attention to attacking MDE models.

The threat of adversarial example on MDE algorithms was first studied by Wong *et al* [12]. They conducted comprehensive and rigorous experiments to explore the impact of pixel attacks on MDE, including attacking the whole depth map of a scene, attacking the depth map of one single object in the scene, and even making a specific object vanished completely on the depth map by adding perturbations.

Although the research of Wong *et al* is pioneering and enlightening on attacking a MDE model, but their method is hard to be realized in the physical world, because the perturbations added on the benign images can not be captured by a camera.

However, it is feasible to attack DNNs by adversarial examples generated with visible patches in the physical world. Many works have emerged to explore this attack paradigm recently. Brown et al. [1] abandon the camouflage of the attack, and they attack DNNs by sticking an apparent, randomly transformed patch on benign images, which is optimized cross-scenes to raise the probability that it is misclassified as the target category. In doing so, they obtained a robust and universal patch. Nevertheless, the adversary’s attack intention will expose easily as a result of the unnatural, conspicuous patches.

Thus, researchers begin to pay attention to the concealment of adversarial examples. Sharif et al. [9] tack the concealment problem by disguising the patches as accessories. Specifically, they attacked a face recognition system by wearing an artificial glasses frame, the pattern of which is optimized by gradient descent with the goal of misrecognition. Besides, to be able to realize the pattern use a printer, they define the non-printability score (NPS) of images to be high for images that contain unreproducible colors, and low otherwise. Finally, they are misrecognized by the victim face recognition system with the printed, real glasses frame.

Similarly, for better concealment, Hu *et al* [4] disguise tha patches as the pattern on shirts to attack human detection models. They first detect all humans in the benign image and pin optimizable patches generated through a GAN on the body areas to obtain adversarial examples, which are then fed into the victim model to optimize the generated patches by reducing the detection rate. Kong et al. [7] also use a GAN to generate patches. however, instead of generating directly, their patches are generated by adding perturbations through the GAN. Concretely, they collect advertisement posters and send them into a GAN to generate adversarial examples by adding perturbations. Finally the perturbed patches are artificially stuck to street billboards, which is inconspicuousness for posters. This method achieves better inconspicuousness, yet the process of sticking the patch to billboards is labor consuming.

The locations of the patches are fixed in [4, 7, 9], conceivably, the locations of patches also have an impact on the attack success rate. Eykholt et al. [3] demonstrate this hypothesis through empirical experiments and they locate this area by visualizing the perturbations added using  $L_1$  regularization. Instead of locating the vulnerable areas by visualizing the perturbations, they locate the vulnerable areas by a attention model, which takes the benign image as input, outputs a mask that indicates where is more vulnerable and pin the patch on this area.

Indeed, affiliated GAN and attention networks enhance the concealment of the adversary, but the time and space complexity also have been increased. In this paper, we take physical realizability and inconspicuousness into account, and design a fast and automatic method to attack MDE

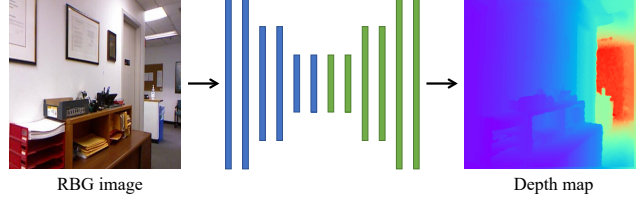


Figure 2. A common stream of MDE algorithms in inference stage.

algorithms. Fig. 1(a) shows the outline of our method. After searching for the optimal location through gradient descent, we alter the patch’s appearance according to the visual effect and ‘stick’ it to the optimal location In pixel level to generate our adversarial examples. Meanwhile, in Fig. 1(b), we print the patch and stick it in the physical world. It can be seen that adversarial example in digital world and physical world can not be distiguished. This indicate that our adversarial examples generated on pixel level can be simulated through sticking a printed patch on the real scene.

The contributions of this work are as follows:

- We are the first to apply patch attack on MDE tasks.
- A fast, automatic, inconspicuous attack method is proposed. In this method, a learnable matrix is designed to update the patch’s location based on attack effect and visual quality using gradient descent.
- We apply perspective transform according to visual effect to alter the patch’s appearance and make it looks inconspicuous.
- Experiments show that our method generate destructive while concealed adversarial examples.

The structure in the rest paper is organized as follows. After a review of MDE and patch attacks in Sec. 2, we introduce our method concretely in Sec. 3. Then we visualize the effect of our location searching phase and attack a MDE network in Sec. 4. Finally We analyze our patch attack method in Sec. 5.

## 2. Background

In this paper, we concentrate on patch attack on MDE models. Therefore, we will give a brief introduction for MDE task.

With the rapid development of DNNs, MDE task achieves satisfactory progress. The problem of predicting a depth map from a single RGB image can be treated as a supervised learning problem. Denote the space of RGB images as  $\mathcal{I}$  and the domain of ground truth depth maps as  $\mathcal{D}$ , given a training set  $\mathbb{T} = (I^{H \times W \times 3}, D^{H \times W})$ ,  $I \in \mathcal{I}$  and

$D^* \in \mathcal{D}$ , a MDE model  $f_{mde}$  can learn a non-linear mapping  $f_{mde} : \mathcal{I} \rightarrow \mathcal{D}^*$  by optimize the objective function:

$$\min \|f_{mde}(I^{adv}) - D^*\| \quad (1)$$

where  $\|\cdot\|$  denotes a metric for distance. Fig. 2 shows a common inference stream of a MDE model: A RGB image is fed into the trained MDE model, which generate a dense depth map from the RGB image.

In this paper, we use Fastdepth [11], an efficient and lightweight encoder-decoder MDE network, as the victim model to study the effect of our adversarial examples.

### 3. Method

To obtain adversarial examples with a patch, there are two problems to be tackled: the patch’s location and the patch’s appearance. Therefore, we design a two-phase method, which is shown in Fig. 3. The first phase named **Location Searching Phase** searches for a vulnerable and inconspicuousness location on the benign image and the second named **Appearance Altering Phase** alters the patch’s appearance to make it inconspicuous.

We first formulize the generation of the adversarial examples as an optimal problem in Sec. 3.1. Then the location searching phase and appearance altering phase are presented in Sec. 3.2 and Sec. 3.3, respectively.

#### 3.1. Problem Statement

Considering that elaborately designed perturbations can hardly be captured by consumer cameras, we propose to obtain adversarial examples  $I^{adv}$ <sup>1</sup> by sticking visible patches  $\delta^{h \times w}$  to benign images  $I$ :

$$I^{adv} = I + \delta \quad (2)$$

Denote the victim MDE model as  $f_{mde}$ , it was trained on the training set  $\mathbb{T} = (I^{H \times W \times 3}, D^{*H \times W})$ . We aim to derive the optimal adversarial examples in an optimization manner:

$$\operatorname{argmin}_{I^{adv}} \mathcal{L}(I^{adv}), \quad (3)$$

where  $\mathcal{L}$  represents the objective function, which will be designed under two considerations: destructiveness and inconspicuousness.

#### 3.2. Location Searching

In this subsection, we will describe our location searching phase in detail, of which consists patch design, objective function, and optimization of the location. The location searching phase can be seen in Fig 3(a).

<sup>1</sup>Eq. 2 is a simplified definition of  $I^{adv}$ . the concrete definition will be given in Sec. 3.2.1.

##### 3.2.1 Patch Design

For better inconspicuousness, we camouflage the patch added in the scene as a graffiti. Therefore, simple line-drawn patches are preferred. We select a drawing from QuickDraw [5], a collection of 50 million hand-writing drawings across 345 categories, as our patch  $\delta$ . The drawing is shown in Fig. 4. To obtain adversarial examples with  $\delta$ , we organize a mask  $M : \{0, 1\}^{H \times W}$  by following process: the patch  $\delta$  is placed in the center of the clean image  $I$ , then we expand it to the size of  $I$  by padding value one around it. Our adversarial examples is composed of the clean image  $I$  and the mask  $M$ :

$$I^{adv} = M \odot I \quad (4)$$

where  $\odot$  is element-wise multiplication. Fig. 5 shows our mask and the representation of our adversarial examples.

Instead adding perturbations on a patch or generating a patch, we select a patch and put it on a specific location of the clean image  $I$  to generate destructive and concealed adversarial examples. Exhaustively searching the location on  $I$  is feasible but time consuming. Our method, however, searches the location in an optimization manner, that is, the patch location is updated according to optimization objectives. To this end, we exploit affine transform to update the location. In particular, we define a  $2 \times 3$  matrix  $\mathbf{A}_\theta$ , with which we can translate the patch ont the clean image:

$$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix} \quad (5)$$

where  $t_x, t_y$  denote the offsets in horizontal and vertical directions, respectively. Denote the  $[t_x, t_y]^T$  as  $\mathcal{T}$ . Initial values of  $\mathcal{T}$  is  $[0, 0]^T$ , which represents that the mask overlaps the clean image completely and the patch locate on the center of the image. The homogeneous coordinate of our mask  $M$  is noted as  $G$ . Given an affine transform matrix, we can obtain a new coordinate  $G'$  from  $G$ :

$$G'^T = A_\theta G^T \quad (6)$$

Denote the coordinate of each pixel of  $M$  as  $G_i = (x_i, y_i, 1)$ , they are moved to a new coordinate:

$$\begin{pmatrix} x_s \\ y_s \end{pmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix} \begin{pmatrix} x_t \\ y_t \\ 1 \end{pmatrix} \quad (7)$$

Based on this, the patch  $\delta$  can perform arbitrary translation on the image  $I$ .

##### 3.2.2 Objective Function

As illustrated before, our patch slides on the image through affine transform in an optimization manner. During optimization, we consider two factors: attack effect and inconspicuousness. Based on this considerations, we design two optimization objectives.

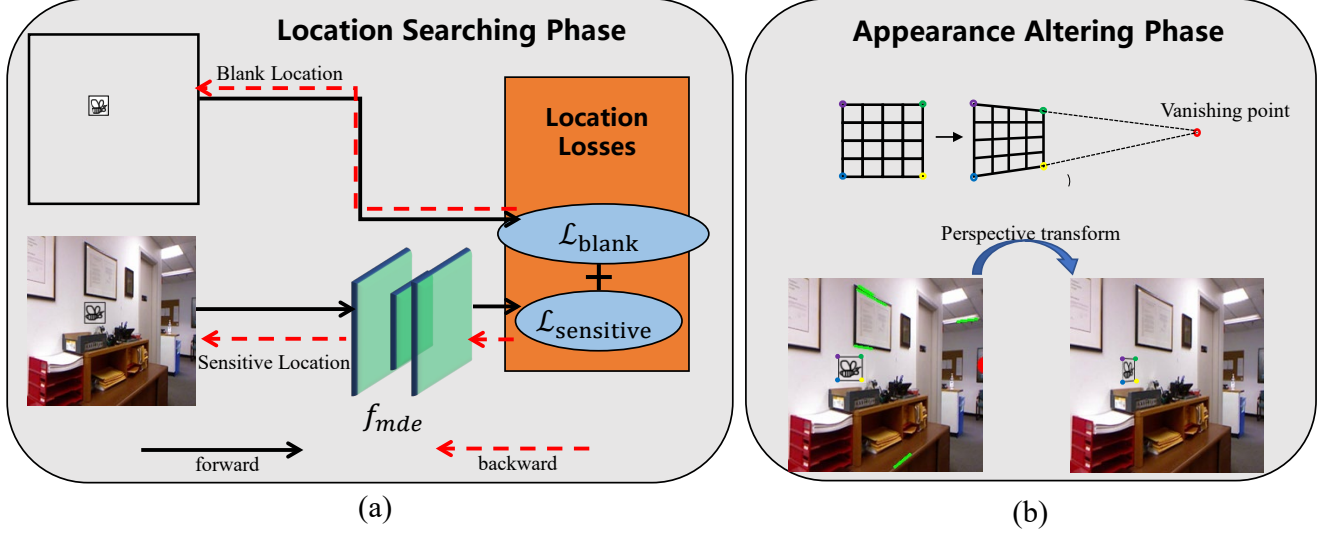


Figure 3. Stream of our MDE attacking method. The whole stream consists of two phases: (a) location searching phase and (b) altering appearance phase. They are described concretely in Sec. 3.2 and Sec. 3.3, respectively.

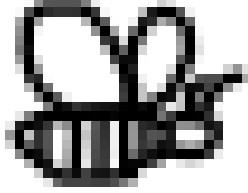


Figure 4. Our patch selected from QuickDraw.

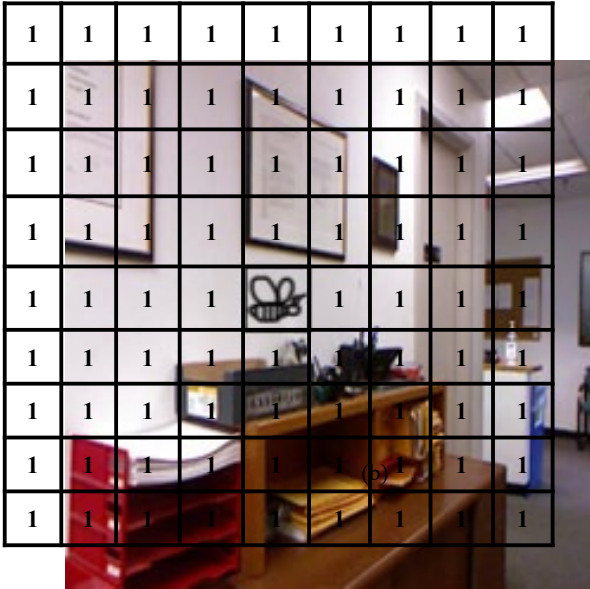


Figure 5. Illustration of our mask  $M$ , with which we can generate adversarial examples through  $M \odot I$ .

**Attack effect** MDE models  $f_{mde}$  generate depth maps, whose pixel values represent the distance from area to sample sensors. To obtain the best attack effect, that is, to maximize the error between depth maps predicted from adversarial examples and ground truth depth maps. We name the location that has better attack effect sensitive location. We design  $\mathcal{L}_{sensitive}$  and search the sensitive location by maximize the function:

$$\mathcal{L}_{sensitive} = \|f_{mde}(I^{adv}) - D^*\|_p \quad (8)$$

where  $P$  represents a distance metric, we use  $L_1, L_2$  and Huber distance in our experiments.

**Inconspicuousness** For better inconspicuousness, we camouflage the patch as a graffiti, which can be usually found on blank walls or furnitures. We design Blank loss to help the patch to find blank areas,  $\mathcal{L}_{blank}$  consists of two optimization sub-terms:

$$\mathcal{L}_{blank} = \mathcal{L}_{smooth} + \mathcal{L}_{white} \quad (9)$$

In Eq. 9,  $\mathcal{L}_{smooth}$  constrain the patch locate on areas with low texture complexity:

$$\mathcal{L}_{smooth} = -\frac{\sqrt{\partial_x^2(I - M \odot I) + \partial_y^2(I - M \odot I)}}{n} \quad (10)$$

where  $\partial_x$  and  $\partial_y$  denote the gradient in  $x$  and  $y$  directions, respectively, and  $n$  is the number of pixels in patch  $\delta$ .  $\mathcal{L}_{white}$  constrain the patch locate on white areas:

$$\mathcal{L}_{white} = \frac{\sum_1^n I - M \odot I}{n} \quad (11)$$



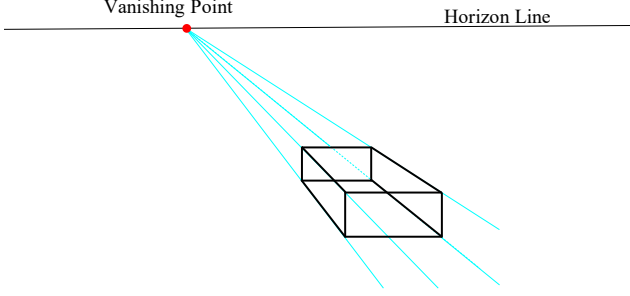


Figure 6. Diagram of single point perspective drawing: All the vanishing lines lead to the vanishing point. The horizontal and vertical lines, however, remain parallel to each other.

The two loss functions are summed up in a linear manner to compose our objective function:

$$\mathcal{L}_{total} = \mathcal{L}_{sensitive} + \mathcal{L}_{blank} \quad (12)$$

### 3.2.3 Location Optimization

As indicated in Eq.7, the target coordinates  $G'_i$  are continuous values. In other words, the value of the target pixels are computed by a sampling method. We adopt bilinear interpolation sampling to determine the value of the target pixel, which is proved to be differentiable in *spatial transformer networks* [6]. By maximizing the  $\mathcal{L}_{total}$ , we can optimize the offsets  $\mathcal{T}$  step by step.

$$\mathcal{T}_0 = [0, 0]^\top, \mathcal{T}_{s+1} = Clip\{\mathcal{T}_s + \alpha \nabla_{\mathcal{T}} \mathcal{L}_{total}\}. \quad (13)$$

where  $\mathcal{T}$  denotes the translation vector  $[t_x, t_y]^\top$ . To avoid patches overflowing from clean images,  $t_x$  and  $t_y$  are clipped between  $(\frac{W-w}{W}, -\frac{W-w}{W})$  and  $(\frac{H-h}{H}, -\frac{H-h}{H})$ , respectively. In our experiments, the size of clean images is  $224 \times 224$  and the size of the patch is  $28 \times 28$ , which means the  $t_x$  and  $t_y$  are both clipped between  $(0.875, -0.875)$ . For all experiments we set  $\alpha = 0.004$ , total optimization step is 50 steps.

### 3.3. Appearance Altering

In location searching phase, our method find a sensitive and concealed location for the patch. At this phase, we alter the appearance of the patch according to the perspective transformation principle to make it looks more natural. The phase can be seen in Fig. 3(b).

For better visualization quality, we follow the principle of perspective drawing: the vanishing point is where all parallel lines intersect and is always on the horizon line. Fig 6 clearly illustrates this principle.

To obtain the vanishing point, we detect straight lines in the adversarial examples with *Hough* transform and extend

them. The intersection of these lines is vanishing point obviously.

As illustrated in Fig 3(b), if we connect the left-top point and vanishing point, the right-top point should lay on this line. Same with the bottom line. Base on this principle, we calculate four corrected corner coordinates. The perspective transforms in this phase are implemented with *OpenCV*.

## 4. Experiments

In this section, we conduct several experiments to exhibit the results of our automatic and inconspicuous attack method on a MDE model. The implementation detail including datasets, data pre-processing and experiment environment is introduced in Sec. 4.1. Then we demonstrate the automatic optimization performance in Sec. 4.2. Finally, our attack effects are shown by visualization and quantitative comparison in Sec. 4.3 and Sec. 4.4.

### 4.1. Implementation Detail

To study the effect of our patch adversarial examples, we examine the robustness of a supervised method Fastdepth [11]. The examined MDE model is trained and evaluated on NYU depth v2 [8], a widely used indoor dataset with 0-10 meters depth range. NYU depth v2 consists of 1449 densely annotated depth and RGB images pairs, according to Eigen *et al* [2], 795 of them are split to train dataset and the rest are split to test dataset. For the image pre-processing, we followed the process in Fastdepth: images are resize to  $250 \times 333$  firstly, secondly, center cropped to  $224 \times 304$ , in the end they are resized to  $224 \times 224$ . As mentioned in 3.2.1, adversarial examples generated with simple line-drawn patches looks more natural. Therefore, we select a size of  $28 \times 28$  bee drawing from QuickDraw [5] to generate our adversarial examples.

We focus on four metrics used in previous works to evaluate the attack effect:

$$RMSE = \sqrt{\frac{1}{|N|} \sum_{i \in N} ||D_i - D_i^*||^2} \quad (14)$$

$$MAE = \frac{1}{|N|} \sum_{i \in N} |D_i - D_i^*| \quad (15)$$

$$Abs Rel = \frac{1}{|N|} \sum_{i \in N} \frac{|D_i - D_i^*|}{D_i^*} \quad (16)$$

$$Threshold : \% \text{ of } D_i \text{ s.t. } \max(\frac{D_i}{D_i^*}, \frac{D_i^*}{D_i}) = \delta < thr \quad (17)$$

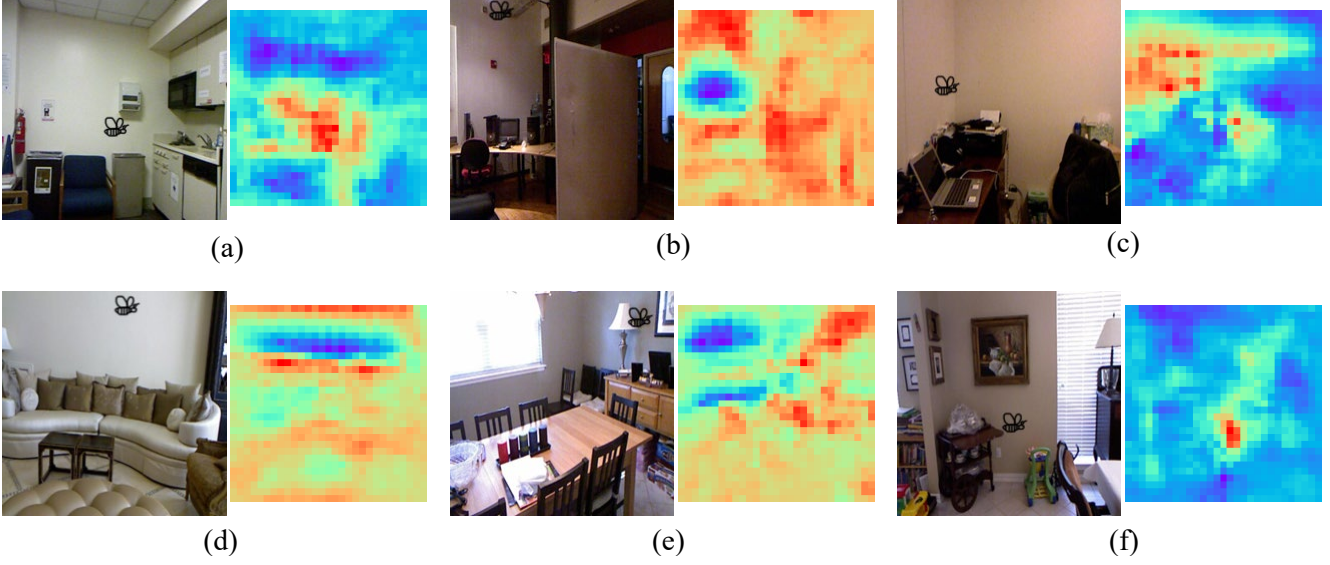


Figure 7. The final attack locations searched by our method (left) and corresponding MAE maps (right). Then calculate MAE maps, the offsets of the patch are clamped between  $-0.875$  and  $0.875$ , which makes the maps a little smaller than the RGB images.

where  $D_i$  represents the predicted depth map from  $I^{adv}$ ,  $N$  is the pixel number of  $I^{adv}$ , and  $thr$  denotes three threshold  $1.25$ ,  $1.25^2$ ,  $1.25^3$ . Besides the accuracy, we also record our runtime, which can show the efficiency of our attack method.

#### 4.2. Automatic Searching

Given a clean image, our attack method can search the sensitive area and stick the patch on this area. To better demonstrate our search effect, we calculate the MAE maps of test images. The MAE maps are obtained using following step: beginning from the left top corner, we use strides 7 to slide the patch step by step and calculate the MAE of each position. The red and violet color in the maps represent higher and lower MAE values, respectively. As you can see from Fig 7, our method can accurately search the sensitive area of the image. Note that to emphasize our search effect, we didn't use the  $\mathcal{L}_{blank}$  Eq.9 in this experiment.

#### 4.3. Visualization Attack Results

For inconspicuousness, we show the adversarial examples in the first and third rows. The black box shows that our adversarial examples follow the principle of perspective drawing. Besides the patches lay on walls and furniture, which makes our examples look like graffiti and more harmonious in indoor scenes.

For destructiveness, we show the prediction error in the second and fourth row. Our method achieves  $320mm$  max error. Prediction results from  $I^{adv}$  are usually nearer than ground truth depth maps in far-range areas, for instance, the blue areas in Fig 8. However, the results are farther in near-

range areas, which means our adversarial examples mislead the MDE system and make the prediction results converge to middle distances.

#### 4.4. Quantitative Attack Results

We show our attack effect by quantitative comparison in Table 1. To the best of our knowledge, we are the first to attack MDE systems using patches. There does not exist much related works. So we compare the baseline MDE method, fastdepth between our method with different  $\mathcal{L}_{attack}$ . We degrade the MAE by 9.13%, as a contrast, Wong [12] achieve 10% accuracy loss (closer or farther) on each pixel of the depth. However, they rely on the guidance of the modified ground truth depth maps, in other words, they modify the ground truth depth value down or up 10% in each pixel, and use the modified maps as supervision. Besides, they use elaborately designed perturbations to generate adversarial examples.

### 5. Conclusion

We attacked an MDE system using our fast, physical-realizable, automatic and inconspicuous patch attack method. Fast because it doesn't lean on any generative networks. Physical-realizable because the effect can be simulated by sticking a real patch on the sensitive spot. Automatic because it is optimized by gradient descent. Inconspicuous because our method conceals the patch as graffiti. Our experiments showed that the depth maps predicted by the target MDE system lost some accuracy. Our work revealed the probability that an accident could happen due to drawing graffiti intentionally or unintentionally. Future

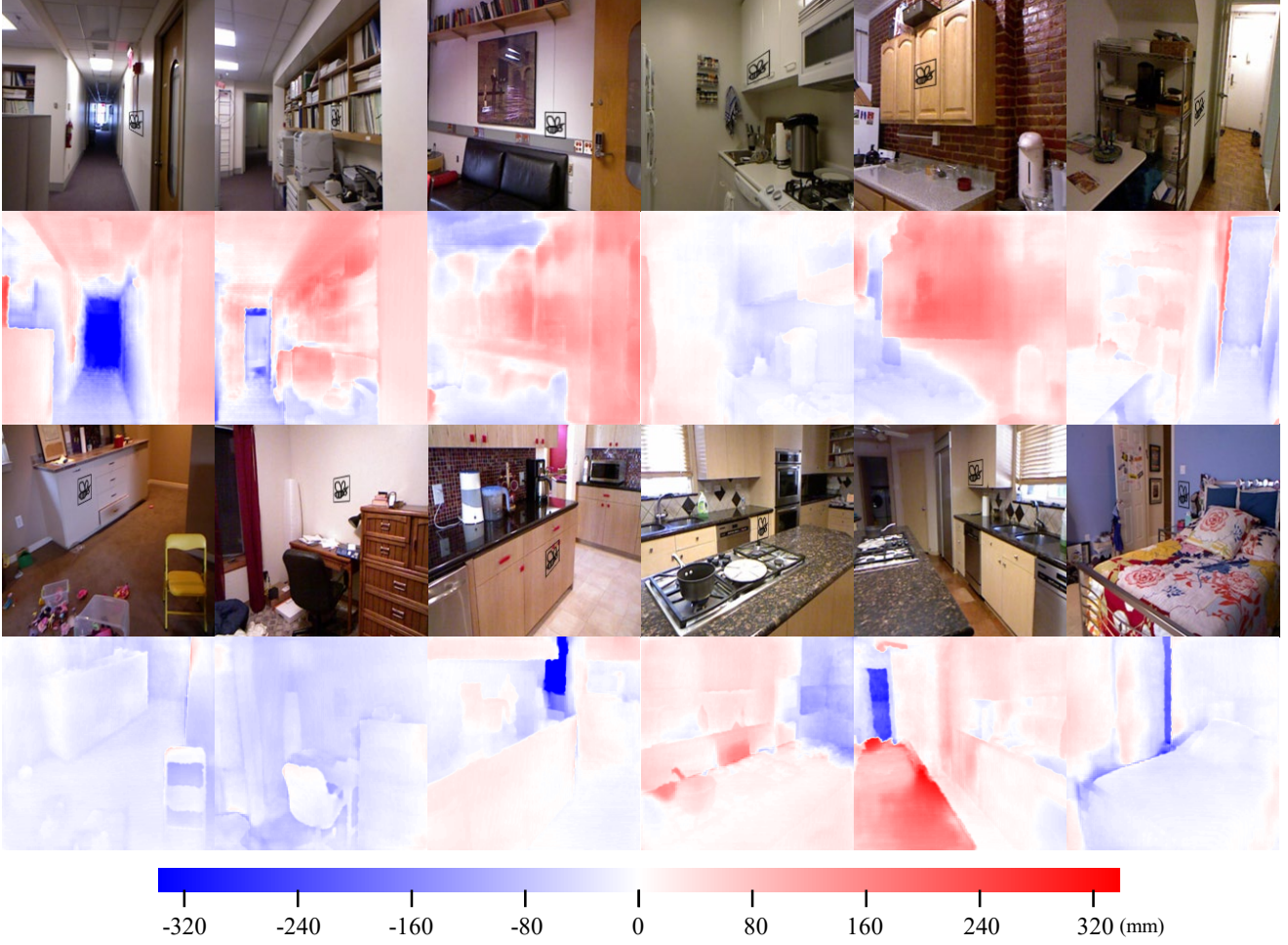


Figure 8. The visualization of our patch attack method. The first and third rows are  $I^{adv}$ . The second and fourth rows are the visualization of prediction error:  $f_{mde}(I^{adv}) - D^*$ . The color bar down the results explicates the error value intuitively. We emphasize the patches using surrounding black boxes, which are transformed in the altering appearance phase also.

Table 1. Accuracy of our method, best results are boldfaced

		RMSE	MAE	Absrel	$\delta_1 < 1.25$	$\delta_2 < 1.25^2$	$\delta_3 < 1.25^3$	Runtime
fastdepth (baseline)		0.600	0.427	0.162	0.771	0.927	0.988	-
Ours	$L_1$	0.643	0.464	0.175	0.738	0.920	0.973	0.772
	$L_2$	0.641	0.461	<b>0.177/9.3%</b>	0.739	0.921	0.973	0.781
	Huber	<b>0.648/6%</b>	<b>0.466/9.1%</b>	0.176	<b>0.737</b>	<b>0.919</b>	<b>0.972</b>	0.772

works should pay more attention to multimodal fusion or provide robust constraints for MDE networks.

## 6. Acknowledgements

xxx

## References

- [1] Tom B Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017. 2
- [2] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran As-

- sociates, Inc., 2014. 5
- [3] Kevin Eykholt, Ivan Evtimov, Earlene 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 (CVPR)*, June 2018. 2
  - [4] 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 (ICCV)*, pages 7848–7857, October 2021. 2
  - [5] T. K. J. K. J. Jongejan, H. Rowley and Fox-Gieg. The quick, draw!, 2016. <https://github.com/googlecreativelab/quickdraw-dataset>. 3, 5
  - [6] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2, NIPS'15*, page 2017–2025, Cambridge, MA, USA, 2015. MIT Press. 5
  - [7] Zelun Kong, Junfeng Guo, Ang Li, and Cong Liu. Physgan: Generating physical-world-resilient adversarial examples for autonomous driving. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2
  - [8] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgb-d images. In *ECCV*, 2012. 5
  - [9] 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, CCS '16*, page 1528–1540, New York, NY, USA, 2016. Association for Computing Machinery. 2
  - [10] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. 1
  - [11] Diana Wofk, Fangchang Ma, Tien-Ju Yang, Sertac Karaman, and Vivienne Sze. Fastdepth: Fast monocular depth estimation on embedded systems. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6101–6108, 2019. 3, 5
  - [12] Alex Wong, Safa Cicek, and Stefano Soatto. Targeted adversarial perturbations for monocular depth prediction. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 8486–8497. Curran Associates, Inc., 2020. 1, 6