Attck Monocular Depth Estimation Model with a Sticker

Meng Pan, Fangjun Huang†
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 [9], 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, knowned 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

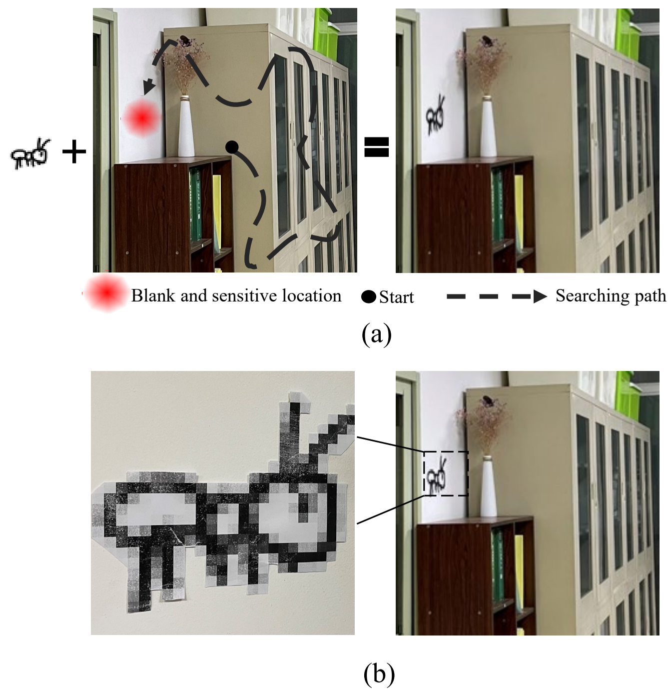


Figure 1. Preview of our patch attack method. The adversarial example generated in pixel leveal by our concealed attack method(a). We stick a printed drawing in the physical world (b), and Note that this was shot in our office, not the dataset scene we used in the experiment part.

attach more and more attention to attacking MDE models.

The threat of adverserial example on MDE algorithms was first studied by Wong *et al* [11]. 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.

†Corresponding author

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. [8] 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. In doing so, they are misrecognized by the victim face recognition system with the printed 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. Kong et al. [6] also use a GAN to generate patches. howerver, instead of generating directly, their patches are generated by adding perturbations through the GAN.

The locations of the patches are fixed in [4, 6, 8], 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.

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 accout, and design a fast and automatic method to attack MDE 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.

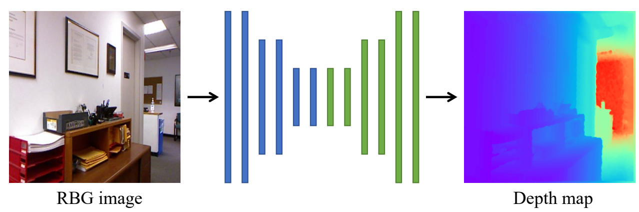


Figure 2. A common stream of MDE algorithms in inference stage.

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 optimizing the objective function:

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

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 [10], an efficient and lightweight encoder-decoder MDE network, as the victim model to study the effect of our adversarial examples.

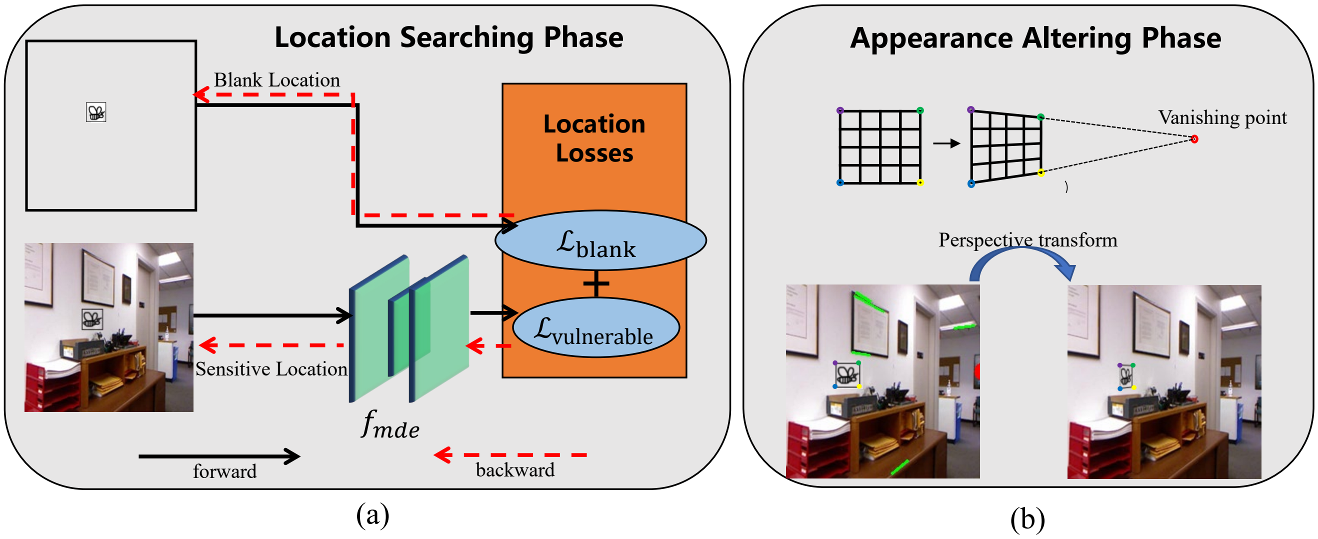


Figure 3. Stream of our MDE attacking method. The whole stream consists of two phases: (a) location searching phase and (b) altering appearance phase. They are described concretely in Sec. 3.1 and Sec. 3.2, respectively.

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.

Then the location searching phase and appearance altering phase are presented in Sec. 3.1 and Sec. 3.2, respectively.

3.1. 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).

3.1.1 Patch Design

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

For better inconspicuousness, we disguise the patch a graffiti, which still not shows attack intention when it is noticed in a scene. Therefore, we select a simple line-drawn drawing from QuickDraw [5], a collection of 50 million hand-writing drawings across 345 categories, as our patch. The drawing is shown in Fig. 4, with which our adversarial examples are defined as follows:

$$I^{adv} = (1 - m) \odot I + m \odot \delta \tag{2}$$

where m is a mask $m : \{0, 1\}^{H \times W}$ that covers the patch area with 1 and others with 0, $\odot$ is element-wise multiplication. We extend δ to size of $H \times W$ by padding 1 around it to conduct the element-wise multiplication. The definition

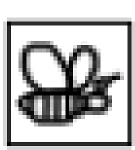
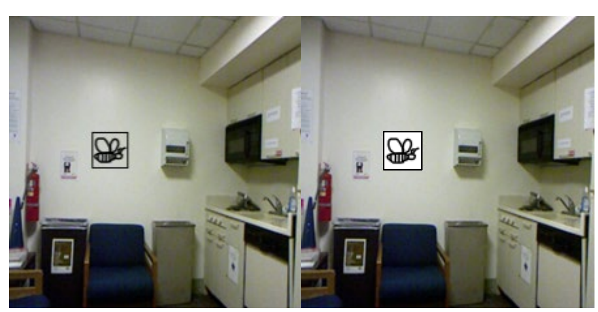


Figure 4. Our patch selected from QuickDraw. We put the patch in a black box for better visualization.



(a) (b)

Figure 5. Two definition of the adversarial examples. (a) is the traditional definition used in prior works. (b) is our definition, which reduces the modification to I and makes the patch looks like a graffiti.

of Eq. 2 modifies the whole patch area, to reduce modifications to I, our final definition of I^{adv} is:

$$I^{adv} = I - I \odot \delta \tag{3}$$

Instead adding perturbations on a patch or generating a patch, we select a patch and put it on a specific location of the benign image to generate destructive and concealed adversarial examples. Exhaustively searching on I for this specific location is feasible but time consuming. Therefore, we propose to search for the location in an optimization manner, which is automatic and consumes less time.

To this end, we first set up a coordinate grid on the benign image, with which the patch can navigate on the benign image. The coordinate grid is shown in Fig. 6.

After that, we exploit affine transform to perform the translation. In particular, we organize a 2×3 matrix $\mathbf{A}_\theta$ to translate the patch on the benign image:

$$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix} \tag{4}$$

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

$$G'^T = A_\theta \, G^T \tag{5}$$

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

$$\begin{pmatrix} x_t \\ y_t \end{pmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix} \begin{pmatrix} x_s \\ y_s \\ 1 \end{pmatrix} \tag{6}$$

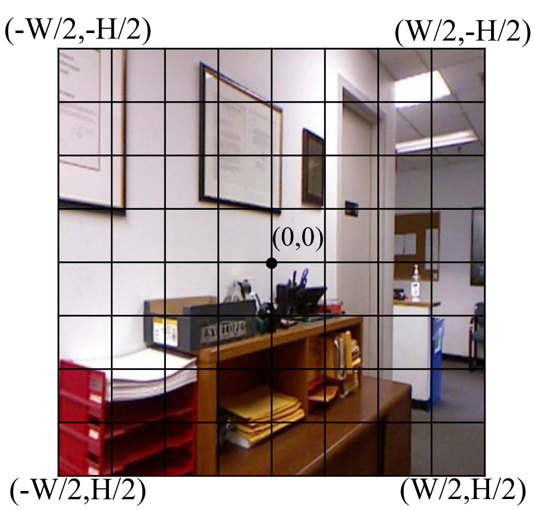


Figure 6. The coordinate grid of the benign image.

Based on this, the patch δ can perform arbitrary translation on the benign image.

3.1.2 Objective Function

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

Destructiveness The MDE model f_{mde} generates depth maps, whose pixel values represent the distance from the sampled scene to the sensor. We aim to obtain the adversarial examples that have strong destructiveness, that is, to maximize the error between depth maps predicted form adversarial examples and ground truth depth maps. We design $\mathcal{L}_{vulnerable}$ to search for the vulnerable location:

$$\mathcal{L}_{vulnerable} = \| f_{mde}(I^{adv}) - D^* \|_p \tag{7}$$

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

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

$$\mathcal{L}_{blank} = \mathcal{L}_{smooth} + \mathcal{L}_{white} \tag{8}$$

In Eq. 8, $\mathcal{L}_{smooth}$ constrain the patch locate on areas that have low texture complexity:

$$\mathcal{L}_{smooth} = -\frac{\sqrt{\partial_x^2(I - m \odot I) + \partial_y^2(I - m \odot I)}}{n} \tag{9}$$

where ∂_x and ∂_y denote the gradient in x and y directions, respectively, and n is the number of valid pixels in the mask m. Because our patch is composed by black lines, which means they are more likely to appear in white areas. So we design the $\mathcal{L}_{white}$ to constrain the patch locate on white areas:

$$\mathcal{L}_{white} = \frac{\sum_1^n m \odot I}{n} \tag{10}$$

The $\mathcal{L}_{vulnerable}$ and $\mathcal{L}_{blank}$ are summed up in a linear manner to compose our final objective function:

$$\mathcal{L}_{total} = \mathcal{L}_{vulnerable} + \mathcal{L}_{blank} \tag{11}$$

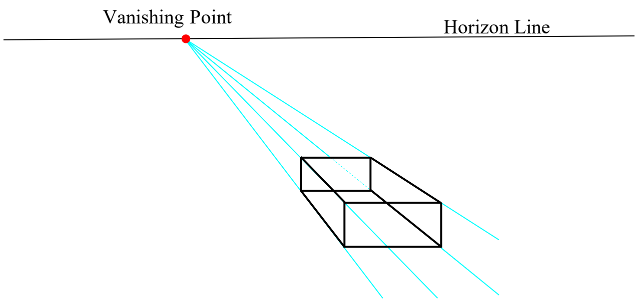


Figure 7. Diagram of single point perspective drawing: All the vanishing lines lead to the vanishing point. The horizontal and vertical lines, however, remain parallel to each other.

3.1.3 Location Optimization

As indicated in Eq.6, the target coordinates of the patch 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. By maximizing the $\mathcal{L}_{total}$, we can optimize the offsets $\mathcal{T}$ step by step.

$$\mathcal{T}_0 = [0,0]^\top, \mathcal{T}_{s+1} = Clip\{\mathcal{T}_s + \alpha \bigtriangledown_{\mathcal{T}} \mathcal{L}_{total}\}. \tag{12}$$

where $\mathcal{T}$ denotes the translation vector $[t_x, t_y]^\top$.

The whole process is proved to be differentiable:

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 224×224 and the size of the patch is 28×28, 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.

3.2. 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 7 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 implemention 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. Implemention Detail

To study the effect of our patch adversarial examples, we examine the robustness of a supervised method Fastdepth [10]. The examined MDE model is trained and evaluated on NYU depth v2 [7], 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 250×333 firstly, secondly, center cropped to 224×304, in the end they are resized to 224×224. As mentioned in 3.1.1, adversarial examples generated with simple line-drawn patches looks more natural. Therefore, we select a size of 28×28 bee drawing from QuickDraw [5] to generate our adversarial examples.

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

$$RMSE = \sqrt{\frac{1}{|N|} \sum\nolimits_{i \in N} ||D_i - D_i^*||^2} \tag{13}$$

$$MAE = \frac{1}{|N|} \sum\nolimits_{i \in N} |D_i - D_i^*| \tag{14}$$

$$Abs\ Rel = \frac{1}{|N|} \sum\nolimits_{i \in N} \frac{|D_i - D_i^*|}{D_i^*} \tag{15}$$

$$Threshold : \%\ of\ D_i\ s.t. \max(\frac{D_i}{D_i^*}, \frac{D_i^*}{D_i}) = \delta < thr \tag{16}$$

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.

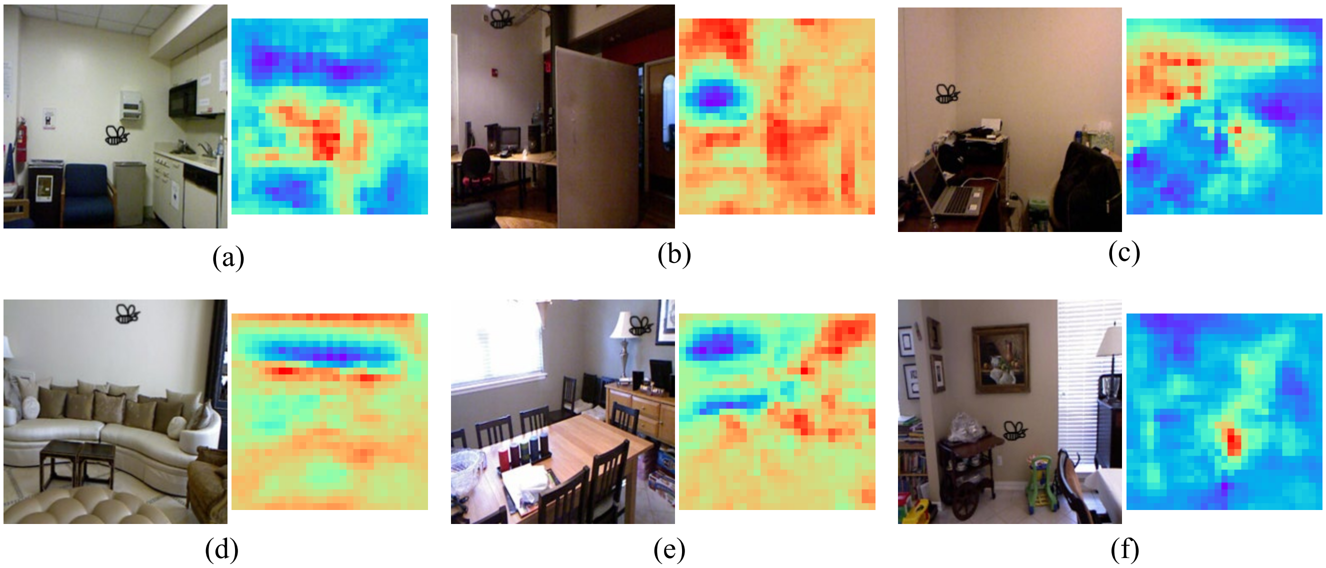


Figure 8. The final attack locations searched by our method (left) and corresponding MAE maps (right). Then calculate MAE maps, the offsets of the patch are clamped between -0.875 and 0.875, which makes the maps a little smaller than the RGB images.

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	**0.177/9.3%**	0.739	0.921	0.973	0.781
	Huber	**0.648/6%**	**0.466/9.1%**	0.176	**0.737**	**0.919**	**0.972**	0.772

4.2. Automatic Searching

Given an 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 8, 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.8 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 sencond 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 9. 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 our knowldege, 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 [11] 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.

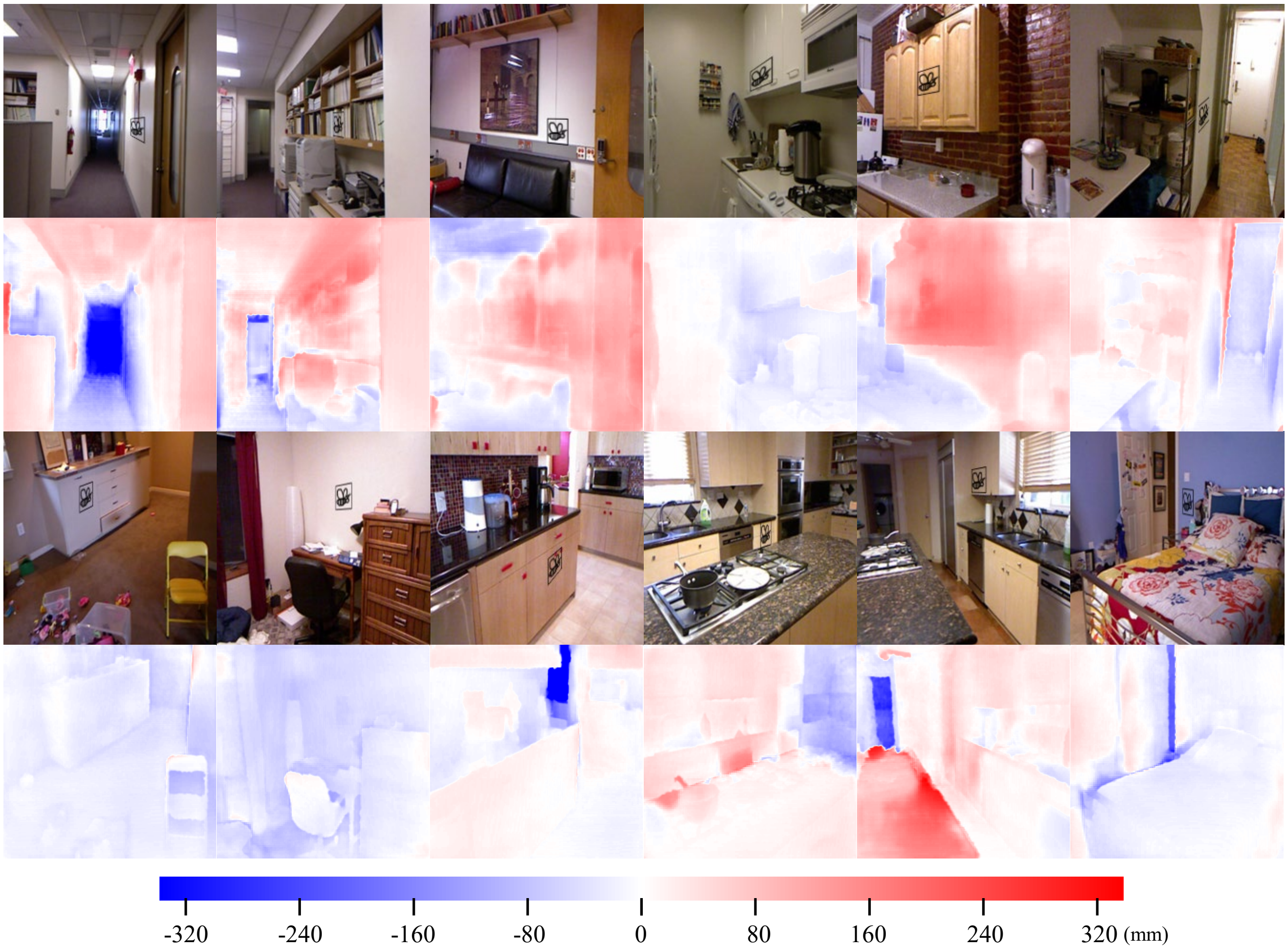


Figure 9. The visualization of our patch attack method. The first and third rows are I^{adv}. The second and fourth rows are the visualization of prediction error: $f_{mde}(I^{adv}) - D^*$. The color bar down the results explicates the error value intuitively. We emphasize the patches using surrounding black boxes, which are transformed in the altering appearance phase also.

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 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 Associates, Inc., 2014. 5

[3] 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 (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] 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

[7] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 5

[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*, CCS '16, page 1528–1540, New York, NY, USA, 2016. Association for Computing Machinery. 2

[9] 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

[10] 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. 2, 5

[11] 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