Attack MDE

xxx
School of Computer Science and Engineering, Sun Yat-sen University, China
{panm9}@mail2.sysu.edu.cn

Abstract

1. Introduction

Monocular Depth Estimation is a crucial 3D scene perception algorithm. Trapped by the lack of sufficient geometric constraints, MDE is still an ill-posed problem. With the rapid development of Deep Neural Networks(DNNs), MDE has emerged many excellent works in recent years. However, according to Szegedy's research [19], DNNs are prone to be attacked by adversarial examples: when a imperceptibly small perturbation is added to an input image, the classifier based on DNNs will classify the image into a wrong category with high probability. This property has been verified not only on classification tasks, but also on logical regression tasks, such as semantic segmentation and object recognition. MDE also failed to get rid of this dilemma.

As a kind of 3D scene perception algorithm, MDE needs to be robust and reliable, so that when an attack occurs in automatic driving, the vehicle can still accurately perceive the depth of the scene. Otherwise, the damage is much more serious than the percentage loss in accuracy. On MDE, this problem was first confirmed and researched by Wong [21]. 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. This research is pioneering and enlightening for attacking MDE.

However, the study of Wong [21] can hardly apply to physical world, it didn't take physical attack into account. The perturbations used in experiments are elaborately designed, which are very hard to realize in the physical world. While attacking a scene by patch is an appropriate solution: sticking a patch into the scene to be attacked is much more reasonable and implementable. Some researches exploit patches to attack DNNs [1], but most of them didn't pay much attention on inconspicuousness. Abrupt and unreal patch in scenes will reveal the attack intention. Indeed, some works noticed this problem and utilize GAN [9,12,16] to make patches more naturalistic. This will increase time and space complexity obviously. In this study, we take physical realizability and inconspicuousness into accout, design an algorithm to attack MDE.

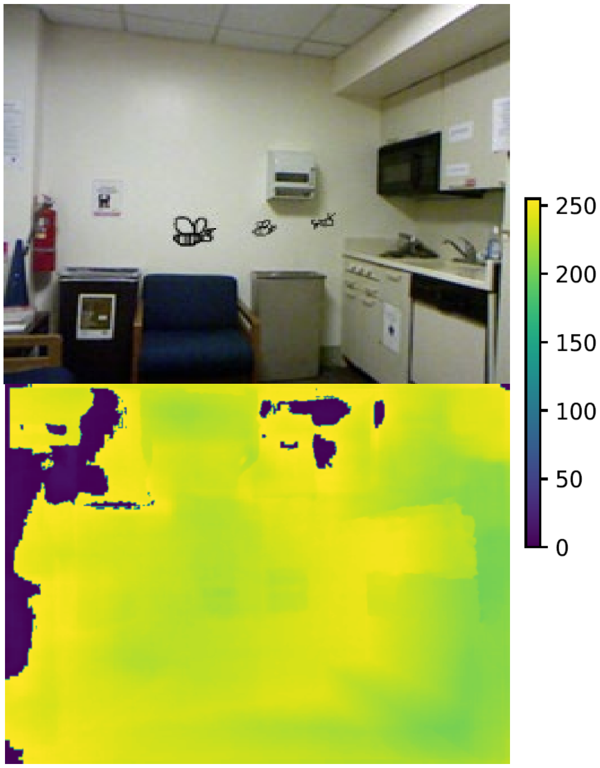

Figure 1. we have achieved a considerable attack success rate While taking into account the concealment and physical implementation.

Specifically, we design a learnable matrix, which can generate reasonable patches according to depth map automatically. The generation processes are driven by three learnable coefficients and don't need any extra generation networks.

contribution.

The structure in the rest of this paper is conducted as:

2. Related Work

In this work, patch attacks of MDE model are concentrated on, so a brief review of MDE and patch attacks will be introduced in this section.

2.1. Monocular Depth Estimation

2.1.1 Supervised

DNNs learn through the supervision of ground truth labels annotated artificially. Since the mapping from a single image to a depth map is very complicated, this supervised learning manner is very suitable for MDE tasks. The initial work adopts this **intuitive endoder-decoder** concept. Eigen et al. [4] designed two hierarchical neural networks to predict the depth map in coarse and fine respectively. A network was proposed subsequently by them [3], in which three different tasks, depth estimation, segmentation and normals prediction was integrated into one model. Lee [13] coined novel local planar guidance layers and locate them at different scale decoding phases. As a result of this, final fine depth maps are densely restored from small to big. Current MDE networks are bulky and inefficient, which can not be equipped on embedding systems. Wofk et al. [20] proposed Fastdepth, a light, fast MDE network. Some researches [2, 6, 15] address this problem by formulizing it as a **classification** problem. Discrete depth labels are prepared by separating continuous depth value, which are then used to supervise the training process. While using **relative depth information** to predict depth indirectly is also a effective solution. Lee [14] proposed a method to predict depth from relative depth. They first estimate relative depth by utilizing rank-1 property, then the final depth maps are reconstructed through these relative depth maps.

2.1.2 Unsupervised

Although there exist some datasets with ground truth depth annotation, dense and accurate depth maps are labor-consuming and rare. In that case, unsupervised methods dominate recently. Godard et al. [7] exploit the left-right consistency of binocular image pairs to tackle this issue: with a decently predicted disparity, the two images sampled from a binocular camera can synthesize each other. A consecutive vedio sequence can provide a constrain for MDE. Given a middle frame and its former, later frame sampled from a video, DNNs can estimate the camera pose and the depth, with which we can restore the other two frames. Based on this constrain, camera pose and depth can be learned jointly [22]. However, there is a flaw in both video and binocular solutions. On the one hand, binocular image pairs methods struggle in occlusion and texture-copy problems, yet, on the other hand, as an alternative method, predicting depth from a video performs unsatisfactorily when it comes to relatively stationary objects. To address this, Godard et al. [8] proposed a multi-scale reconstruction loss and a automasking approach to ignore relatively stationary objects.

2.2. Patch Attack

Adding perturbations to images has a limit: the imperceptive noise can not be captured by camera. Recently, researchers turn to generating adversarial examples with patches. It can be captured by camera and it can implement to physical world. The only question is, how to make it concealed.

Sharif et al. [18] attacked FRS using real printed glass frames, with which the FRS can not recognize the person or recognize him as someone else in the system. The pattern of these frames are generated by optimizing a composite objective function, including gradient decenting, smoothness of perturbations and non-printability score (NPS) optimal terms. Tom et al. [1] abandon the camouflage of the attack, they attack DNNs with an apparent, universal patch, which is updated with different locations and transformations settings. To generate robust adversaries, eykholt et al. [5] sampled from both actual physical and synthetic transformation images. Besides, they found that there exist vulnerable and robust areas in an image and identified these areas by computing perturbations using the L_1 regularization. Then they generate perturbations using L_2 regularization on vulnerable areas, finally reshape the perturbations to a graffiti shape. In doing so, they achieve a high successful rate under various environmental conditions. Liu et al. [16] also attacked the traffic signs. Instead of locating the vulnerable areas by computing, they locate the patch by a attention model, which takes as input the traffic sign, outputs an mask that indicates where to put the patch. The patch is generated through a GAN, which consists of a generator to generate a misclassification patch and a discriminator to discriminate whether the patch is original or adversarial. Kong et al. [12] also use a GAN to generate perturbation patches. In particular, they manipulate advertisement posters as original patch, which is artificially stuck on a street billboard after it through a GAN. This method produces better inconspicuousness yet the process of sticking the patch to billboard is labor consuming. In [9], the locations of patches are artificially determined and fixed to make a better camouflage. They first detect all humans in an image and pin a generated patch on their shirt. This attack makes the adversarial examples look as natural as a natural shirt pattern. Forward-mentioned attack methods need to access to the objects they attacked. The advantage is that adversarial examples are more inconspicuous, it is hard to deployment in the physical world though. To this end, Zolfi et al. [23] proposed a contactless attack method. Several translucent patches are generated by updating the affine transformation matrix, then

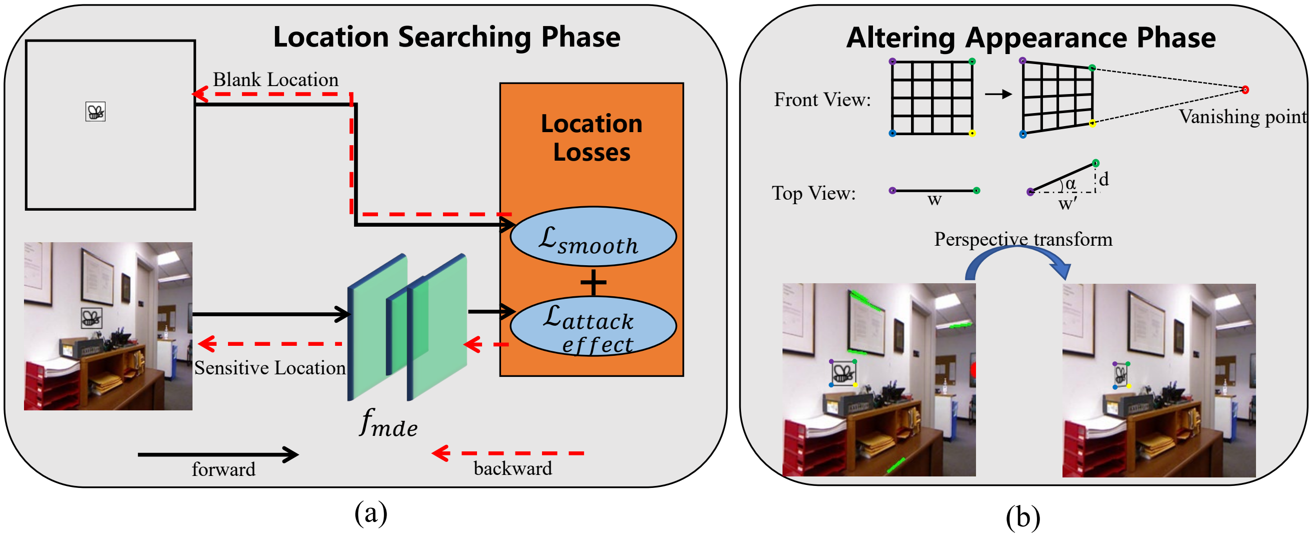

Figure 2. Stream of our MDE attacking method. The whole stream consists of two phases: (a) location searching phase and (b) altering appearance phase.

they are deployed on the camera's lens.

3. Method

In this section, we are going to present our MDE attacking method in detail. we first formulize the problem in Sec 3.1. Then we introduce how we design our patch to attack the MDE system in Sec 3.2. Finally we explain how the patches are optimized in Sec 3.3.

3.1. Problem Statement

Given a MDE data distribution: $\mathbb{R}^{H\times W\times 3} \to \mathbb{R}^{H\times W}$, a MDE model f_{mde} can learn a map that simulate the distribution within an error range. Considering elaborately designed perturbations can hardly be detected by consumer cameras, we are going to attack f_{mde} by sticking inconspicuous patternsδ on appropriate areas.

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

where$I$ is clean original image, $I^{'}$ is adversarial example, m is a mask that covers all non-zero pixels of the patch δ. Our goal, attacking a MDE system, is to maximize the error of ground truth depth map D^* and depth map generated from $I^{'}$. $\|\cdot\|_p$ denotes the l_p norm.

$$\max \|f_{mde}(I^{'}) - D^*\|_p \tag{2}$$

3.2. Patch Design

location To obtain an adversarial example $I^{'}$, the first problem we encountered is where to stick the patches. Exhaustive search is feasible but time consuming. We expect to complete this this process automatically and quickly. So we turn to optimizing the patch location by gradient decenting. We define a 2×3 matrix $\mathbf{A}_\theta$ for each patch to implement this location search process. By optimizing this $\mathbf{A}_\theta$, a patch can update its location automatically.

$$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix} \tag{3}$$

where $[O_x, O_y]^\top$ denote the offsets in horizontal and vertical directions, respectively. Initial values are $[0, 0]$, which represent that the center of the patch is on the center of the image. To avoid overflow from RGB images, $[t_x, t_y]^\top$ are clamped between $(-0.875, 0.875)$.

Specifically, we first place the patch δ in the center of the clean image I then pad 0 around it to the size of I. The homogeneous coordinate of this patch is noted as G and its size is $H \times W \times 3$. For each pixel of padded δ, $G_i = (x_i, y_i, 1)$. This location searching process can be represented as:

$$G'^T = A_\theta\, G^T \tag{4}$$

$$\begin{pmatrix} x_s \\ y_s \end{pmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix} \begin{pmatrix} x_t \\ y_t \\ 1 \end{pmatrix} \tag{5}$$

Appearance The location problem is settled by optimizing an affine transform matrix $\mathbf{A}_\theta$, next we address altering the appearance of the patch. The appearance altering phase can be seen in Fig 2(b).

When we alter the appearance of the patch, for better visualization quality, we followed the principle of perspective drawing: the vanishing point is where all parallel lines

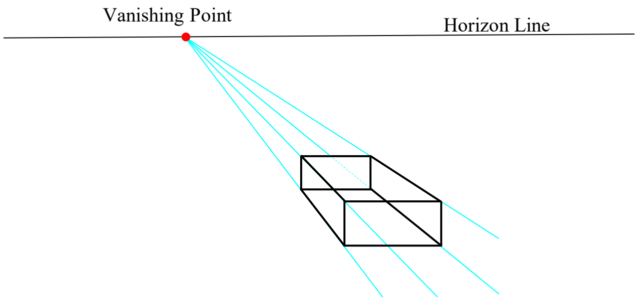

Figure 3. Diagram of single point perspective drawing: All the vanishing lines lead to the vanishing point. The horizontal and vertical lines, however, remain parallel to each other.

intersect and is always on the horizon line. Fig 3 clearly illustrate this principle.

We first detect straight lines in the attacked scene with *Hough* transform, then we obtain the vanishing point by extending this lines and computing their intersection. Based on the principle of perspective drawing, as illustrated in Fig 2(b), if we connect the left-top point and vanishing point, the right-top point should lay on this line. Same on the bottom line. Meanwhile, the connection lines, which converge to vanishing points, will look shorter than their original length. We explain this fact in Fig 2(b): when seen from the front view, the width of patches w will shrink to w'.

$$w' = \sqrt{w^2 - d^2} \tag{6}$$

Since this task we concentrate on is just MDE, we possess the depth of each pixel. The patches' depth d can be got by subtracting fronter boarder depth from back boarder depth. We set $w = 0.1m$ in our experiments. We get four corrected corner coordinates, with which we alter the appearance of the patch. The perspective transforms in this phase are implemented with *OpenCV*.

3.3. Objective Function

The aim of this research is to attack MDE models inconspicuously with patches, we consider three factors: attack effect, which area of the adversarial example is more inconspicuous and what appearance of the patches are more inconspicuous. The last consideration about the appearance is addressed by perspective geometry, while the attack effect and inconspicuous areas are optimized by two loss funtions respectively. The two loss functions consisit our objective function in a linear manner:

$$\mathcal{L}_{total} = \lambda_1 \mathcal{L}_{attack} + \lambda_2 \mathcal{L}_{blank} \tag{7}$$

We are going to explain the two loss functions in the following part of this section.

3.3.1 attack effect

MDE f_{mde} systems generate depth maps, whose pixel values represent the distance from area to sample sensors. To obtain best attack effect, that is, to maximize the error between prediction maps and annotated maps:

$$\mathcal{L}_{attack} = \|f_{mde}(I^{'}) - D^*\|_p \tag{8}$$

We use L_1, L_2 and Huber loss in our experiments.

3.3.2 Visualization Quality

For better inconspicuousness, we camouflage the patch as graffiti, which can be usually found on blank walls. Blank means with no pictures, marks or decoration on white walls. Based on this we design Blank loss to help the patch to find blank areas, the Blank loss consisits of two optimazation sub-terms:

$$\mathcal{L}_{blank} = \mathcal{L}_{smooth} + \mathcal{L}_{white} \tag{9}$$

$\mathcal{L}_{smooth}$ constrain the patch locate on areas with low texture complexity:

$$\mathcal{L}_{smooth} = -\frac{\sqrt{\partial_x(M \odot I) + \partial_y(M \odot I)}}{N} \tag{10}$$

where M denotes a full 1 mask that has the same size as δ. ∂_x and ∂_y denote the gradient in x and y directions respectively. N is the number of pixels in M. $\mathcal{L}_{white}$ constrain the patch locate on white areas:

$$\mathcal{L}_{white} = \frac{\sum_1^N M \odot I}{N} \tag{11}$$

3.4. location Optimization

As indicated in Eq.5, the coordinates of the target pixels G_i' are continuous values. We adopt bilinear interpolation to determine the value of the target pixel. Bilinear interpolation mechanism is proved to be differentiable in *spatial transformer networks* [11]. By maximize the $\mathcal{L}_{attack}$, we can optimize the offsets step by step.

$$I_0^{adv} = I, X_{N+1}^{adv} = Clip_{I,\epsilon}\{I_N^{adv} + \alpha \bigtriangledown_I J(\theta, x, y)\}. \tag{12}$$

The stream of location searching phase can be seen in Fig 2(a).

4. Experiments

In this section, we conduct several experiments to exhibit the results of our automatic and inconspicuous attack method on the fastdepth [20] MDE system. We introduce

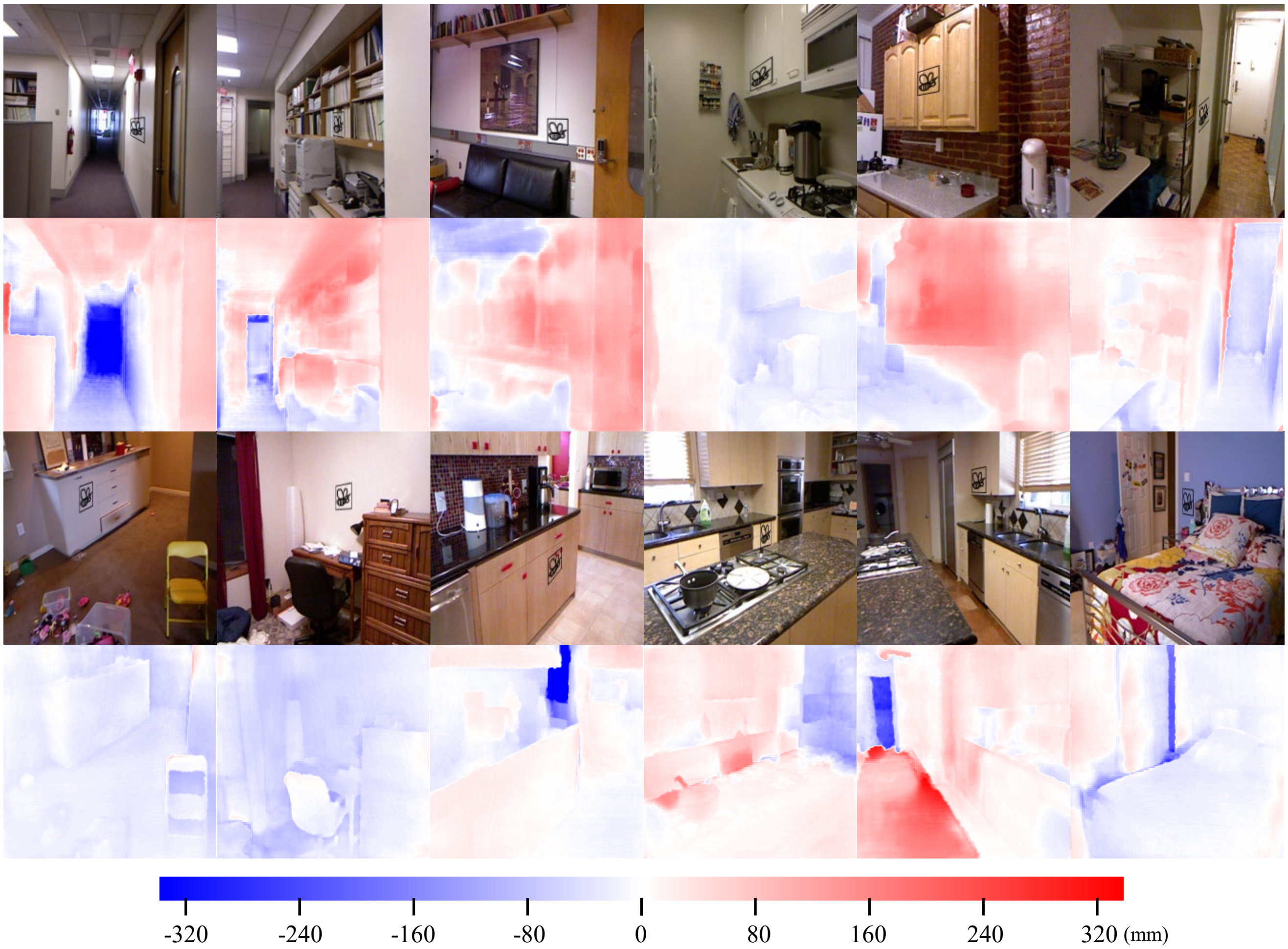

Figure 4. The visualization of our patch attack method. The first and third rows are I^{adv}. The second and fourth rows are the visualization of prediction error: $f_{mde}(I^{adv}) - D^{*}$. We emphasize the patches using surrounding black boxes, which are transformed in the altering appearance phase also.

our implemention detail, including datasets and hyper parameters in Sec 4.1. In Sec 4.2 and Sec 4.3, our attack effects are shown by visualization and quantitative comparison. Then we demonstrate the automatic searching process driven by our method in Sec 4.4. Finally we show and analyze the visual quality of our adversarial examples.

4.1. Implemention Detail

For the MDE system, we use Fastdepth [20]. We also use the same dataset NYU Depth v2 [17] used in the Fastdepth [20]. There are 1449 densely annotated depth and RGB images pairs, 795 of which are split to train dataset and the rest are split to test dataset. For the image preprocessing, we followed the process in Fastdepth: images are resize to 250 × 333 firstly, secondly, center cropped to 224 × 304, in the end they are resize to 224 × 224.

For the patch dataset, we choose QuickDraw [10]. QuickDraw is a collection of 50 million hand-writing drawings across 345 categories. We select several categories, including bee, ant, dolphin, crocodile and so on. The size of them are all 28 × 28.

4.2. Visualization Attack Results

As mentioned above

4.3. Quantitative Attack Results

4.4. Automatic Searching

In this section you are going to find that given an image our attack method can search the sensitive zone and stick the patch on the zone. To better demonstrate our search effect, we calculate the MAE maps of test images. Beginning from the left top corner, where is (-0.875,-0.875), we use strides 7 to move the patch step by step and calculate the MAE of each position, finally getting the MAE maps. The red and violet colors in the maps represent higher and lower MAE values, respectively. As you can see from Fig 5, our method can accurately search the sensitive area of the image. Note that to emphasize our search effect, we didn't use the Blank (eq 9) loss function in this experiment.

We visualize the serach path in Fig 6.

We use different stickers to calculate their MAE maps on the same image, to investigate the impact of different patches on the sensitive area's location in the image. As shown in Fig 7, the distribution of sensitive and robust areas in (c) (f) is similar, the sensitive areas are concentrated in the center, among them (d) and (f) almost looked the same. However there are some differences between (h),(g) and (c) (f). Intuitively, this is caused by the variations of patch⑤ and ⑥ in saptial structure. Based on this, we conclude that, to similar patches, the distribution of an image's sensitive and robust area is usually the same. The similarity is more significant along with the smaller size.

4.5. Visual Quality

5. Conclusion

References

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

[2] Yuanzhouhan Cao, Zifeng Wu, and Chunhua Shen. Estimating depth from monocular images as classification using deep fully convolutional residual networks. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(11):3174–3182, 2017. 2

[3] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015. 2

[4] 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. 2

[5] 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

[6] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep Ordinal Regression Network for Monocular Depth Estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[7] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 270–279, 2017. 2

[8] Clement Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 2

[9] 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. 1, 2

[10] T. K. J. K. J. Jongejan, H. Rowley and Fox-Gieg. The quick, draw!, 2016. https://github.com/googlecreativelab/quickdraw-dataset. 5

[11] 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. 4

[12] 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. 1, 2

[13] Jin Han Lee, Myung-Kyu Han, Dong Wook Ko, and Il Hong Suh. From big to small: Multi-scale local planar guidance for monocular depth estimation. *arXiv preprint arXiv:1907.10326*, 2019. 2

[14] Jae-Han Lee and Chang-Su Kim. Monocular depth estimation using relative depth maps. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2019. 2

[15] Ruibo Li, Ke Xian, Chunhua Shen, Zhiguo Cao, Hao Lu, and Lingxiao Hang. Deep attention-based classification network for robust depth prediction. In C.V. Jawahar, Hongdong Li, Greg Mori, and Konrad Schindler, editors, *Computer Vision – ACCV 2018*, pages 663–678, Cham, 2019. Springer International Publishing. 2

[16] Aishan Liu, Xianglong Liu, Jiaxin Fan, Yuqing Ma, Anlan Zhang, Huiyuan Xie, and Dacheng Tao. Perceptual-sensitive gan for generating adversarial patches. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19. AAAI Press, 2019. 1, 2

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

[18] 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

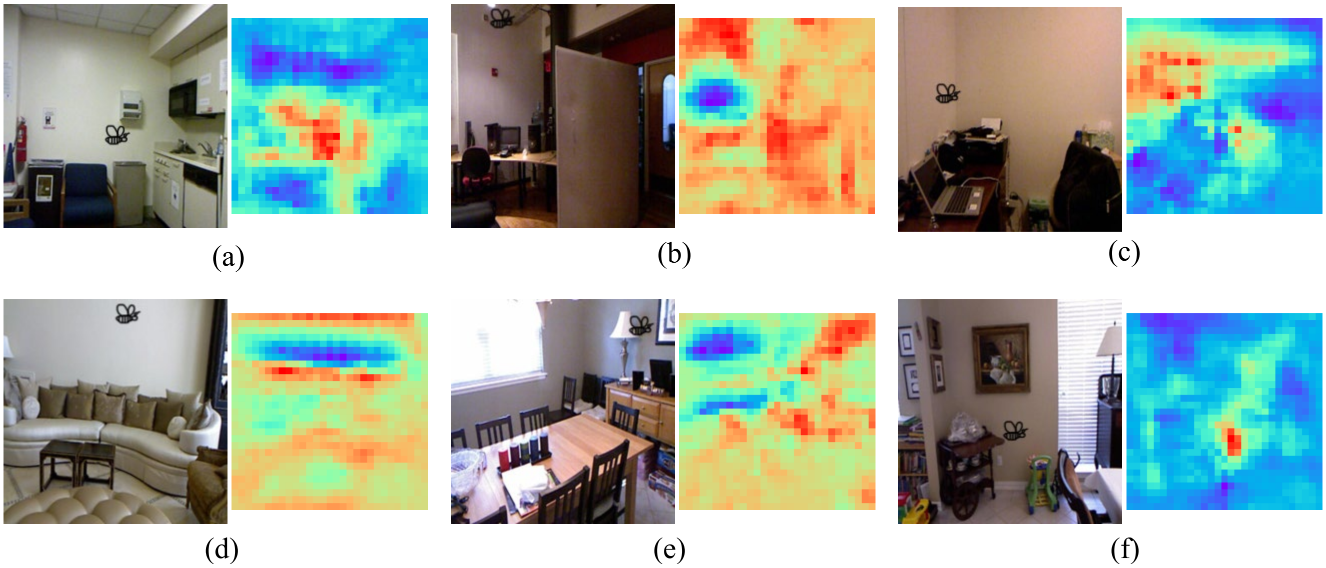

Figure 5. The final attack locations searched by our method (left) and corresponding MAE maps (right). The offsets of the maps are clamped between -0.875 and 0.875, as a contrast, the width and height of the RGB images extend from -1 to 1, which makes the maps a little smaller than the RGB images.

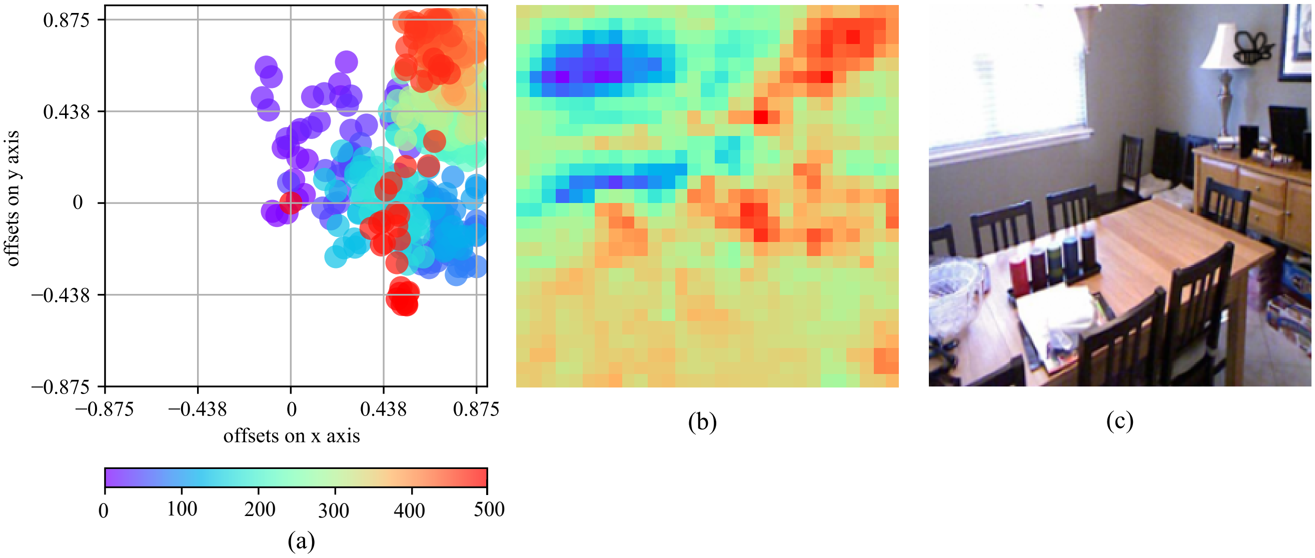

Figure 6. The search path of 5 (e), the color from violet to red represent the search steps from 0 to 500.

[19] 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

[20] 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, 4, 5

[21] 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

[22] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6612–6619, 2017. 2

[23] Alon Zolfi, Moshe Kravchik, Yuval Elovici, and Asaf Shab-

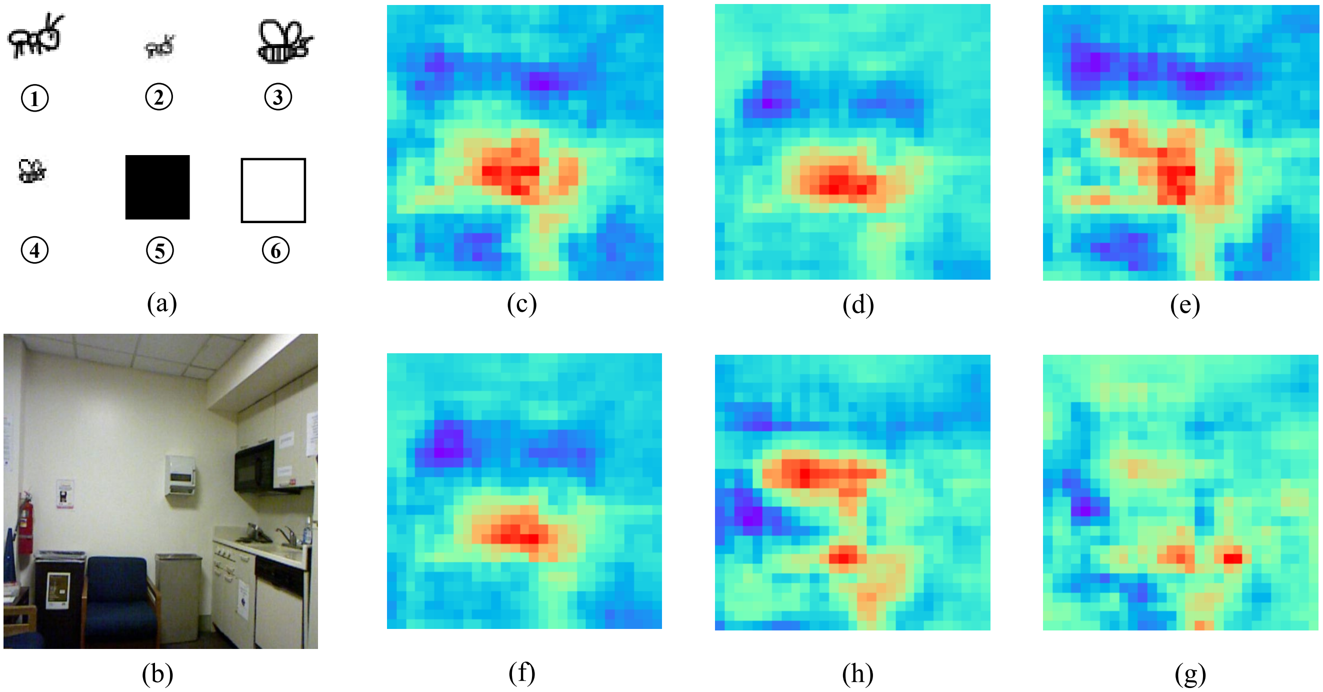

Figure 7. There are 6 different patches in (a), they are stuck in (b) to calculate MAE maps (c g). The placement of (c g) is the same as the patches' placement in (a). Patch ② and ④ are down sampled from ① and ③ with factor 0.5. Patch ⑤ is a black patch filled with 0 and patch ⑥ is a white patch filled with 1. The black box is for visualization this white patch.

tai. The translucent patch: A physical and universal attack on object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15232–15241, June 2021. 2