

Self-supervised Adversarial Training of Monocular Depth Estimation against Physical-World Attacks

Zhiyuan Cheng, Cheng Han, James Liang, Qifan Wang, Xiangyu Zhang, Dongfang Liu

Abstract—Monocular Depth Estimation (MDE) plays a vital role in applications such as autonomous driving. However, various attacks target MDE models, with physical attacks posing significant threats to system security. Traditional adversarial training methods, which require ground-truth labels, are not directly applicable to MDE models that lack ground-truth depth. Some self-supervised model hardening techniques (*e.g.*, contrastive learning) overlook the domain knowledge of MDE, resulting in suboptimal performance. In this work, we introduce a novel self-supervised adversarial training approach for MDE models, leveraging view synthesis without the need for ground-truth depth. We enhance adversarial robustness against real-world attacks by incorporating L_0 -norm-bounded perturbation during training. We evaluate our method against supervised learning-based and contrastive learning-based approaches specifically designed for MDE. Our experiments with two representative MDE networks demonstrate improved robustness against various adversarial attacks, with minimal impact on benign performance. Our code: <https://github.com/Bob-cheng/DepthModelHardening>.

Index Terms—Self-supervised Learning, Adversarial Training, Monocular Depth Estimation, and Adversarial Robustness.

1 INTRODUCTION

MONOCULAR Depth Estimation (MDE) is a deep neural network (DNN)-based task that estimates depth from a single image, allowing for 2D-to-3D projection by predicting the distance for each pixel in a 2D image [2]. This makes it a cost-effective alternative to pricey Lidar sensors. Applications of MDE are vast, including autonomous driving [3], visual SLAM [4], and visual relocalization [5], *etc.* Specifically, self-supervised MDE has become increasingly popular in the industry (*e.g.*, Tesla Autopilot [3]) due to its ability to achieve comparable accuracy without requiring ground-truth depth data from Lidar during training. However, due to the known vulnerabilities in DNNs [6], [7], several digital-world [8]–[10] and physical-world adversarial attacks [11] has been developed against MDE. These attacks primarily use optimization-based methods to create adversarial input that deceive the MDE models. Given the significance and widespread use of MDE models, these adversarial attacks pose a substantial risk to the security of applications such as autonomous driving. As a result, there is an urgent need to develop and strengthen MDE models against these threats.

Adversarial training [6] is a widely recognized and effective defense against adversarial attacks, which hardens tar-

get models by exposing them to adversarial examples during the training process. However, it typically necessitates ground-truth labels during training, making it unsuitable for hardening MDE models that lack depth ground truth. Although contrastive learning has recently garnered significant attention and has been employed for self-supervised adversarial training [12], [13], it does not take into account the domain knowledge of depth estimation, resulting in suboptimal outcomes (as demonstrated in §4.2). Furthermore, many existing adversarial training methods do not account for certain characteristics of real-world attacks [11], [14], such as adversarial patches with intense perturbations and perspective variations of target objects. Therefore, in this paper, we concentrate on the challenge of *strengthening MDE models against physical-world attacks using self-supervised adversarial training without the need for ground-truth depth*.

A straightforward proposal for enhancing MDE models is to perturb 3D objects across various scenes while ensuring that the estimated depths remain accurate. However, implementing such adversarial training is challenging. Firstly, 3D perturbations are difficult to achieve in the physical world. Even if a model is trained in simulation, a high-fidelity simulator and a powerful scene rendering engine are required to accurately project 3D perturbations into 2D variations. Secondly, since self-supervised MDE training lacks ground-truth depth, even if realistic 3D perturbations could be obtained and utilized in training, the model might converge on incorrect (but robust) depth estimations.

In this paper, we introduce a novel self-supervised adversarial training approach for MDE models. Figure 1a offers a conceptual depiction of our method. A board A displaying the 2D image of a 3D object (*e.g.*, a car) is positioned at a fixed location (next to the car at the top-right corner). Two cameras (close to each other at the bottom) C_t and C_s provide two adjacent views (*i.e.* stereo view [15])

- Z. Cheng and X. Zhang are with Purdue University (Email: cheng443@purdue.edu, xyzhang@cs.purdue.edu)
- C. Han is with University of Missouri – Kansas City and Rochester Institute of Technology (Email: chk9k@umsystem.edu; ch7858@rit.edu)
- D. Liu is with Rochester Institute of Technology (Email: dongfang.liu@rit.edu)
- J. Liang is with Rochester Institute of Technology and U.S. Naval Research Laboratory (Email: jcl3689@rit.edu)
- Q. Wang is with Meta AI (Email: wqfcr@fb.com)
- A preliminary version of this work has been published on ICLR 2023 as Spotlight [1].
- Corresponding author: Dongfang Liu and Xiangyu Zhang

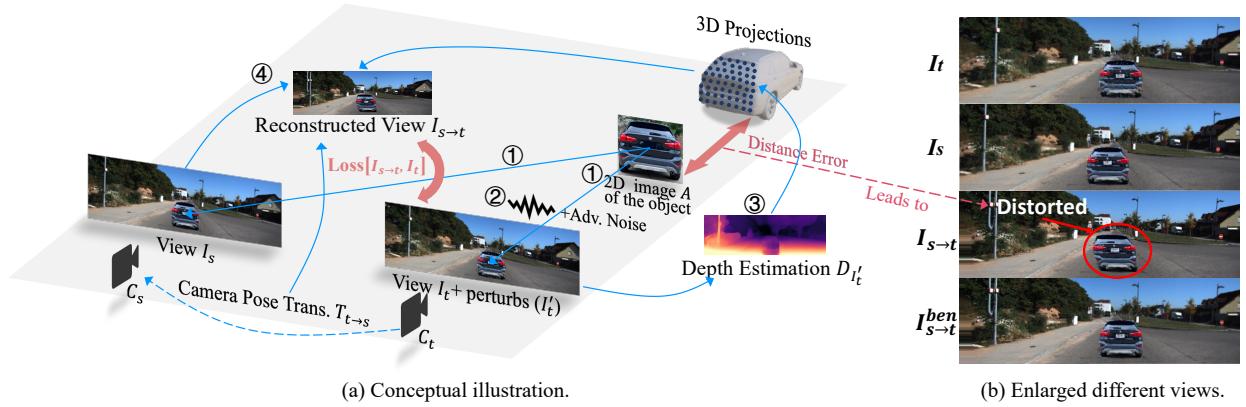


Fig. 1: Self-supervised adversarial training of MDE with view synthesis.

of the board (images I_t and I_s in Figure 1b). There are fixed geometric relationships between pixels in the two 2D views captured by each camera, allowing the image from one view to be transformed into the image from the other view. Intuitively, the target view image I_t can be obtained by shifting the source view image I_s to the right. If two cameras are unavailable, two temporally adjacent frames in a video stream can be used to form the two views.

During training, four main steps are taken. ①: Camera C_t captures an image I_t of the original 2D image board A , and camera C_s takes a picture I_s of the board as well. ②: Adversarial noise is applied to A , aiming to deceive the target MDE model into overestimating its depth, resulting in a perturbed view I'_t . ③: The perturbed image $I_t + \text{perturbs}$ (I'_t) is input into the MDE model for depth estimation, then we project the estimated distance $D_{I'_t}$ to 3D space. The blue dots on the car denote the projected pixels of the target object. Due to the perturbations, a susceptible model generates distance errors, as indicated by the red arrow between A and the projected pixels in Figure 1a. ④: Given the relative pose transformations $T_{t \rightarrow s}$ between the two cameras, we map the projected pixels in 3D space back onto camera C_s 's view to locate the corresponding pixels in the adjacent image I_s , then we attempt to reconstruct target image I_t from source image I_s by rearranging the pixels.

Owing to the distance error, the reconstructed image $I_{s \rightarrow t}$ (shown in Figure 1b) differs from I_t , causing part of the car (the upper part inside the red circle) to be distorted. In contrast, Figure 1b also displays the reconstructed image $I_{s \rightarrow t}^{ben}$ without the perturbation, which closely resembles I_t . The objective of our training (for the target MDE model) is to minimize the differences between original and reconstructed images. The above process is conceptual and would require significant physical-world overhead to be realized faithfully. In §3, we describe how to circumvent the majority of the physical-world costs through image synthesis and training on synthesized data.

Traditional adversarial training often assumes bounded perturbations in L_2 or L_∞ norm (i.e., measuring the overall perturbation magnitude across all pixels), while real-world attacks are typically unbounded in these norms. These attacks tend to be stronger in order to persist amid varying environmental conditions. To strengthen MDE models against such attacks, we employ a loss function that can efficiently approximate the L_0 norm (measuring the number

of perturbed pixels regardless of their perturbation magnitude) while remaining differentiable. Adversarial samples produced by minimizing this loss can effectively simulate physical attacks. Also, the “board” (A in Figure 1a) serves as a pragmatic approximation of a 3D model of the target object. This simplification, using a 2D board, is both realistic and practical, since adversarial patches are commonly affixed to flat surfaces in real-world attacks. We manipulate the board with random distances and viewing angles during training to simulate perspective shifts encountered in real-world patch attacks, resulting in a more robust defense. Moreover, this approach significantly curtails the training overhead in view synthesis and boosts efficiency.

We evaluate our approach and compare it with a supervised learning baseline and a contrastive learning baseline, adapted from state-of-the-art adversarial contrastive learning [13]. Our results demonstrate that our method achieves superior robustness against various adversarial attacks, with minimal degradation in model performance. The average depth estimation error for an adversarial object with 1/10 area of perturbation is reduced from 6.08 m to 0.25 m using our technique, outperforming the 1.18 m error from the supervised learning baseline. Furthermore, the contrastive learning baseline significantly degrades model performance. A video of our real-world experiments is available at https://youtu.be/_b7E4yUFB-g. Our main contributions are as follows:

- We develop a novel technique for synthesizing 2D images that adhere to real-world constraints (e.g., relative camera positions), and directly apply perturbations to these images during adversarial training. This approach significantly reduces the physical world costs involved.
- Our approach leverages the *reconstruction consistency* between two different views to facilitate self-supervised adversarial training without ground-truth depth labels.
- We generate L_0 -bounded perturbations using a differentiable loss function and randomize camera and object settings during the synthesis process to simulate real-world attacks and enhance the robustness of the model.

This work significantly extends our conference paper [1] in various aspects:

- (1) We broaden our approach to include a general training framework that can be applied to any monocular depth estimation method against physical-world attacks (§3) and provide more adversarial attack examples and qualitative

defense results to evaluate the performance (§4.2.6).

(2) To ensure the effectiveness of view synthesis, we enhance the view synthesis process by considering inconsistent environmental lighting between the scene and object and perform additional ablative experiments on the impact of varying viewing angles during training, demonstrating the strength of our view synthesis algorithm (§4.3.2).

(3) We conduct a human evaluation to assess the quality and reality of our synthesized images (§4.2.2), and investigate the effects of inaccurate projections and unrealistic object positions (§4.3.8).

(4) We present additional transferability evaluation (§4.3.3) for our hardened models applied to other target objects and validate models' robustness against a wider range of adversarial attacks including black-box ones (§4.3.4).

(5) We compare the differences between fine-tuning and training from scratch (§4.3.5), explore the model performance when combining various methods (§4.3.6), and contrast the supervised baseline trained with pseudo-depth labels and ground-truth depth labels (§4.3.7).

(6) We extend our method to indoor scenes and more advanced MDE networks such as Manydepth (§4.4).

(7) We present an in-depth analysis of our method's limitations and potential in §5, which we hope will inspire further research collaboration.

2 RELATED WORK

This section provides an overview of the most pertinent research on Self-supervised MDE (§2.1), MDE Attack and Defense (§2.2), and Adversarial Robustness (§2.3).

2.1 Monocular Depth Estimation

Due to the benefits of training without depth ground truth, MDE has recently attracted significant interest in a self-supervised manner. In this kind of training approach, monocular videos and/or stereo image pairs serve as input. Essentially, two images captured by a camera or cameras from neighboring positions are used in each optimization iteration. A depth network and a pose network are employed to estimate the depth map of one image and the transformation between the two camera poses, respectively. Using the depth map and pose transformation, pixel correspondence across the images is calculated, and then an attempt is made to rearrange the pixels in one image to reconstruct the other. Both the pose network and depth network are updated simultaneously to minimize reconstruction error. Please refer to [16] for a detail explanation of this process.

Garg *et al.* [17] first proposed using color consistency loss between stereo image pairs in training. Zhou *et al.* [16] enabled video-based training with two networks (one depth network and one pose network). Many subsequent works improved self-supervision with new loss terms [18]–[24] or incorporated temporal information [25]–[28]. Among them, Monodepth2 [29] significantly enhanced performance with several innovative designs, such as minimum photometric loss selection, masking out static pixels, and multi-scale depth estimation. Depthhints [30] further improved upon Monodepth2 by incorporating additional depth suggestions obtained from stereo algorithms. While unsupervised training has proven effective, enhancing its robustness against physical attacks remains an open challenge.

2.2 MDE Attack and Defense

Mathew *et al.* [31] employed a deep feature annihilation loss to launch perturbation and patch attacks. Zhang *et al.* [8] designed a universal attack using a multi-task strategy, while Wong *et al.* [9] generated targeted adversarial perturbations on images, which can arbitrarily alter the depth map. Hu *et al.* [32] proposed a defense method against perturbation attacks by masking out non-salient pixels, requiring an additional saliency prediction network. Regarding physical-world attacks, Cheng *et al.* [11] created a printable adversarial patch to make a vehicle disappear. To the best of our knowledge, our work is the first to focus on enhancing the robustness of MDE models against physical-world attacks.

2.3 Adversarial Robustness

Deep neural networks (DNNs) are known to be vulnerable to adversarial attacks, which are usually optimized perturbations on the model input that could lead to model misbehavior [33]–[35]. Adversarial training is a popular defensive approach that can harden DNN models against adversarial attacks [34], [36], [37]. The core concept involves integrating adversarial examples into the training process by augmenting the training data, thereby pre-exposing the model to adversarial noise and improving the run-time robustness. Typically, during adversarial training, the process alternates between conducting model training to minimize the training loss and generating adversarial examples that can maximize the loss. This approach has been applied to various domains, including image classification [34], [36], object detection [38]–[40], and segmentation [41]–[43], etc. However, adversarial training often requires supervision, as generating adversarial examples necessitates ground truth, and most tasks demand labels for training. Some semi-supervised adversarial learning methods [44], [45] use a small portion of labeled data to enhance robustness. Contrastive learning [12], [13] is also employed with adversarial examples for self-supervised learning. In this work, we investigate self-supervised adversarial training for MDE without ground-truth depth and compare our method with contrastive learning-based and supervised learning-based methods for MDE. In the realm of autonomous driving, there are other works focusing on attacking Lidar or multi-sensor fusion-based systems [46]–[49]. These approaches use sensor spoofing or adversarial shapes to deceive Lidar hardware or AI models. In contrast, our work considers fully vision-based systems in which MDE is the key component.

3 METHODOLOGY

Our approach is composed of several components. The first component (§3.1) deals with self-supervised adversarial training pipeline and the view reconstruction process (steps ④ in Figure 1a). The second component (§3.2) involves view synthesis, which generates two views I_t and I_s of the object, corresponding to step ① in Figure 1a. The third component (§3.3) focuses on robust adversarial perturbation, altering I_t to produce maximum distance errors (step ② in Figure 1a). We discuss the specifics of these components as follows.

3.1 Self-supervised MDE Training

Figure 2 depicts an overview of our self-supervised adversarial training pipeline. From left to right, the pipeline starts

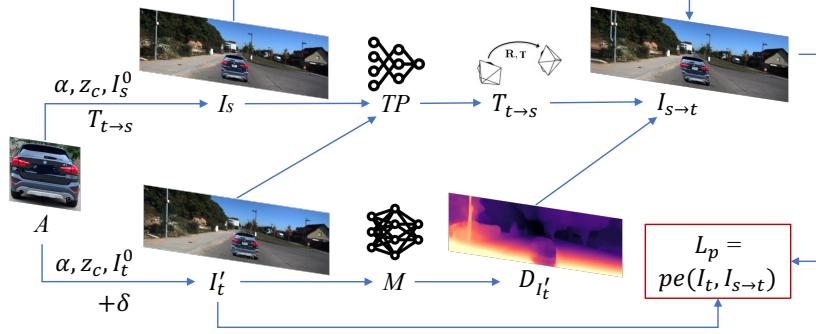


Fig. 2: The pipeline of self-supervised adversarial training of monocular depth estimation.

with the object image A . First, we synthesize it onto two adjacent street view backgrounds, I_s^0 and I_t^0 , sampled from the dataset to create the two views of the object, I_s and I_t . During synthesis, we use various distances z_c and viewing angles α . The synthesis process circumvents physical-world costs significantly, and we explain the details in §3.2. Next, robust adversarial perturbations δ are generated onto A while synthesizing I_t , forming the adversarial sample I_t' , which is then fed into the depth network M to obtain depth $D_{I_t'}$. This process is detailed in §3.3. Recall that we need the camera pose transformation matrix $T_{t \rightarrow s}$ between the coordinate systems of the two cameras. We predict it by a camera transposing model TP with the two street views I_s^0 and I_t^0 as input. This TP network is adapted from Monodepth2 [29]. Its architecture is formed from a ResNet18, modified to accept a pair of color images (or six channels) as input, and to predict a single 6-DoF (Degrees of Freedom) relative pose. For a comprehensive understanding of the model, we refer readers to the original publication. Next, we reconstruct a version of the target view I_t from the source view I_s , using $D_{I_t'}$ and the output $T_{t \rightarrow s}$ of the TP network. We refer to the resultant image as $I_{s \rightarrow t}$. Intuitively, I_t' induces depth errors which distort the reconstruction from I_s to $I_{s \rightarrow t}$, making the latter appear different from I_t . At last, we use a loss L_p of the reconstruction error to train the two models M and TP .

Specifically, the reconstruction of the target I_t is achieved by utilizing the pixel-to-pixel correspondence between I_s and I_t . The function for projecting a pixel (u^t, v^t) in image I_t to a pixel (u^s, v^s) image I_s is defined as follows:

$$\begin{aligned} [x^t \ y^t \ z^t \ 1]^\top &= D_{I_t'}[u^t, v^t] \cdot K^{-1} \cdot [u^t \ v^t \ 1]^\top, \\ [x^s \ y^s \ z^s \ 1]^\top &= T_{t \rightarrow s} \cdot [x^t \ y^t \ z^t \ 1]^\top, \\ [u^s \ v^s \ 1]^\top &= 1/z^s \cdot K \cdot [x^s \ y^s \ z^s \ 1]^\top. \end{aligned} \quad (1)$$

Conceptually, there exist relationships between 2D image pixels and 3D coordinates, as illustrated by the first and third formulas in Equation 1. The 3D coordinates also have correlations determined by camera poses, as demonstrated by the second formula. Basics of the camera projections can be found at [50]. It is important to note that the first 2D-to-3D relationship, corresponding to step ③ in Figure 1a, is parameterized on $D_{I_t'}$, which is the depth estimation of I_t' . Let $[u^s \ v^s] = P_{D_{I_t'}, T_{t \rightarrow s}}(u^t, v^t)$ be the transformation function that maps a pixel in I_t to a pixel in I_s , derived from Equation 1. $I_{s \rightarrow t}$ is reconstructed as:

$$I_{s \rightarrow t}[u, v] = I_s[P_{D_{I_t'}, T_{t \rightarrow s}}(u, v)]. \quad (2)$$

Intuitively, this method reorganizes the pixels in I_s to create $I_{s \rightarrow t}$. $I_{s \rightarrow t}$ is then compared with I_t , and the core training objective for our approach is defined as follows:

$$\min_{\theta_M, \theta_{TP}} \mathcal{L}_p = pe(I_t, I_{s \rightarrow t}), \quad (3)$$

which is to adjust the weight values of M and TP to minimize the photometric reconstruction error, represented by $pe()$. The specific design of $pe()$ varies in the literature, and we use the same one as Monodepth2 [29] in our experiments. Nevertheless, alternative formulations of $pe()$ could conceivably be integrated within the framework of our model hardening approach.

3.2 View Synthesis To Avoid Physical Scene Mutation

As discussed in §1, conceptually, we need to place an image board of the 3D object at various physical locations on streets and use two cameras to capture images of the board. To achieve robustness during training, we must vary the object's image (*e.g.*, different vehicles), the position of the image board, the camera positions and viewing angles, and the street view. This would involve significant overhead in the physical world. Consequently, we propose an innovative view synthesis technique that minimizes physical world overhead. Instead, it directly synthesizes I_s and I_t by incorporating an object into some street views, considering different settings of the aforementioned configurations.

In particular, we consider a camera C_t and a board attached with an object image A in the physical world. The board's physical width and height are W and H , respectively. As illustrated in Figure 3 (b), the four corners of the board have the 3D coordinates (x_0^t, y_0^t, z_0^t) , ..., and (x_3^t, y_3^t, z_3^t) in camera C_t 's coordinate system with the camera as the origin. The board is placed at a distance z_c from the camera, with an angle of α (see Figure 3 (a)). The board's size is true to the rear of the actual object, which is crucial for realistic physical-world view synthesis. Subsequently, we establish a projection function, defined by variables such as W , H , z_c , α , etc., that enables the transformation of a pixel from A to a pixel in the view I_t of camera C_t . This function includes a 3D-to-2D projection process, which is designed in such a way that by adjusting z_c and α , we can generate numerous instances of I_t with varying perspectives of A .

Obtaining I_s , which is intended to construct a stereo pair with I_t , does not require a second physical camera. Instead, we utilize two successive frames from a street view video, such as those found in the KITTI dataset [51], denoted as I_s^0 and I_t^0 . These frames can mimic a stereo pair captured

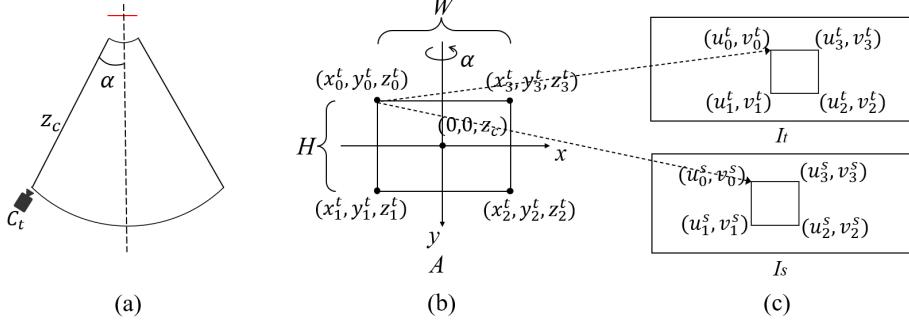


Fig. 3: (a) A top-down bird view of the relative positions of the camera and the target object. (b) The 3D coordinates of the object’s four corners in the camera frame. (c) Projection of the physical-world object onto the two views.

by two adjacent cameras. The distinctions between the two images inherently denote the relative positions of the two cameras. A significant advantage of this approximation is the ease with which we can simulate a wide range of camera placements and street views by selecting different consecutive video frames. This is in line with existing research [29], [30]. We substitute the portion of I_t that does not belong to A , i.e., the object’s background, with the corresponding part in I_t^0 . In essence, we create a realistic I_t by overlaying the image of the object onto a background image I_t^0 , while maintaining respect for physical world constraints.

The adjacent view I_s with A is also synthesized using I_s^0 in a similar way, but this time, we need to consider the camera pose transformation from C_t to C_s . This transformation, represented by $T_{s \rightarrow t}$ and established from both I_s^0 and I_t^0 , converts the 3D coordinates of A in the coordinate system of camera C_t to that of camera C_s . Then, the same 3D-to-2D projection is utilized to synthesize I_s (Figure 3(c)). Consequently, the resulting view of A in I_s aligns with the camera pose represented by I_s^0 . I_t and I_s are then incorporated into model hardening.

Formally, when the center of camera C_t ’s view is in alignment with the center of the image board A (see Figure 3), the relationship between a pixel (u^A, v^A) in A and its corresponding 3D coordinate (x^t, y^t, z^t) can be defined as:

$$\begin{bmatrix} x^t \\ y^t \\ z^t \\ 1 \end{bmatrix} = \begin{bmatrix} \cos \alpha & 0 & -\sin \alpha & 0 \\ 0 & 1 & 0 & 0 \\ \sin \alpha & 0 & \cos \alpha & z_c \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} W/w & 0 & -W/2 \\ 0 & H/h & -H/2 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} u^A \\ v^A \\ 1 \end{bmatrix}, \quad (4)$$

where w and h represent the pixel dimensions (i.e., width and height, respectively) of A . The remaining variables (e.g., α, z_c) are explained in Figure 3. The 3D coordinates can then be further transformed into pixels in I_t and I_s as follows:

$$\begin{aligned} [u^t \ v^t \ 1]^\top &= 1/z^t \cdot K \cdot [x^t \ y^t \ z^t \ 1]^\top, \\ [x^s \ y^s \ z^s \ 1]^\top &= T_{t \rightarrow s} \cdot [x^t \ y^t \ z^t \ 1]^\top, \\ [u^s \ v^s \ 1]^\top &= 1/z^s \cdot K \cdot [x^s \ y^s \ z^s \ 1]^\top, \end{aligned} \quad (5)$$

K represents the camera’s intrinsic parameters [50]. By consolidating Equation 4 with Equation 5, we can formulate the projections from a pixel (u^A, v^A) of the object image to a pixel (u^t, v^t) in I_t and to a pixel (u^s, v^s) in I_s . Let $[u^t \ v^t \ 1]^\top = P_{z_c, \alpha}^{A \rightarrow t}(u^A, v^A)$ and $[u^s \ v^s \ 1]^\top = P_{z_c, \alpha, T_{t \rightarrow s}}^{A \rightarrow s}(u^A, v^A)$, we synthesize I_t and I_s :

$$I_t[u, v] = \begin{cases} A[u^A, v^A], [u \ v \ 1]^\top = P_{z_c, \alpha}^{A \rightarrow t}(u^A, v^A) \\ I_t^0[u, v], \quad \text{otherwise} \end{cases}, \quad (6)$$

$$I_s[u, v] = \begin{cases} A[u^A, v^A], [u \ v \ 1]^\top = P_{z_c, \alpha, T_{t \rightarrow s}}^{A \rightarrow s}(u^A, v^A) \\ I_s^0[u, v], \quad \text{otherwise} \end{cases}, \quad (7)$$

where I_t^0 and I_s^0 are the background images that indirectly represent the relative positions of the cameras. By adjusting variables such as $I_t^0, I_s^0, z_c, \alpha, A$, we can generate a vast number of I_t and I_s instances which are then utilized in the hardening process. The production of this synthetic data incurs practically no expense compared to generating a comparably diverse dataset from the physical world.

3.3 Robust Adversarial Perturbations

We employ an optimization-based method to create robust adversarial perturbations, denoted as δ , on the object image A . This results in the corresponding adversarial object $A + \delta$, and we synthesize I'_t by substituting A with $A + \delta$ in Equation 6. The synthesized I'_t is subsequently utilized in adversarial training. We limit the perturbations using the L_0 -norm, which effectively controls the number of perturbed pixels. This is a departure from digital-world attacks that typically use L_∞ -norm or L_2 -norm-bounded perturbations, like FGSM [6], Deepfool [35], and PGD [34].

Physical-world attacks usually employ adversarial patches [52] with no restrictions on the magnitude of perturbations within the patch area. This is because larger perturbations are often required to cause consistent model errors in the face of diverse environmental conditions, such as changes in lighting, viewing angles, distance, and camera noise. Therefore, L_0 -norm is a better fit for physical-world attacks as it restricts the number of pixels to be perturbed without limiting the perturbation magnitude of individual pixels. However, the calculation of L_0 -norm is non-differentiable by definition, making it unsuitable for optimization. For this reason, we adopt a ‘soft’ version of it, as proposed in [53]. The core idea is to separate the perturbations into positive or negative components and utilize the long-tail effects of the tanh function in the normalization term. This effectively models the two extremes of a pixel’s value change: zero perturbation or arbitrarily large perturbation. As a result, a pixel tends to either have a large perturbation or no perturbation at all.

$$\delta = \max_p \cdot (\text{clip}(b_p, 0, 1) - \text{clip}(b_n, 0, 1)). \quad (8)$$

$$\begin{aligned} \mathcal{L}_{pixel} = & \sum_{h,w} \left(\max_c \left(\frac{1}{2} (\tanh(\frac{b_p}{\gamma}) + 1) \right) \right) \\ & + \sum_{h,w} \left(\max_c \left(\frac{1}{2} (\tanh(\frac{b_n}{\gamma}) + 1) \right) \right). \end{aligned} \quad (9)$$

TABLE 1: Benign performance of original and hardened models on depth estimation.

Models	Monodepth2					DepthHints				
	ABSE \downarrow	RMSE \downarrow	ABSR \downarrow	SQR \downarrow	$\delta \uparrow$	ABSE \downarrow	RMSE \downarrow	ABSR \downarrow	SQR \downarrow	$\delta \uparrow$
Original	2.125	4.631	0.106	0.807	0.877	2.021	4.471	0.100	0.728	0.886
LO+SelfSup (Ours)	2.16	4.819	0.105	0.831	0.874	2.123	4.689	0.103	0.777	0.877
LO+Sup	2.162	4.648	0.110	0.846	0.876	2.015	4.453	0.100	0.734	0.887
LO+Contras	3.218	6.372	0.155	1.467	0.782	3.626	6.742	0.209	1.561	0.694
PGD+SelfSup	2.169	4.818	0.105	0.826	0.874	2.120	4.680	0.103	0.774	0.877
PGD+Sup	2.153	4.637	0.109	0.838	0.876	2.019	4.460	0.101	0.736	0.886
PGD+Contras	3.217	6.083	0.194	1.825	0.756	3.928	7.526	0.213	2.256	0.701

* For hardened models, A+B denotes generating adversarial perturbation with method A and training with method B.

In particular, the perturbation is specified in Equation 8, and the normalization term is represented by \mathcal{L}_{pixel} in Equation 9. Here, b_p and b_n are the positive and negative components, respectively. The `clip()` function limits the variable within the range of [0,1]; h, w, c stand for the image's height, width, and channels, respectively; and γ is a scaling factor. We refer [53] for comprehensive explanations.

Equation 10 provides the formal representation of our method for generating perturbations. In this equation, S_p represents a distribution of physical-world distances and viewing angles (for instance, mirroring the relationships between cameras and vehicles during actual driving scenarios); S_b is the collection of background scenes (such as different street views); $M()$ is the MDE model that predicts the estimated depth; and $MSE()$ is the mean square error.

$$\begin{aligned} \min_{b_n, b_p} E_{z_c, \alpha \sim S_p, I_t^0 \sim S_b} & \left[MSE \left(M(I_t')^{-1}, 0 \right) \right] + \mathcal{L}_{pixel}, \\ \text{s.t. } L_0(\delta) & \leq \epsilon. \end{aligned} \quad (10)$$

Our adversarial objective is to make the target object appear more distant, so our aim is to maximize the depth estimation (*i.e.*, minimize its reciprocal). Intuitively, we synthesize I_t' with random I_t^0 and varied α and z_c of the object A and employ the expectation of transformations (EoT) method [54] to enhance physical-world robustness. In the adversarial loss term, we aim to minimize the mean square error between zero and the reciprocal of the depth estimation for the synthesized scenario, while using \mathcal{L}_{pixel} as the normalization term for perturbations. The parameter ϵ is a predefined L_0 -norm threshold for perturbations, which signifies the maximum proportion of pixels that can be perturbed (*e.g.*, $\epsilon = 1/10$ means at most 1/10 of the pixels within an image can be perturbed for adversarial attack).

4 EVALUATION

In this section, we evaluate the performance of our method in white-box, black-box, and physical-world attack scenarios, and discuss the ablations. Our code is available at: <https://github.com/Bob-cheng/DepthModelHardening>

4.1 Experimental Setup

Networks and Dataset. We use Monodepth2 [29] and DepthHints [30] as our subject networks to harden. They are representative and popular self-supervised MDE models that are widely used as benchmarks in the literature. Both models are trained on the KITTI dataset [51] and our methods fine-tune the original models publicly available.

Baselines To the best of our knowledge, no previous research has been devoted to harden MDE models, hence

there are no immediate benchmarks at our disposal. Therefore, we opt to expand upon current contrastive learning-based and supervised learning-based adversarial training techniques to MDE, utilizing these as our benchmarks.

(1) *Adversarial Contrastive Learning.* Contrastive learning has gained considerable traction as a technique specifically designed for self-supervised learning environments [55]–[60], and it's been employed in combination with adversarial examples for two primary reasons: firstly, to bolster model resilience against adversarial attacks [13], and secondly, to amplify the efficacy of contrastive learning itself [12]. In the context of this study, we've taken a leading-edge contrastive learning-based adversarial training technique [13] and adapted it to fortify MDE models against physical threats. This adaptation serves as a comparative baseline for our own methodology.

In line with the approach used in previous studies [13], our contrastive learning strategy identifies benign examples (represented as I_t) and their corresponding adversarial examples (I_t') as positive pairs, with additional augmentation achieved by modifying color. However, our approach differs in that we do not necessitate negative pairs. Instead, we leverage a learning strategy introduced in SimSiam [61] that only calls for positive pairs, capable of achieving competitive results with reduced batch sizes. The fundamental objective is to heighten the similarity between the embeddings of benign and adversarial examples, such that their decoded depth map outputs are analogous. The parameters of the MDE model's encoder and the predictive MLP network are updated in an iterative manner during training. As we aim to maintain similarity among positive samples, we opt for color augmentation over other transformations like resizing and rotation, as these may influence the depth map output while a change in color wouldn't. The setup of our MLP network and other settings align with those used in SimSiam [61], and we direct readers to the SimSiam [61] study for an in-depth understanding.

(2) *Supervised Adversarial Learning with Estimated Depth.* In a self-supervised setting, where we do not have access to actual depth ground truth, an alternative method for adversarial training involves using depth estimations generated by the original model with benign sample inputs, serving as pseudo ground truth or pseudo labels, and then carrying out supervised adversarial training. We employ the mean square error (MSE) as the loss function to update the MDE model parameters, aiming to minimize the discrepancy between the model's output for adversarial samples and the pseudo ground truth.

The use of pseudo ground truth, as determined by an



Fig. 4: More examples of view synthesis with different background scenes, target objects and distance z_c of the object. Object mask is used to remove background of the 2D object image.

existing model, has been shown to be a straightforward and effective strategy within the realm of semi-supervised learning (SSL) [62]. It has been applied in adversarial training [63] and self-supervised MDE [64] to enhance model performance. Especially within MDE, the use of pseudo ground truth has been demonstrated to be quite satisfactory when compared to the utilization of actual ground truth. Like our supervised baseline, some studies use depth estimations from an existing MDE model (or pseudo depth labels) to supervise subsequent MDE model training. These investigations have found that a model trained under pseudo-supervision can perform comparably to, or even better than, a model trained using ground truth depth. In our work, we carry out experiments to compare the performance of a supervised baseline trained with pseudo depth labels and ground truth depth labels. This experimentation substantiates that choosing a pseudo-supervised baseline does not represent a compromise in quality. The detailed results of this comparison can be found in §4.3.7.

Training Configurations In adversarial training, the ranges of distance z_c and viewing angle α are sampled randomly from 5 to 10 meters and -30 to 30 degrees, respectively. The view synthesis uses EoT [54]. We generate the adversarial perturbations with two methods: L_0 -norm-bounded with $\epsilon = 1/10$ and L_∞ -norm-bounded (*i.e.*, PGD [34]) with $\epsilon = 0.1$. The latter is for comparison purposes. We train with our self-supervised method and two baseline methods based on contrastive learning and supervised learning. Hence, there are 6 approaches combining the 2 perturbation generation methods with the 3 training methods. With these approaches, we fine-tune the original model for 3 epochs on

the KITTI dataset and produce 6 hardened models for each network. In our main experiments, we do not intentionally filter out unrealistic paintings like objects on top of others, since those samples that could induce incorrect distance estimation of the target object are infrequent in the entire dataset. Nevertheless, we conduct additional experiments to evaluate the impact on performance in §4.3.8.

We train our model with one GPU (Nvidia RTX A6000) that has a memory of 48G and the CPU is Intel Xeon Silver 4214R. For each model, doing adversarial training from scratch takes around 70 hours. It includes 20 epochs of training on the KITTI dataset. The fine-tuning of 3 epochs takes about 10 hours. The input resolution of our MDE model is 1024*320 and the original monodepth2 and depthhints models we used for fine-tuning are the official versions trained with both stereo images and videos. In our hardening, we use stereo image pairs with fixed camera pose transformation $T_{t \rightarrow s}$. In perturbation generation, we use 10 steps and a step size of $2.5 \cdot \epsilon/10$ in L_2 and L_∞ -bounded attacks to ensure that we can reach the boundary of the ϵ -ball from any starting point within it [34] and a batch size of 12. In MDE training, the batch size is 32, and the learning rate is 1e-5. We use Adam as the optimizer and other training setups are the same as the original model.

As for the selection of 2D images of objects, as shown in Figure 3 (a) and Figure 3 (b), we have assumptions about the initial relative positions between the target object and the camera (*i.e.*, the 3D coordinates of the center of the object is $(0, 0, z_c)$ in the camera's coordinate system and the viewing angle α of the camera is 0 degree). Hence, for a more realistic and high-quality synthesis, the camera should look at the

TABLE 2: Defensive performance of original and hardened models under attacks.

Attacks	Original		L0+SelfSup (Ours)		L0+Sup		L0+Contras		PGD+SelfSup		PGD+Sup		PGD+Contras		
	ABSE↓	δ↑	ABSE↓	δ↑	ABSE↓	δ↑	ABSE↓	δ↑	ABSE↓	δ↑	ABSE↓	δ↑	ABSE↓	δ↑	
Monodepth2	L0 1/20	4.71	0.65	0.18	0.99	<u>0.44</u>	<u>0.98</u>	1.30	0.67	0.49	0.95	0.58	0.94	0.69	0.94
	L0 1/10	6.08	0.51	<u>0.25</u>	<u>0.98</u>	<u>0.94</u>	<u>0.93</u>	1.75	0.54	<u>0.82</u>	0.91	1.18	0.79	0.96	0.89
	L0 1/5	8.83	0.39	<u>0.34</u>	<u>0.9</u>	1.59	<u>0.85</u>	2.32	0.46	2.33	0.70	2.72	0.51	<u>1.11</u>	0.85
	L0 1/3	9.99	0.34	<u>0.52</u>	<u>0.96</u>	2.08	<u>0.78</u>	2.65	0.41	4.32	0.51	4.09	0.41	<u>1.75</u>	0.70
	PGD 0.05	4.74	0.56	<u>0.82</u>	<u>0.97</u>	1.29	0.80	6.61	0.38	<u>0.67</u>	<u>0.98</u>	0.82	0.95	1.82	0.67
	PGD 0.1	11.68	0.34	<u>1.53</u>	<u>0.85</u>	2.53	0.71	12.74	0.24	<u>1.38</u>	<u>0.95</u>	1.64	0.76	2.66	0.53
	PGD 0.2	17.10	0.23	<u>3.46</u>	<u>0.69</u>	6.14	0.50	20.14	0.15	<u>3.81</u>	<u>0.58</u>	5.04	0.32	3.97	0.42
	Patch	2.71	0.77	<u>0.39</u>	<u>0.98</u>	1.35	0.89	6.40	0.52	<u>0.40</u>	<u>0.98</u>	0.84	0.92	0.50	0.95
DepthHints	L0 1/20	2.33	0.66	0.19	0.99	0.34	0.96	1.06	0.83	0.22	0.99	0.58	0.89	0.40	0.99
	L0 1/10	3.19	0.59	0.27	0.99	0.48	0.95	1.56	0.77	<u>0.42</u>	<u>0.98</u>	1.03	0.79	0.60	0.97
	L0 1/5	4.77	0.42	0.40	0.98	0.96	0.82	1.85	0.75	0.83	0.92	1.93	0.68	<u>0.66</u>	<u>0.95</u>
	L0 1/3	6.03	0.36	<u>0.48</u>	<u>0.98</u>	1.64	0.68	2.60	0.69	1.45	0.79	3.06	0.57	<u>1.16</u>	0.82
	PGD 0.05	3.11	0.48	0.62	0.98	1.23	0.75	4.05	0.55	<u>0.64</u>	<u>0.98</u>	0.93	0.79	<u>1.16</u>	0.79
	PGD 0.1	6.44	0.36	<u>1.27</u>	<u>0.86</u>	2.37	0.62	7.59	0.36	<u>1.21</u>	<u>0.92</u>	1.76	0.67	1.74	0.62
	PGD 0.2	18.37	0.23	<u>3.09</u>	<u>0.60</u>	7.13	0.41	13.59	0.24	6.14	0.68	4.22	0.37	2.60	0.49
	Patch	0.70	0.91	<u>0.46</u>	<u>0.95</u>	0.53	0.93	6.90	0.49	0.46	0.95	<u>0.36</u>	0.99	<u>0.34</u>	<u>0.98</u>

*Bold and underlining indicate the best and second-best performance in each row. Hardened models are named the same as in Table 1.



Fig. 5: The reference images used in our human study.

center of the target object at the same height while taking the 2D image of the object. The width w and height h of the 2D image of the object should be proportional to the physical size W and H of it: $w/W = h/H$. Moreover, when we prepared the 2D image of the object, we also prepared a corresponding mask to “cut out” the main body of the object for projection and we take the object together with its shadow to preserve reality. Examples of object masks can be found in Figure 4.

We train models with L_0 and L_∞ -bounded (*i.e.*, PGD) perturbations in our evaluation but not L_2 norm because [34] has demonstrated that models hardened with L_∞ -bounded perturbations are also robust against L_2 -bounded attacks and our experiments in §4.3.4 also validate the robustness of our models. In addition, physical-world attacks with adversarial patches have more resemblance to L_0 -bounded attacks that only restrict the ratio of perturbed pixels rather than the magnitude of the perturbation.

Attacks. We conduct various kinds of attacks to evaluate the robustness of different models. They are L_0 -norm-bounded attacks with $\epsilon = 1/20, 1/10, 1/5$ and $1/3$, L_∞ -norm-bounded (PGD) attacks with $\epsilon = 0.05, 0.1$ and 0.2 (image data are normalized to $[0,1]$), and an adversarial patch attack in [31]. Adversarial perturbation or patch is applied to an object image. The patch covers $1/10$ of the object at the center. Each attack is evaluated with 100 randomly selected background scenes. The object is placed at a distance range of 5 to 30 meters and a viewing angle range of -30 to 30 degrees. We report the average attack performance over different background scenes, distances, and viewing angles for each attack. In addition, we conduct the SOTA physical-world attack [11] with the printed optimal patch and a real

vehicle in driving scenarios. Adversarial examples are in §4.2.6. Evaluation with more attacks are in §4.3.4.

Evaluation Metrics In particular, the evaluation metrics we used in our evaluation are defined as follows, where we use $X = \{x_1, x_2, \dots, x_n\}$ to denote the estimated depth map and $Y = \{y_1, y_2, \dots, y_n\}$ to denote the reference depth map and $I()$ is the indicator function that evaluates to 1 only when the condition is satisfied and 0 otherwise.

$$\begin{aligned} ABSE &= \frac{1}{n} \sum_{i=1}^n |x_i - y_i|, \quad RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (x_i - y_i)^2} \\ ABSR &= \frac{1}{n} \sum_{i=1}^n \left(\frac{|x_i - y_i|}{y_i} \right), \quad SQR = \frac{1}{n} \sum_{i=1}^n \frac{(x_i - y_i)^2}{y_i} \\ \delta &= \frac{1}{n} \sum_{i=1}^n I(\max\left\{\frac{x_i}{y_i}, \frac{y_i}{x_i}\right\} < 1.25) \end{aligned} \quad (11)$$

The mean absolute error (ABSE) and root mean square error (RMSE) are common metrics and are easy to understand. Intuitively, the relative absolute error (ABSR) is the mean ratio between the error and the ground truth value, and the relative square error (SQR) is the mean ratio between the square of error and the ground truth value. δ denotes the percentage of pixels of which the ratio between the estimated depth and ground-truth depth is smaller than 1.25.

4.2 Main Results

4.2.1 Benign Performance

Together with the original model, we have 7 models under test for each network. We evaluate the depth estimation performance on the KITTI dataset using the Eigen split and report the results in Table 1. As shown, self-supervised and supervised methods have little influence on the models’ depth

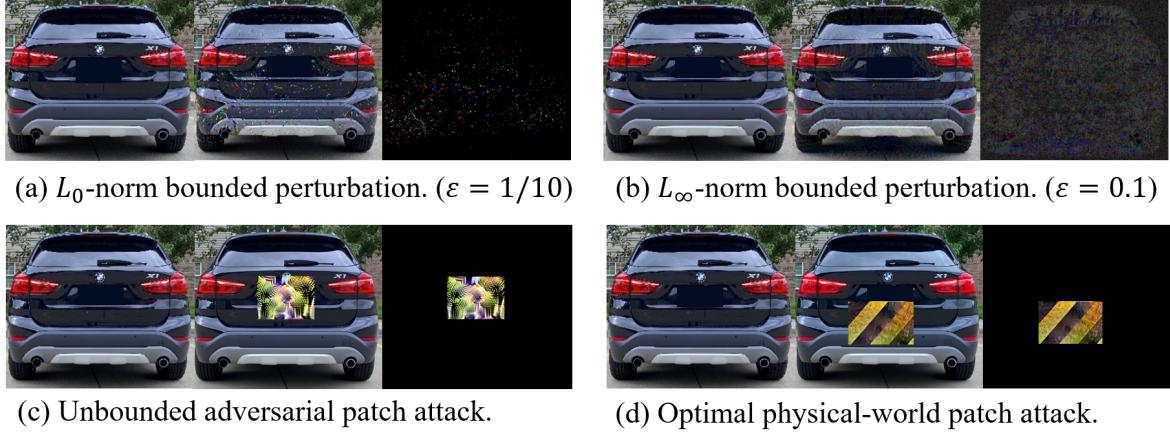


Fig. 6: Examples of adversarial attacks in our robustness evaluation.

TABLE 3: Human evaluations of the quality of our synthesized images. We show the number of participants who gave a score in the corresponding range in each row.

Score Ranges	Size	Location	Lightning	Overall
1-2	0	0	1	0
5-6	4	6	6	3
7-8	16	24	23	18
9-10	42	48	40	45
Total	100	100	100	100
Average Score	7.7	7.21	7.43	7.74

estimation performance, which means these approaches can harden the model with nearly no benign performance degradation. In contrast, the contrastive learning-based approach performs the worst. The ABSE of estimated depth is over 1 m worse than the original model. The reason could be that contrastive learning itself does not consider the specific task (*i.e.*, MDE) but fine-tunes the encoder to filter out the adversarial perturbations. Thus the benign performance is sacrificed during training. The other two methods consider the depth estimation performance either by preserving the geometric relationship of 3D space in synthesized frames or by supervising the training with estimated depth.

4.2.2 Quality of the view synthesis

To demonstrate the quality of our view synthesis, we show more examples of the synthesized images in Figure 4. I_s and I_t are two adjacent views of the same scene. Different target objects (e.g., the white sedan, black SUV and traffic barrier) are synthesized into the views with various distances z_c and background scenes. To evaluate the veracity of these images, we use Amazon Mechanical Turk to conduct human evaluations of our synthesized images. Participants are requested to evaluate the quality of the synthesized objects from 4 distinct perspectives: size, consistency of location, lighting, and overall quality. The score ranges from 1 to 10 for each metric. We use a score of 1 to indicate scenes synthesized with random projections and a score of 10 for real scenes taken by stereo cameras. Examples of perfect (score of 10) and unrealistic (score of 1) scenes for each metric are provided as references (See Figure 5). The results of the experiment with 100 participants can be found in Table 3. In each row, we show the number of participants who gave a score within the corresponding range and summarized the average in the last row. As demonstrated, We got an average score of 7.7 regarding the size, 7.21 regarding the location,

TABLE 4: Defensive performance of original and hardened models under black-box attacks.

Target Source	Original		$L_0+SelfSup$ (Ours)		L_0+Sup		$L_0+Contras$	
	ABSE	$\delta \uparrow$	ABSE	$\delta \uparrow$	ABSE	$\delta \uparrow$	ABSE	$\delta \uparrow$
Original	-	-	0.25	0.99	0.55	0.95	1.35	0.71
$L_0+SelfSup$	0.52	0.93	-	-	0.24	0.98	0.45	0.95
L_0+Sup	1.09	0.80	0.29	0.99	-	-	0.65	0.89
$L_0+Contras$	2.65	0.50	0.22	0.99	0.27	0.98	-	-

*Bold indicates the best performance in each row or column.

7.43 regarding the lighting, and 7.74 regarding the overall quality. Compared with the real scene, our synthesized scene is slightly inferior but still much better than random projection. More importantly, it works well to harden models against real attacks at a low cost.

4.2.3 White-box Attacks

We conduct various white-box attacks on each model to evaluate the robustness of hardened models. Specifically, for each model, we compare the estimated depth of the adversarial scene (*i.e.*, I'_t) with that of the corresponding benign scene (*i.e.*, I_t in Equation 6) and larger difference means worse defensive performance. Table 2 shows the result. As shown, all the hardened models have better robustness than the original models against all kinds of attacks, and it is generic on the two representative MED networks, which validates the effectiveness of our adversarial training approaches. Comparing different approaches, $L_0+SelfSup$ has the best performance in general. It reduces the ABSE caused by all-level L_0 -norm-bounded attacks from over 4.7 m to less than 0.6 m. Specifically, the self-supervision-based method outperforms the contrastive learning-based and the supervision-based methods regardless of the perturbation generation method used. It is because the self-supervision-based method follows the original training procedure that is carefully designed for the network and has been evaluated thoroughly. It is not surprising that models adversarially trained with L_0 -norm-bounded perturbation (our method) achieve better robustness against L_0 -norm-bounded attacks and so do PGD-based methods, but more importantly, L_0 -norm-based training also has good defensive performance against PGD attacks. The robustness of $L_0+SelfSup$ is only slightly worse than $PGD+SelfSup$ on some PGD attacks and even better than it on stronger PGD attacks. An explanation is

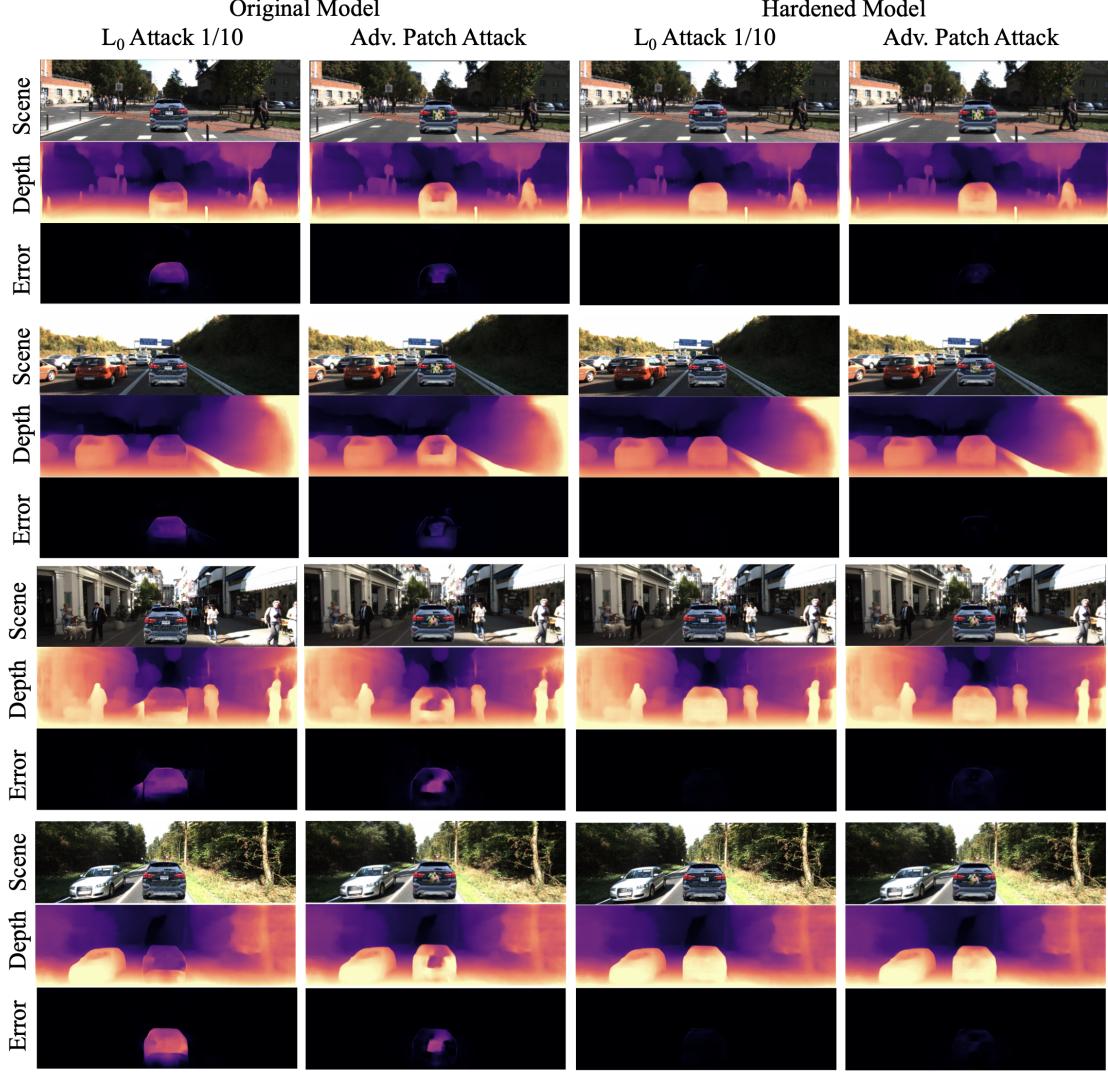


Fig. 7: Qualitative results of the defensive performance of our hardened model.

TABLE 5: Ablations on the range of angles in adversarial training with L0+SelfSup.

Attacks	0° 10° 20° 30° 40°				
	ABSE↓	ABSE↓	ABSE↓	ABSE↓	ABSE↓
L0 1/10	0.271	0.260	0.253	0.251	0.249
L0 1/5	0.363	0.352	0.351	0.347	0.350
PGD 0.1	1.662	1.543	1.544	1.539	1.536
PGD 0.2	3.587	3.472	3.465	3.467	3.470

that L_0 -norm does not restrict the magnitude of perturbation on each pixel, and stronger PGD attacks are closer to this setting (*i.e.*, high-magnitude perturbations) and can be well-defended using the L_0 -norm-based adversarial training. Monodepth2 is vulnerable to the patch attack, and this kind of attack can be well-defended by our methods. L0+SelfSup also performs the best. Depthhints itself is relatively robust to the patch attack, and our methods can further reduce the attack effect. Our defense generalizes well to complex scenes including various road types, driving speed, and the density of surrounding objects. Qualitative results are in §4.2.6.

4.2.4 Black-box Attacks

We also validate our methods against black-box attacks. We use the original Monodepth2 model and the models fine-

TABLE 6: Defensive performance on various target objects.

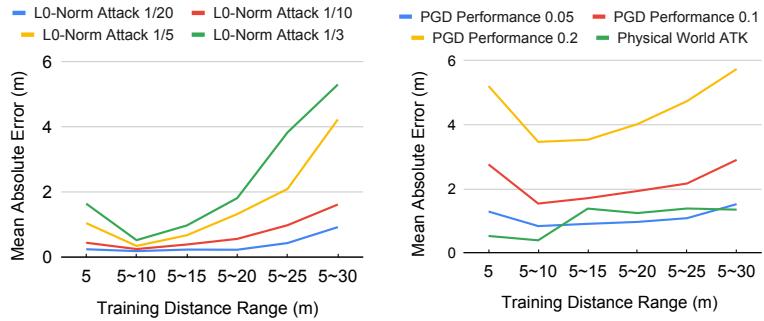
Objects	Original		L0+SelfSup (Ours)		L0+Sup		L0+Contras	
	ABSE	$\delta \uparrow$	ABSE	$\delta \uparrow$	ABSE	$\delta \uparrow$	ABSE	$\delta \uparrow$
SUV Black	6.079	0.512	0.248	0.988	0.949	0.934	1.642	0.552
SD Blue	4.259	0.456	0.651	0.905	1.097	0.833	1.701	0.606
SD White	4.262	0.437	1.408	0.776	2.602	0.632	2.324	0.505
SUV Grey	6.379	0.535	1.565	0.889	1.859	0.836	2.476	0.552
Barrier	4.479	0.375	1.619	0.54	2.3	0.483	0.813	0.767

*Abbreviations. SD: Sedan.

tuned with L_0 -norm-bounded perturbations and the three training methods. We perform L_0 -norm-bounded attacks on each model with $\epsilon = 1/10$ and apply the generated adversarial object to other models evaluating the black-box attack performance. The first column in Table 4 denotes the source models and other columns are the target models' defensive performance. Looking at each column, adversarial examples generated from L0+SelfSup have the worst attack performance, which indicates the low transferability of adversarial examples generated from the model trained with our self-supervised methods. From each row, we can observe that L0+SelfSup has the best defensive performance against adversarial examples generated from each source model, which further validates the robustness of L0+SelfSup. In summary, the self-supervised training method can produce



Fig. 8: Physical-world attack and defense. Video: https://youtu.be/_b7E4yUFB-g



(a) L_0 -norm attack.

Fig. 9: Defensive performance with different training distance ranges.



Fig. 10: The various target objects in the transferability evaluation.

safer and more robust models. Their examples have low transferability, and they defend against black-box attacks well.

4.2.5 Physical-world Attacks

Our evaluation with the state-of-the-art physical-world MDE attack [11] validates the effectiveness of our method in various real-world lighting conditions, driving operations, road types, etc. The experimental settings are the same as [11]. Figure 8 shows the result. The first row is the real-world adversarial scene, in which a car is moving with an adversarial patch attached to the rear. The second row is the depth predicted by the original Monodepth2 model and the third row is predicted by our hardened model ($L_0 + \text{SelfSup}$). The “hole” in the output of the original model caused by the adversarial patch is fixed in our hardened model. It is known that the adversarial attacks designed for the physical world are still generated in the digital world and they have better digital-world attack performance than physical-world performance because additional environmental variations in the real world degrade the effectiveness of adversarial patterns [11], [65], [66]. Hence, defending attacks in the digital world is more difficult and our success in digital-world defense in previous experiments has implied effectiveness in the real world.

4.2.6 Qualitative Results against Adversarial Attack

Figure 6 gives examples of the three kinds of adversarial attacks we conducted in our robustness evaluation. The first column is the original object; the second column is the adversarial one and the third column is the adversarial perturbations. We scale the adversarial perturbations of the L_∞ -norm-bounded attack for better visualization. L_0 -norm restricts the number of perturbed pixels. L_∞ -norm restricts the magnitude of perturbation of each pixel. Adversarial patch attack perturbs pixels within the patch area.

Figure 7 shows the qualitative results of our evaluation. The vertical axis denotes different street views and the horizontal axis denotes different models and attacks. Here we compare the performance of the original Monodepth2



Fig. 11: Performance of fine-tuning and training from scratch.

model with our model hardened by **L0+SelfSup**. For each street view, model, and attack, the first row is the adversarial scene (*i.e.*, the scene with the adversarial object), the second row is the corresponding depth estimation and the third row is the depth estimation error caused by the adversarial perturbations. As shown, our method mitigates the effectiveness of attacks significantly and the attacks can hardly cause any adversarial depth estimation error on our hardened model. In addition, our method works on different scenes including complex ones that have multiple pedestrians or vehicles at different speeds. It is because we do not have assumptions about the scene geometry or surrounding objects in our method. Both the scene synthesis and depth estimation are single-image-based and the scene geometry will not affect the quality of synthesis.

4.3 Ablations

4.3.1 Distance Range in Training.

While synthesizing views in training, the range of distance z_c of the target object is randomly sampled from d_1 to d_2 meters. In this ablation study, we evaluate the effect of using different ranges of distance in training. We use **L0+SelfSup** to fine-tune the original Monodepth2 model. The ranges of

TABLE 7: Defensive performance of original and hardened models under **more attacks**.

Attacks	Original		L0+SelfSup (Ours)		L0+Sup		L0+Contras	
	ABSE \downarrow	$\delta \uparrow$						
L_2 -PGD $\epsilon = 8$	1.403	0.76	0.294	0.996	1.161	0.741	0.66	0.919
L_2 -PGD $\epsilon = 16$	6.491	0.522	0.597	0.984	2.516	0.479	1.437	0.734
L_2 -PGD $\epsilon = 24$	13.018	0.354	0.932	0.913	3.613	0.387	2.92	0.7
APGD $\epsilon = 0.05$	5.557	0.739	0.423	0.976	1.614	0.793	2.71	0.859
APGD $\epsilon = 0.1$	10.216	0.46	0.928	0.945	3.279	0.603	4.578	0.793
Square Attack $\epsilon = 0.1$ N=5000	0.924	0.924	0.422	0.991	0.712	0.934	0.568	0.973
Gaussian Blur	0.323	0.996	0.191	0.997	0.288	0.997	0.264	0.997
AdvLight	0.512	0.988	0.493	0.991	0.513	0.987	0.504	0.988

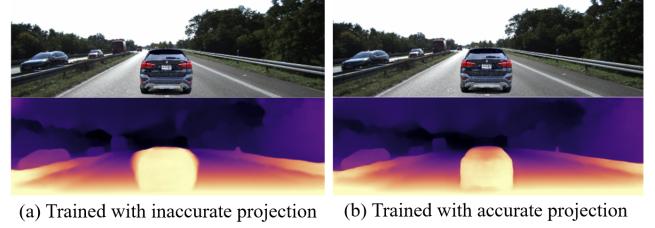
TABLE 8: Benign performance of models trained with methods combinations.

	ABSE \downarrow	RMSE \downarrow	ABSR \downarrow	SQR \downarrow	$\delta \uparrow$
Original	2.125	4.631	0.106	0.807	0.877
SelfSup+Con	2.172	4.84	0.105	0.847	0.875
SelfSup+Sup	2.161	4.819	0.105	0.836	0.875
Con+Sup	2.408	4.986	0.122	0.93	0.849
All	2.171	4.83	0.105	0.841	0.875

distance we use in training are [5, 5], [5, 10], [5, 15], [5, 20], [5, 25] and [5, 30] (Unit: meter). Note that, for a fair comparison, the range of distance we use in model evaluation is always from 5 to 30 meters. The results are shown in Figure 9. As shown, the model trained with a distance range of 5-10 meters has the best robustness and a larger or smaller distance range could lead to worse performance. It is because further distances lead to smaller objects on the image and fewer pixels are affected by the attack. Thus the attack performance is worse at further distances and training with these adversarial examples is not the most effective. If the distance in training is too small (e.g. 5 meters), the model cannot defend various scales of attack patterns and cannot generalize well to further distances. In our experiments, the range of 5-10 meters makes a good balance between training effectiveness and generality. Other ablation studies about viewing angles range in training are in §4.3.2, transferability to unseen target objects is in §4.3.3, comparing training from scratch and fine-tuning is in §4.3.5 and the performance of method combinations can be found in §4.3.6.

4.3.2 Viewing Angles Range in Training

Randomizing the degree of synthesis will make the model more robust in the physical-world settings because the camera’s viewing angle is not fixed toward the target object in practice, and the range we use (-30 degrees to 30 degrees) covers the most common situations. Ablations of the range of angles during adversarial training do not have as much effect as changing the range of distance because distance has a dominant influence on the size (*i.e.*, number of pixels) of the adversarial object on the synthesized scene image (*i.e.*, the further object looks smaller) and the number of adversarial pixels affects the attack performance a lot [52]. We conduct experiments with different viewing angle ranges, and other settings are the same as the ablation study of distance ranges in Training. Table 5 shows the result. As shown, ranges of viewing angles have less influence on the defensive performance, and using a fixed setting (*i.e.*, 0°) still performs the worst, which is consistent with our claims.

Fig. 12: **Influence of inaccurate projection.** The outline of the depth estimation of the target object is blurred.

4.3.3 Transfer to other target objects

The target object under attack may not be used in training. In this experiment, we evaluate how the adversarially trained models perform on unseen target objects. We evaluate with four vehicles and one traffic barrier using the original and the three hardened Monodepth2 models and we perform L_0 -norm-bounded attack on each object with $\epsilon = 1/10$. Figure 10 shows the examples of target objects that we used in our transferability evaluation. Figure 10(a) is the object we used in adversarial training and others are target objects under attack in the robustness evaluation. Table 6 shows the result. The first column is the target objects under attack and the following columns are the performance of different models. The first object (BMW SUV Black) is an object used in training and our hardened models are most robust on it. The others are unseen objects during training and the hardened models can still mitigate the attack effect a lot compared with the original model, which validates the transferability of our hardened models. L0+SelfSup still has the best performance in general.

4.3.4 Robustness against more attacks

In addition to the attacks evaluated in our main paper (PGD attacks, l_0 attacks, adversarial patch attack and physical-world optimal patch attack), we have also evaluated the robustness of our models against more attacks: the L_2 -bounded PGD attack [34], APGD attacks in AutoAttack [67], Square attack [68](a query-based black-box attack), Gaussian Blur [69] and AdvLight [70]. The subject network is Monodepth2. Results can be found in Table 7. As shown, Our model is still more robust than the original model in all cases, and our method outperforms all others. Attacks with an L_∞ -bound of 0.1 can only cause less than 1 m depth estimation error on our hardened models while more than 10 m on the original model. The black-box attack does not have a good performance in the MDE task and only causes a mean depth estimation error of 0.924 m with 5000 queries to the original model. Although the Gaussian Blur and AdvLight are more stealthy and natural attacks, they

TABLE 9: Defensive performance of models trained with methods combinations.

Attacks	Original		SelfSup+Con		SelfSup+Sup		Con+Sup		All		
	ABSE \downarrow	$\delta \uparrow$									
Monodepth2	L0 1/20	4.711	0.652	0.182	0.998	0.193	0.989	0.529	0.917	<u>0.187</u>	0.998
	L0 1/10	6.088	0.516	<u>0.244</u>	<u>0.991</u>	0.256	0.988	0.849	0.761	0.24	0.991
	L0 1/5	8.83	0.393	<u>0.357</u>	<u>0.967</u>	0.394	0.969	1.293	0.714	0.288	0.986
	L0 1/3	9.996	0.344	<u>0.491</u>	<u>0.958</u>	0.568	0.954	1.803	0.667	0.419	0.97
	PGD	4.747	0.56	0.681	0.988	0.816	0.982	1.666	0.72	<u>0.709</u>	<u>0.986</u>
	PGD	11.684	0.343	1.332	<u>0.869</u>	<u>1.586</u>	0.839	2.832	0.646	1.637	0.884
	PGD	17.109	0.233	<u>3.382</u>	<u>0.591</u>	3.546	0.65	4.612	0.472	3.056	<u>0.621</u>
	Patch	2.714	0.778	0.758	0.945	0.331	<u>0.977</u>	1.106	0.798	<u>0.277</u>	0.995

*Bold indicates the best performance in each row and underlining means the second best.

TABLE 10: Comparison between supervised baseline trained with pseudo depth label ($L_0 + \text{Sup}$ (Pseudo)) and ground-truth depth label ($L_0 + \text{Sup}$ (GT)).

Attacks	Original		$L_0 + \text{SelfSup}$ (Ours)		$L_0 + \text{Sup}$ (Pseudo)		$L_0 + \text{Sup}$ (GT)	
	ABSE	$\delta \uparrow$	ABSE	$\delta \uparrow$	ABSE	$\delta \uparrow$	ABSE	$\delta \uparrow$
L0 1/10	6.08	0.51	0.25	0.98	0.94	0.93	0.75	0.92
L0 1/5	8.83	0.39	0.34	0.9	1.59	0.85	0.98	0.83
PGD 0.1	11.68	0.34	1.53	0.85	2.53	0.71	2.44	0.68
PGD 0.2	17.1	0.23	3.46	0.69	6.14	0.50	5.11	0.32

are not as effective as other methods in the task of MDE (*i.e.*, per-pixel-based regression task).

4.3.5 Training From Scratch

We also compare the robustness performance of training from scratch and fine-tuning an existing model with our self-supervised adversarial training method using L_0 -norm-bounded perturbation. We evaluate on Monodepth2 and Figure 11 shows the ABSE result. As shown, training from scratch can provide more robustness, especially for higher-level attacks. As for the benign performance (*i.e.*, depth estimation performance), the ABSE of the fine-tuned model is 2.16 while the ABSE of the model trained from scratch is 2.468, meaning that the model trained from scratch has slightly worse than benign performance.

4.3.6 Adversarial Training Methods Combination

We introduced three adversarial training methods in §3 and evaluated them separately in our main experiments. In this experiment, we explore the method combinations. Specifically, we combine different loss terms together as a total loss in adversarial training and there are four combinations in total. The perturbation in training is L_0 -norm-bounded with $\epsilon = 1/10$ and other evaluation setups are the same as our main experiments. The depth estimation performance (*i.e.*, clean performance) of each model is shown in Table 8. Results show that models trained with the self-supervised method have equivalent performance as the original model and Con+Sup performs slightly worse than the original one. Table 9 shows the robustness performance under attacks. As shown, combining self-supervised learning with contrastive learning could achieve better robustness than self-supervised learning itself, and combining all three methods further improves the robustness.

4.3.7 Supervised Baseline with Ground Truth Depth

Although in the scope of this paper, we discuss self-supervised scenarios and assume the ground-truth depth is not available in training, we still conduct experiments comparing the defensive performance of our supervised

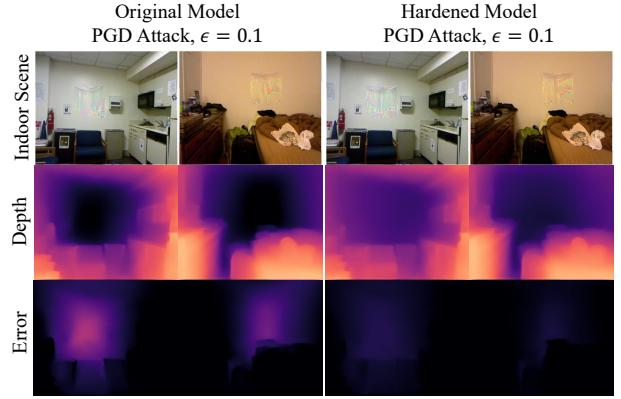


Fig. 13: Qualitative examples of the defensive performance on adversarial indoor scenes.

baseline trained with pseudo-ground truth and that trained with ground-truth depth. We use Monodepth2 as the subject model. The results in Table 10 show that using ground-truth depth ($L_0 + \text{Sup}$ (GT)) has a similar performance to using pseudo ground truth ($L_0 + \text{Sup}$ (Pseudo)), and they are still worse than our self-supervised approach. Hence, whether to use ground-truth depth is not the bottleneck, and our pseudo-supervised baseline is not a weak choice, which also has significant defensive performance against attacks.

4.3.8 Influence of Inaccurate View Synthesis

Incorrect Projections. As stated in §3.2, we project a 2D image of the target object to two adjacent views considering the camera pose transformation $T_{t \rightarrow s}$ between C_s and C_t (Equation 6 and 7), then we enable the self-supervised training of MDE network by reconstructing I_t from I_s using the estimated depth D_{I_t} (Equation 2). In this section, we explore what influence the inaccurate projections of the two camera views would have on training results. In this experiment, we only consider half of the camera pose transformation (*i.e.*, $T_{t \rightarrow s}/2$) while projecting the object to I_s instead of the true value $T_{t \rightarrow s}$, which leads to a wrong I_s intentionally. For example, the distance between the stereo cameras in the KITTI dataset is 0.54 m, but we use 0.27 m to synthesize I_s . I_t is still correctly synthesized. We use L_0 -bounded perturbation with $\epsilon = 1/10$ and self-supervised training to harden the Monodepth2 model. Figure 12 shows the result. As shown, the outline of the estimated depth of the target object is blurred when the MDE model is trained with inaccurate projection. It remains clear for the model trained with accurate projection.

Unrealistic Object Positions. As stated in the training configurations in §4.1, in our main experiments, we did not

TABLE 11: Defensive performance of the model trained **without object synthesis**.

Attacks	Original		L0+SelfSup (Ours)		L0+SelfSup (w/o Obj. Syn.)	
	ABSE ↓	$\delta \uparrow$	ABSE ↓	$\delta \uparrow$	ABSE ↓	$\delta \uparrow$
L0 1/20	4.71	0.65	0.18	0.99	0.36	0.98
L0 1/10	6.08	0.51	0.25	0.98	0.82	0.93
PGD 0.05	4.74	0.56	0.82	0.97	1.08	0.85
PGD 0.1	11.68	0.34	1.53	0.85	2.17	0.77

exclude unrealistic images like objects on top of others from the synthesized training set because they are infrequent. In this section, we evaluate the influence of the unrealistic object positions on model hardening performance. In the new experiment, we employ a semantic segmentation model (Mask2Former [71]) on the sampled stereo image pairs first, then we exclude the synthesized images in which the target object is on top of others. We conduct adversarial training with Monodepth2. Results show that the model hardened with filtered images has similar defensive performance with models in our original setting. Specifically, the ABSE of the new model under l_0 -norm-bounded attack with $\epsilon = 1/10$ and PGD attack with $\epsilon = 0.1$ are 0.248 and 1.535 respectively. In comparison, the ABSE of our previous model under those attacks are 0.251 and 1.533. The benign performance of the new model is slightly improved by 0.88% with the ABSE being 2.141. Overall, it appears that the rare cases of unrealistic object positions in training data have minimal impact on performance.

4.3.9 Training without Object Synthesis

As outlined in our introduction, we simulate the perspective shifts in real-world patch attack on objects by synthesizing objects onto stereo image pairs and adjusting the distance and viewing angles during view synthesis. In this section, we conduct ablation studies to assess how omitting our object synthesis technique affects model hardening performance. Differing from our initial approach, we generate adversarial perturbations directly onto I_t^0 of the stereo images taken from the dataset without synthesizing any target objects. Consistent with our usual protocol, we continue to use the proposed L_0 -norm-bounded perturbation method (i.e., Equation 10) with $\epsilon = 1/10$, along with self-supervised adversarial training. All other training and evaluation configurations remain unchanged. Table 11 presents the defensive capabilities of models trained without object synthesis. The results indicate that these models have reduced defensive effectiveness on various attacks compared to those hardened using our standard method, confirming that our object synthesis technique plays a significant role in enhancing adversarial resilience.

4.4 Extensions

4.4.1 Indoor Scenes

Although we focus on outdoor scenarios like autonomous driving, a domain where self-supervised MDE is widely adopted (e.g., Tesla Autopilot), our technique hardens the MDE networks and does not have any assumptions about indoor or outdoor scenes. Actually, both Monodepth2 and DepthHints models have been proven to be directly applicable on indoor scenes (e.g., NYU-Depth-v2 Dataset) without any retraining [72], [73], which implies the indoor-scene

TABLE 12: Defensive performance on the **indoor scenes** with the NYU-Depth-v2 Dataset.

Attacks	Original		L0+SelfSup (Ours)		L0+Sup		L0+Contras	
	ABSE	$\delta \uparrow$	ABSE	$\delta \uparrow$	ABSE	$\delta \uparrow$	ABSE	$\delta \uparrow$
L0 1/20	1.243	0.772	0.099	0.993	0.363	0.976	0.695	0.812
L0 1/10	3.866	0.54	0.164	0.985	0.49	0.924	1.12	0.733
PGD 0.05	1.784	0.727	0.12	0.986	0.456	0.849	1.641	0.717
PGD 0.1	4.598	0.426	0.288	0.912	0.962	0.775	4.779	0.42

TABLE 13: Defensive performance of the original and hardened **Manydepth models**.

Attacks	Original		L0+SelfSup	
	ABSE↓	$\delta \uparrow$	ABSE↓	$\delta \uparrow$
L0 1/20	1.554	0.78	0.221	0.996
L0 1/10	2.684	0.662	0.301	0.994
PGD 0.05	7.112	0.417	1.270	0.855
PGD 0.1	9.452	0.339	1.614	0.83

capability of our hardened models. We further conduct additional experiments with indoor scenes in the NYU-Depth-v2 Dataset. We launch adversarial attacks in a square area at the center of each scene to mimic a physical patch in the scene and evaluate the defensive performance of the original and hardened Monodepth2 models. Results are shown in Table 12. As shown, our hardened models reduce the depth estimation error caused by the attack significantly and the model trained with our self-supervised method has the best defensive performance. Qualitative examples are presented in Figure 13. The first row illustrates adversarial indoor scenes, while the second and third rows depict depth estimation and errors induced by the adversarial examples, respectively. As demonstrated, our hardened model offers improved defense against adversarial attacks and reduces the error in depth estimation. Therefore, the robustness of our hardened model still holds for indoor scenes.

4.4.2 Advanced Networks

Monodepth2 and DepthHints are the two popular and representative self-supervised MDE networks, and they are widely used as baselines in the literature, so we use them as our subject networks. However, we do not have any assumptions about the MDE network structure, and our method should also work on state-of-the-art MDE models. Hence we conduct additional experiments with Manydepth [28]. We use our L_0 -bounded perturbation with self-supervised adversarial training to harden the original Manydepth model, and Table 13 shows the defensive performance of the original and hardened models. As shown, the original models are still vulnerable to both L_0 -bounded and PGD attacks. The model hardened with our techniques is a lot more robust. The mean depth estimation error is reduced by over 80%, which validates that our techniques are generic and also work on state-of-the-art MDE models.

5 BROADER IMPACT AND LIMITATIONS

Our adversarial training method hardens the widely used monocular depth estimation networks and makes applications like autonomous driving more secure with regard to adversarial attacks. Compared to original training, the hardening of such models with our method does not require additional data or resources and the computational cost is affordable. The adversarial attacks we studied are from

existing works and we do not pose any additional threats. Some limitations we could think of about our method are as follows. In our synthesis of different views, we assume the physical-world object is a 2D image board instead of a 3D model to avoid using an expensive scene rendering engine or high-fidelity simulator and to improve efficiency, which could induce small errors in synthesis though. However, since most physical-world attacks are based on adversarial patches attached to a flat surface, this 2D board assumption is realistic and practical. Precise synthesis should consider lighting factors such as glare, reflections, shadows, etc., but how to do that at a low-cost is an open problem and we leave it as our future work. In addition, although our modeling of the relative positions and viewing angles of the camera and physical object does not cover all real-world conditions, it considers the most common situations. There might be some corner cases in which the adversarial attack could escape from our defense, but we still mitigate the threats in the most common situations and improve the overall security a lot. Additionally, our methodology is primarily designed to counteract adversarial attacks that exploit meticulously crafted adversarial noise to undermine the integrity of DNN-based MDE models. These adversarial perturbations are finely tuned to resonate with the target model, thereby facilitating the attainment of the attack goal. Our approach is designed to enable the model to disregard these adversarial noises during its prediction process. Given that the variations or distortions introduced in the image, such as those caused by a droplet of water on the camera lens, do not constitute the optimised adversarial noise aimed at a specific attack goal, they fall outside the purview of our defense mechanism and, as such, may not be effectively mitigated by our proposed solution. We leave it as a future direction to explore.

6 CONCLUSION

We address the issue of enhancing the resilience of self-supervised Monocular Depth Estimation (MDE) models against real-world attacks without the necessity for depth ground truth data. Our strategy entails a self-supervised adversarial training process, leveraging view synthesis in conjunction with camera poses. Additionally, we incorporate L_0 -norm-limited perturbation creation during the training phase to boost the model's robustness to attacks in the physical world. When benchmarked against conventional supervised learning and contrastive learning techniques, our method demonstrates superior resilience to a range of adversarial attacks, both in digital and physical environments, while maintaining virtually unchanged performance.

Acknowledgement. This research was supported, in part, by IARPA TrojAI W911NF-19-S-0012, NSF 1901242, 1910300 and 2242243, ONR N000141712045, N000141410468, and N000141712947. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Naval Research Laboratory (NRL) or the U.S. Government.

REFERENCES

- [1] Z. Cheng, J. Liang, G. Tao, D. Liu, and X. Zhang, "Adversarial training of self-supervised monocular depth estimation against physical-world attacks," *Int. Conf. Learn. Representations*, 2023.
- [2] Y. Ming, X. Meng, C. Fan, and H. Yu, "Deep learning for monocular depth estimation: A review," *Neurocomputing*, vol. 438, pp. 14–33, 2021.
- [3] A. Karpathy, "Tesla use per-pixel depth estimation with self-supervised learning," 2020, <https://youtu.be/hx7BXih7zx8?t=1334>.
- [4] F. Wimbauer, N. Yang, L. von Stumberg, N. Zeller, and D. Cremers, "Monorec: Semi-supervised dense reconstruction in dynamic environments from a single moving camera," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [5] L. von Stumberg, P. Wenzel, N. Yang, and D. Cremers, "Lm-reloc: Levenberg-marquardt based direct visual relocalization," in *International Conference on 3D Vision (3DV)*, 2020.
- [6] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.
- [7] Z. Cheng, H. Choi, S. Feng, J. C. Liang, G. Tao, D. Liu, M. Zuzak, and X. Zhang, "Fusion is not enough: Single modal attack on fusion models for 3d object detection," in *The Twelfth International Conference on Learning Representations*, 2024.
- [8] Z. Zhang, X. Zhu, Y. Li, X. Chen, and Y. Guo, "Adversarial attacks on monocular depth estimation," *arXiv preprint arXiv:2003.10315*, 2020.
- [9] A. Wong, S. Cicek, and S. Soatto, "Targeted adversarial perturbations for monocular depth prediction," in *Advances Neural Inf. Process. Syst.*, 2020.
- [10] Z. Cheng, Z. Liu, T. Guo, S. Feng, D. Liu, M. Tang, and X. Zhang, "Badpart: Unified black-box adversarial patch attacks against pixel-wise regression tasks," in *ICML*, 2024.
- [11] Z. Cheng, J. Liang, H. Choi, G. Tao, Z. Cao, D. Liu, and X. Zhang, "Physical attack on monocular depth estimation with optimal adversarial patches," in *Proc. Eur. Conf. Comput. Vis.*, 2022.
- [12] C.-H. Ho and N. Nivasconcelos, "Contrastive learning with adversarial examples," in *Advances Neural Inf. Process. Syst.*, 2020.
- [13] M. Kim, J. Tack, and S. J. Hwang, "Adversarial self-supervised contrastive learning," in *Advances Neural Inf. Process. Syst.*, 2020.
- [14] K. Eykholz, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, "Robust physical-world attacks on deep learning visual classification," in *CVPR*, 2018.
- [15] "Stereoscopy," <https://en.wikipedia.org/wiki/Stereoscopy>.
- [16] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *CVPR*, 2017.
- [17] R. Garg, V. K. Bg, G. Carneiro, and I. Reid, "Unsupervised cnn for single view depth estimation: Geometry to the rescue," in *Proc. Eur. Conf. Comput. Vis.*, 2016.
- [18] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017.
- [19] J. Bian, Z. Li, N. Wang, H. Zhan, C. Shen, M.-M. Cheng, and I. Reid, "Unsupervised scale-consistent depth and ego-motion learning from monocular video," in *Advances Neural Inf. Process. Syst.*, 2019.
- [20] C. Wang, J. M. Buenaposada, R. Zhu, and S. Lucey, "Learning depth from monocular videos using direct methods," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018.
- [21] Z. Yin and J. Shi, "Geonet: Unsupervised learning of dense depth, optical flow and camera pose," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018.
- [22] M. Ramamonjisoa, M. Firman, J. Watson, V. Lepetit, and D. Turmukhambetov, "Single image depth prediction with wavelet decomposition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [23] N. Yang, L. v. Stumberg, R. Wang, and D. Cremers, "D3vo: Deep depth, deep pose and deep uncertainty for monocular visual odometry," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.
- [24] Y. Lu, Q. Wang, S. Ma, T. Geng, Y. V. Chen, H. Chen, and D. Liu, "Transflow: Transformer as flow learner," in *CVPR*, 2023.
- [25] R. Wang, S. M. Pizer, and J.-M. Frahm, "Recurrent neural network for (un-)supervised learning of monocular video visual odometry and depth," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019.
- [26] Y. Zou, P. Ji, Q.-H. Tran, J.-B. Huang, and M. Chandraker, "Learning monocular visual odometry via self-supervised long-term modeling," in *Proc. Eur. Conf. Comput. Vis.*, 2020.

- [27] L. Tiwari, P. Ji, Q.-H. Tran, B. Zhuang, S. Anand, and M. Chandraker, "Pseudo rgb-d for self-improving monocular slam and depth prediction," in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [28] J. Watson, O. Mac Aodha, V. Prisacariu, G. Brostow, and M. Firman, "The temporal opportunist: Self-supervised multi-frame monocular depth," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [29] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, "Digging into self-supervised monocular depth prediction," in *Proc. IEEE Int. Conf. Comput. Vis.*, October 2019.
- [30] J. Watson, M. Firman, G. J. Brostow, and D. Turmukhambetov, "Self-supervised monocular depth hints," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019.
- [31] A. Mathew, A. P. Patra, and J. Mathew, "Monocular depth estimators: Vulnerabilities and attacks," *arXiv preprint arXiv:2005.14302*, 2020.
- [32] J. Hu and T. Okatanai, "Analysis of deep networks for monocular depth estimation through adversarial attacks with proposal of a defense method," *arXiv preprint arXiv:1911.08790*, 2019.
- [33] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.
- [34] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Int. Conf. Learn. Representations*, 2018.
- [35] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016.
- [36] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Oakland S&P*, 2017.
- [37] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," in *Int. Conf. Learn. Representations*, 2018.
- [38] H. Zhang and J. Wang, "Towards adversarially robust object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019.
- [39] X. Chen, C. Xie, M. Tan, L. Zhang, C.-J. Hsieh, and B. Gong, "Robust and accurate object detection via adversarial learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [40] P.-C. Chen, B.-H. Kung, and J.-C. Chen, "Class-aware robust adversarial training for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [41] X. Xu, H. Zhao, and J. Jia, "Dynamic divide-and-conquer adversarial training for robust semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [42] W.-C. Hung, Y.-H. Tsai, Y.-T. Liou, Y.-Y. Lin, and M.-H. Yang, "Adversarial learning for semi-supervised semantic segmentation," *arXiv preprint arXiv:1802.07934*, 2018.
- [43] A. Arnab, O. Miksik, and P. H. Torr, "On the robustness of semantic segmentation models to adversarial attacks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018.
- [44] Y. Carmon, A. Raghunathan, L. Schmidt, J. C. Duchi, and P. S. Liang, "Unlabeled data improves adversarial robustness," in *Advances Neural Inf. Process. Syst.*, 2019.
- [45] J.-B. Alayrac, J. Uesato, P.-S. Huang, A. Fawzi, R. Stanforth, and P. Kohli, "Are labels required for improving adversarial robustness?" in *Advances Neural Inf. Process. Syst.*, 2019.
- [46] J. Tu, M. Ren, S. Manivasagam, M. Liang, B. Yang, R. Du, F. Cheng, and R. Urtasun, "Physically realizable adversarial examples for lidar object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.
- [47] J. Tu, H. Li, X. Yan, M. Ren, Y. Chen, M. Liang, E. Bitar, E. Yumer, and R. Urtasun, "Exploring adversarial robustness of multi-sensor perception systems in self driving," in *5th Annual Conference on Robot Learning*, 2021.
- [48] Y. Cao, C. Xiao, B. Cyr, Y. Zhou, W. Park, S. Rampazzi, Q. A. Chen, K. Fu, and Z. M. Mao, "Adversarial sensor attack on lidar-based perception in autonomous driving," in *CCS*, 2019.
- [49] Y. Cao, N. Wang, C. Xiao, D. Yang, J. Fang, R. Yang, Q. A. Chen, M. Liu, and B. Li, "Invisible for both camera and lidar: Security of multi-sensor fusion based perception in autonomous driving under physical-world attacks," in *Oakland S&P*, 2021.
- [50] "Camera Resectioning," https://en.wikipedia.org/wiki/Camera_resectioning.
- [51] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.
- [52] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer, "Adversarial patch," *arXiv preprint arXiv:1712.09665*, 2017.
- [53] G. Tao, G. Shen, Y. Liu, S. An, Q. Xu, S. Ma, P. Li, and X. Zhang, "Better trigger inversion optimization in backdoor scanning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [54] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, "Synthesizing robust adversarial examples," in *Proc. Int. Conf. Mach. Learn.*, 2018.
- [55] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, 2020.
- [56] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.
- [57] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018.
- [58] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [59] M. Ye, X. Zhang, P. C. Yuen, and S.-F. Chang, "Unsupervised embedding learning via invariant and spreading instance feature," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019.
- [60] I. Misra and L. v. d. Maaten, "Self-supervised learning of pretext-invariant representations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.
- [61] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [62] D.-H. Lee et al., "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *ICML Workshop*, 2013.
- [63] Z. Deng, L. Zhang, K. Vodrahalli, K. Kawaguchi, and J. Y. Zou, "Adversarial training helps transfer learning via better representations," *Advances Neural Inf. Process. Syst.*, 2021.
- [64] A. Petrovai and S. Nedevschi, "Exploiting pseudo labels in a self-supervised learning framework for improved monocular depth estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [65] A. Braunegg, A. Chakraborty, M. Krundick, N. Lape, S. Leary, K. Manville, E. Merkhofer, L. Strickhart, and M. Walmer, "Apricot: A dataset of physical adversarial attacks on object detection," in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [66] Z. Wu, S.-N. Lim, L. S. Davis, and T. Goldstein, "Making an invisibility cloak: Real world adversarial attacks on object detectors," in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [67] F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," in *Proc. Int. Conf. Mach. Learn.*, 2020.
- [68] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, "Square attack: a query-efficient black-box adversarial attack via random search," in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [69] J. Rauber, R. Zimmermann, M. Bethge, and W. Brendel, " Foolbox native: Fast adversarial attacks to benchmark the robustness of machine learning models in pytorch, tensorflow, and jax," *Journal of Open Source Software*, 2020.
- [70] R. Duan, X. Mao, A. K. Qin, Y. Chen, S. Ye, Y. He, and Y. Yang, "Adversarial laser beam: Effective physical-world attack to dnns in a blink," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [71] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *CVPR*, 2022.
- [72] R. Peng, R. Wang, Y. Lai, L. Tang, and Y. Cai, "Excavating the potential capacity of self-supervised monocular depth estimation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [73] K. Zhou, L. Hong, C. Chen, H. Xu, C. Ye, Q. Hu, and Z. Li, "Devnet: Self-supervised monocular depth learning via density volume construction," in *Proc. Eur. Conf. Comput. Vis.*, 2022.



Zhiyuan Cheng is currently a Ph.D. student at Purdue University, and he received his bachelor's degree in Computer Science and Technology from Wuhan University in 2019. His research interest lies at the intersection of AI and security, and he specializes in trustworthy machine learning. Specifically, he focuses on enhancing the robustness and security of AI-driven applications. He has publications in both top-tier security and machine learning conferences, such as NDSS, ICLR, ICML and ECCV.



Cheng Han received his bachelor's degree from Tianjin University (TJU) in 2019, and his master's degree from the Pennsylvania State University (PSU) in 2021. He is currently a Ph.D. candidate at the Rochester Institute of Technology (RIT), and an incoming assistant professor in the School of Science and Engineering at the University of Missouri – Kansas City (UMKC). His research interests include computer vision, deep learning, and adaptable and sustainable intelligence.



James Liang received his bachelor's degree in Mathematics and Statistics from the University of Illinois at Urbana-Champaign, in 2021. He is currently a Ph.D. candidate in the Department of Computer Engineering at the Rochester Institute of Technology and a student trainee at the U.S. Naval Research Laboratory. His research interests include computer vision, deep learning, and machine learning.



Qifan Wang received his Ph.D. degree in computer science from Purdue University in 2015. He is currently a Senior Staff Research Scientist at Meta AI, leading a team building innovative Deep Learning and Natural Language Processing models for Recommendation Systems. He has co-authored over 60 publications in top-tier conferences and journals, including NeurIPS, SIGKDD, WWW, SIGIR, AAAI, IJCAI, ACL, EMNLP, WSDM, CIKM, ECCV, TPAMI, TKDE, and TOIS.



Xiangyu Zhang is a Samuel Conte Professor at Purdue University, West Lafayette. He received his Ph.D. degree in computer science from the University of Arizona, Tucson, in 2006. His current research interest lies in AI security, software security, program analysis, and software engineering. He has published over 80 top-tier publications, including NDSS, USENIX Security, S&P, CCS, ICML, etc. He has received NSF Career Award and a few Best Paper Awards from top conferences as well.



Dongfang Liu is an assistant professor in the Department of Computer Engineering at the Rochester Institute of Technology (RIT). He earned his Ph.D. degree from Purdue University. Dr. Liu has dedicated his research efforts to developing AI-based solutions for interdisciplinary research to address challenges with societal relevance. His publication portfolio includes flagship conferences in the AI and robotics domains, such as CVPR, ICCV, ECCV, AAAI, IJCAI, WACV, WWW, and IROS.