# Revisiting Non-Parametric Matching Cost Volumes for Robust and Generalizable Stereo Matching

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

Stereo matching is a classic challenging problem in computer vision, which has recently witnessed remarkable progress by Deep Neural Networks (DNNs). This paradigm shift leads to two interesting questions that have not been addressed well. First, it is unclear whether stereo matching DNNs really learn to perform matching well. This paper studies this problem from the lens of adversarial attacks. It presents a method of learning stereo-constrained photometrically-consistent attacks, which by design are weaker adversarial attacks. State-of-the-art stereo matching DNNs are, however, vulnerable against them. This observation suggests that DNNs may not learn to perform matching well in the sense that they should otherwise achieve potentially even better after stereo-constrained perturbations are introduced. Second, stereo matching DNNs are typically trained under the simulation-to-real (Sim2Real) pipeline due to the data hungry of DNNs. Thus, alleviating the impacts of the Sim2Real photometric gap in stereo matching DNNs becomes a pressing need. Towards adversarially robust and domain generalizable stereo matching, this paper proposes to rethink the role of DNNs. It presents a method that casts stereo matching as a cost aggregation problem (solved by training a DNN) over a non-parametric cost volume (that truly focuses on matching) with parametric contextual features. In experiments, the proposed method is tested in the SceneFlow dataset, the KITTI2015 dataset, and the Middlebury dataset. It significantly improves the adversarial robustness, while retaining accuracy performance comparable to state-of-the-art methods. It also shows better Sim2Real generalizability.

## 1 Introduction

Stereo matching remains a long-standing problem in computer vision that has been studied for several decades. As shown in the left of Fig. 1, given a pair of (rectified) stereo images, the reference left image $I^L$ and the right image $I^R$, for a pixel $I^L(x, y)$, its corresponding pixel in $I^R$ is constrained to be at $I^R(x + \mathcal{D}(x, y), y)$, where $\mathcal{D}(x, y)$ is called the **disparity** of the pixel $(x, y)$ in the reference left image. The disparity is inversely proportional to the depth $\mathcal{Z}(x, y)$, $\mathcal{D}(x, y) = \frac{f \times b}{\mathcal{Z}(x,y)}$, where $f$ is the focal length of the left camera and $b$ is the baseline (i.e., the distance between the two cameras). High-performing stereo matching will enable highly reliable and cost effective depth estimation from stereo images without leveraging expensive Lidar sensors, which has great potential in a wide range of applications such as autonomous driving and robot autonomy. To infer the disparity, the basic approach is to first compute the matching costs between $I^L(x, y)$ and $I^R(x + d, y)$ for different values of $d$ and then find the best match that corresponds to the minimum cost. Due to occlusions and textureless regions, the pixel correspondences can not be solved individually and the disparity map needs to be inferred in its entirety via prior-constrained (e.g. local disparity smoothness) global matching cost optimization, which remains a challenging problem (see Appendix. A).

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

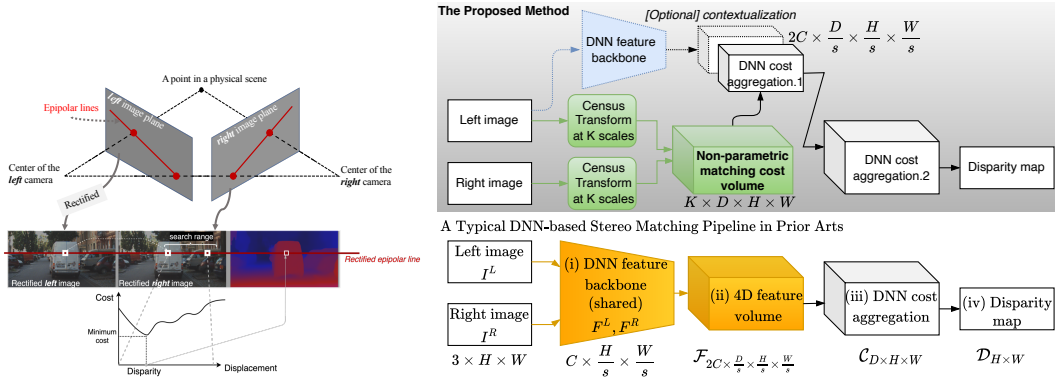


Figure 1: ***Left*:** Illustration of the stereo matching problem, which conventionally emphasizes the matching component between the (reference) left image, $I^L$ and the right image, $I^R$. ***Right-Bottom:*** A typical workflow of DNN-based stereo matching in prior arts, which often consists of four components (see text). ***Right-Top:*** The proposed method casts stereo matching as a cost aggregation/optimization problem (solved by training a DNN) over a non-parametric cost volume (that truly focuses on matching) with (optional) parametric contextual features from the reference left image only. The non-parametric cost volume using the census transform (CT) features [1] enables blocking the gradients in learning white-box adversarial attacks (or significantly eliminating the effects even when a differentiable approximation of CT is used). Due to the binary nature of CT, it also helps improve the Sim2Real generalizability. To handle occluded and/or textureless regions better, the proposed method also utilize the contextual features computed from the left image only. See text for details.

Stereo matching has recently witnessed remarkable progress using Deep Neural Networks (DDNs). As shown in the right-bottom of Fig. 1, DNN-based stereo matching methods amortize the need of inducing proper priors and of designing global matching cost optimization algorithms, which often consist of four components: (i) Extracting DNN features, $F^L$ and $F^R$ for matching with the spatial downsampling rate, $s$, (ii) Computing the 4D feature volume $\mathcal{F}$ by concatenating features $F^L(u, v)$ and $F^R(u + d, v)$ w.r.t. each disparity level $d \in [0, \frac{D}{s}]$ where $D$ is the predefined maximum disparity level, (iii) Computing and aggregating the matching cost volume $\mathcal{C}$ for each disparity level at the input resolution, which is typically realized via 3D convolution under an U-Net type of encoder-decoder architecture, and (iv) Estimating the final disparity map $\mathcal{D}$.

For the end-to-end fully differentiable learning of stereo matching in prior arts, one interesting question naturally arises: **How well do DNNs learn to perform matching from the component (ii) to (iii)**? We can do some deductive reasoning: If they indeed learn to performance matching, their performance should potentially increase after we add the same perturbations at $I^L(x, y)$ and $I^R(x + \mathcal{D}(x, y), y)$ using the ground-truth disparity $\mathcal{D}(x, y)$ (i.e., the stereo-constrained photometric consistency), in regardless of what the perturbations are (e.g., adversarial attacks). As we shall show, DNNs do not learn to perform matching well and state-of-the-art stereo matching DNNs are vulnerable even when the photometric consistency is preserved against adversarial attacks (Section 2). Since stereo matching methods are widely used in autonomous driving, adversarial vulnerabilities in these models can lead to catastrophic consequences, and thus addressing adversarial vulnerabilities has become a pressing need in deep learning.

To defend against adversarial attacks, most methods rely on adversarial training [2], often at the expense of decreasing performance on clean images, long training time, and potential over-fitting to specific attacks and datasets (*e.g.* cannot transfer the robustness to other datasets as shown in the later sections). In contrast, **we propose to utilize domain-specific knowledge to facilitate the built-in robustness of the neural networks.** Because of the strong *photometric consistency between stereo images*, stereo matching provides an ideal case to defend against adversarial attacks through the design of the neural network (see our proposed method in the right-top of Fig. 1, and Fig. 5 and Table 9 in the Appendix).

**For non-occluded regions** in stereo images, the corresponding pixels of the same physical point have similar colors. We observe that by using DNN features for matching, attacks will increase the matching costs for features that belong to the same physical point (see the appendix for details). Therefore, we propose to remove DNN features for matching and use hand-crafted features that will preserve the color differences to construct the cost volume, then the DNN is used to aggregate/optimize a non-parametric cost volume. To further make the cost as hard to alter as possible, we use local binary patterns that compare each pixel intensity to their neighbors (i.e., Census Transform [1, 3])

as the feature descriptor. **For occluded regions,** the photometric consistency does not hold. **For textureless non-occluded regions**, the local photometric features are weak for matching, even though the underlying consistency is retained. Thus, high-level semantic information learned by DNN features will be entailed. We use a DNN feature backbone for the reference (left) image only to contextualize the feature volume, which turns out to be helpful for improving adversarial robustness. This shows that the majority of the vulnerability actually comes from the matching part rather than the end-to-end trainable contextual information.

The non-parametric feature volume and the contextual DNN features will be fed through a head sub-network playing the role of a learnable optimizer that seeks the best matching result (Fig. 1). In essence, we cast stereo matching as a cost aggregation/optimization problem over a non-parametric cost volume with parametric contextual features. In experiments, we show that this more transparent approach improves adversarial robustness significantly while maintaining high accuracy.

Parallel to the adversarial vulnerability, **cross-domain generalizabilty** also is an important problem in stereo matching: DNN-based stereo matching is typically pre-trained under the so-called simulation to real (Sim2Real) pipeline due to the high cost of collecting ground-truth matching results in practice and the data-hungry aspect of DNNs. It has been shown that DNNs may learn shortcut solutions that are strongly biased by the training dataset [4]. Removing the DNN feature backbone for matching and utilizing the binary patterns of CT will induce the DNN to be a more general cost volume optimizer, thus alleviating the opportunity of shortcut learning in the feature space and resulting in better performance in cross-domain deployment, especially when no fine-tuning is used. These are verified in our experiments from the SceneFlow dataset [5] to the KITTI benchmark [6] and the MiddleBury dataset [7] when no fine-tuning is used.

With a comprehensive study of the proposed method and thorough comparisons with prior arts in experiments (Section 4), **we have some interesting observations as follows**: (1) By removing feature backbones (i.e., (ii) in the right-bottom of Fig. 1) that are trained on certain datasets and using non-parametric cost volumes as inputs, we can improve the cross-domain genearlizability through training the DNN as a general optimizer. (2) Even without contextual information (i.e., the component shown in dashed arrows and boxes in the right-top of Fig. 1), DNNs can do reasonably well on aggregating/optimizing the raw cost volume, which is nontrivial due to the loss of image information. (3) The vulnerability mostly comes from the matching part instead of the learned contextual information. (4) Since the vanilla CT can block the gradients in learning attacks, the obfuscated gradient problem [8] of CT is studied using a differentiable approximation. With this approximation, the overall DNN output can still be altered, showing the internal vulnerability of DNNs. (5) The robustness of adversarial training in stereo matching does not transfer well to other datasets or attacks, thus inducing the built-in robustness is very important, as done by the CT based non-parameteric cost volume in our proposed method (Section 3.2).

**Our Contributions.** This paper makes two main contributions to the field of stereo matching: (i) It proposes a novel design for stereo matching by utilizing DNNs to aggregate/optimize non-parametric cost volumes with parametric contextual features, which shows significantly better adversarial robustness and improved cross-domain (Sim2Real) generalizability when no-fine tuning is used. (ii) It presents the stereo-constrained projected gradient descent (PGD) attack method, which by design preserves photometric consistency to show the more serious vulnerabilities of state-of-the-art DNN-based stereo matching methods.

# 2 Attacking Stereo Matching Deep Neural Networks

## 2.1 The Proposed Stereo-Constrained PGD Attacks

To study the brittleness of DNN based stereo matching models, we intentionally develop a physically realizable attacking method based on the widely-used powerful PGD method [2], which retains the underlying photometric consistency in stereo matching by changing the intensities of the same physical point in both images. By design, it is a weaker attack method compared with the unconstrained counterpart. As we shall show, its strong attack performance for state-of-the-art stereo matching methods clearly calls for alternative designs of stereo matching networks that exploit the nature of the stereo matching for better robustness.

In learning the proposed stereo-constrained attacks, the same amounts of perturbations are added to each pair of correspondence pixels in the left and right images simultaneously while occluded areas will not be modified. Since the left image is the reference image for computing the disparity

loss, we disallow to attack and evaluate occluded regions of the reference image, which prevents the perturbation to attack the regions where the estimation does not rely on matching.

Given a perturbation map $P(x, y), (x, y) \in \Lambda$ ($\Lambda$ denotes the image lattice), the distorted intensities for each pixel location $(x, y)$ are computed as:

$$\begin{aligned} I^L_{adv}(x, y) &= I^L(x, y) + P(x - \mathcal{D}(x, y), y), \\ I^R_{adv}(x, y) &= I^R(x, y) + P(x, y), \end{aligned} \tag{1}$$

where $\mathcal{D}(x, y)$ is the ground-truth disparity map. Intuitively, this is a weaker attack since we actually leak the underlying disparity information to the algorithm using the pair $(P(x, y), P(x - \mathcal{D}(x, y), y))$ which may potentially help the stereo matching, instead of attacking it. Consider two corresponding patches on the left and the right images containing the same physical points, the absolute sum of difference between these two patches will remain the same after the attack. The proposed stereo-constrained attacks uses the ground-truth disparity map in generating the perturbations, which is designed (i) to understand better the underlying vulnerability in stereo matching methods, and (ii) to simulate the physical scenarios, as the adversarial stereo-constrained patch attack experiments shown in Section 2.2.

We use the $L_\infty$ norm to measure similarities between images. Two images will appear visually identical under a certain threshold. To learn a $L_\infty$ bounded adversarial perturbation $P^{adv}$, the iterative PGD method is used,

$$P^{adv}_{t+1} = clip^{\epsilon}_{P}\{P^{adv}_t + \alpha \cdot sign(\nabla_P \ell(P^{adv}_t))\}, \tag{2}$$

where $t = 0, 1, 2, \cdots, T$ and $P^{adv}_0$ starts with all zeros. $\ell(\cdot)$ denotes the mean absolute error for a perturbation $P^{adv}_t$ between the predicted disparity map for the perturbed images (Eqn. 1) and the ground-truth disparity map. And, $clip^{\epsilon}_{P}$ clips the perturbation to be within the $\epsilon$-ball of the corresponding zero-plane and the maximum color range. In our experiments, we set $\epsilon = 0.06$ or $0.03$ (the larger it is, the stronger the attack is), $\alpha = 0.01$ and $T = 20$. Appendix C.1 shows a toy example.

| | EPE | | | | Bad 1.0 | | | | Bad 3.0 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | CL | CT-0.03 | CT-0.06 | UCT-0.03 | CL | CT-0.03 | CT-0.06 | UCT-0.03 | CL | CT-0.03 | CT-0.06 | UCT-0.03 |
| PSMNet [9] | 0.28 | 29.05 | 84.04 | 91.08 | 2.00 | 84.75 | 90.41 | 92.75 | 0.16 | 54.80 | 83.68 | 89.91 |
| GANet-Deep [10] | **0.25** | **3.93** | **9.75** | 23.75 | **1.42** | **70.64** | 84.68 | 89.48 | **0.10** | 29.94 | 68.70 | 79.11 |
| LEAStereo [11] | 0.37 | 4.02 | 11.38 | **14.71** | 4.54 | 71.20 | **83.24** | **82.42** | 0.42 | **29.09** | **63.61** | **64.31** |
| PSMNet + adv. | 0.46 | 0.70 | 1.02 | 1.06 | 8.04 | 17.78 | 33.54 | 36.50 | 0.66 | **1.40** | **3.08** | **4.14** |
| GANet + adv. | **0.42** | **0.65** | **0.98** | **1.05** | **6.47** | **14.99** | **28.56** | **31.22** | **0.63** | **1.40** | 3.76 | 4.36 |
| LEAStereo + adv. | 0.51 | 0.81 | 1.23 | 1.30 | 9.89 | 21.73 | 38.72 | 42.06 | 0.99 | 2.34 | 5.59 | 6.07 |

Table 1: State-of-the-art stereo matching methods have catastrophic performance drop w.r.t. PGD attacks, and adversarially trained versions can not counter the drop. The results are tested in the KITTI2015 training dataset [6]. For each metric (Section 4), the four columns show that metric on *CL*ean images, stereo-constrained attacked images (*CT*, $\epsilon = 0.03$, and *CT*, $\epsilon = 0.06$), and unconstrained attacked images (*UCT*, $\epsilon = 0.03$).

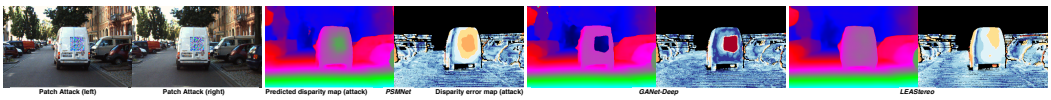


Figure 2: Illustration of the physical realizability of the proposed stereo-constrained attack method using adversarial patch attacks. *Best viewed in color and magnification.*

## 2.2 Attacking Results in KITTI2015

We test the proposed stereo-constrained PGD attack method and the unconstrained counterpart using three state-of-the-art methods in the KITTI2015 benchmark dataset.

Table 1 shows the results. Since different methods use different training and validation splits when finetuning in the KITTI benchmark, the performance in Table 1 are computed on the entire dataset (200 images in total). Both the stereo-constrained attacks and the unconstrained ones can lead to catastrophic drop of performance. Such vulnerability may raise serious concerns for the deployment of DNNs in safety critical applications. The three methods cover DNN architectures by both sophisticated manual design and differentiable neural architecture search. This clearly shows alternative approaches worth exploring. As we shall show, the proposed network is a promising one. We can see that the stereo-constrained attacks are indeed relatively weaker attacks, but more physically realizable than the unconstrained ones in stereo matching settings.

**The Physical Realizability of the Proposed Stereo-Constrained Attack.** To test if the adversarial vulnerability can be intentionally exploited in a more realistic setting, such as autonomous driving,

we constructed the patch attack experiment to demonstrate the possibility of such attempts. We select 10 scenarios where $40 \times 40$ adversarial patches can be put on more flat surfaces (Fig. 2, details and all images are provided in Appendix C.2). To preserve the depth of the scene, the ground truth disparities of the patches are the same as the corresponding part of the original image. For each image pair, we apply stereo-constrained PGD attacks with $\epsilon = 0.03$ and $T = 100$. Table 2 shows the results supporting the physical realizability.

| | Clean | | | Synthetic Patch Attack | | |
|---|---|---|---|---|---|---|
| Models | EPE | Bad 1.0 | Bad 3.0 | EPE | Bad 1.0 | Bad 3.0 |
| PSMNet | 0.28 | 2.30 | 0.21 | **0.80** | 15.41 | **4.46** |
| GANet-Deep | **0.25** | **1.46** | **0.12** | 1.84 | **12.52** | 9.09 |
| LEAStereo | 0.41 | 5.60 | 0.69 | 1.34 | 20.32 | 6.18 |
| PSMNet + adv. | 0.58 | 12.56 | 1.44 | **1.44** | 26.55 | **8.59** |
| GANet + adv. | **0.55** | **11.24** | **1.32** | 2.31 | **23.5** | 11.98 |
| LEAStereo + adv. | 0.71 | 15.87 | 2.24 | 1.96 | 30.88 | 11.72 |

Table 2: Adversarial patch attack results in the KITTI2015 training dataset with photometric consistency retained in attack. Note that the errors are computed using the whole image, while only a small portion of the image is attacked.

### 2.3 Adversarial Training

To further support the pressing need of exploring alternative networks towards robust stereo matching, we show that prior arts reinforced by adversarial training still do not perform sufficiently well against the attacks. Table 1 and Table 2 show the results of attacks at the image level and the patch level respectively. While being relatively more robust after adversarial training, the performance drop on clean images. The proposed robust stereo matching method will significantly improve this.

To compare with adversarial training, we fine-tune each method on the `KITTI2015` training images perturbed by 3-step unconstrained PGD attacks for 20 epochs [1], denoted as **+adv.** in tables.

## 3 The Proposed Stereo Matching Network

Strong attacking results on state-of-the-art stereo matching methods and the lack of sufficient correction via adversarial training in Section 2 motivate us to explore alternative design of stereo matching networks that enjoy built-in robustness. In this section, we present the details of the proposed method.

### 3.1 Harnessing the Best of Classic Designs and DNNs

Fig. 1 illustrates the proposed workflow consisting of four main components:

**Extracting Features and Computing the Feature Volume Using Multi-Scale Census Transform.** Most current stereo matching methods use DNN-based features to form the 4D feature volume. In terms of matching, DNNs can increase the uniqueness of the feature for each pixel, but they also suffer from the inherent adversarial vulnerability. In contrast, traditional methods often use simple window-based similarity functions to initialize the costs, then rely on the optimization or cost aggregation stage to integrate all local cost information [12]. Following the same philosophy, we propose to use hand-crafted feature descriptors and similarity functions that are less sensitive to adversarial perturbations to initialize the costs, then rely on DNNs to integrate the local cost information. Specifically, we want the feature descriptor to change as little as possible when local intensities are perturbed. This specific requirement lead us to the Census Transform (CT), a traditional feature descriptor that is developed to eliminate the issue of radiometric differences caused by different exposure timing or non-Lambertian surfaces. Previous studies find that CT is the most robust and well-rounded cost function with global or semi-global methods [1, 3].

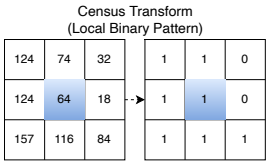


Figure 3: Illustration of the CT using a $3 \times 3$ input patch.

We use grey-scale raw intensity values in computing the census transform. Given a local window patch $W$ centered at a pixel $u \in \Lambda$ (Fig. 3), the census transform computes the local binary pattern by comparing each neighboring pixel $v \in W$ with $u$ such that it equals 1 if $I(v) >= I(u)$ and 0 otherwise. Hamming Distance (i.e. the number of different values in two bit strings) is used to compute the cost between two patches.

Unlike in traditional semi-global or global methods in which the cost of each pair can only be a scalar, we take advantage of the flexibility of DNNs and design the multi-scale census transform to incorporate the context at different scales. We use local windows with sizes from $k_1$ to $k_2$ (e.g. $k_1 = 3, k_2 = 11$ in our experiments) so there are $K = k_2 - k_1 + 1$ costs associated with each matching candidate pairs. To normalize the cost at each scale, we divide the Hamming Distance by the number of pixels of each local window. To reduce the overall computational cost and the memory

[1] Due to the time-consuming adversarial training from scratch and the concerns of reproducing results of the prior arts by retraining them from scratch, we use this post-hoc adversarial training strategy.

footprint, we use 3D convolutions to down-scale the cost volume to $1/3$ of its spatial size. Details of the architecture are included in the Appendix B.

**Contextualizing the Feature Volume and Computing and Aggregating the Cost Volume.** Although being robust to adversarial attacks, the census transform based feature volume alone is not sufficiently powerful to handle occlusion, textureless (flat) regions, and more challenging semantic information, such as transparent objects and specular reflections. We introduce a 2-stack Hourglass [13] module with 2D convolutions to extract context information from the left reference image, resulting in a $H_1 \times W_1 \times C$ feature map which is unsqueezed along the second dimension (i.e., copying the feature map $D_1$ times ) to form a same size tensor as the down-scaled CT feature volume. The two are then concatenated along the second last dimension. The contextualized feature volume will be fed into an encoder-decoder for the cost volume computation and aggregation stage, $\mathcal{C}_{H_0 \times W_0 \times D}$, which is realized by a 3-stack Hourglass [13] module with 3D convolutions.

If no feature backbone is used to extract context information from the left image (Fig. 1), our model will be denoted as **"ours w/o ctx."** in tables and figures.

**Disparity Map Prediction.** To predict the final disparity map $\mathcal{D}(x, y), \forall (x, y) \in \Lambda$, the output of each module in the decoder (i.e., a stack in the Hourglass module) of the cost aggregation is first up-sampled to the original size $H_0 \times W_0 \times D$, denoted as $\mathcal{D}_s(x, y, d)$ where $s$ is the stack index. Then, similar to the method used in [14], the predicted disparity map $\mathcal{D}_s(x, y)$ is computed by,

$$\mathcal{D}_s(x, y) = \sum_{d=1}^{D} d \times Softmax(\mathcal{D}_s(x, y, d)), \tag{3}$$

where $Softmax$ is applied along the last dimension in $\mathcal{D}_s$.

**The Loss Function.** In training, we use the smooth $L_1$ loss, $\ell_{L_1}(z) = \frac{z^2}{2}, (\text{if } z < 1); |z| - 0.5, (\text{otherwise})$ due to its robustness at disparity discontinuities and low sensitivity to outliers [9, 10]. Given the ground-truth disparity map $\mathcal{D}^*(u)$, the loss is defined by,

$$\mathcal{L}(\Theta; \mathcal{D}^*) = \sum_{s=1}^{S} \beta_s \cdot \frac{1}{|\Lambda|} \sum_{\mathbf{x} \in \Lambda} \ell_{L_1}(\mathcal{D}_s(\mathbf{x}) - \mathcal{D}^*(\mathbf{x})), \tag{4}$$

where $\Theta$ collects all parameters in our model, $\beta_s$ represents the weight for the output from a stack $s$ (e.g., 0.5, 0.7, and 1 are used for the 3-stack Hourglass module in our experiments), $\mathbf{x} = (x, y) \in \Lambda$.

### 3.2 Attacking Census Transform and Its Built-in Robustness

Since Census Transform contains the non-differentiable comparison operator, the gradients from the constructed cost volume are blocked to the input images thus leading to an illusion of safety, *i.e.* the obfuscated gradient problem [8]. For fair comparisons with differentiable methods, we combine subtraction and the sigmoid function as a differentiable approximation of the comparison operator.

$$a > b \approx sigmoid(a - b) \cdot C \tag{5}$$

We use a large constant (*i.e.* $C = 10^5$) such that the output is close to either zero or one.

**Built-in Robustness of Census Transform.** Without using this differentiable approximation, our method without the contextual feature backbone will be ***unattackable*** since the gradient flows are completely blocked. From the perspective of attacks, the binary patterns generated by Census Transform is more difficult to alter due to the comparison operator. Given a threshold of maximum pixel difference in perturbation $\epsilon$, *neighbors will not be altered if their difference with the center is larger than* $2\epsilon$. Let $CT(u, v)$ denote the local binary value at $u$ with the neighbor $v$, $I$ be the image, and $CT'$ be the binary pattern after the perturbation, we have:

$$CT'(u, v) = CT(u, v), \quad \text{if } |I(u) - I(v)| > 2\epsilon \tag{6}$$

If the attack does not violate photometric consistency, it will be even harder to alter the cost between binary patches of corresponding pairs. Specifically, if a neighboring pixel appears in both the left and the right binary patches, its relative magnitude relationship with the center pixel will be the same for both patches, no matter how its intensities change. It is our interest to test if this highly non-linear operator can defend the DNNs against attacks.

Due to its binary nature, the cost curves will be less affected by the different color distributions of different datasets, thus improving the Sim2Real generalizability of the proposed stereo matching network when no finetuning is used.

| Models (trained on SceneFlow) | KITTI 2015 | | | KITTI 2012 | | | Middlebury | | |
|---|---|---|---|---|---|---|---|---|---|
| | EPE | Bad 1.0 | Bad 3.0 | EPE | Bad 1.0 | Bad 3.0 | EPE | Bad 1.0 | Bad 3.0 |
| PSMNet | 6.89 | 72.93 | 31.55 | 5.90 | 71.59 | 28.42 | 4.33 | 73.01 | 19.01 |
| GANet | 1.66 | 42.12 | 10.48 | 1.48 | 31.61 | 9.51 | 2.26 | 27.45 | 11.40 |
| LEAStereo | 2.00 | 51.29 | 13.90 | 1.91 | 44.26 | 14.28 | 3.47 | 32.67 | 14.81 |
| **Ours w/o ctx** | **1.25** | **25.95** | **6.12** | **1.23** | **19.66** | **6.80** | **1.71** | **18.72** | **9.16** |
| Ours | 1.26 | 27.92 | 6.31 | 1.28 | 20.62 | 7.16 | 1.96 | 20.09 | 10.05 |

Table 3: Comparisons for the Sim2Real cross-domain generalizability from the SceneFlow trained models to the KITTI 2015, KITTI 2012 and Middleburry datasets in testing without any fine-tuning.

| | EPE | | | | Bad 1.0 | | | | Bad 3.0 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | CL | CT-0.03 | CT-0.06 | UCT-0.03 | CL | CT-0.03 | CT-0.06 | UCT-0.03 | CL | CT-0.03 | CT-0.06 | UCT-0.03 |
| Ours w/o ctx. | 0.38 | 1.13 | 1.43 | 2.36 | 4.14 | 24.64 | 30.69 | 41.34 | 0.32 | **2.46**(↑26.63) | 8.05 | 16.30 |
| Ours | **0.36** (↓0.11) | **0.88** (↑2.05) | **1.16**(↑8.59) | **1.81**(↑12.90) | **3.61**(↓2.19) | **21.20**(↑49.44) | **29.19**(↑54.05) | **36.42**(↑46) | **0.27**(↓0.17) | 3.75 | **6.17**(↑57.44) | **11.29**(↑53.02) |
| Ours w/o ctx. + adv. | 0.42 | 0.78 | 0.90 | 1.26 | 5.95 | 16.83 | 21.42 | 32.27 | 0.73 | 2.88 | 3.83 | 7.51 |
| Ours + adv. | **0.41**(↑0.01) | **0.61**(↑0.04) | **0.69**(↑0.27) | **0.88**(↑0.17) | **5.77**(↑0.7) | **13.46**(↑1.53) | **16.29**(↑12.27) | **22.93**(↑8.29) | **0.52**(↑0.11) | **1.39**(↑0.01) | **2.00**(↑1.08) | **3.99**(↑0.15) |

Table 4: Stereo-constrained and unconstrained PGD attack results in the KITTI2015 training dataset [6]. Compared with the best result (in bold) of each metric by prior arts in Table 1, without adversarial training, the proposed method obtains slightly worse performance (as shown by ↓) on clean images, but achieves significantly better adversarial robustness (as shown by the ↑). With adversarial training, the proposed method consistently outperforms prior arts on both clean images and attacked images.

# 4 Experiments

We first present the results on the Sim2Real cross-domain generalizability, followed by showing results on the adversarial robustness. The proposed method significantly improves the robustness and the Sim2Real no-finetuning generalizability in comparison with prior arts. **Our PyTorch source code is provided in the supplementary material.** More implementation details are in the appendix.

*Data.* We evaluate our method on the `SceneFlow` [5] and `KITTI2015` [6] datasets. We also test pretrained models on the `KITTI2012` [15] and the `Middlebury` [7] dataset at quarter resolution.

*Evaluation Metrics.* We adopt the provided protocols in the two datasets. There are three metrics: **EPE [px]** which measures the end-point error in pixels, **Bad 1.0 [%]** and **Bad 3.0 [%]** which represents the error rate of errors larger than 1 pixel and 3 pixels respectively.

*Baseline Methods.* We compare with state-of-the-art deep stereo matching methods: the PSMNet [9], the GANet [10], and the LEAStereo [11]. We use their publicly released codes and trained model checkpoints in comparisons.

## 4.1 The Sim2Real Cross-Domain Generalizability

To verify the conjecture that cross-domain generalizability in stereo matching can be induced by removing the dependency between the cost volume computation and the dataset-dependent feature backbone, we evaluate all models pretrained on SceneFlow directly on the KITTI training datasets and the Middlebury training dataset [7]. As shown in Table 3, **our method outperforms prior art by a large margin**. This result shows that our proposed design of combining a non-parametric cost volume formed by the multi-scale census transform and a generalized cost aggregation/optimization DNN is indeed more cross-domain consistent. It also shows that the head sub-network DNN indeed learns to play the role of a domain-independent optimizer over a given cost volume.

## 4.2 The Adversarial Robustness

**Attacks at the image level.** To evaluate the adversarial robustness in KITTI2015, we directly test the trained models on the entire training dataset (200 images). Due to the GPU memory limitation, we only use the $240 \times 384$ center part of each image. Because of cropping, we also ignore those pixels where their correspondences are outside of the cropped images.

Table 4 shows the results. Compared with prior arts (see Table 1), **our method shows significantly better robustness on both stereo-constrained and unconstrained attacks.** We note that although the approximation used in attacking the CT is highly non-linear (Eqn. 5), adversarial attacks can still find ways to perturb the input images, which further demonstrate the vulnerability of the DNNs. As aforementioned, our method with the contextualized feature volume is more robust than its counterpart, showing that **the majority of the vulnerability actually comes from the matching part rather than the contextual information.**

**Attacks at the patch level.** Table 5 shows the comparisons. Fig. 4 shows an example. *Our method is very robust against adversarial patches.* In contrast, other methods perform poorly, even with adversarial training (see Table 2).

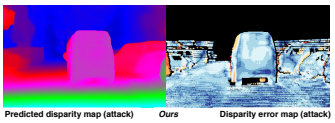


Figure 4: An example of adversarial patch attack on our method. Compared with Fig. 2, it is much more robust.

| Models | Clean EPE | Clean Bad 1.0 | Clean Bad 3.0 | Synthetic Patch Attack EPE | Synthetic Patch Attack Bad 1.0 | Synthetic Patch Attack Bad 3.0 |
|---|---|---|---|---|---|---|
| Ours w/o ctx. | 0.46 | 5.51 | 0.63 | **0.48**(↑0.86) | **6.42**(↑6.1) | **0.81**(↑5.37) |
| Ours | **0.40**(↓0.15) | **4.79**(↓3.33) | **0.46**(↓0.34) | 0.54 | 7.58 | 0.85 |
| Ours w/o ctx. + adv. | **0.52**(↑0.03) | **8.95**(↑2.29) | **1.03**(↑0.29) | **0.54**(↑1.42) | **9.45**(↑14.05) | **1.18**(↑7.41) |
| Ours + adv. | 0.53 | 11.25 | 1.17 | 0.58 | 12.01 | 1.40 |

Table 5: Adversarial stereo-constrained patch attack results in the KITTI2015 training dataset. Numbers in the bracelets with ↑ or ↓ are the difference between the best results in Table 2.

| Models | SceneFlow EPE [px] | SceneFlow Bad 3.0 [%] | KITTI15 (Sim2Real) EPE [px] | KITTI15 (Sim2Real) Bad 3.0 | KITTI15 Attack ($\epsilon = 0.03$) EPE [px] | KITTI15 Attack ($\epsilon = 0.03$) Bad 3.0 [%] |
|---|---|---|---|---|---|---|
| multi-scale SAD | 1.02 | 4.02 | 1.71 | 9.69 | 2.30 | 18.20 |
| CT (w=11) | 1.18 | 4.77 | 1.28 | 6.38 | 1.88 | 7.22 |
| Ours w/ feat. | 0.85 | 3.75 | 3.25 | 26.76 | 2.13 | 15.23 |
| Ours w/o ctx. | 1.10 | 4.40 | **1.25** | **6.12** | 1.13 | **2.46** |
| Ours | **0.84** | **3.70** | 1.26 | 6.31 | **0.88** | 3.75 |

Table 7: Comparison between single-scale CT and multi-scale SAD.

| Models | Non-Occlusion FG | Non-Occlusion Avg All | All Areas FG | All Areas Avg All |
|---|---|---|---|---|
| GCNet [14] | 5.58 | 2.61 | 6.16 | 2.87 |
| PSMNet [9] | 4.31 | 2.14 | 4.62 | 2.32 |
| GANet-15 [10] | 3.39 | 1.84 | 3.91 | 2.03 |
| GANet-Deep [10] | **1.34** | 1.63 | **1.48** | 1.81 |
| LEAStereo [11] | 2.65 | **1.51** | 2.91 | **1.65** |
| Ours | 3.54 | 2.09 | 4.16 | 2.39 |

Table 8: KITTI2015 leaderboard using the Bad 3.0 [%] metric in the protocol.

### 4.3 Improvement via Adversarial Training

As shown in Table 4, our method without adversarial training shows comparable adversarial robustness with $\epsilon = 0.03, 0.06$, especially for EPE and Bad 1.0. In comparisons, for prior arts, as shown in Table 1, the difference between with and without adversarial training are significantly larger.

For the patch attack experiment, our method w/o adversarial training is already much more robust than prior arts with adversarial training (see Tabel 5). With adversarial training, our method has a stronger robustness than all other methods, showing that our approach is orthogonal to adversarial training and they could be jointly used to further improved robustness.

**Transferability of the reinforcement.** To show the importance of built-in robustness, we also test on KITTI2012 using models trained on KITTI2015 to see how the adversarial robustness generalize on unseen data. In Table. 6, our method shows a stronger cross-domain adversarial robustness than other adversarially trained methods. Similarly, our method with adversarial training is the most robust over all methods.

| Models | Clean EPE | Clean Bad 1.0 | Clean Bad 3.0 | PGD Attack ($\epsilon = 0.03$) EPE | PGD Attack ($\epsilon = 0.03$) Bad 1.0 | PGD Attack ($\epsilon = 0.03$) Bad 3.0 | PGD Attack ($\epsilon = 0.06$) EPE | PGD Attack ($\epsilon = 0.06$) Bad 1.0 | PGD Attack ($\epsilon = 0.06$) Bad 3.0 |
|---|---|---|---|---|---|---|---|---|---|
| PSMNet + adv. | 0.8 | 18.24 | 3.03 | 1.37 | 33.87 | 6.72 | 2.07 | 51.00 | 12.28 |
| GANet + adv. | 0.80 | 18.82 | 3.03 | 1.45 | 36.79 | 7.75 | 2.29 | 54.07 | 15.42 |
| LEAStereo + adv. | 0.85 | 20.17 | 3.56 | 1.48 | 38.24 | 8.60 | 2.3 | 55.67 | 16.24 |
| Ours w/o ctx. | 0.49 | 6.40 | 1.23 | 1.58 | 26.69 | 9.66 | 1.92 | 32.52 | 12.87 |
| Ours | 0.46 | 6.07 | 1.00 | 1.37 | 22.45 | 7.29 | 1.80 | 29.61 | 10.50 |
| Ours w/o ctx. + adv. | 0.65 | 11.18 | 2.25 | 1.28 | 24.62 | 6.64 | 1.48 | 28.93 | 8.31 |
| Ours + adv. | **0.56** | **10.14** | **1.73** | **0.89** | **18.35** | **4.14** | **1.09** | **22.64** | **5.95** |

Table 6: Transferrability of Adversarial Robustness: stereo-constrained 20-step PGD Attack Results in the KITTI2012 training dataset using adversarially trained neural networks on KITTI2015.

### 4.4 Ablation Study

**How important is the non-parametric cost volume?** To check whether the importance of the non-parametric cost volume can be justified, we modify our model by replacing the CT component with the concatenation of normal DNN features from the left and right images, while keeping other components unchanged (denoted by **"ours w/ feat."**). We compare with traditional Sum of Absolute Difference (SAD) and CT with a single scale. Table 7 shows that the ones with non-parametric cost volumes generalize much better than the one with DNN features. The contextualized CT feature volume is consistently better than the DNN feature volume.

**Why using the CT?** It is chosen due to its non-differentiability and the fact that it is a well-rounded choice in the literature. We use multi-scale representations to respect the common recognition of its expressivity, and to alleviate choosing window size as a dataset-dependent hyper-parameter. From Table 7, we can see that CT is indeed much more robust than SAD due to its non-differentiability. CT with multiple scales has a stronger robustness than the single-scale version, while having a slightly better accuracy due to its flexibility.

### 4.5 KITTI Leaderboard Comparisons

Table 8 shows the comparisons. Our method is competitive compared to other state-of-the-art methods, while having a much stronger adversarial robustness and cross-domain generalizability.

## 5 Related Work

**Deep Stereo Matching.** After the first deep learning approach for stereo matching was proposed in [16], the first end-to-end trainable DNN-based method (DispNet) was developed by [5], together with the synthetic SceneFlow dataset. The GCNet [14] further extends the end-to-end approach by concatenating features in the cost volume stage, using 3D convolutional layers for cost aggregation, and introducing the soft $\arg\min$ operator to compute the expected disparity. Most subsequent approaches followed these design choices and use SceneFlow for pretraining [6].

The cost aggregation stage was further studied in [9, 17, 18] using the Spatial Pyramid Pooling module for feature extraction and the stacked Hourglass structures [13]. In [19], fast stereo matching is studied by by building the cost volume purely using highly optimized hand-crafted features (e.g. Census Transform and Sum of Absolute Differences) at the expense of accuracy performance. They did not study adversarial attack and defense, and the Sim2Real generalizability, which are the focuses of this paper. Spatial cost propagation layers are studied in [10, 20] to reduce the number of 3D convolutional layers. Neural Architecture Search techniques are used in [11] to automatically find optimal architectures for each stage and further improve the performance. These approaches are the current state-of-the-art in the KITTI 2015 benchmark [6]. However, they significantly suffer from the proposed adversarial attacks.

**Adversarial Attacks and Defense.** Assuming full access to DNNs pretrained with clean images, white-box targeted attacks are powerful ways of investigating the brittleness of DNNs. In autonomous driving, although physically realizable attacks are investigated in many tasks [21, 22, 23, 24, 25], attacking stereo matching has not been well studied. [26] show that DNN-based stereo matching methods are vulnerable against unconstrained adversarial attacks on both images separately. Without enforcing photometric consistency, these attacks will violate the underlying physical properties of binocular vision and thus are not realizable in practice. For example, unconstrained attacks cannot compute adversarial patches to fool stereo systems. They also only studied the relatively simpler FGSM attacks [27]. To find out whether stereo matching methods are indeed vulnerable in a physically realizable setting, we propose the stereo-constrained projected gradient descent (PGD) attack [2] and show that state-of-the-art methods are vulnerable even when the color differences between corresponding pixels are preserved. [22] studied adversarial patch based attacks in optical flow, which is inherently different to attacking stereo matching due to the underlyingly different matching nature of the problems. They did not study adversarial defense for optical flow methods.

Towards defense, adversarial training is the most widely used method to improve adversarial robustness [2, 28]. However, it also suffers from the disadvantages of dropping accuracy, long training time, and over-fitting to specific attacks and datasets. While adversarial training is universal to all kinds of DNNs, our method increases the built-in robustness by utilizing the photometric consistency of stereo matching, thus avoiding the mentioned disadvantages. It can also be combined with adversarial training to further improve robustness.

## 6 Limitations and Potential Negative Impacts of the Proposed Work

One main limitation is that while being more robust and Sim2Real generalizable, the performance of the proposed method on clean images has some room to improve. One direction is to leverage neural architecture search to find more suitable DNN aggregation component. One potential negative impact of the proposed attack method is that since it is easy to implement the proposed physically realizable attacks they could be used in some unintended way to computer vision systems relying on the conventional DNN-based stereo matching.

## 7 Conclusions

This paper presents a novel design for stereo matching, which utilizes DNNs to aggregate/optimize non-parametric cost volumes with parametric contextual features. It harnesses the best of classic features (multi-scale census transform) and end-to-end trainable DNNs for adversarially-robust and cross-domain generalizable stereo matching. The proposed method is motivated by the observation that DNN-based stereo matching methods can be deceived by a type of physically realizable attacks that entail stereo constraints in learning the perturbation. In experiments, the proposed method is tested in SceneFlow and KITTI2015 datasets with significantly better adversarial robustness and Sim2Real cross-domain generalizability achieved.

# References


[1] Heiko Hirschmuller and Daniel Scharstein. Evaluation of stereo matching costs on images with radiometric differences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(9):1582–1599, 2008.

[2] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

[3] Michael Bleyer and Sylvie Chambon. Does color really help in dense stereo matching. Citeseer, 2010.

[4] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[5] Nikolaus Mayer, Eddy Ilg, Philip Häusser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. arXiv:1512.02134.

[6] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[7] Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nešić, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *German conference on pattern recognition*, pages 31–42. Springer, 2014.

[8] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 274–283, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[9] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5410–5418, 2018.

[10] Feihu Zhang, Victor Prisacariu, Ruigang Yang, and Philip HS Torr. Ga-net: Guided aggregation net for end-to-end stereo matching. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[11] Xuelian Cheng, Yiran Zhong, Mehrtash Harandi, Yuchao Dai, Xiaojun Chang, Hongdong Li, Tom Drummond, and Zongyuan Ge. Hierarchical neural architecture search for deep stereo matching. *Advances in Neural Information Processing Systems*, 33, 2020.

[12] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[13] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 483–499. Springer, 2016.

[14] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. In *Proc. of International Conference on Computer Vision (ICCV)*, pages 66–75, 2017.

[15] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[16] Jure Zbontar and Yann LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, 2016.

[17] Xianzhi Du, Mostafa El-Khamy, and Jungwon Lee. Amnet: Deep atrous multiscale stereo disparity estimation networks. *arXiv*, 2019.

[18] Xiaoyang Guo, Kai Yang, Wukui Yang, Xiaogang Wang, and Hongsheng Li. Group-wise correlation stereo network. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[19] Kyle Yee and Ayan Chakrabarti. Fast deep stereo with 2d convolutional processing of cost signatures. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, March 2020.

[20] Xinjing Chen, Peng Wang, and Ruigang Yang. Learning depth with convolutional spatial propagation network. *arXiv*, 2018.

[21] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[22] Anurag Ranjan, Joel Janai, Andreas Geiger, and Michael J. Black. Attacking optical flow. In *ICCV*, 2019.

[23] Yulong Cao, Ningfei Wang, Chaowei Xiao, Dawei Yang, Jin Fang, Ruigang Yang, Qi Alfred Chen, Mingyan Liu, , and Bo Li. 3d adversarial object against msf-based perception in autonomous driving. In *Third Conference on Machine Learning and Systems (MLSys)*, March 2020.

[24] James Tu, Mengye Ren, Sivabalan Manivasagam, Ming Liang, Bin Yang, Richard Du, Frank Cheng, and Raquel Urtasun. Physically realizable adversarial examples for lidar object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13716–13725, 2020.

[25] James Tu, Huichen Li, Xinchen Yan, Mengye Ren, Yun Chen, Ming Liang, Eilyan Bitar, Ersin Yumer, and Raquel Urtasun. Exploring adversarial robustness of multi-sensor perception systems in self driving, 2021.

[26] Alex Wong, Mukund Mundhra, and Stefano Soatto. Stereopagnosia: Fooling stereo networks with adversarial perturbations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 2021.

[27] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[28] Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang. Recent advances in adversarial training for adversarial robustness. In *IJCAI*, 2021.

[29] H. Hirschmuller. Accurate and efficient stereo processing by semi-global matching and mutual information. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 807–814 vol. 2, 2005.

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

[31] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 5019–5031, 2018.

[32] Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, and John Duchi. Unlabeled data improves adversarial robustness. *Advances in neural information processing systems*, 2019.

# Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See the license file in the code folder in the supplementary material.

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 6.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 6.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See the code in the supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See the Appendix D
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] Code is provided.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

# A Background on Stereo Matching

The left of Fig. 1 illustrates the stereo matching setup. The rectified left image is used as the reference image to infer the disparity map. For each pixel in the left image, the goal of stereo matching is to find the target pixel on the rectified epipolar line in the right image. The search range (i.e., disparity levels) often is predefined and fixed to a sufficiently large value in the cost volume computation stage. The matching is based on minimizing the cost between features centered at the source pixel and the target pixel respectively. The challenge of stereo matching is to seek the globally optimal matching for all pixels in the left reference image and to handle many uncertainties such as in the appearance features (textureless regions and specularities), the cost function, and unknown repeated patterns and occlusion situations.

Let $\Lambda$ be an image lattice (e.g., $540 \times 960$) on which the rectified left and right images are defined, denoted by $I^L$ and $I^R$ respectively. Denote by $D(x, y)$ be the disparity map for the reference image $I^L$. In traditional methods, stereo matching is formulated as an energy/cost minimization problem,

$$\min_D E_d(I^L, I^R, D) + \lambda \cdot E_s(D), \tag{7}$$

where the first term is the data energy/cost, $E_d(I^L, I^R, D) = \sum_{(x,y)\in\Lambda} Cost(F_{I^L}(x, y), F_{I^R}(x + D(x, y), y))$ capturing the matching cost between a source pixel $(x, y)$ in the left reference image and the target pixel $(x + D(x, y), y)$ on the rectified epipolar line (i.e., the same row) in the right image. The cost is measured based on features $F_{I^L}$ and $F_{I^R}$ extracted for the source and target pixel respectively. The second term represents the prior/regularity of a disparity map such as the pairwise smoothness assumption, $E_s(D) = \sum_{(u,v)\in\mathcal{N}} S(D(u), D(v))$ where $u, v \in \Lambda$ and $\mathcal{N}$ the set of neighboring pixels (e.g., the 4-connected neighborhood). The challenges in the traditional formulation are in two-fold: what are the good features and the cost functions in the data term? And, what is the good prior that are sufficiently expressive to capture the disparity structures while facilitating efficient optimization (e.g., by the dynamic programming algorithm or semi-global method [29])?

Deep learning approaches mitigate the aforementioned challenges by exploiting the highly-expressive representational power and the end-to-end learning capability of DNNs. A cost volume is first computed which represents the solution space with respect to the data term in Eqn. 7. The prior/regularity term of a disparity map is made implicitly by the supervised loss function. The optimization algorithms (e.g., the traditional global or semi-global methods [29]) are also implicitly realized by a head sub-network (e.g., the stacked Hourglass sub-network in Fig. 5).

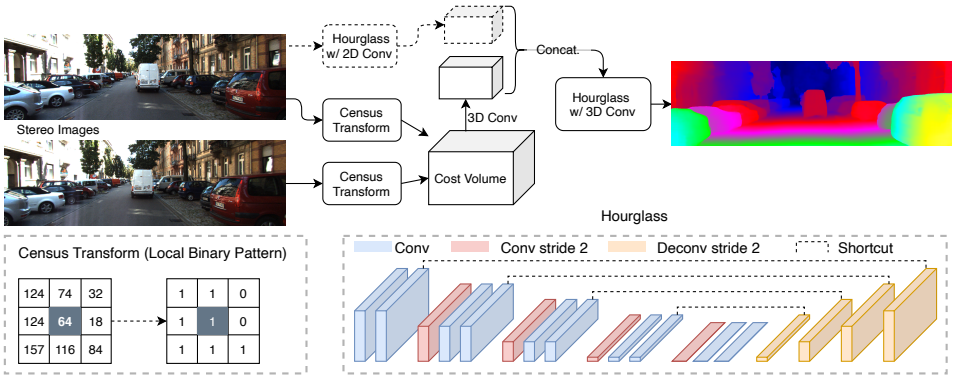


Figure 5: Network architecture.

# B Network Architecture

We show the detailed configuration of the proposed architecture in Table 9 with the workflow reproduced in Fig. 5. Besides how layers are wired, one key difference is the down-sample scale of the cost volume. While our method uses the same $1/3$ scale as GANet, PSMNet uses $1/4$ and LEAStereo uses $1/6$. Intuitively, a smaller down-sample size should lead to a stronger robustness, because more perturbations are averaged out in the cost volume. Therefore, our method is not taking

advantage of the dowm-sampled size. In fact, it is much more robust than PSMNet and LEAStereo with a finer resolution.

| Name | Layer description | Output dimension |
|---|---|---|
| input | normalized image pairs | $H \times W \times 3$ |
| **Backbone** for the left reference image | | |
| conv_start | $3 \times 3$ Conv, stride 1<br>$5 \times 5$ Conv, stride 3<br>$3 \times 3$ Conv, stride 1 | $1/3H \times 1/3W \times 32$ |
| conv_1a | $3 \times 3$ Conv, stride 2 | $1/6H \times 1/6W \times 48$ |
| conv_2a | $3 \times 3$ Conv, stride 2 | $1/12H \times 1/12W \times 64$ |
| conv_3a | $3 \times 3$ Conv, stride 2 | $1/24H \times 1/24W \times 64$ |
| deconv_1a | $3 \times 3$ Deconv, stride 2, add conv_2a | $1/12H \times 1/12W \times 64$ |
| deconv_2a | $3 \times 3$ Deconv, stride 2, add conv_1a | $1/6H \times 1/6W \times 48$ |
| deconv_3a | $3 \times 3$ Deconv, stride 2, add conv_start | $1/3H \times 1/3W \times 32$ |
| conv_1b | $3 \times 3$ Conv, stride 2, add deconv_2a | $1/6H \times 1/6W \times 48$ |
| conv_2b | $3 \times 3$ Conv, stride 2, add deconv_1a | $1/12H \times 1/12W \times 64$ |
| conv_3b | $3 \times 3$ Conv, stride 2, add conv_3a | $1/24H \times 1/24W \times 64$ |
| deconv_1b | $3 \times 3$ Deconv, stride 2, add conv_2b | $1/12H \times 1/12W \times 64$ |
| deconv_2b | $3 \times 3$ Deconv, stride 2, add conv_1b | $1/6H \times 1/6W \times 48$ |
| deconv_3b | $3 \times 3$ Deconv, stride 2, add deconv_3a | $1/3H \times 1/3W \times 32$ |
| backbone output | repeat deconv_3b $\ell/3$ times (dim=2) | $1/3H \times 1/3W \times 1/3\ell \times 32$ |
| **Multi-Scale Census Transform** | | |
| census_transform | census transform of the input with window size 11 | $H \times W \times 120$ |
| **Cost Volume** | | |
| init_cost_volume | the initial cost volume | $H \times W \times \ell \times 9$ |
| conv_3d_0a | $5 \times 5$ Conv3D, stride 3 | $1/3H \times 1/3W \times 1/3\ell \times 32$ |
| concat | [optional] concatenate with the backbone output (as context) | $1/3H \times 1/3W \times 1/3\ell \times 64$ |
| conv_3d_0b | $3 \times 3$ Conv3D | $1/3H \times 1/3W \times 1/3\ell \times 32$ |
| **Cost Aggregation** | | |
| conv_3d_1 | $[3 \times 3$ Conv3D, stride $1]_{\times 2}$ | $1/3H \times 1/3W \times 1/3\ell \times 32$ |
| conv_3d_2 | $3 \times 3$ Conv3D, stride 2<br>$[3 \times 3$ Conv3D, stride $1]_{\times 2}$ | $1/6H \times 1/6W \times 1/6\ell \times 32$ |
| conv_3d_3 | repeat above | $1/12H \times 1/12W \times 1/12\ell \times 32$ |
| conv_3d_4 | repeat above | $1/24H \times 1/24W \times 1/24\ell \times 32$ |
| conv_3d_5 | repeat above | $1/48H \times 1/48W \times 1/48\ell \times 32$ |
| deconv_3d_1 | $3 \times 3$ Deconv, stride 2, add conv_3d_4 | $1/24H \times 1/24W \times 1/24\ell \times 32$ |
| deconv_3d_2 | $3 \times 3$ Deconv, stride 2, add conv_3d_3 | $1/12H \times 1/12W \times 1/12\ell \times 32$ |
| deconv_3d_3 | $3 \times 3$ Deconv, stride 2, add conv_3d_2 | $1/6H \times 1/6W \times 1/6\ell \times 32$ |
| cost_agg_1 | $3 \times 3$ Deconv, stride 2, add conv_3d_1 | $1/3H \times 1/3W \times 1/3\ell \times 32$ |
| cost_agg_2 | repeat cost aggregation (input: cost_agg_1) | $1/3H \times 1/3W \times 1/3\ell \times 32$ |
| cost_agg_3 | repeat cost aggregation (input: cost_agg_2) | $1/3H \times 1/3W \times 1/3\ell \times 32$ |
| **Disparity Regression** | | |
| output_1 | $5 \times 5$ Deconv, stride 3 (input: cost_agg_1)<br>disparity regression (Eqn. 1 in the submission) | $H \times W \times 1$ |
| output_2 | repeat above (input: cost_agg_2) | $H \times W \times 1$ |
| output_3 | repeat above (input: cost_agg_3) | $H \times W \times 1$ |

Table 9: Details of the proposed network architecture. All convolution (Conv and Conv3D) and deconvolution (Deconv) layers are followed by batch normalization and ReLU.

# C More Results

## C.1 A Toy Experiment

We conduct a toy experiment to show that state-of-the-art stereo matching methods can be easily attacked even by the simplest form of stereo-constrained attack, *i.e.* shifted patches (Fig. 6).

We create five synthetic toy stereo image pairs. In a stereo pair, the left reference image is composed by superposing a white-noise patch onto a constant background. The right image is created using the same patch and the same background in which the patch is horizontally shifted with respect to a given disparity level (such as $\ell = 20$). So, the ground-truth disparity for the entire patch will be the specified $\ell$. The background is excluded from the evaluation.

As shown in Table 10, for the clean synthetic images, state-of-the-art stereo matching methods work very well using the SceneFlow trained model checkpoints. Our model shows better performance for all disparity levels. After applying the proposed stereo-constrained PGD attack only to the patches (similar in spirit to the adversarial patches [30]), state-of-the-art methods' performance drop significantly except for the LEAStereo method [11] at two disparity levels (60 and 100). Fig. 7 further shows the effects of applying the same adversarial patch at different disparity levels, showing that this simplest form of attack has a certain transferability with different depth levels.

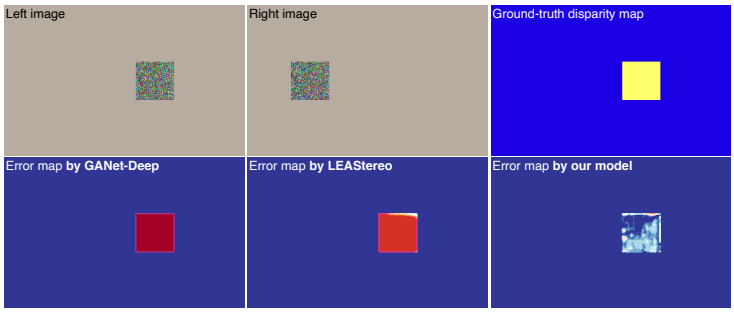


Figure 6: Illustration of adversarial vulnerability of deep stereo matching methods using a toy example: GANet-Deep [10], LEAStereo [11], and the proposed method. See text for detail.

| | Clean / After Attack, (EPE [px]) | | | | |
|---|---|---|---|---|---|
| Models | $\ell$ =20 | 60 | 100 | 140 | 180 |
| PSMNet | 1.89/78.85 | 1.22/63.57 | 0.45/3.29 | 0.94/4.59 | 0.48/36.10 |
| GANet | 2.36/9.94 | 6.74/27.72 | 7.37/80.39 | 13.56/115.77 | 15.09/31.95 |
| LEAStereo | 0.54/31.86 | 0.32/1.13 | 0.36/1.18 | 0.33/98.23 | 0.96/146.68 |
| Ours w/o ctx. | **0.057/0.051** | **0.20/0.20** | 0.29/**0.25** | **0.25/0.24** | 0.26/**0.26** |
| Ours | 0.36/3.12 | 0.28/5.56 | **0.13**/0.44 | 0.37/9.47 | **0.094**/1.77 |

Table 10: Result comparisons on synthetic adversarial patches at different disparity levels $\ell$.

**Through this toy experiment, we can observe:** (i) Although not trained with this synthetic setting, state-of-the-art stereo matching methods are capable of recovering matching results when no attacks are applied. (ii) However, the matching capabilities are not stable even with respect to the much weaker stereo-constrained attacks. This may indicate that state-of-the-art methods could learn shortcut solutions in computing the cost volume, while our methods that directly utilize local rank information in computing the cost volume are more robust, either with or without the feature backbone in computing the cost volume.

### C.2 Examples of Adversarial Patch Attacks

We show all of the 10 scenarios for adversarial patch attack in figures 8, 9, 10, 11, and 12. All scenarios are selected where the adversarial patches can be put on more flat surfaces, but they are not necessarily horizontal to the image plane. Note that the ground truth disparities of the patches are the same as the corresponding part of the original image. The first row shows the attacked image pairs for our method. Other methods will have patches on the same location but the texture will be

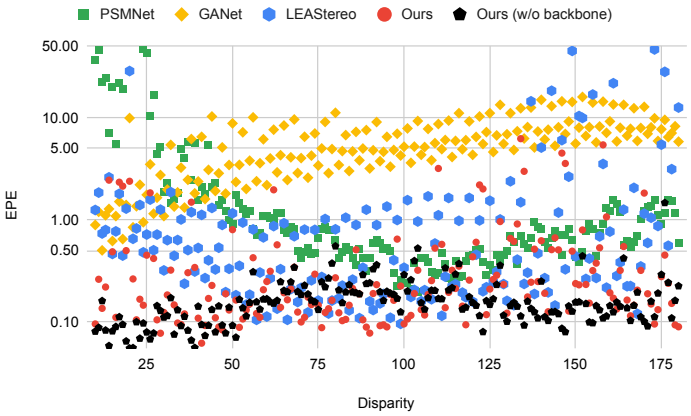


Figure 7: Test results of of shifting an adversarial patch on the left image at disparities from 10 to 180, while fixing the right image. Each point represents a testing pair with a different displacement.

different. PSMNet [9], LEAStereo [11], and GANet [10], and our method are shown on the second, third, fourth, and fifth row respectively. The first and the third columns are the after attack disparity maps, and the second and the fourth columns are the after attack error maps from the ground truth.

From the results, we can see that our method is significantly more robust than others in this physically realizable attack setting.

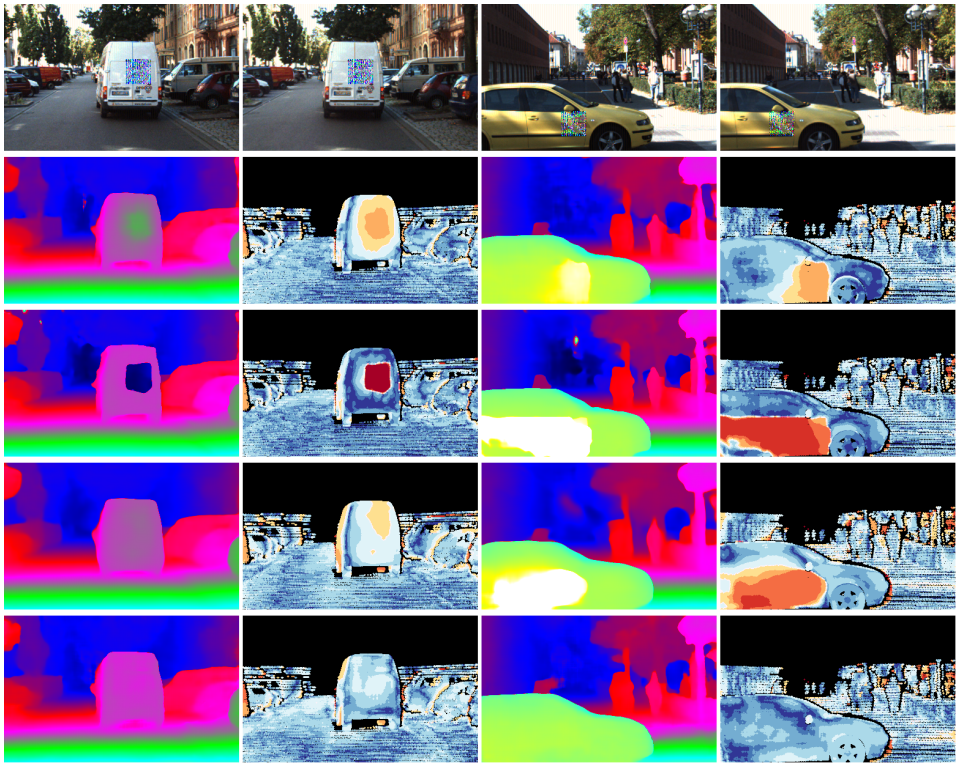

Figure 8: Illustration of the adversarial patch attack (1/5). The first row shows the attacked image pairs. PSMNet [9], GANet [10], LEAStereo [11], and our method are shown on the second, third, fourth, and fifth row respectively. The first and the third columns are the after attack disparity maps, and the second and the fourth columns are the after attack error maps from the ground truth.

### C.3 Ablation Study on Adversarial Training

Here we modify our model by using two feature backbones while keeping other components fixed and use this setting to study the effects of different PGD iteration steps and training epochs on adversarial training in Table 11. Our method is not suitable for justifying these hyper-parameters since it has significantly stronger robustness.

In this experiment, we test the modified counterpart under FGSM attack, the unconstrained PGD attacks with different iterations (3 and 9) and training epochs (20 and 50). In order for adversarial training to be effective, the attack should be as strong as possible [2]. Table 11 shows that all the model trained with PGD attacks are indeed more robust than the one with FGSM attack. However, it does not make much difference for using 9 iterations or 50 epochs, showing that 3 iterations with 20 epochs are sufficient for the adversarial training.

Besides justifying the hyper-parameters, this result also shows that our method is indeed more robust than its counterpart as it uses the proposed multi-scale Census Transform for the matching.

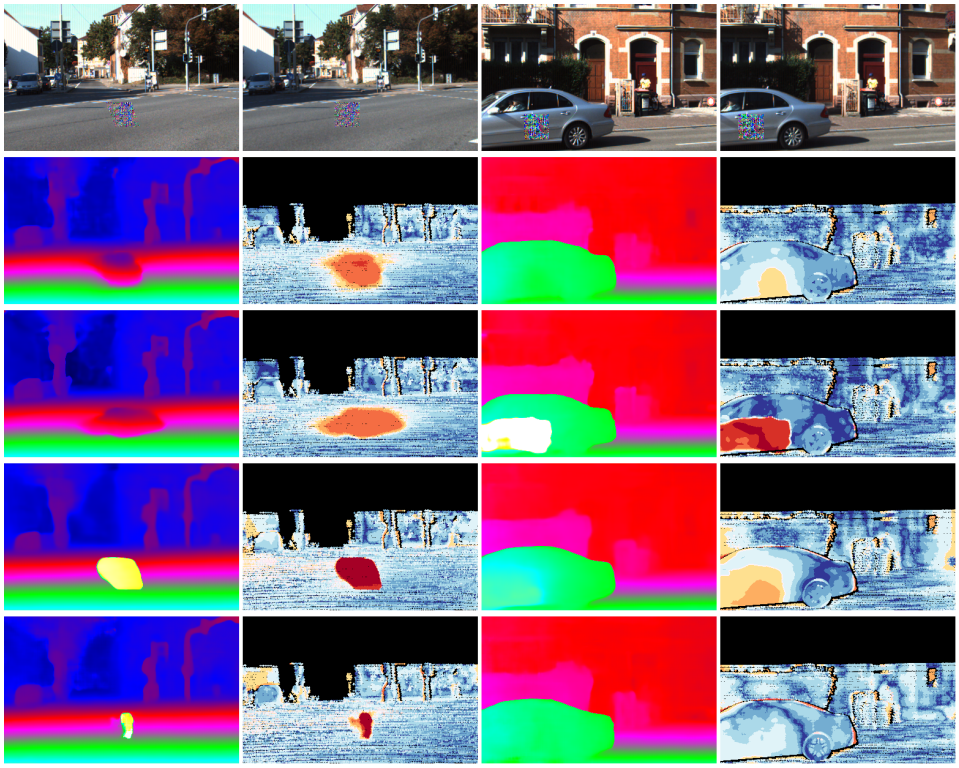

Figure 9: Illustration of the adversarial patch attack (2/5). The first row shows the attacked image pairs. PSMNet [9], GANet [10], LEAStereo [11], and our method are shown on the second, third, fourth, and fifth row respectively. The first and the third columns are the after attack disparity maps, and the second and the fourth columns are the after attack error maps from the ground truth.

| | clean | | | PGD Attack ($\epsilon = 0.03$) | | |
|---|---|---|---|---|---|---|
| Models | EPE | Bad 1.0 | Bad 3.0 | EPE | Bad 1.0 | Bad 3.0 |
| Ours w/o CT. | 0.38 | 4.24 | 0.34 | 2.13 | 56.34 | 15.23 |
| fgsm 20-epoch | 0.44 | 6.97 | 0.65 | 0.86 | 23.92 | 2.46 |
| adv-3 20-epoch | 0.45 | 7.67 | 0.72 | 0.79 | 20.91 | 1.91 |
| adv-9 20-epoch | 0.52 | 10.36 | 1.06 | 0.79 | 21.50 | 2.18 |
| adv-9 50-epoch | 0.52 | 10.36 | 1.06 | 0.74 | 19.26 | 1.98 |
| Ours | **0.36** | **3.61** | **0.27** | 0.88 | 21.20 | 3.75 |
| Ours + adv. | 0.41 | 5.77 | 0.52 | **0.61** | **13.46** | **1.39** |

Table 11: Stereo-constrained attack results under different adversarial training settings. The first 5 rows show the results of our modified version with two feature backbones.

### C.4 Supplementary Results on the Experiments

Here we provide the EPE and Bad 1.0 for the adversarial patch attack and the transferability of adversarial robustness experiments. From Tables 12 and 13, our method shows significantly stronger robustness also in EPE and Bad 1.0, especially in Bad 1.0.

In addition, we show two different versions of the unconstrained attacks in KITTI2015, where one use the ground truth to attack, and the other use the original neural network prediction. Results are shown in Table 14. It shows that using prediction is weaker than using the ground truth for unconstrained attacks.

### C.5 Occluded Regions

For the proposed stereo-constrained PGD attack, we disallow to attack and evaluate occluded regions of the reference image, which prevents the perturbation to attack the regions where the estimation does not rely on matching. Nonetheless, it is still possible to make perturbation on the occluded regions of the right image to hinder the matching, e.g. by creating false positive correspondence. We

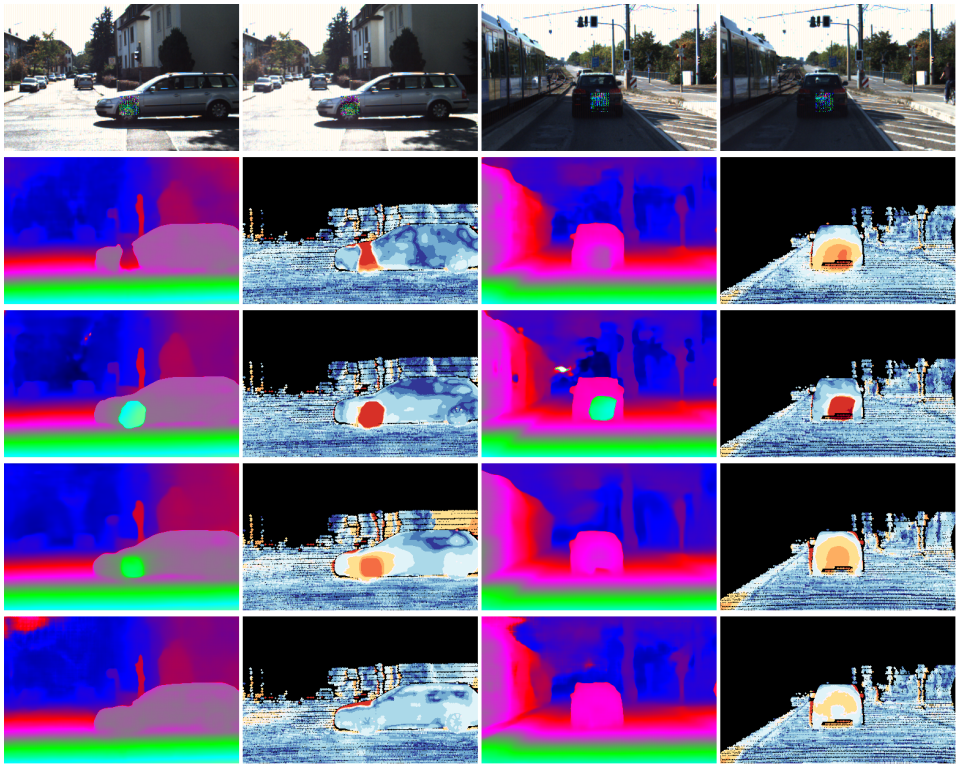

Figure 10: Illustration of the adversarial patch attack (3/5). The first row shows the attacked image pairs. PSMNet [9], GANet [10], LEAStereo [11], and our method are shown on the second, third, fourth, and fifth row respectively. The first and the third columns are the after attack disparity maps, and the second and the fourth columns are the after attack error maps from the ground truth.

| | Clean | | | Synthetic Patch Attack | | |
|---|---|---|---|---|---|---|
| Models | EPE | Bad 1.0 | Bad 3.0 | EPE | Bad 1.0 | Bad 3.0 |
| PSMNet | 0.28 | 2.30 | 0.21 | 0.80 | 15.41 | 4.46 |
| GANet-Deep | **0.25** | **1.46** | **0.12** | 1.84 | 12.52 | 9.09 |
| LEAStereo | 0.41 | 5.60 | 0.69 | 1.34 | 20.32 | 6.18 |
| Ours | 0.40 | 4.79 | 0.46 | 0.54 | 7.58 | 0.85 |
| Ours w/o ctx. | 0.46 | 5.51 | 0.63 | **0.48** | **6.42** | **0.81** |
| PSMNet + adv. | 0.58 | 12.56 | 1.44 | 1.44 | 26.55 | 8.59 |
| GANet + adv. | 0.55 | 11.24 | 1.32 | 2.31 | 23.5 | 11.98 |
| LEAStereo + adv. | 0.71 | 15.87 | 2.24 | 1.96 | 30.88 | 11.72 |
| Ours w/o ctx. + adv. | **0.52** | **8.95** | **1.03** | **0.54** | **9.45** | **1.18** |
| Ours + adv. | 0.53 | 11.25 | 1.17 | 0.58 | 12.01 | 1.40 |

Table 12: Adversarial Patch Attack Results in the KITTI2015 training dataset with photometric consistency retained in attack.

also consider this situation and experiment with *an even weaker attack* such that the occluded regions of both the left and right images will not be attacked.

The results are shown in Table 15 Table 17, which are consistent with those reported in the submission. Our methods are the best in all metrics except for EPE in ScenFlow with $\epsilon = 0.03$.

### C.6 Results in SceneFlow

We first compare the adversarial robustness. Table 16 and Table 17 shows the comparisons: **our models are much more robust than prior arts**. As the original SceneFlow dataset does not have ground truth occlusion, we use a subset provided by the same authors [5] with occlusions. We randomly select $1,000$ images from the test data. Our methods show significantly better robustness against attacks. In fact, our methods are the best in all metrics except for EPE in ScenFlow with $\epsilon = 0.03$ in Table 17. We note that if we do not allow to use the proposed differentiable approximation

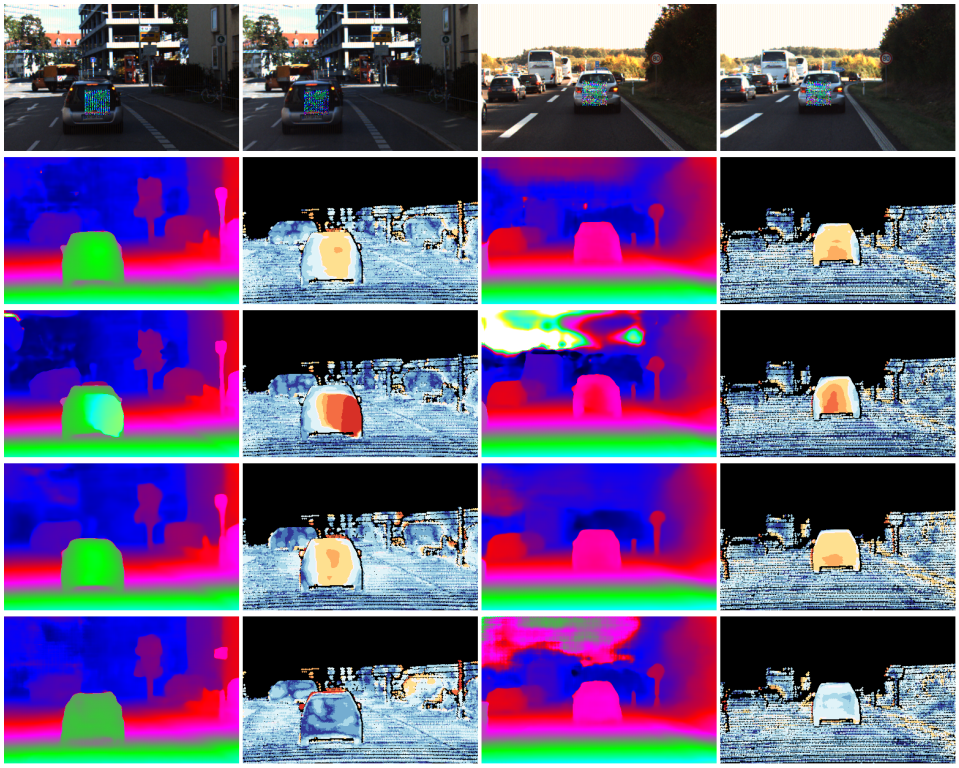

Figure 11: Illustration of the adversarial patch attack (4/5). The first row shows the attacked image pairs. PSMNet [9], GANet [10], LEAStereo [11], and our method are shown on the second, third, fourth, and fifth row respectively. The first and the third columns are the after attack disparity maps, and the second and the fourth columns are the after attack error maps from the ground truth.

| | Clean | | | PGD Attack ($\epsilon = 0.03$) | | | PGD Attack ($\epsilon = 0.06$) | | |
|---|---|---|---|---|---|---|---|---|---|
| Models | EPE | Bad 1.0 | Bad 3.0 | EPE | Bad 1.0 | Bad 3.0 | EPE | Bad 1.0 | Bad 3.0 |
| PSMNet + adv. | 0.8 | 18.24 | 3.03 | 1.37 | 33.87 | 6.72 | 2.07 | 51.00 | 12.28 |
| GANet + adv. | 0.80 | 18.82 | 3.03 | 1.45 | 36.79 | 7.75 | 2.29 | 54.07 | 15.42 |
| LEAStereo + adv. | 0.85 | 20.17 | 3.56 | 1.48 | 38.24 | 8.60 | 2.3 | 55.67 | 16.24 |
| Ours w/o ctx. | 0.49 | 6.40 | 1.23 | 1.58 | 26.69 | 9.66 | 1.92 | 32.52 | 12.87 |
| Ours | 0.46 | 6.07 | 1.00 | 1.37 | 22.45 | 7.29 | 1.80 | 29.61 | 10.50 |
| Ours w/o ctx. + adv. | 0.65 | 11.18 | 2.25 | 1.28 | 24.62 | 6.64 | 1.48 | 28.93 | 8.31 |
| Ours + adv. | **0.56** | **10.14** | **1.73** | **0.89** | **18.35** | **4.14** | **1.09** | **22.64** | **5.95** |

Table 13: Transferrability of Adversarial Robustness: stereo-constrained 20-step PGD Attack Results in the KITTI2012 training dataset using adversarially trained neural networks on KITTI2015.

of Census Transform, our method shows much better robustness, thanks to the non-differentiable cost volume computation.

We also compare the results using the entire image, instead of only non-occlusion regions in evaluating attack performance. Table 18 shows the comparisons. Our method obtains competitive performance against state-of-the-art methods. Recent work suggests there exists an inherent conflict between accuracy and robustness [31, 32]. From this perspective, the comparable performance on clean images and the significantly better robustness of our method show that the proposed design for stereo matching is effective.

# D Environment

All the experiments were done on a Nvidia DGX server running Ubuntu 18.04.5, which equips 4 Tesla V-100 GPUs, each has 32 gigabytes of memory. For Pytorch, we use version 1.8.0 with CUDA 11.1.

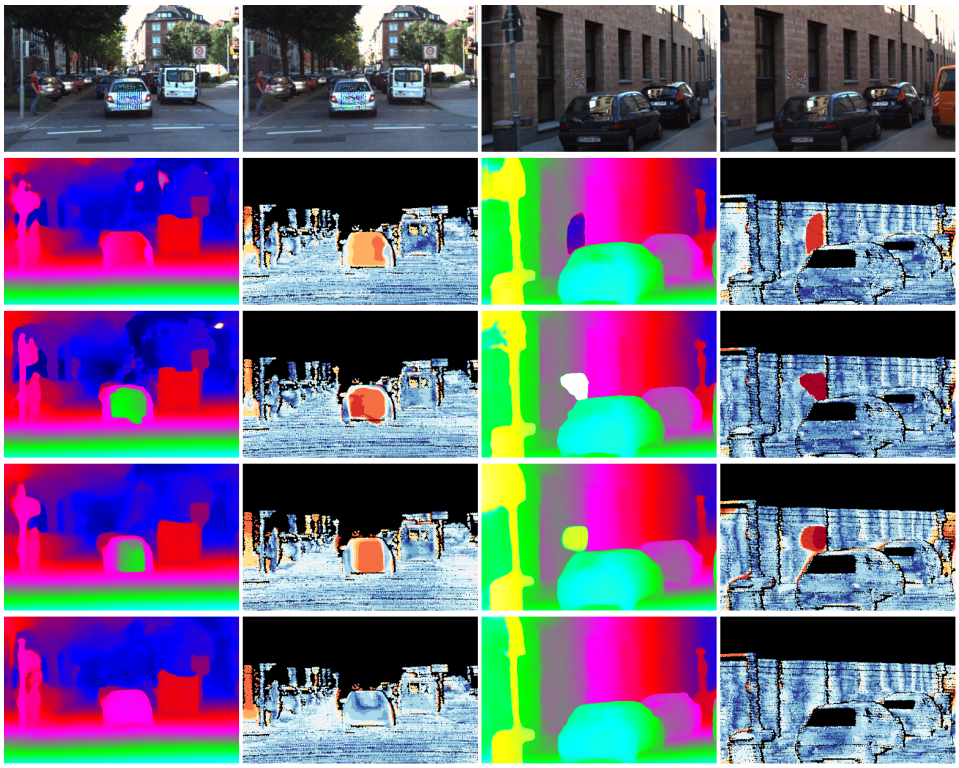

Figure 12: Illustration of the adversarial patch attack (5/5). The first row shows the attacked image pairs. PSMNet [9], GANet [10], LEAStereo [11], and our method are shown on the second, third, fourth, and fifth row respectively. The first and the third columns are the after attack disparity maps, and the second and the fourth columns are the after attack error maps from the ground truth.

| | PGD Attack ($\epsilon = 0.03$) w/ GT | | | PGD Attack ($\epsilon = 0.03$) w/ Prediction | | |
|---|---|---|---|---|---|---|
| Models | EPE [px] | Bad 1.0 [%] | Bad 3.0 [%] | EPE [px] | Bad 1.0 [%] | Bad 3.0 [%] |
| PSMNet | 91.08 | 92.75 | 89.91 | 58.75 | 84.11 | 70.49 |
| GANet | 23.75 | 89.48 | 79.11 | 19.49 | 68.01 | 51.58 |
| LEAStereo | 14.71 | 82.42 | 64.31 | 12.03 | 74.21 | 54.00 |
| Ours w/o ctx. | 2.36 | 41.34 | 16.30 | 2.27 | 34.98 | 14.74 |
| Ours | **1.81** | **36.42** | **11.29** | **1.53** | **29.25** | **10.09** |

Table 14: Vanilla 20-step PGD Attack Results in the KITTI2015 training dataset [6]. The PGD attacks are learned using either the GT disparity map or the predicted disparity map from clean images in the loss function used to compute PGD. The performance are still measured in terms of the GT disparity map.

| | Clean | | | PGD Attack ($\epsilon = 0.03$) | | | PGD Attack ($\epsilon = 0.06$) | | |
|---|---|---|---|---|---|---|---|---|---|
| Models | EPE | Bad 1.0 | Bad 3.0 | EPE | Bad 1.0 | Bad 3.0 | EPE | Bad 1.0 | Bad 3.0 |
| PSMNet | 0.28 | 2.00 | 0.16 | 1.29 | 45.79 | 5.12 | 5.19 | 70.73 | 30.08 |
| GANet | **0.25** | **1.42** | **0.10** | 1.25 | 39.76 | 5.92 | 2.93 | 63.88 | 25.84 |
| LEAStereo | 0.37 | 4.54 | 0.42 | 1.44 | 46.57 | 7.61 | 3.20 | 66.26 | 27.45 |
| Ours w/o ctx. | 0.38 | 4.14 | 0.32 | 0.60 | 12.95 | 1.16 | 0.64 | 14.39 | 1.42 |
| Ours | 0.36 | 3.61 | 0.27 | **0.53** | **9.90** | **0.75** | **0.59** | **11.97** | **0.93** |

Table 15: Stereo-Constrained 20-step PGD Attack Results in the KITTI2015 training dataset [6]. Attacks on occluded regions of both the left and the right image are disallowed.

| | Clean (Non-occlusion regions) | | | After PGD Attack ($\epsilon = 0.03$) | | | After PGD Attack ($\epsilon = 0.06$) | | |
|---|---|---|---|---|---|---|---|---|---|
| Models | EPE | Bad 1.0 | Bad 3.0 | EPE | Bad 1.0 | Bad 3.0 | EPE | Bad 1.0 | Bad 3.0 |
| PSMNet | 1.56 | 20.62 | 5.36 | 12.37 | 68.12 | 30.75 | 25.51 | 75.66 | 51.13 |
| GANet | 1.04 | 10.82 | 3.37 | 11.52 | 51.47 | 36.10 | 28.54 | 72.75 | 64.03 |
| LEAStereo | 1.03 | 8.87 | **2.69** | 12.12 | 55.69 | 33.82 | 22.30 | 68.04 | 53.63 |
| Ours | **1.02** | **8.85** | 3.28 | 9.87 | 31.94 | 25.67 | 20.74 | 45.84 | 41.37 |
| Ours w/o ctx. | 1.16 | 9.49 | 3.55 | **9.23** | **30.61** | **25.15** | **10.85** | **32.88** | **27.84** |
| Ours$^{\dagger}$ | 1.02 | 8.85 | 3.28 | 2.25 | 12.64 | 6.71 | 5.79 | 18.09 | 13.04 |

Table 16: Stereo-Constrained 20-step PGD Attack Results in SceneFlow [5]. $^{\dagger}$ shows results by our method without using the modified census transform in learning attacks, which are much more resistant to attacks. See text for detail.

| | Clean | | | After PGD Attack ($\epsilon = 0.03$) | | | After PGD Attack ($\epsilon = 0.06$) | | |
|---|---|---|---|---|---|---|---|---|---|
| Models | EPE | Bad 1.0 | Bad 3.0 | EPE | Bad 1.0 | Bad 3.0 | EPE | Bad 1.0 | Bad 3.0 |
| PSMNet | 1.56 | 20.62 | 5.36 | 12.32 | 68.15 | 30.76 | 20.15 | 74.50 | 46.74 |
| GANet | 1.04 | 10.82 | 3.37 | 14.37 | 57.90 | 42.03 | 23.59 | 69.28 | 57.90 |
| LEAStereo | 1.03 | 8.87 | **2.69** | **9.32** | 54.86 | 31.16 | 16.51 | 64.26 | 45.77 |
| Ours w/o ctx. | 1.16 | 9.49 | 3.55 | 11.63 | **34.18** | **29.26** | **13.09** | **36.55** | **32.21** |
| Ours | **1.02** | **8.85** | 3.28 | 11.70 | 35.36 | 29.56 | 16.08 | 42.14 | 37.21 |

Table 17: Stereo-Constrained 20-step PGD Attack Results in SceneFlow [5]. Attacks on occluded regions of both the left and the right image are disallowed.

| Models | Params [M] | EPE [px] | Bad 1.0 [%] | Bad 3.0 [%] |
|---|---|---|---|---|
| PSMNet | 3.5M | 1.49 | 20.6 | 5.9 |
| GANet | 6.6M | **0.82** | 9.0 | 3.5 |
| LEAStereo | 1.8M | 0.83 | **8.0** | **3.3** |
| Ours w/o ctx. | 1.9M | 1.10 | 9.7 | 4.4 |
| Ours | 2.7M | 0.84 | 8.8 | 3.7 |

Table 18: Result comparisons using clean images in the SceneFlow dataset [5].