Title
Stereopagnosia: Fooling Stereo Networks with Adversarial Perturbations

Author
Mundhra, Mukund

Publication Date
2021

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

Stereopagnosia:
Fooling Stereo Networks with
Adversarial Perturbations

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Science in Computer Science

by

Mukund Mundhra

2021

ABSTRACT OF THE THESIS

Stereopagnosia:
Fooling Stereo Networks with
Adversarial Perturbations

by

Mukund Mundhra
Master of Science in Computer Science
University of California, Los Angeles, 2021
Professor Stefano Soatto, Chair

This work studies the effect of adversarial perturbations of images on the estimates of disparity by deep learning models trained for stereo. We show that imperceptible additive perturbations can significantly alter the disparity map, and correspondingly the perceived geometry of the scene. These perturbations not only affect the specific model they are crafted for, but transfer to models with different architecture, trained with different loss functions. We show that, when used for adversarial data augmentation, our perturbations result in trained models that are more robust, without sacrificing overall accuracy of the model. This is unlike what has been observed in image classification, where adding the perturbed images to the training set makes the model less vulnerable to adversarial perturbations, but to the detriment of overall accuracy. We test our method using the most recent stereo networks and evaluate their performance on public benchmark datasets.

The thesis of Mukund Mundhra is approved.

Quanquan Gu

Cho-Jui Hsieh

Stefano Soatto, Committee Chair

University of California, Los Angeles

2021

To my family . . .

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to Prof. Stefano Soatto for giving me the opportunity to work as a part of the UCLA Vision Lab and advising me during my graduate career at UCLA. I am thankful to Prof. Soatto for guiding me throughout this project, for several ideas used here and for his constant words of encouragement. I would like to thank Prof. Cho-Jui Hsieh and Prof. Quanquan Gu for their support and feedback which guided me through this thesis. I am very thankful to Alex Wong for his constant guidance, support, ideas and several brainstorming sessions which made this work possible.

I also want to thank other faculty and students of the UCLA Computer Science department who I've learned a lot from during my time as a graduate student here. Finally, I want to thank my parents, Sangeeta and Srikrishna Mundhra, my sister, Anushree Mundhra, and my friends for their emotional support, especially during the trying times of a global pandemic when I wrote this thesis.

CHAPTER 1

Introduction

Deep Neural Networks are being extensively used in fields like Computer Vision, Natural Language Processing, Reinforcement Learning, etc., for a wide variety of tasks like image classification, dialogue systems, autonomous driving, etc. These networks, however, are seen as fragile, in the sense that small perturbations of their input, for instance an image, can cause a large change in the output, for instance the inferred class of objects in the scene [18] or its depth map [28]. This is not too surprising, since there are infinitely many scenes that are consistent with the given image, so at inference time one has to rely on the complex relation between that image and *different scenes* portrayed in the training set. This is not the case for stereo: Under mild assumptions discussed below, a depth map can be uniquely inferred point-wise from two images. There is no need to *learn* stereo, as the images of a particular scene are sufficient to infer its depth without relying on images of different scenes. (The reason we *do* use learning is to regularize the reconstruction where the assumptions mentioned below are violated, for instance in regions of homogeneous reflectance.) It would therefore be surprising if one could perturb the images in a way that forces the model to over-rule the evidence and alter the perceived depth map, especially if such perturbations affect regions of non-uniform reflectance. In this work, we show that this *can* be done and refer to this phenomenon as *stereopagnosia*, a geometric analogue of prosopagnosia [2].

Specifically, we consider stereo networks, that are functions that take as input a calibrated stereo pair and produce a depth map as output. A stereo pair consists of two images captured by cameras in known relative configuration (position and orientation), with non-zero parallax (distance between the optical centers), projectively rectified so that corresponding points (points in the two image planes whose pre-image under perspective projections intersect in space) lie on corresponding scan-lines. A depth map is

a function that associates to each pixel in a rectified image a positive real number corresponding to the distance of the point of first intersection in the scene from the optical center.

Equivalently, the network can output *disparity*, the displacement between corresponding points in the two images, from which depth can be computed in closed form. Wherever a point in the scene is supported on a surface that is Lambertian, locally smooth, seen under constant illumination, co-visible from both images, and sporting a *sufficiently exciting* reflectance (there exist region statistics that exhibit isolated extrema, so the region around the point is "distinctive" [14]), its distance from the images can be computed in closed-form [15]. Where such assumptions are violated, disparity is either not defined, for instance in occluded regions that are visible from one image but not the other, or ill-posed, for instance in regions with constant reflectance where any disparity is equally valid (the so-called "aperture problem"). To impute disparity to these regions, regularization can be either generic (*e.g.*, minimal-surface assumptions [10]) or data-driven, exploiting known relations between stereo pairs and disparity in scenes other than the one in question. This is where stereo networks come in.

The main contributions of this work can be summarized as follows:

- We show that *stereo networks are vulnerable to adversarial perturbations,* which are small additive changes in the input images (either one or both), designed for a specific image pair in a way that maximally changes the output of a specific trained deep network model. The fact that it is possible to alter the disparity, *even in regions that satisfy the assumptions discussed above* (Fig. 3.3), where disparity is uniquely defined and computable in closed form, is surprising since the network is forced to ignore the evidence, rather than simply exploit the unbounded hypothesis space available in an ill-posed problem. (Chap. 3)

- We show that, despite being crafted for a specific model, the perturbations can affect the behavior of other models, with different network architecture, trained with different loss functions and optimization methods. However, transferability is not symmetric, for instance perturbations constructed for AANet [32] can wreak havoc if used with DeepPruner [5], but not vice-versa. Models that incorporate

explicit matching, such as correlation, are more robust than those that are agnostic to the mechanics of correspondence, and are instead based on stacking generic features. (Chap. 4)

- Our third contribution is more constructive, and establishes that adversarial perturbations can be used to beneficial effects by augmenting the dataset and function as regularizers. Unlike in single-image classification and monocular depth perception where such regularization trades off robustness to perturbations with overall accuracy, in our case we obtain models that are more robust while retaining the performance of the original model. (Chap. 5)

To achieve these results, we extend the Fast Gradient Sign Method [7] and its iterative versions [4, 13], developed for single frame classification, to two-frame stereo disparity estimation. We evaluate the robustness of recent stereo methods (PSMNet, DeepPruner, AANet) on the standard benchmark stereo datasets (KITTI 2012 [6] and 2015 [17]).

CHAPTER 2

Background and Related Work

2.1 Background

2.1.1 Adversarial Attacks

Adversarial perturbations are visually imperceptible perturbations added to the inputs of deep networks that fool them into producing incorrect predictions. To ensure that the perturbations are imperceptible, we use some norm constraints depending on the type of perturbations required. Adversarial perturbations have been extensively studied for single image tasks like image classification, with several existing methods to generate perturbations. One of the earliest and simplest methods is the Fast Gradient Sign Method [7] proposed for the classification task. Assuming access to the network f_θ and its loss function $\ell(f_\theta(x), y_{gt})$, the perturbations are computed as the sign of gradient with respect to the input:

$$v = \epsilon \cdot \texttt{sign}(\nabla_x \ell(f_\theta(x), y_{gt}), \tag{2.1}$$

where y_{gt} is the groundtruth class and ϵ is the max-norm of the perturbations. [13] proposed an iterative version of the same, called iterative-FGSM (I-FGSM). To craft perturbations v for the input image x using iterative FGSM, we begin with $v^0 = 0$ and accumulate the sign of gradient with respect to the input for N steps:

$$g^{n+1} = \nabla_x \ell(f_\theta(x + v^n), y_{gt}), \tag{2.2}$$

$$v^{n+1} = \texttt{clip}\big(\alpha \cdot \texttt{sign}(g^{n+1}), -\epsilon, \epsilon\big), \tag{2.3}$$

where n is the step, α is the step size and the $\texttt{clip}(\cdot, -\epsilon, \epsilon)$ operation sets any value less than $-\epsilon$ to $-\epsilon$ and any value larger than ϵ to ϵ. The output perturbation is obtained after the N-th step, $v = v^N$.

[4] proposed MI-FGSM, a momentum variant of the iterative version to leverage gradients from previous steps. We replace the gradients (Eqn. 2.2) with normalized gradients and a momentum term weighted by a positive scalar β for N steps:

$$g^{n+1} = \frac{\nabla_x \ell(f_\theta(x + v^n), y_{gt})}{\|\nabla_x \ell(f_\theta(x + v^n), y_{gt})\|_1}, \tag{2.4}$$

$$m^{n+1} = \beta \cdot m^n + (1 - \beta) \cdot g^{n+1}, \tag{2.5}$$

$$v_I^{n+1} = \texttt{clip}\big(\alpha \cdot \texttt{sign}(m^{n+1}), -\epsilon, \epsilon\big), \tag{2.6}$$

where $m^0 = 0$, $v = v^N$.

There are several other ways to generate adversarial perturbations but we consider only the above three and repurpose them for stereo which is a dense regression task.

2.1.2 Stereo Matching

Stereo Matching refers to the task of finding pixels in a stereo pair of images which correspond to the same 3D point in the scene. The input stereo pair is rectified, which simplifies the process by warping the images such that the correspondences lie on the same horizontal line. Given a rectified pair of stereo images, a stereo network outputs the disparity of each pixel in the image, which is the horizontal distance between a pixel and its corresponding match in the other image. For points that are co-visible from the two images, there exists a unique minimizer. Methods like Semiglobal Matching [9] which use a pixelwise mutual information based matching cost along with fast approximation by pathwise optimizations do stereo matching without any learning. The reason we do learn stereo using deep networks is for regularization in regions where the assumptions mentioned in the introduction (Chap. 1) are not satisfied. For these regions, we impute disparity from known relations between stereo pairs and disparities as seen in the training set. Moreover, a single image is compatible with infinitely many scenes; hence, the single image depth prediction network learns a prior on the scene given an image so it is easy to perturb pixel intensities to alter prediction. Whereas, in stereopsis, the scene is observable from the stereo pair under suitable conditions; therefore, fooling stereo networks is equivalent to forcing them to ignore the evidence present in the images.

2.2 Related Work

2.2.1 Adversarial Perturbations

Adversarial Perturbations [26] have been extensively studied for classification [7, 26] with many iterative methods to boost the effectiveness of the attacks [4, 13, 18]. [19] further extended the attacks to the universal setting, where the same perturbations can be added to each image in a dataset to fool a network; [22] showed that unrecognizable noise can result in high confidence predictions. To defend against such attacks, [13, 27] proposed training with adversarial data augmentation and [31] improved it with randomization.

Recently, [21] studied transferability of perturbations across datasets and models and [30] improved transferability across networks by deforming the image. [24] demonstrated lower bounds on the magnitudes of perturbations needed to fool a network and [11] showed that the existence of adversarial perturbations can be attributed to non-robust features.

While there are many adversarial works on classification, there exist only a few for dense-pixel prediction tasks (e.g. semantic segmentation, depth, optical flow). [29] designed attacks for detection and segmentation. [8] demonstrated targeted universal attacks for semantic segmentation, where the network is fooled to predict a specific target. [28] used targeted attacks to provide explainability for single image depth prediction networks; whereas [3] probed them by inserting vehicles into input images. [20] examined universal attacks for segmentation and single image depth. [25] studied patch attacks for optical flow.

Unlike [20, 28], we study *non-targeted* adversarial perturbations for stereo matching. While [25] also use multiple frames, they apply the *same visible* patch to the same locations in both images, whereas our attacks are *visually imperceptible* and crafted separately for each image.

2.2.2 Deep Stereo Matching

Deep Stereo Matching [33, 34] leveraged deep networks to extract features and perform matching separately. Recent works implement the entire stereo pipeline as network layers

trained end-to-end. [16] used correlation layers to create a 2D cost volume. [23] extended [16] to a cascade residual learning framework. AANet [32] also used correlation, but instead introduced adaptive sampling to replace convolutions (regular grid sampling) when performing cost aggregation to avoid sampling at discontinuities. [12] proposed to concatenate features together to build a 3D cost volume for performing cost aggregation. PSMNet [1] added spatial pyramid pooling layers and introduced a stacked hourglass architecture. DeepPruner [5] followed the 3D cost volume architectures proposed by [1, 12] and proposed differentiable patch matching over deep features to construct a sparse 3D cost volume.

In this work, we consider adversaries for PSMNet [1], DeepPruner [5] and AANet [32]. PSMNet is an exemplar of modern stereo networks (stacked hourglass, cost volume, 3D convolutions), but uses feature stacking without explicit matching. DeepPruner shares the general architecture of PSMNet, but performs explicit matching. AANet is the state of the art and represents the 2D convolution and correlation architecture. In choosing these methods, we (i) examine their individual robustness against adversaries, (ii) study the transferability of perturbations between similar and different architectures, and (iii) apply defenses to increase robustness against adversaries. To the best of our knowledge, we are the first to study adversarial perturbations for stereo. As mentioned in the introduction, it is not a given that adversarial perturbations, known to exist for single image reconstruction, would exist for stereo, where the geometry of the scene is uniquely determined from the data, at least in the regions that satisfy the unique correspondence assumptions.

CHAPTER 3

Attacking Stereo Networks

3.1 Generating Adversarial Perturbations

Given a pretrained stereo network $f_\theta(x_L, x_R)$ that predicts the disparity between the left x_L and right x_R images of a stereo pair, our goal is to craft perturbations $v_L, v_R \in^{H\times W\times 3}$, such that when added to (x_L, x_R), $f_\theta(x_L, x_R) \neq f_\theta(x_L + v_L, x_R + v_R)$. To ensure that the perturbations are visually imperceptible, we subject them to the norm constraints $\|v_I\|_\infty \leq \epsilon$ for $I \in \{L, R\}$. To demonstrate such perturbations exist, we extend white-box methods Fast Gradient Sign Method (FGSM) [7], iterative-FGSM (I-FGSM) [13], and its momentum variant (MI-FGSM) [4], originally for classification, to the stereo matching task. We note that it is also possible to perturb only one of the two images (e.g. let $v_L = 0$ or $v_R = 0$); the effect is less pronounced, but nonetheless present and shown later in this chapter.

3.1.1 FGSM

Assuming access to the network f_θ and its loss function $\ell(f_\theta(x_L, x_R), y_{gt})$, the perturbations for the left and right images are computed as the sign of gradient with respect to the images separately:

$$v_I = \epsilon \cdot \texttt{sign}(\nabla_{x_I} \ell(f_\theta(x_L, x_R), y_{gt}), \tag{3.1}$$

where $y_{gt} \in \mathbf{R}_+^{H\times W}$ is the groundtruth and $I \in \{L, R\}$.

3.1.2 I-FGSM

To craft perturbations v_L and v_R for the stereo pair x_L and x_R using iterative FGSM, we begin with $v_L^0 = 0$ and $v_R^0 = 0$ and accumulate the sign of gradient with respect to each

image for N steps:

$$g_I^{n+1} = \nabla_{x_I} \ell(f_\theta(x_L + v_L^n, x_R + v_R^n), y_{gt}), \tag{3.2}$$

$$v_I^{n+1} = \texttt{clip}\big(\alpha \cdot \texttt{sign}(g_I^{n+1}), -\epsilon, \epsilon\big), \tag{3.3}$$

where n is the step, α is the step size and the $\texttt{clip}(\cdot, -\epsilon, \epsilon)$ operation sets any value less than $-\epsilon$ to $-\epsilon$ and any value larger than ϵ to ϵ. The output perturbation is obtained after the N-th step, $v_L = v_L^N$ and $v_R = v_R^N$.

3.1.3 MI-FGSM

To leverage gradients from previous steps, we follow [4] and replace the gradients (Eqn. 3.2) with normalized gradients and a momentum term weighted by a positive scalar β for N steps:

$$g_I^{n+1} = \frac{\nabla_{x_I} \ell(f_\theta(x_L + v_L^n, x_R + v_R^n), y_{gt})}{\|\nabla_{x_I} \ell(f_\theta(x_L + v_L^n, x_R + v_R^n), y_{gt})\|_1}, \tag{3.4}$$

$$m_I^{n+1} = \beta \cdot m_I^n + (1 - \beta) \cdot g_I^{n+1}, \tag{3.5}$$

$$v_I^{n+1} = \texttt{clip}\big(\alpha \cdot \texttt{sign}(m_I^{n+1}), -\epsilon, \epsilon\big), \tag{3.6}$$

where $m_I^0 = 0$, $v_L = v_L^N$, and $v_R = v_R^N$.

3.1.4 DI^2-FGSM and MDI^2-FGSM

Besides crafting perturbations for specific models, we also study their transferability to different models. To this end, we take $(x_L + v_L, x_R + v_R)$ optimized for one model (e.g. PSMNet) and feed it as input to another (e.g. AANet). However, while iterative methods (I-FGSM, MI-FGSM) are more effective than FGSM at corrupting the target model, their perturbations are unlikely to transfer across models because they tend to overfit to the target model. To increase the transferability across models, we leverage diverse inputs [30] as data augmentation when crafting perturbations using I-FGSM and MI-FGSM.

Diverse inputs (DI) for iterative methods aims to reduce overfitting by randomly resizing the input images by a factor of $h \in [h_{\min}, h_{\max}]$ in height and $w \in [w_{\min}, w_{\max}]$ in width with probability p. To maintain the original resolution, the inputs are randomly padded with zeros on each side such that the total padding along the height is $(H - h \cdot H)$

and along the width ($W - w \cdot W$), respectively. We denote this procedure as $\phi(x, h, w)$. However, unlike [30], where the transformed image maps to a single class label, ground-truth disparity y_{gt} is dense or semi-dense, so a matching transformation must be applied to y_{gt}. Moreover, the scale of y_{gt} also needs to be adjusted with respect to the resized width ($w \cdot W$) of the image. Hence, we extend diverse inputs to support stereo networks by:

$$\hat{x}_I = \phi(x_I, h, w), \tag{3.7}$$

$$\hat{v}_I = \phi(v_I, h, w), \tag{3.8}$$

$$\hat{y}_{gt} = w \cdot \phi(y_{gt}, h, w) \tag{3.9}$$

To incorporate this into iterative methods, we modify their respective gradient computations, g_I^{n+1}. For I-FGSM, we can re-write Eqn. 3.2 as:

$$g_I^{n+1} = \nabla_{\hat{x}_I} \ell(f_\theta(\hat{x}_L + \hat{v}_L^n, \hat{x}_R + \hat{v}_R^n), \hat{y}_{gt}), \tag{3.10}$$

Similarly, for MI-FGSM, we can modify Eqn. 3.4 to be:

$$g_I^{n+1} = \frac{\nabla_{\hat{x}_I} \ell(f_\theta(\hat{x}_L + \hat{v}_L^n, \hat{x}_R + \hat{v}_R^n), \hat{y}_{gt})}{\|\nabla_{\hat{x}_I} \ell(f_\theta(\hat{x}_L + \hat{v}_L^n, \hat{x}_R + \hat{v}_R^n), \hat{y}_{gt})\|_1}. \tag{3.11}$$

3.2 Experiment Setup

3.2.1 Datasets

We evaluate adversarial perturbations (robustness, transferability, defense) for recent stereo methods (PSMNet, DeepPruner, AANet) on the standard benchmark datasets: KITTI 2015 stereo [17] and KITTI 2012 [6] validation sets. KITTI 2015 is comprised of 200 training stereo pairs and KITTI 2012 consists of 194 (all at 376×1240 resolution) with ground-truth disparities obtained using LiDAR for outdoor driving scenes. Following KITTI validation protocol, the KITTI 2015 training set is divided into 160 for training and 40 for validation, and the KITTI 2012 training set is split into 160 for training and 34 for validation. Due to computational limitations, we downsampled all images to 256×640; hence, there are slight increases in errors (Eqn. 3.14) compared to those reported by baseline methods.

3.2.2 Implementation

We implemented our approach in PyTorch and used the publicly available code and pretrained models of PSMNet [1], DeepPruner [5], and AANet [32]. We are unable to obtain the necessary computational resources to craft adversarial perturbations for each of the models on the KITTI 2012 [6] and 2015 [17] datasets at full image resolution (376×1240). Therefore, we resize the images to 256×640 resolution for PSMNet and DeepPruner, and 252×636 for AANet, when we craft adversarial perturbations. Because of the resizing, the baseline errors we report are slightly higher than those reported by the authors of PSMNet, DeepPruner and AANet. We note that PSMNet and AANet released separate pretrained models for KITTI 2012 and 2015, but DeepPruner released a single model trained on both KITTI 2012 and 2015. Therefore, for our experiments on KITTI 2012, we use the KITTI 2012 models for PSMNet and AANet, and the KITTI 2012, 2015 model for DeepPruner. Additionally, DeepPruner provided two pretrained models: DeepPruner-Best and DeepPruner-Fast, both trained on KITTI 2012 and 2015. We used DeepPruner-Best for all our experiments. AANet also provided two model variants: AANet and AANet+. We used AANet for all our experiments.

3.2.3 Hyper-parameters

We study perturbations under four different upper norms, $\epsilon = \{0.02, 0.01, 0.005, 0.002\}$. $\epsilon = 0.002$ is where adversaries have little effect on the networks and $\epsilon = 0.02$ is the norm needed to achieve 100% errors on benchmark datasets. We show the hyper-parameters used for each perturbation method in Table 3.1. When optimizing with I-FGSM and DI^2-FGSM, we used $N = 40$ and $\alpha = 1/N \cdot \epsilon$ for $\epsilon = \{0.01, 0.005, 0.002\}$ and $\alpha = 0.10\epsilon$ for $\epsilon = 0.02$. For MI-FGSM and MDI^2-FGSM, $\alpha = 1/N \cdot \epsilon$ for all ϵ and chose $\beta = 0.47$ for momentum. We explored larger number of steps N, but found little difference. Smaller N results in less effectiveness. We investigated $\alpha \in [\epsilon, \frac{1}{N}\epsilon]$ and found that larger α for smaller ϵ tend to be ineffective. This is likely because the accumulated perturbations quickly saturate at a small ϵ due to clipping – additional steps nullify the effect of the perturbations. Hence we chose $\alpha = 1/N \cdot \epsilon$ for $\epsilon = \{0.01, 0.005, 0.002\}$. Within the search range of α, we found that $\alpha = 0.10$ performed the best (highest error) for $\epsilon = 0.02$.

	I-FGSM	MI-FGSM	DI^2-FGSM	MDI^2-FGSM
N	40			
α, $\epsilon = 0.002$	0.00005			
α, $\epsilon = 0.005$	0.000125			
α, $\epsilon = 0.01$	0.00025			
α, $\epsilon = 0.02$	0.002			
β	-	0.47	-	0.47
$h_{\min}$	-	-	0.90	0.90
$h_{\max}$	-	-	1.00	1.00
$w_{\min}$	-	-	0.90	0.90
$w_{\max}$	-	-	1.00	1.00
p	-	-	0.50	0.50

Table 3.1: Hyper-parameters used by iterative methods

3.2.4 Run-time

We used an Nvidia GTX 1080Ti for our experiments. Crafting perturbations using FGSM requires on average ≈0.87s in addition to the time needed for a forward pass through the stereo model (PSMNet, DeepPruner, AANet). I-FGSM on average requires an extra ≈29.68s and MI-FGSM requires ≈32.23s more. DI^2-FGSM and MDI^2-FGSM on average require ≈30.78s and ≈33.16s, respectively, in addition to the time needed for a forward pass. While these perturbations can degrade performance and also transfer to other models, they cannot be crafted in real time. Hence, in this work we focus on leverage adversarial perturbations to learn more robust stereo networks through adversarial data augmentation.

3.2.5 Evaluation

To evaluate the robustness of stereo networks, we use the official KITTI D1-all (the average number of erroneous pixels in terms of disparity and end-point error) metric:

$$\delta(i,j) = |f_\theta(\cdot)(i,j) - y_{gt}(i,j)|, \tag{3.12}$$

$$d(i,j) = \begin{cases} 1 & \text{if } \delta(i,j) > 3, \frac{\delta(i,j)}{y_{gt}(i,j)} > 5\%, \\ 0 & \text{otherwise} \end{cases} \tag{3.13}$$

$$\text{D1-all} = \frac{1}{\|\Omega_{gt}\|} \sum_{i,j \in \Omega_{gt}} d(i,j), \tag{3.14}$$

where Ω_{gt} is a subset of the image space Ω with valid ground-truth disparity annotations, $y_{gt} > 0$.

3.3 Results

3.3.1 KITTI 2015

We begin with naive attacks on stereo networks (PSMNet, DeepPruner, AANet) by perturbing the input stereo pair (x_L, x_R) with Gaussian $\mathcal{N}(0, (\epsilon/4)^2)$ and uniform $\mathcal{U}(-\epsilon, \epsilon)$ noise for $\epsilon \in \{0.02, 0.01, 0.005, 0.002\}$. Fig. 3.1-(a) shows that such noise cannot degrade performance as the measured error stayed approximately constant under various ϵ. All methods are thus robust to Gaussian (G) and Uniform (U) noise. This demonstrates that the deep features extracted for matching are robust to random noises and fooling a stereo network requires non-trivial perturbations. Hence, we examine the robustness of stereo networks against perturbations specifically optimized for each network using our variants of FGSM, I-FGSM, MI-FGSM.

FGSM. Fig. 3.1-(b) shows errors after attacking the networks with FGSM (Eqn. 3.1) where perturbations are optimized over a single time step. For large upper norm $\epsilon = 0.02$, the perturbations can degrade performance significantly – from 1.33% (AANet), 1.15% (DeepPruner) and 3.27% (PSMNet) mean error to 42.09%, 59.86%, and 97.33%, respectively. The larger the upper norm, the more potent the attack, but even with small $\epsilon = 0.002$, this attack can still increase AANet to 4.18% error, DeepPruner to 5.93%, and PSMNet to 38.11%. Fig. 3.2 shows a comparison of FGSM attacks on PSMNet

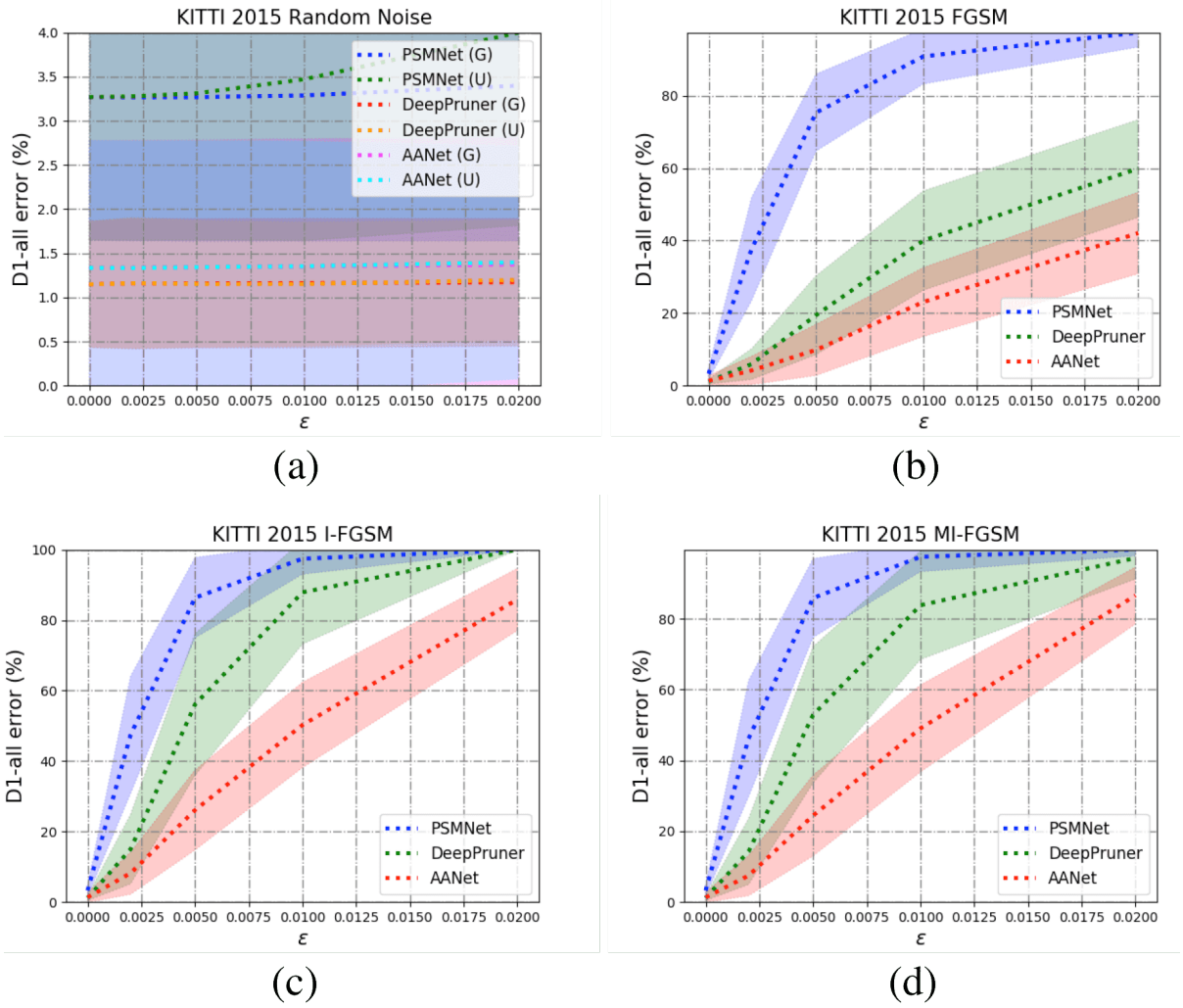


Figure 3.1: Attacks on Stereo Models for KITTI 2015

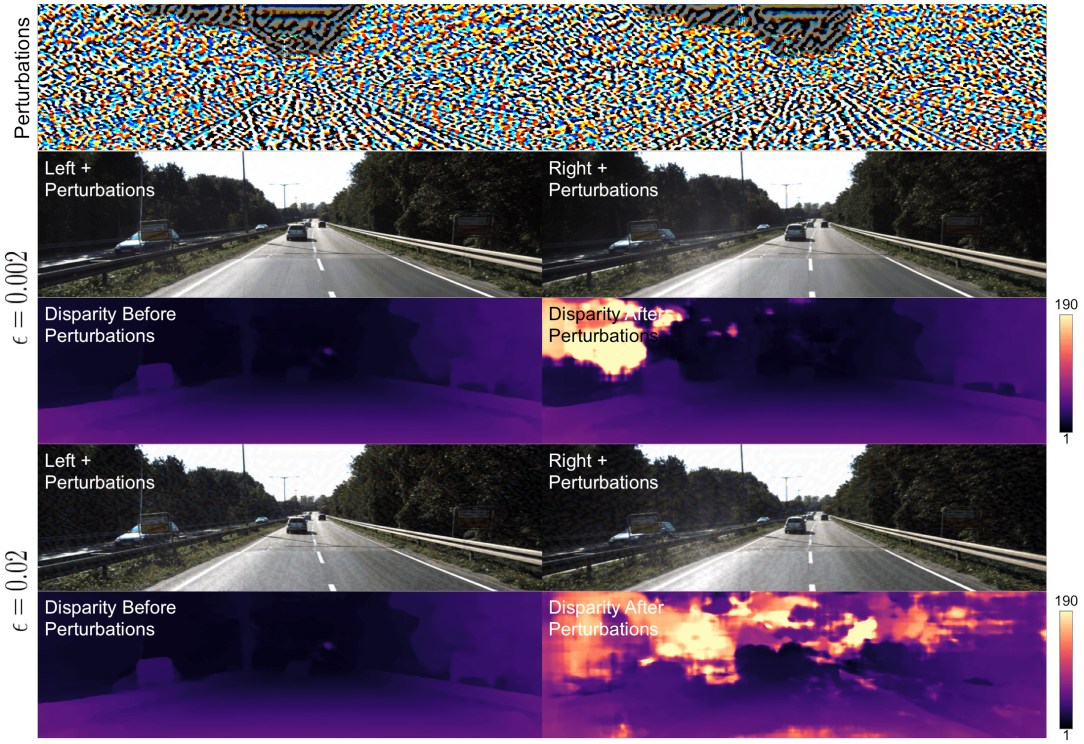


Figure 3.2: FGSM with upper norms of 0.002 and 0.02

using upper norms of 0.002 and 0.02. With visually imperceptible noise, $\epsilon = 0.002$, PSMNet is fooled to predict much larger disparities (closer depths) in the top left corner region. Using $\epsilon = 0.02$, the perturbations corrupt the geometry of the entire scene. For $\epsilon = 0.002$, most of the damage is localized (e.g. top left region of image space); whereas for $\epsilon = 0.02$, the entire predicted scene is corrupted. The localized damage from small norm perturbations can be attributed to the observability of the scene. We hypothesize that training affects inference where the radiance of the surfaces is not sufficiently exciting i.e. the regularizer fills in in a manner that depends on training experience. So, small norm perturbations can corrupt regions where the radiance is less informative (sky, uniform textures, foliage etc.); whereas, other regions require larger norms.

I-FGSM, MI-FGSM. Fig. 3.1-(c, d) show that I-FGSM and MI-FGSM both affect performance similarly. Because of the multiple optimization steps, when $\epsilon = 0.02$, even the more robust AANet succumbs to the attacks – increasing error to ≈87%. For PSMNet and DeepPruner, both I-FGSM and MI-FGSM can cause them to reach 100% error. AANet is consistently more robust to adversarial noise than PSMNet and DeepPruner. While we cap the upper norm at ϵ, we note that iterative methods are able to introduce

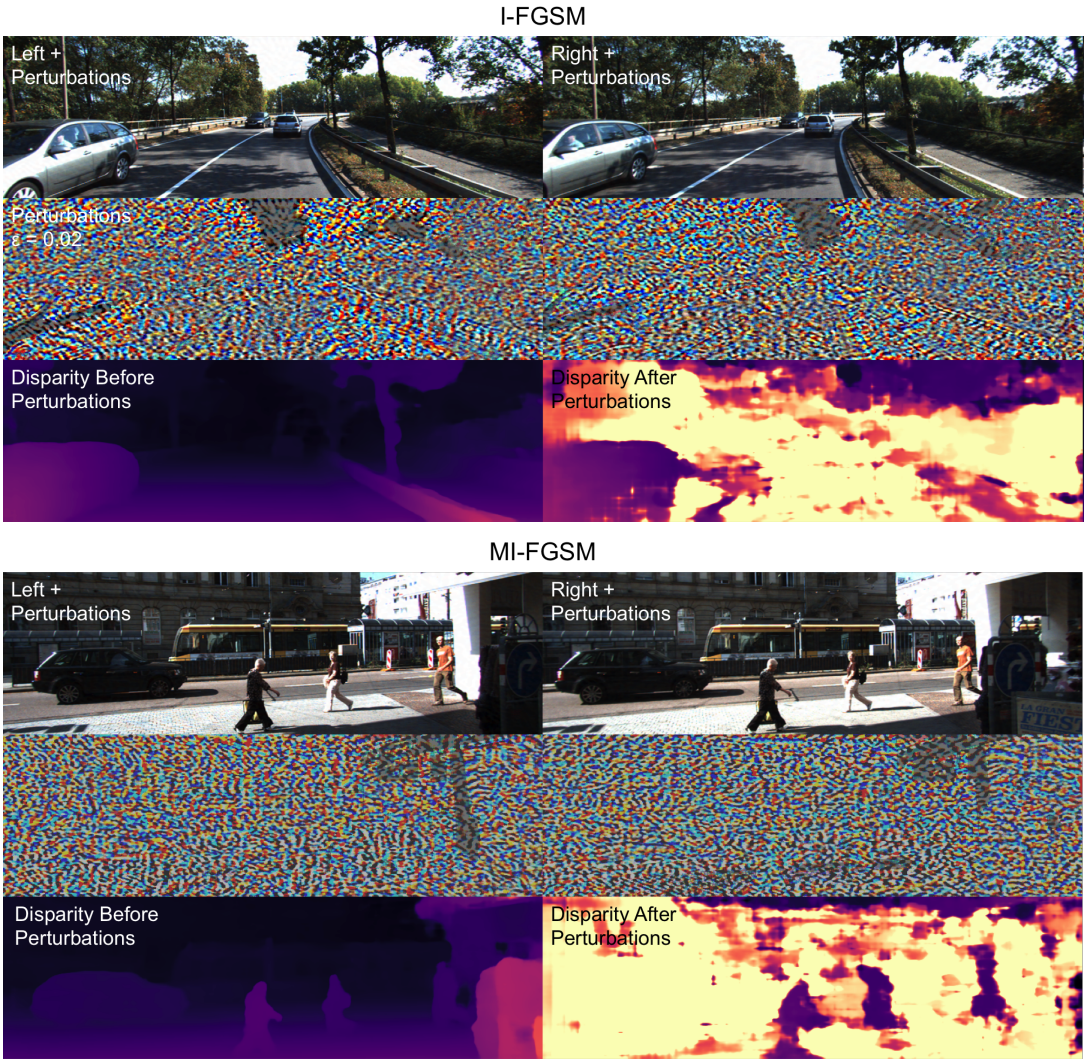


Figure 3.3: I-FGSM and MI-FGSM on PSMNet

more errors while using a much smaller perturbation magnitude. For $\epsilon = 0.02$, FGSM achieves 97.33% error on PSMNet with $\|v_L\|_1 = 0.0569$ and $\|v_R\|_1 = 0.0568$; whereas, I-FGSM achieves 100% error with only $\|v_L\|_1 = 0.0213$ and $\|v_R\|_1 = 0.0196$ – less than half of the L1 norm used by FGSM, making it less perceptible. Fig. 3.3 shows examples of I-FGSM and MI-FGSM on PSMNet. When compared to rows 2 and 4 of Fig. 3.2 ($\epsilon = 0.02$), both are less perceptible. Moreover, I-FGSM and MI-FGSM can fool the networks in textured regions where disparity can be obtained simply by matching. Unlike FGSM(Fig. 3.2) where most of the changes in disparity were concentrated on low-texture regions (no disparity signal and the perturbation drives the matching), perturbations crafted using iterative methods (Fig. 3.3) degrade high texture regions.

Interesting Observations. Even though error reaches 100% for I-FGSM and MI-

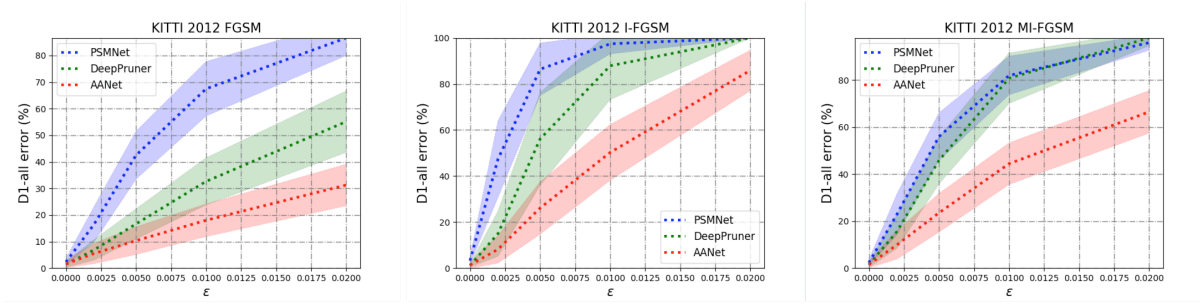
FGSM, the shape (albeit incorrect) of some salient objects like cars and humans still persists (Fig. 3.3). This is because disparity is largely driven by the data term and hence there exist unique correspondences for such objects with sufficiently exciting texture. However, while the general shape persists, the disparity is incorrect (as it is filled in by the regularizer). Another phenomenon is that the noise required to perturb white regions (white walls, sky, Fig. 3.2, Fig. 3.3) is much less than that required to attack other color intensities (e.g. uniform black). While radiance (being less informative) is a factor, we hypothesize that this is a special case due to white regions being on the upper support of image intensities, which results in high activations; hence, the adversary only needs to add small noise to adjust the activations to the needed values to corrupt the scene. We will leave the numerical analysis of this "white-pixel" phenomenon to future work.

Thus, stereo networks are indeed vulnerable to adversarial perturbations. Each architecture exhibits different levels of robustness against adversaries. Feature stacking (PSMNet) is the least robust, followed by patch-matching (DeepPruner) with correlation (AANet) being the most robust. This is because DeepPruner and AANet both find correspondences based on similarity between deep features via explicit matching (well-defined data fidelity, so the perturbations corrupt the regularizer); whereas PSMNet relies purely on learned convolutional filters to produce matches and is more susceptible to the attacks.

3.3.2 KITTI 2012

We repeat all experiments performed on the KITTI 2015 dataset for KITTI 2012 dataset. Fig. 3.4 shows the effect of perturbations crafted using FGSM, I-FGSM, and MI-FGSM. Fig. 3.5 shows representative examples of FGSM and I-FGSM attacks on DeepPruner and AANet, respectively.

For FGSM attacks (Fig. 3.4-(a)), PSMNet proves to be the most susceptible as perturbations with $\epsilon = 0.002$ can degrade performance to 16.45% and with $\epsilon = 0.02$ error increases to 86.16%. For I-FGSM (Fig. 3.4-(b)), perturbations can achieve 100% error on both PSMNet and DeepPruner using the highest upper norm and 86.07% on AANet. Fig. 3.4-(c) shows perturbations optimized using MI-FGSM achieves 95.95%, 97.95%, and 66.49% on PSMNet, DeepPruner and AANet, respectively. In Fig. 3.5, on the left



(a) (b) (c)

Figure 3.4: Attacks on Stereo Models for KITTI 2012

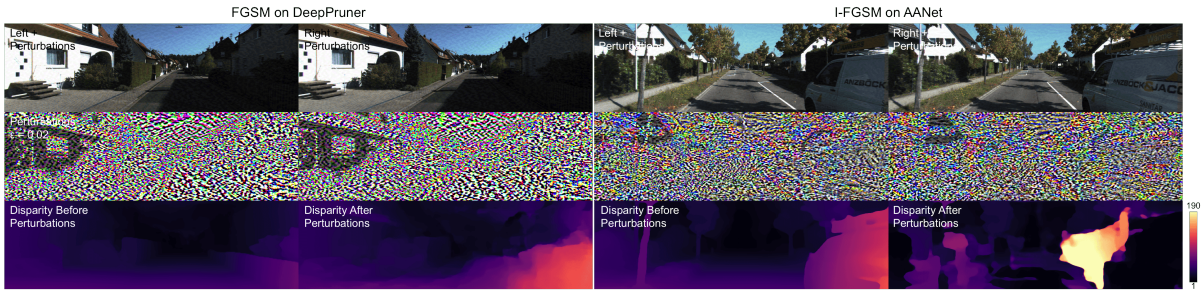


Figure 3.5: Qualitative examples of FGSM and I-FGSM attacks on stereo networks for KITTI 2012

we show an FGSM attack on DeepPruner using $\epsilon = 0.02$. The perturbations mainly corrupted the right side of the output scene. On the right, we show an I-FGSM attack on AANet using $\epsilon = 0.02$. Unlike the FGSM attack, which tends to be localized, the I-FGSM attack corrupts the entire scene. In general, explicit matching methods (DeepPruner and AANet) are more robust than implicit matching or feature stacking methods (PSMNet). Overall, AANet is the most robust out of the three stereo models evaluated. This is similar to the results demonstrated for KITTI 2015.

3.3.3 Attacking Only One Image

We demonstrate on the KITTI 2015 dataset that it is possible to fool a stereo network by only perturbing one of the two images in a stereo pair (either the left, $v_R = 0$, or the right, $v_L = 0$, image only). Even with perturbations on one of the two images, we can still fool the network into predicting drastically incorrect depths.

Fig. 3.6 shows a quantitative comparison between attacking both images versus attacking just one of them on the KITTI 2015 dataset. Blue curve shows the error when

attacking both images, green curve shows the error when attacking just the left image ($v_R = 0$) and red curve just the right image ($v_L = 0$). Surprisingly, perturbing just one of image is sufficient to fool stereo network into predicting incorrect disparities – although consistently with less effectiveness than perturbing both images. Perturbing both images using **(a)** FGSM, **(b)** I-FGSM, and **(c)** MI-FGSM is consistently more effective than perturbing just one of them. We note that perturbing just the left image using FGSM on PSMNet yields similar results (96.98%) at the largest upper norm $\epsilon = 0.02$ (note that FGSM attacks are much weaker against DeepPruner and AANet in this setting). While for smaller upper norms, all attacks on a single image are less effective than attacking both images, we note that iterative methods (I-FGSM, MI-FGSM) using $\epsilon = 0.02$ can achieve similar error percentages to attacking both images for PSMNet and DeepPruner. When attacking one of the two images, trends similar to those for attacking both images are present. The larger the norm, the more effective the perturbations. AANet proves to be more robust than PSMNet and DeepPruner in this problem setting.

Fig. 3.7 shows I-FGSM attacks on PSMNet using $\epsilon = 0.02$ where the perturbations are only located on either the left and right images. As we can see, the network is still fooled into predicting incorrect disparities. We note that the shapes of salient objects observed in Fig. 3.3 are also observed here. While we see the shape of the vehicle on the left panel, the disparities of the vehicle are incorrect. This is likely due to the vehicle being co-visible and hence the shape is observed. This also shows that the network is able to learn general shapes of objects that exist in the scene.

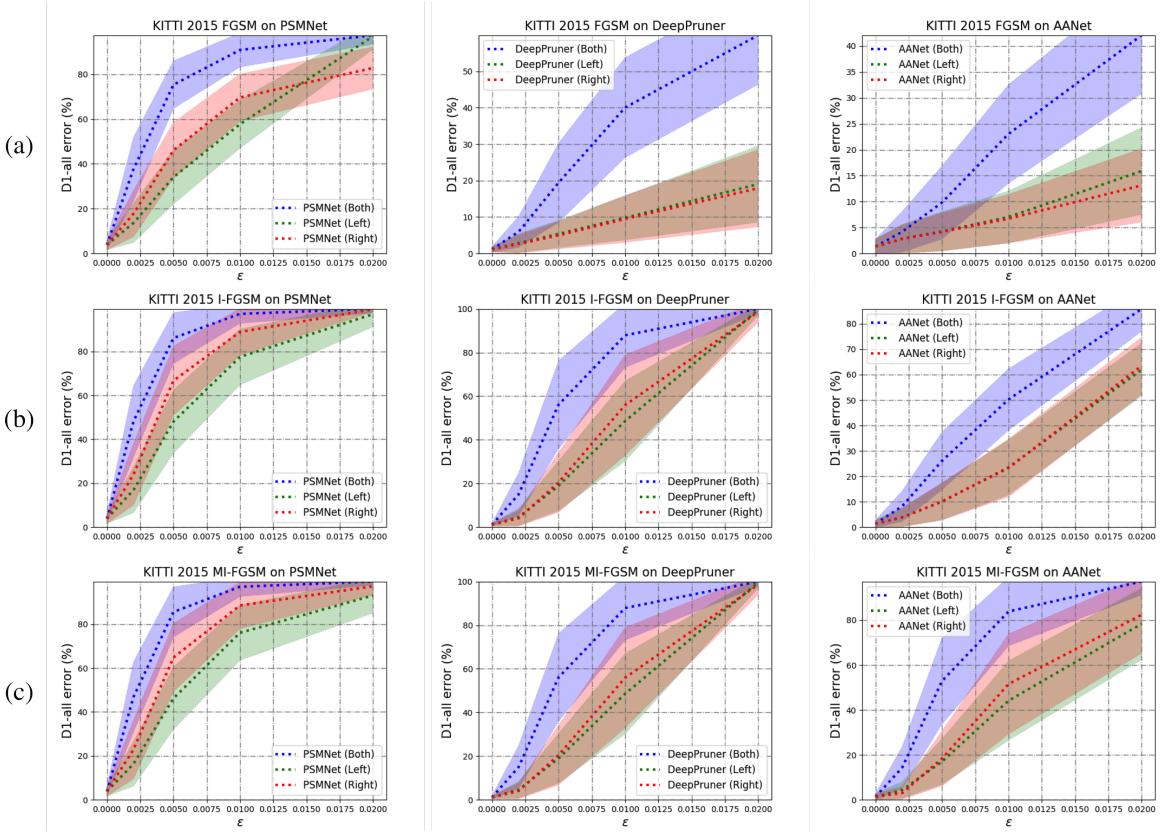


Figure 3.6: Comparison between attacking both images and attacking just one of the two images

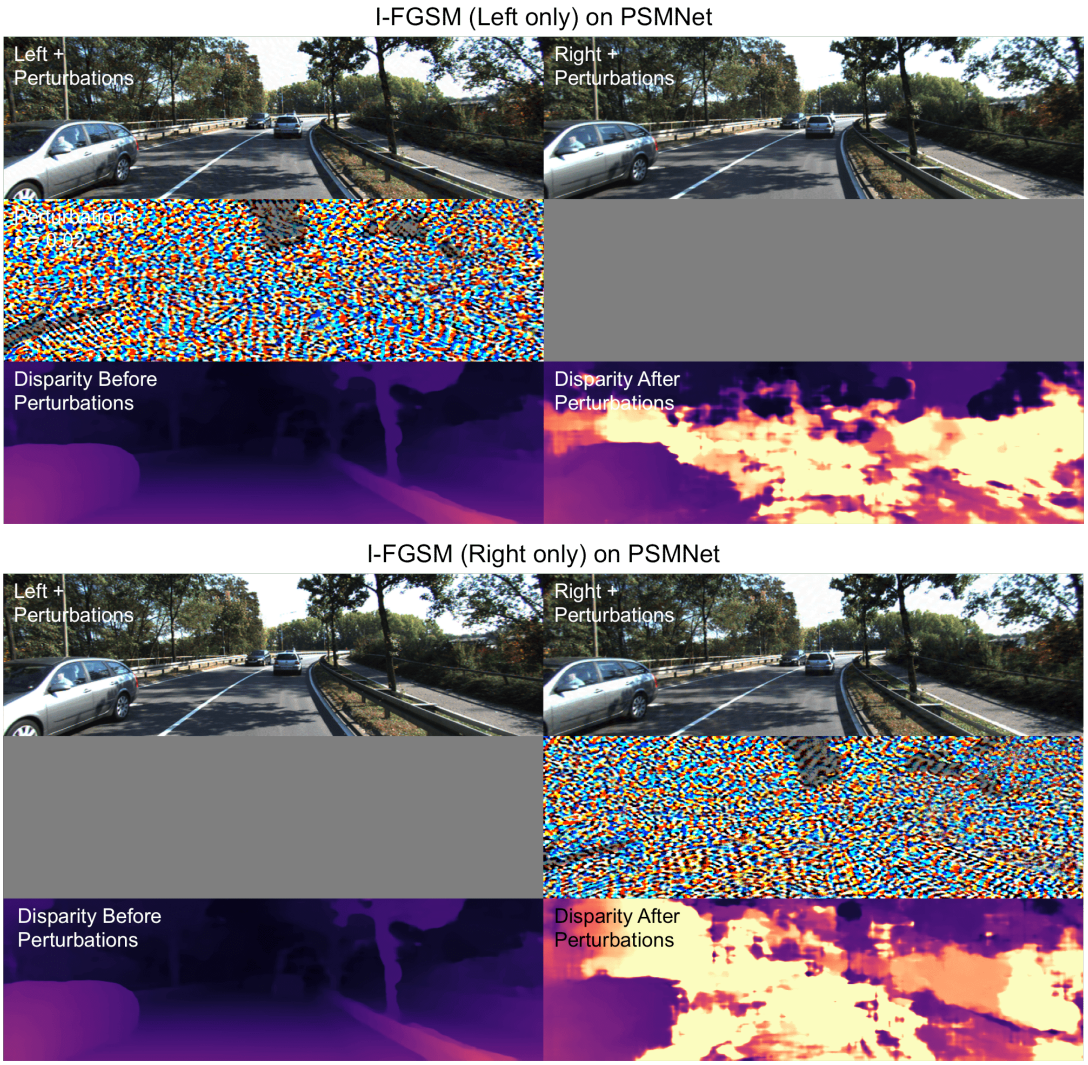


Figure 3.7: I-FGSM attacks on stereo networks for KITTI 2015 by perturbing only the left or right image

CHAPTER 4

Transferability Across Models

To study transferability, we (i) optimize perturbations for PSMNet, DeepPruner and AANet separately, (ii) add each set of model-specific perturbations to the associated input stereo pair for another model i.e. add perturbations for PSMNet to the inputs of DeepPruner and AANet, and (iii) measure the resulting error using Eqn. 3.14.

4.1 FGSM-based Methods

We begin with methods studied above. Fig. 4.1-(a) shows the transferability of FGSM from different models (red, green), for which the perturbations were optimized, to a target model (blue). We found that the perturbations do transfer, but with reduced effects e.g. for $\epsilon = 0.02$, perturbations optimized for DeepPruner and AANet achieve 55.47% and 53.55% error on PSMNet, respectively, while perturbations optimized for PSMNet achieves 97.33%. The potency of the perturbations also grows with the upper norm; hence, one can increase ϵ of an adversary to further degrade new models.

Fig. 4.1-(b) shows the transferability of I-FGSM and Fig. 4.1-(c), MI-FGSM. Unlike FGSM, iterative methods transfer much less. For instance, I-FGSM ($\epsilon = 0.02$) perturbations for DeepPruner and AANet achieve 31.20% and 48.08%, respectively, on PSMNet; whereas, FGSM achieves 55.47% and 53.55%, respectively. In general, iterative methods transfer less than FGSM because the perturbations tend to overfit to the model they were optimized for. We note that AANet is the most robust against perturbations from other models and yet has the highest transferability, which interestingly shows that transferability is not symmetric. While perturbations for DeepPruner and AANet achieve 31.20% and 48.08% on PSMNet, PSMNet and AANet achieve 13.74% and 22.25% on DeepPruner, and those for PSMNet and DeepPruner only achieve 8.46% and 5.81% on

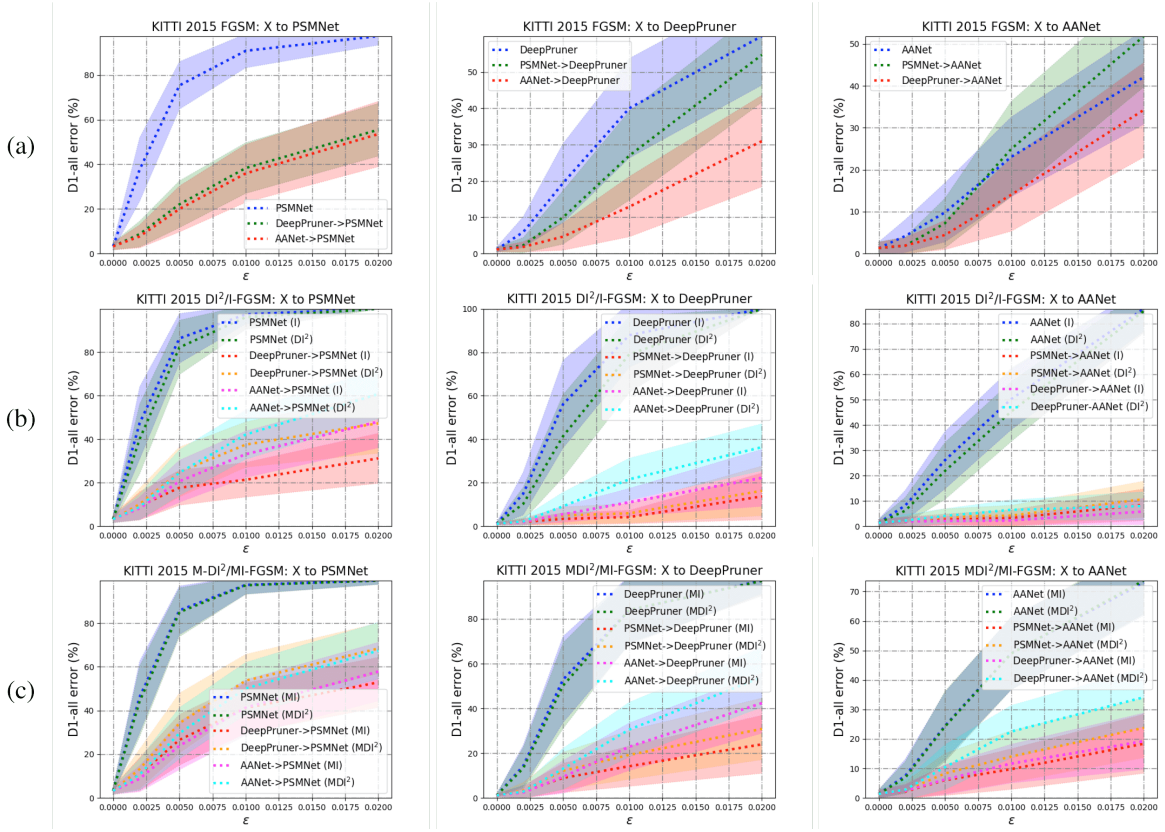


Figure 4.1: Transferability for KITTI 2015

AANet.

4.2 Improvement with Input Diversity

To increase transferability to other stereo networks, we additionally optimized perturbations using DI^2-FGSM and MDI^2-FGSM. Fig. 4.1-(b) shows that DI^2-FGSM (green) consistently degrades the target model's performance less than I-FGSM (blue). This is largely due to the noise in the gradients introduced by random resizing and padding. Fig. 4.1-(c) shows that perturbations from MDI^2-FGSM achieve errors similar to MI-FGSM since each iteration still retains the momentum from previous time steps.

Fig. 4.1-(b, c) shows that DI^2-FGSM and MDI^2-FGSM consistently transfer better to new models than I-FGSM and MI-FGSM. The best performing iterative method is MDI^2-FGSM, which not only achieves comparable numbers to MI-FGSM on the model it is optimized for, but also transfers well to new models. We note the trends observed in I-FGSM and MI-FGSM are also observed here. In Fig. 4.2, we craft perturbations

Figure 4.2: Transferability between AANet and DeepPruner

for AANet and DeepPruner using I-FGSM and DI^2-FGSM with $\epsilon = 0.02$. DI^2-FGSM transfers better than I-FGSM. Perturbations crafted for AANet transfer well to DeepPruner (AANet→DeepPruner), but those for DeepPruner have less effect on AANet (DeepPruner→AANet).

4.3 Discussion

While the mere existence of adversaries indicates (possible common) flaws in stereo networks, the reason that perturbations are transferable is because disparity is generic i.e. a surface 1m away generates the same disparity whether it belongs to a cat, a dog or a tree. Yet, transferability is not symmetric and AANet is yet again the most robust with the highest transferability. Fig. 4.2 shows that perturbations optimized for AANet fools DeepPruner, but those optimized for DeepPruner have less effect on AANet. We hypothesize that architectural differences between AANet (2D convolutions) and PSMNet and DeepPruner (3D convolutions) play a role in transferability. A possible reason for why perturbations for AANet transfers better to others (but less the other way around) may be because they are optimized to attack 2D convolutional layers, which PSMNet and Deeppruner also use to build their cost volumes. However, perturbations for PSMNet and Deeppruner are optimized to disrupt 3D convolutions as well, which are not present in AANet.

4.4 Results for KITTI 2012

Fig. 4.3-(a) shows the transferability of perturbations crafted by FGSM. Unlike our findings in the transferability experiments conducted on the KITTI 2015 dataset, PSMNet transfers the best to DeepPruner and AANet, matching performance for perturbations optimized for DeepPruner and outperforming perturbations optimized for AANet. The latter is an *interesting phenomenon* also observed in the KITTI 2015 dataset – perturbations crafted for PSMNet using FGSM are more effective than those crafted for AANet when applied to AANet. This phenomenon is isolated to only this case and we hypothesize that this is likely due to architectural similarities in the feature extraction step (general enough to DeepPruner and AANet, but not the other way around because DeepPruner and AANet features are more specific to explicit patch matching architecture). We leave additional analysis on why this occurs to future work.

Fig. 4.3-(b) shows I-FGSM and DI^2-FGSM attacks on stereo networks. Similar to our findings in KITTI 2015, I-FGSM perturbations crafted for a specific model are much less effective than FGSM when transferred to another. This is due to overfitting to the network it is optimized for. Input diversity improves transferability. Unlike Fig. 4.1-(b) for KITTI 2015, perturbations crafted for PSMNet transfer the best here. Similar trends can be observed in Fig. 4.3-(c). Perturbations crafted using momentum methods transfer the best. Transferability between models is, again, not symmetric. While AANet is still the most robust out of the three models, PSMNet transfers the best for KITTI 2012.

4.5 Transferring Perturbations to Next Frame

To investigate whether or not the adversarial perturbations crafted for a specific stereo pair is pathological – in that they can only do damage to the stereo pair that they are optimized for, we perform a simple experiment of transferring (adding) the perturbations crafted for a stereo pair at time step t to the temporally adjacent stereo pair at time step $t+1$. Because groundtruth disparity is not available for stereo pairs taken at time $t+1$, we only evaluate this section qualitatively in Fig. 4.4. Fig. 4.4 shows examples of stereo pairs at time $t+1$ with added perturbations crafted for the stereo pair at time t, and the

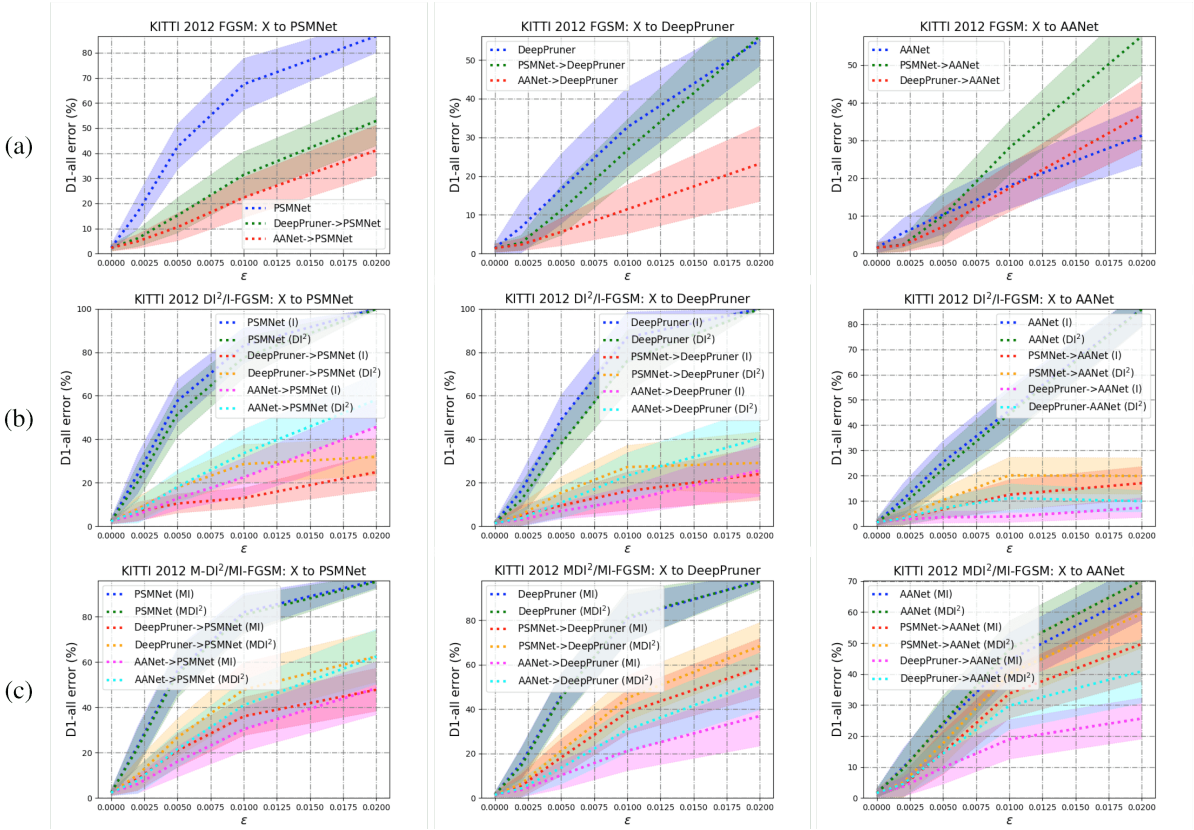


Figure 4.3: Transferability for KITTI 2012

predictions for the images taken at $t+1$ – before and after adding the perturbations.

Fig. 4.4 shows that the perturbations have the ability to transfer to the frames taken at the next time step; however, it depends on the structures in the scene. For this experiment, we crafted FGSM perturbations for PSMNet, DeepPruner and AANet using the smallest norm $\epsilon = 0.002$. Because the perturbations tend to concentrate around edges, we chose the smallest upper norm such that mis-match in edges between the stereo pair at time t and the next time step $t+1$ will not cause perturbations to be visually perceptible. We found that depending on the scene, the perturbations can transfer. For instance, if the perturbations are added onto similar structures that are present in the previous time step, then the effect remains intact. However, if the motion is large and the intended structures are occluded or new structures are disoccluded, then it loses the effect. Fig. 4.4 demonstrates this phenomenon. The predictions for the sky and some of the vegetation regions are still corrupted by the perturbations crafted for the stereo pair from the previous time step. Based on our observations, we do not expect the perturbations optimized for one stereo pair to retain the same effect on a stereo network

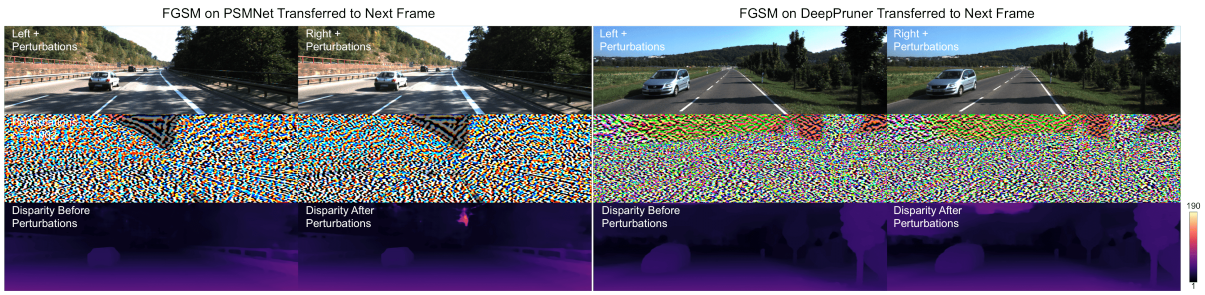


Figure 4.4: Investigating whether an adversarial example crafted for a specific stereo pair is pathological

when added to a different stereo pair of a different scene.

CHAPTER 5

Defenses Against Adversaries

We begin with a basic defense, Gaussian blur, against adversaries. Fig. 5.1-(a) shows Gaussian blurring (3×3 kernel) with various σ does not mitigate the effect of adversaries (top), but exacerbates them – further degrading performance. In addition, simply applying Gaussian blur on clean images (bottom) also decreases performance. Hence, we aim to learn more robust representations by harnessing adversarial examples to fine-tune the models. Fig. 5.1-(b, c, d) show the performance of stereo methods after fine-tuning on a combination of clean and perturbed images (using FGSM with various ϵ). As a sanity check, we also fine-tuned on just clean images ($\star$) to ensure that differences are result of adversarial examples.

5.1 Training

For defending against adversaries, we fine-tuned the models on a combination of clean and perturbed images (using FGSM with various ϵ). We used 160 images (and their perturbed versions) from the KITTI 2015 training set for fine-tuning, and the remaining 40 stereo pairs (and their perturbed versions) for the validation set. A similar distribution was used for experiments on KITTI 2012 dataset as well (34 stereo pairs instead of 40). All images (clean and perturbed) were resized to 256×640 resolution. PSMNet and DeepPruner took a 256×512 crop of the image during training, while AANet took a 252×636 crop of the image. We chose a learning rate of 0.001 for PSMNet and 0.0001 for DeepPruner and AANet after experimenting with smaller and larger learning rates. We did not use pseudo ground truth supervision in AANet during fine-tuning. We fine-tuned the models for 150 epochs.

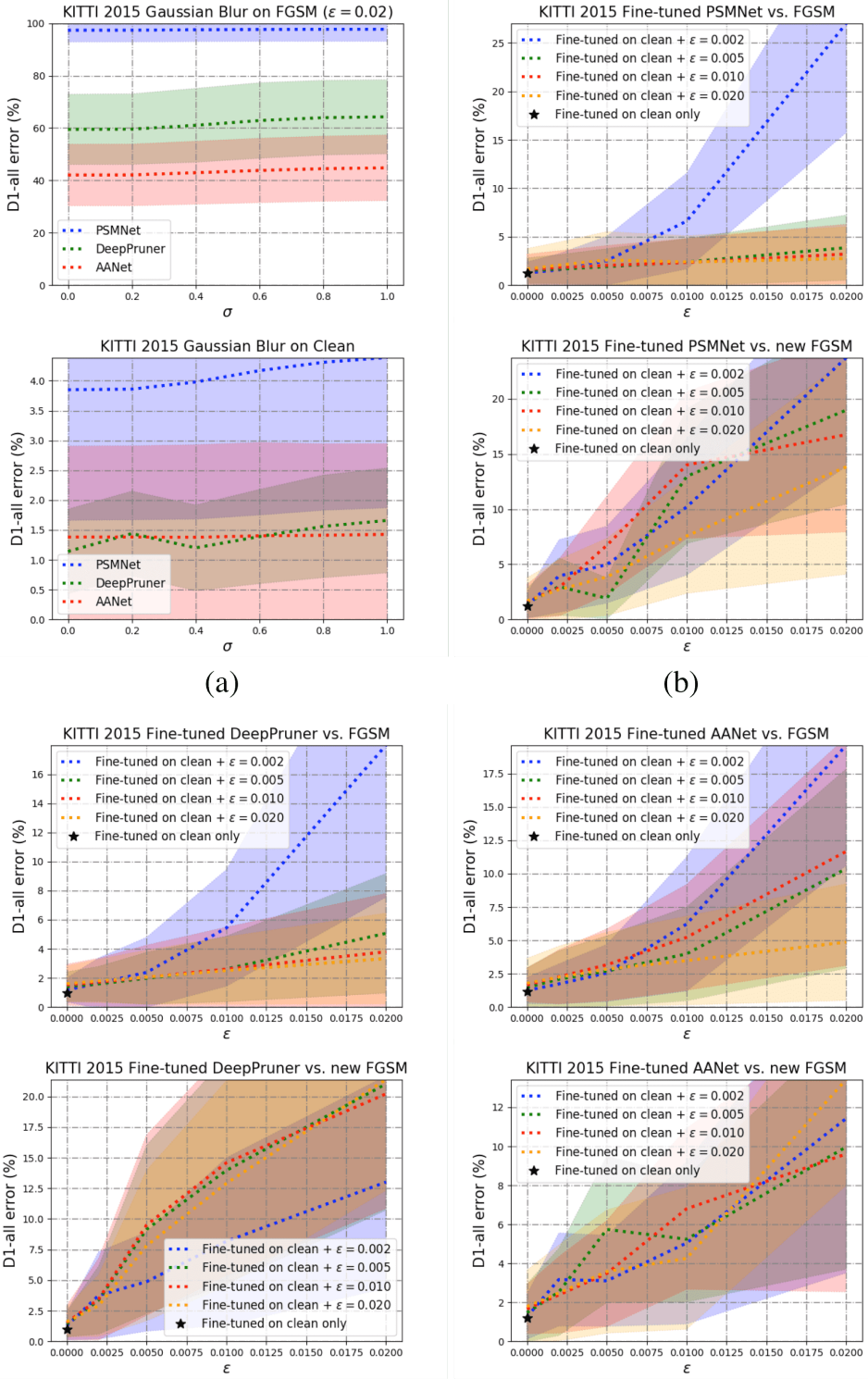


Figure 5.1: Defenses against attacks for KITTI 2015

5.2 Results on KITTI 2015

Adversarial data augmentation increases robustness for all models. For FGSM $\epsilon = 0.02$, PSMNet decreases error from 97.33% (Fig. 3.2) to 2.74% against the adversary it is trained on. Moreover, training on a smaller norm ($\epsilon = 0.002$) can increase robustness against larger norm ($\epsilon = 0.02$) attacks e.g. FGSM $\epsilon = 0.02$ can degrade PSMNet to only 27.03% error. Also the models are more robust against new adversaries. For this, we attacked each fine-tuned model and found that a new adversary (FGSM $\epsilon = 0.02$) can only degrade a PSMNet trained on FGSM $\epsilon = 0.02$ to 13.84% error and 23.74% when PSMNet is trained on FGSM $\epsilon = 0.002$. We also observe these trends in DeepPruner and AANet (Fig. 5.1-(c, d)).

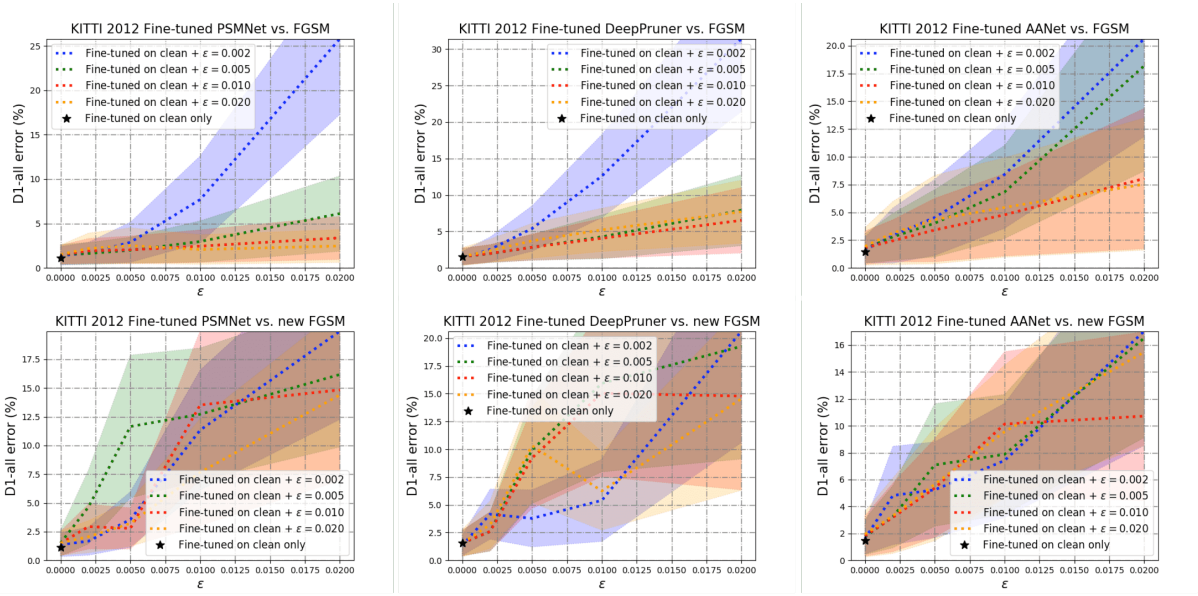
Contrary to findings reported in classification [7, 13], augmenting the training set with adversarial examples have little negative effect on performance of stereo models for clean images. When training with $\epsilon = 0.002$ (blue), performance for PSMNet and AANet are essentially unchanged (compared to $\star$); for the largest $\epsilon = 0.02$ (orange), errors increased by $\approx$0.4%. The smaller the norm, the less it affects performance on clean images. This is likely due to the mis-match in image intensity distributions between natural and adversarial examples. To avoid loss in performance, one can train on $\epsilon = 0.002$ and still observe the benefits on robustness. Fig. 5.1-(b, c, d) shows that all models are (i) robust to perturbations at $\epsilon = 0.002$ and 0.005, (ii) comparable to leveraging larger norm perturbations when facing new adversaries, and (iii) retains original performance on clean images.

While training on larger norms increases robustness against both existing and new adversaries, the model generally performs worse against a new adversary. For $\epsilon = 0.02$, a fine-tuned PSMNet achieves 2.74% against the adversary it is trained on and 13.84% against a new adversary; similarly, DeepPruner achieves 3.33% and 21.39% respectively. In contrast, when training on smaller norms ($\epsilon = 0.002$), the model keeps the same level robustness against existing and new adversaries. In fact, both PSMNet and DeepPruner perform better against new adversaries. For FGSM $\epsilon = 0.02$, PSMNet fine-tuned on $\epsilon = 0.002$ achieves 27.03% against the existing adversary and 23.74% against a new adversary; similarly, DeepPruner achieves 17.99% and 13.01%, respectively. This phenomenon is

likely because the network is overfitting to the intensity patterns of the large norm noise (also related to the slight degrade in performance). But for small norms, the network learns the underlying pattern without needing to alter its decision boundaries significantly since the intensity distribution is closer to natural images – resulting in a more regularized model. Perhaps a strategy to learning robust stereo networks is to iteratively craft various small norm perturbations and train on them with a mixture of clean images.

5.3 Results on KITTI 2012

We repeat all experiments performed on the KITTI 2015 dataset for the KITTI 2012 dataset. Fig. 5.2 shows the errors of each stereo method, after fine-tuning on an adversary of a specific upper norm, when attacked by adversaries optimized for the original model (top row) and by new adveraries optimized for the fine-tuned model (bottom row). Again, as a sanity check to ensure that any performance difference is due to the adversarial data augmentation, we also fine-tuned each method on the clean data (denoted as $\star$). Fig. 5.2 shows that after training on a mixture of clean and adversarial perturbed images, all methods are now more robust against the adversary designed for the original model. Moreover, all methods are also more robust against new adversaries. Unlike findings reported in adversarial works in classification, training on adversarial examples does not compromise performance on clean images when using smaller norm ($\epsilon \in \{0.002, 0.005\}$) perturbations; when training on larger norm perturbations ($\epsilon \in \{0.01, 0.02\}$), performance only degrades slightly (performance of all methods on clean images is very close to $\star$ in Fig. 5.2), where the change in error is $\approx$0.4%. This is likely due to the observability of 3D from the input stereo pair e.g. one does not need to learn stereo, classic matching methods can estimate disparity without any learning. Hence, the adversarial examples serve as regularization. We note while the change in performance for larger norm is small, it is nonetheless performance degradation; we hypothesize that this is due to the mis-match in intensity distribution between the clean and perturbed images.



(a) (b) (c)

Figure 5.2: Training with adversarial data augmentation for KITTI 2012

CHAPTER 6

Discussion

6.1 Summary

Stereo networks are indeed vulnerable to adversarial perturbations. This is unexpected because the problem setting is quite unlike adversarial perturbations for single image tasks where there is no unique minimizer (a single image does not constraint the latent, the training set does). Here, the geometry of the scene can be directly observed in co-visible regions as the data term is well defined and would have a unique minimizer under mild conditions. This means that stereo matching does not require learning; learning affects regularization. So it is surprising that, despite a uniquely identifiable latent variable (disparity), the training manages to produce such a strong bias that makes the overall system susceptible to perturbations, and local perturbations to boot. What is more interesting is that, not only can these perturbations drastically alter predictions on the stereo models they are optimized for, they can also transfer across models (although with reduced potency). However, given that it is rare for a malicious agent to have full access to a network and its loss, these attacks are not feasible in practice. Yet, the fact they exist gives us an opportunity to leverage them offline and train more robust stereo networks.

Previous works in single image based tasks have demonstrated that augmenting the training set with adversarial examples can improve robustness, but at the expense of performance on clean images. Yet, for stereo networks, we show that adversarial data augmentation can increase robustness *without* compromising performance on clean images – critical for designing robust and accurate systems. This too is likely related to the observability of the scene geometry from images where texture is sufficiently exciting. So, whereas in single image based tasks, training with adversarial perturbations alters

the low-level filters to the point of hampering precision, in stereo precision is dictated by the data term, which is largely unaffected by training (correlation is an architectural inductive bias in deep stereo matching networks and is precisely why DeepPruner and AANet use it). While indeed, adversarial perturbations wreck havoc (reaching as much as 100% in D1-all error) on networks trained only on clean images, stereo networks can recover by learning the distribution of adversarial noise through data augmentation with adversarially perturbed images and the matching process takes care of the rest.

6.2 Future Work

Our work here is just a first step. We only studied transferability across models and not datasets. We also do not consider the universal setting, where a constant additive image can degrade performance across all images within a dataset. Computationally, crafting perturbations using iterative methods adds an average of ≈29s on top of the time needed for forward passes; hence, they cannot be computed in real time. Amongst our findings, we also observed the "white-pixel" phenomenon, where very little perturbations are needed to alter regions with white pixels. This is an interesting phenomenon that is present across all methods. We believe this is due to white being on the upper support of image intensities; we leave the numerical analysis of this to future work. While there is still much to do, we hope that our work can lay the foundation for harnessing adversarial perturbations to train more robust stereo models.

6.3 Ethical Impact

As deep learning models are widely deployed for various tasks, adversarial examples have been treated as a threat to the security of such applications. While demonstrating that adversaries exist for stereo seems to add to this belief (since stereo is widely used in autonomous agents), we want to assure the reader that these perturbations cannot (and should not) cause harm outside of the academic setting. Cameras are not the only sensors on an autonomous agent, they are generally equipped with range sensors as well. Hence, corrupting the depth or disparity map will not cause the system to fail since it can still

obtain depth information from other sources. Also, as mentioned in our Introduction section, crafting these perturbations is computationally expensive and cannot be done in real time. Thus, we see little opportunities for negative ethical implications, but of course where there is a will there is a way.

More importantly, we see adversarial perturbations as a vehicle to develop better understanding of the behavior of black-box models. By identifying the input signals to which the output is most sensitive, we can ascertain properties of the map, as others have recently begun doing by using them to compute the curvature of the decision boundaries, and therefore the fragility of the networks and the reliability of their output.

What we want to stress in this work is that the mere existence of these perturbations tells us that there is a problem with the robustness of stereo networks. Therefore, we treat them as an opportunity to investigate and ultimately to improve stereo models. Our findings in our Defenses against Adversaries section shed light on the benefits of harnessing adversarial examples and potential direction towards more robust representations.

CHAPTER 7

Conclusion

As Deep Neural Networks become more widely used in critical applications, there is a need to re-examine the robustness of these networks. Adversarial perturbations have been used to test the fragility of such networks. Given a rectified stereo pair of images, deep stereo networks predict the disparity of each pixel in the image. The existence of adversarial perturbations for stereo is unexpected because unlike single image tasks, stereo does not require learning (we only use learning for regularization). Since the geometry of the scene is directly observable, there exists a unique minimizer for co-visible regions. Thus, it is surprising when we demonstrate the existence of adversarial perturbations for three popular stereo methods namely PSMNet, DeepPruner, and AANet. We generate these perturbations by repurposing FGSM based attacks from the single image case to stereo which is a dense regression task. We show that these attacks can wreck havoc on the performance of the stereo networks. We also demonstrate that these attacks are transferable, that is, they are also effective (although with reduced potency) against a model with a different architecture than the one they were designed for. Transferability of these perturbations is, however, not symmetric. Finally, we propose to use these perturbations for adversarial data augmentation to improve the robustness of stereo networks. We show that fine-tuning with a mixture of clean and perturbed images makes the model more robust to old and new adversaries, without a trade-off in the performance on clean images. We present this work as a first step towards building more robust representations for stereo.

APPENDIX A

Additional Examples

In the main text, we have primarily shown examples of perturbations with upper norms $\epsilon \in \{0.002, 0.02\}$ (smallest and largest norms considered) crafted using various methods. Fig. A.1 to A.10 show examples of attacks on PSMNet, DeepPruner, and AANet using $\epsilon \in \{0.005, 0.01\}$. Fig. A.1 and A.2 compare FGSM attacks for the two norms on the same stereo pairs, Fig. A.3 and A.4 compare I-FGSM attacks, Fig. A.5 and A.6 compare MI-FGSM attacks, Fig. A.7 and A.8 compare DI^2-FGSM attacks and lastly, Fig. A.9 and A.10 compare MDI^2-FGSM attacks. Increasing the upper norm from 0.005 to 0.01 does not make the perturbations visible; however, it does increase the effect of perturbations to fool the stereo networks into predicting drastically different scenes.

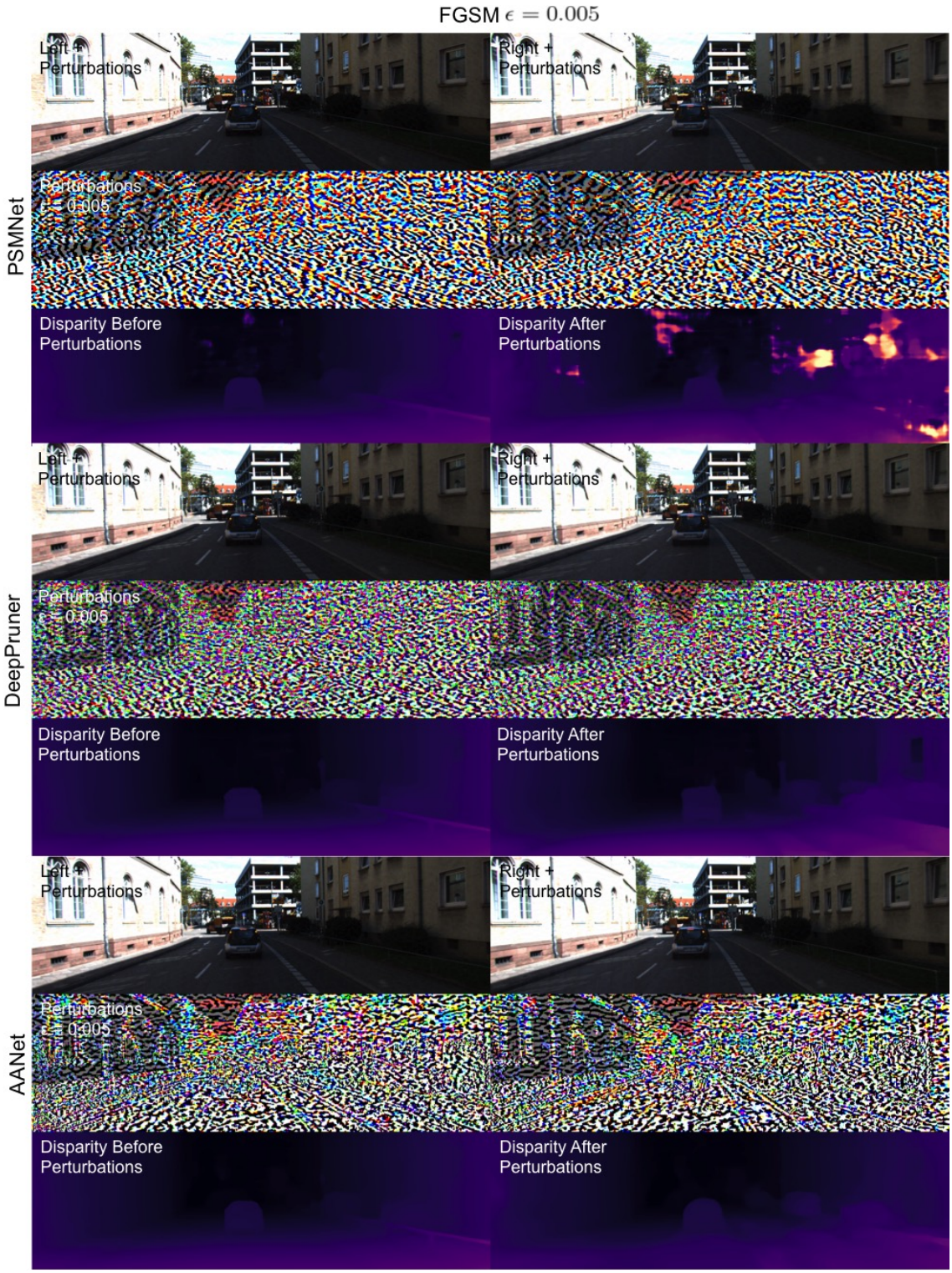


Figure A.1: Examples of FGSM attacks using $\epsilon = 0.005$.

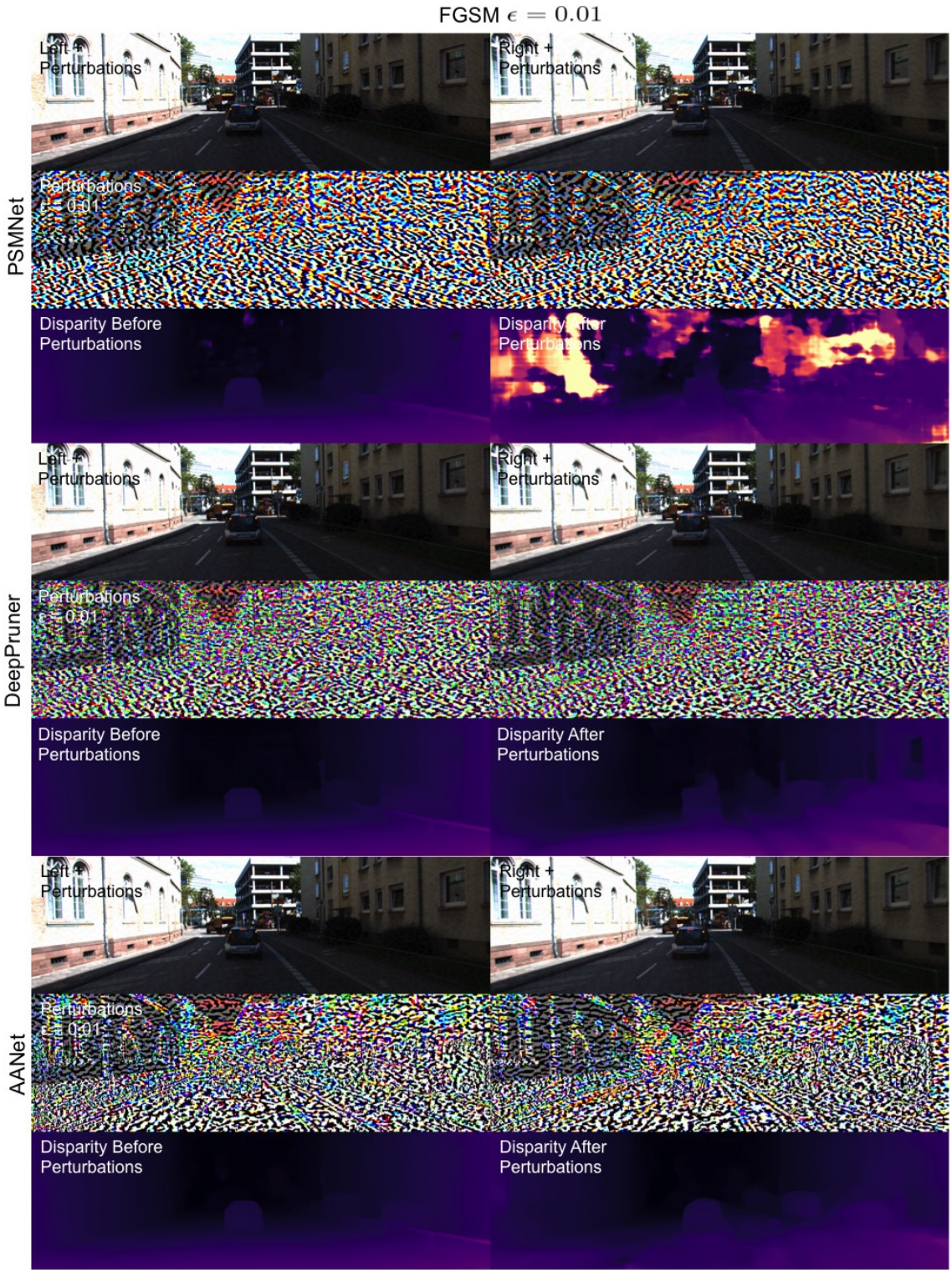


Figure A.2: Examples of FGSM attacks using $\epsilon = 0.01$.

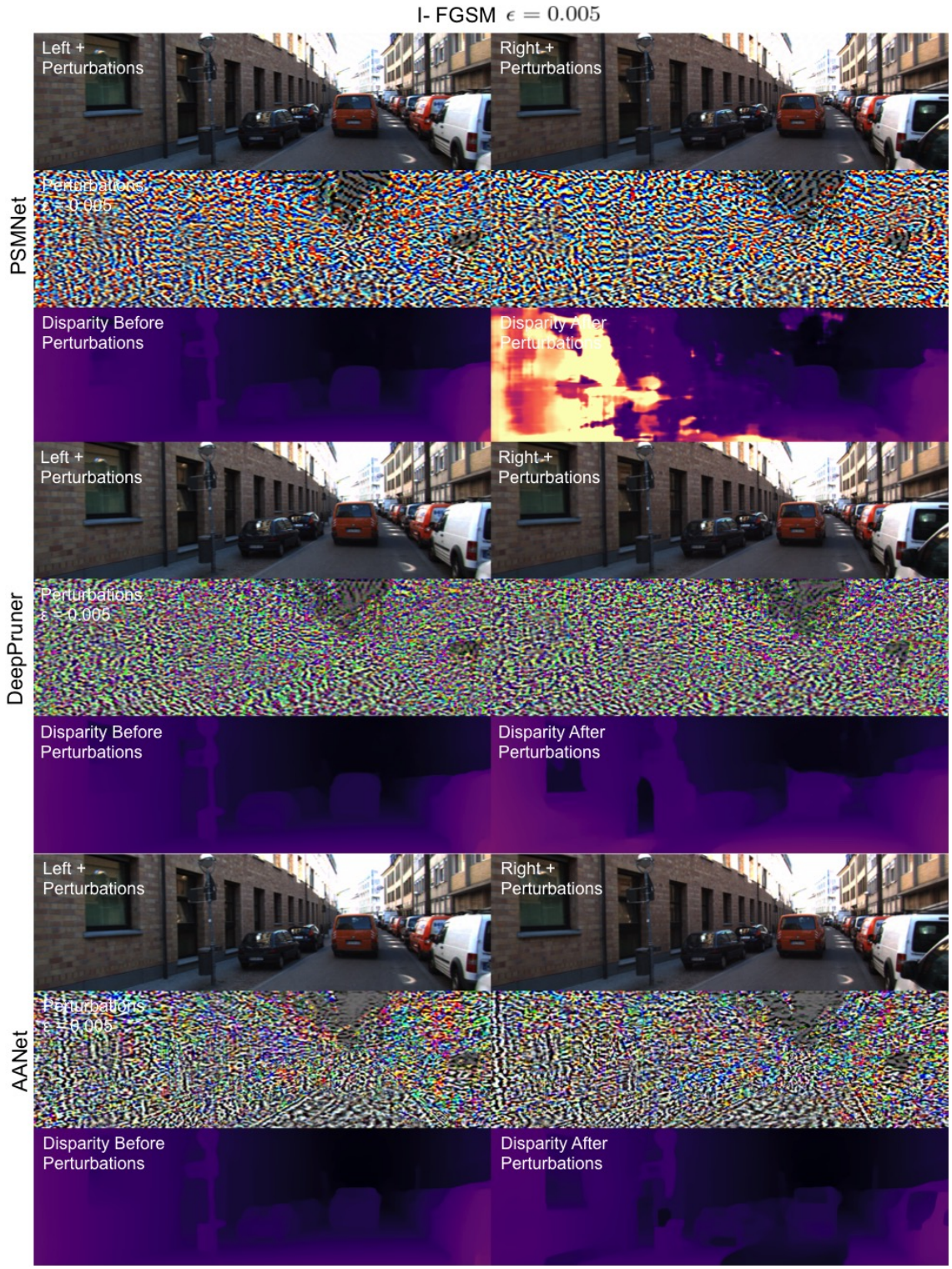


Figure A.3: Examples of I-FGSM attacks using $\epsilon = 0.005$.

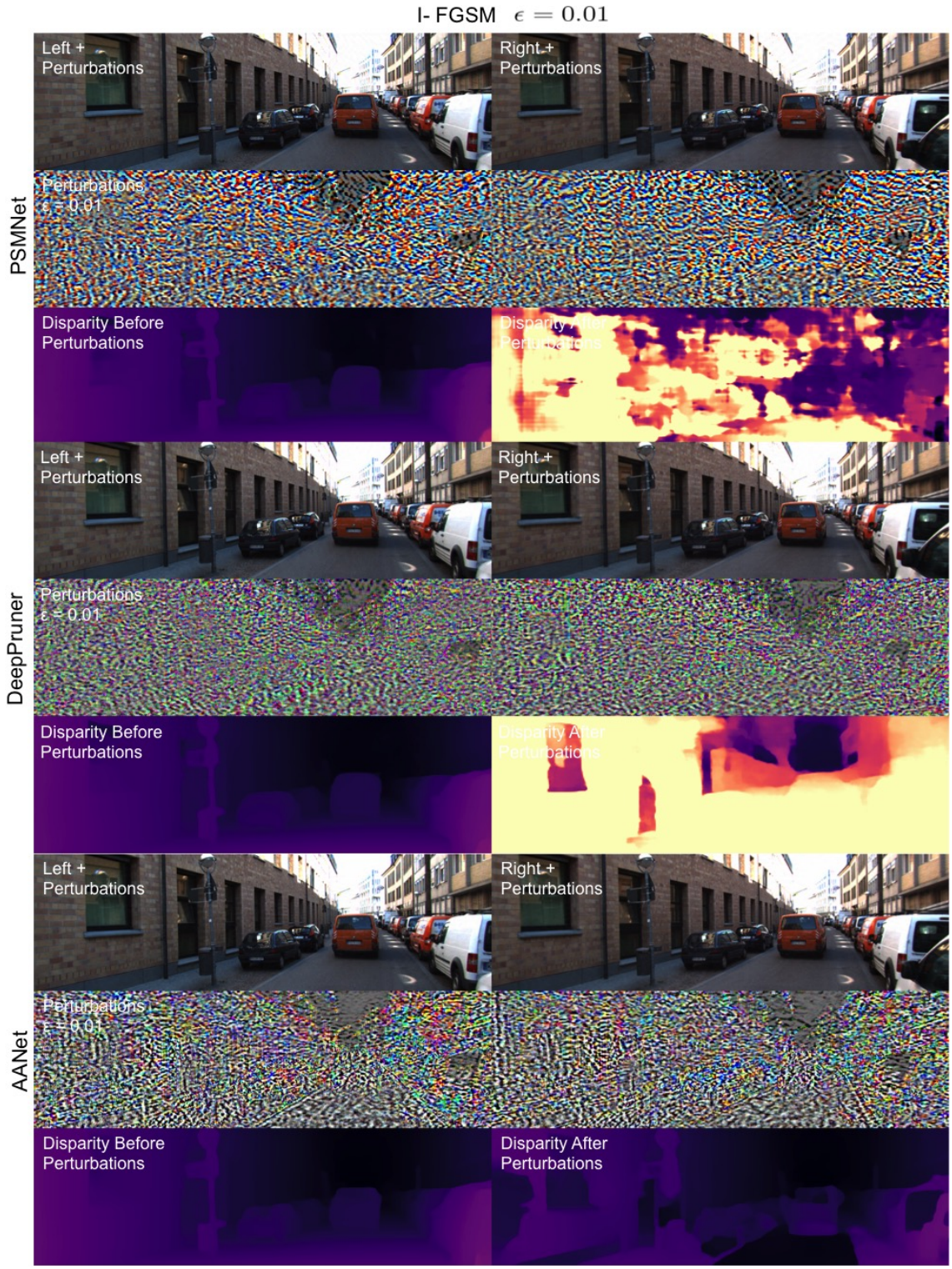


Figure A.4: Examples of I-FGSM attacks using $\epsilon = 0.01$.

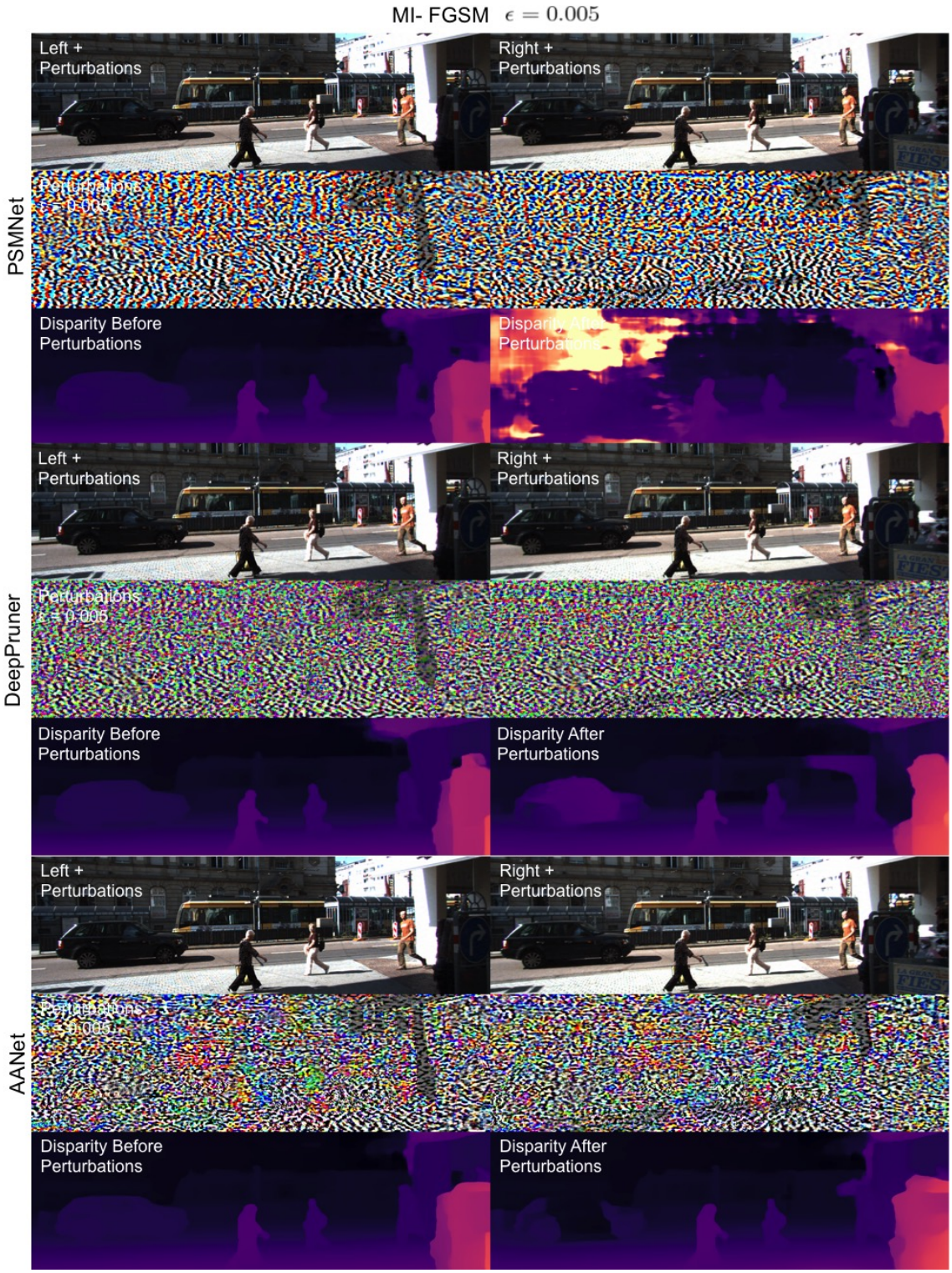


Figure A.5: Examples of MI-FGSM attacks using $\epsilon = 0.005$.

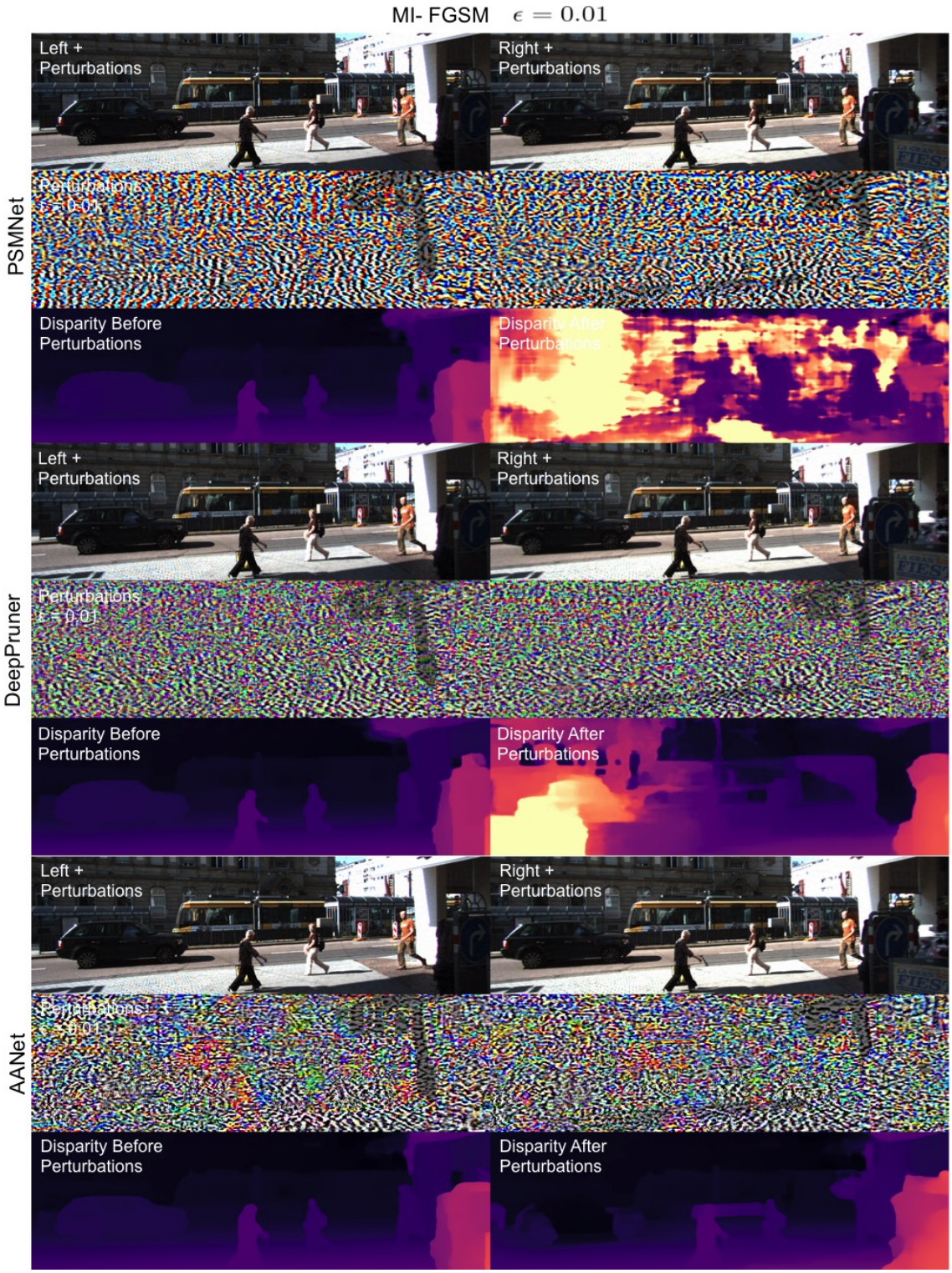


Figure A.6: Examples of MI-FGSM attacks using $\epsilon = 0.01$.

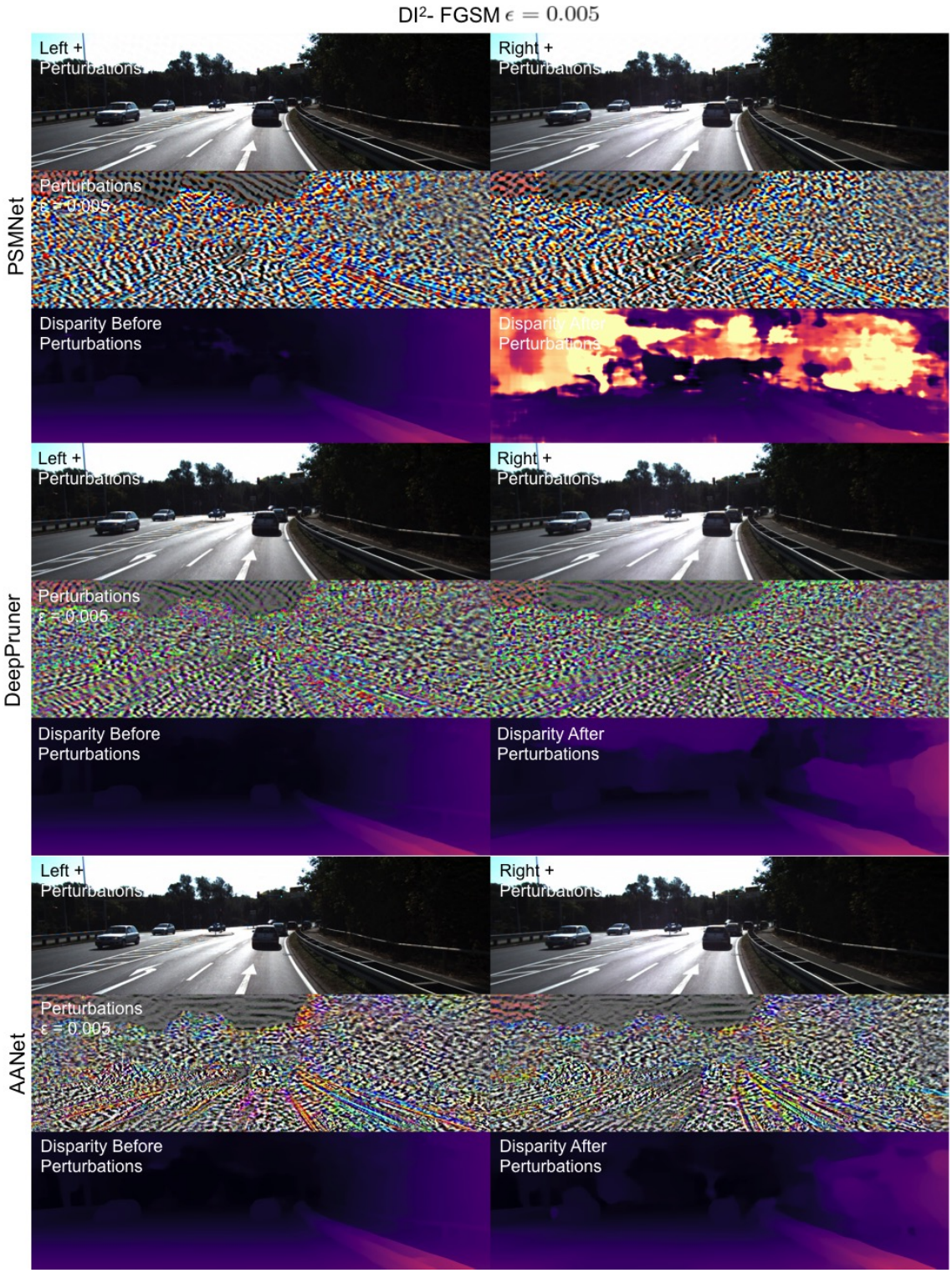


Figure A.7: Examples of DI^2-FGSM attacks using $\epsilon = 0.005$.

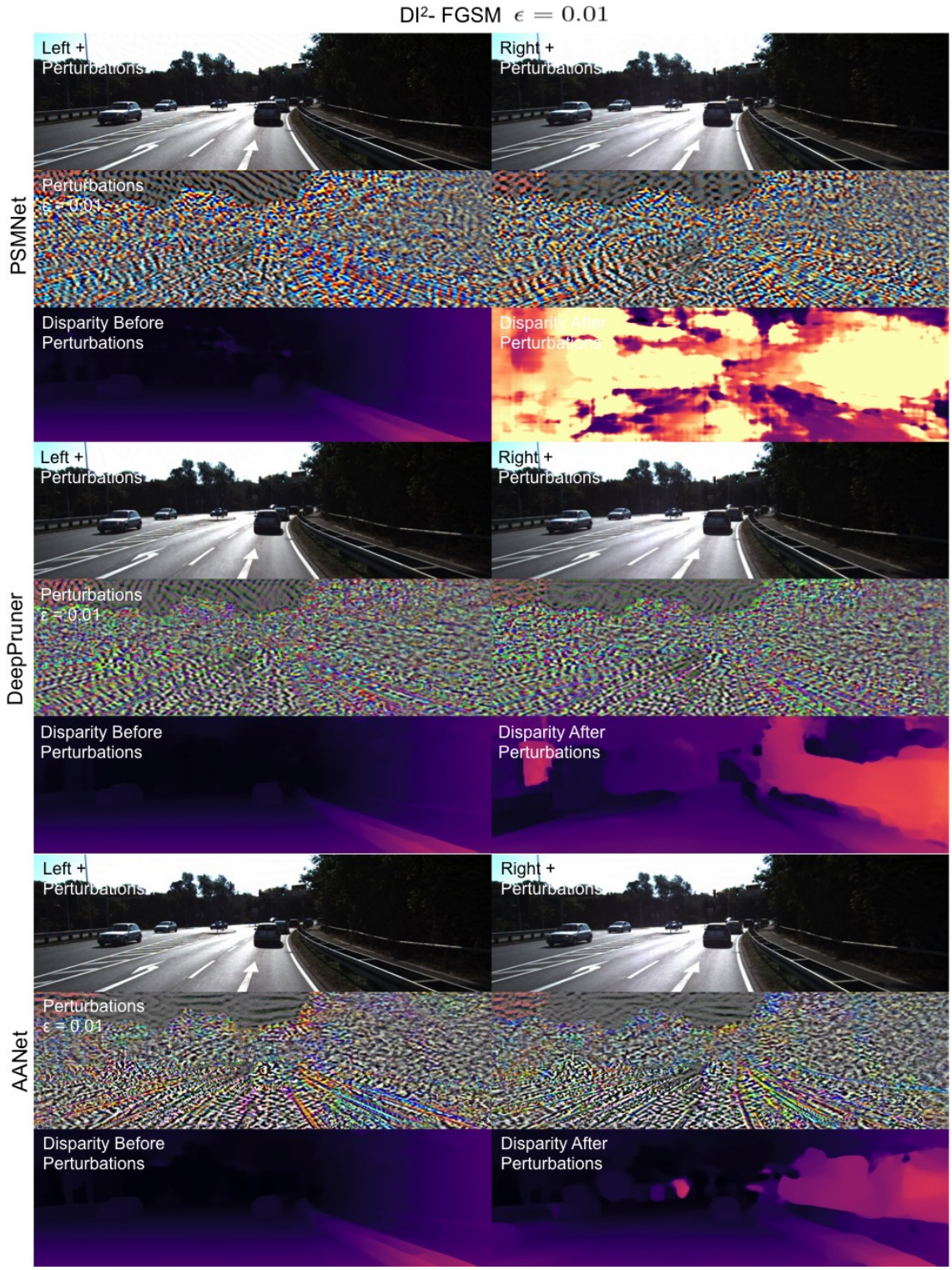


Figure A.8: Examples of DI^2-FGSM attacks using $\epsilon = 0.01$.

Figure A.9: Examples of MDI^2-FGSM attacks using $\epsilon = 0.005$.

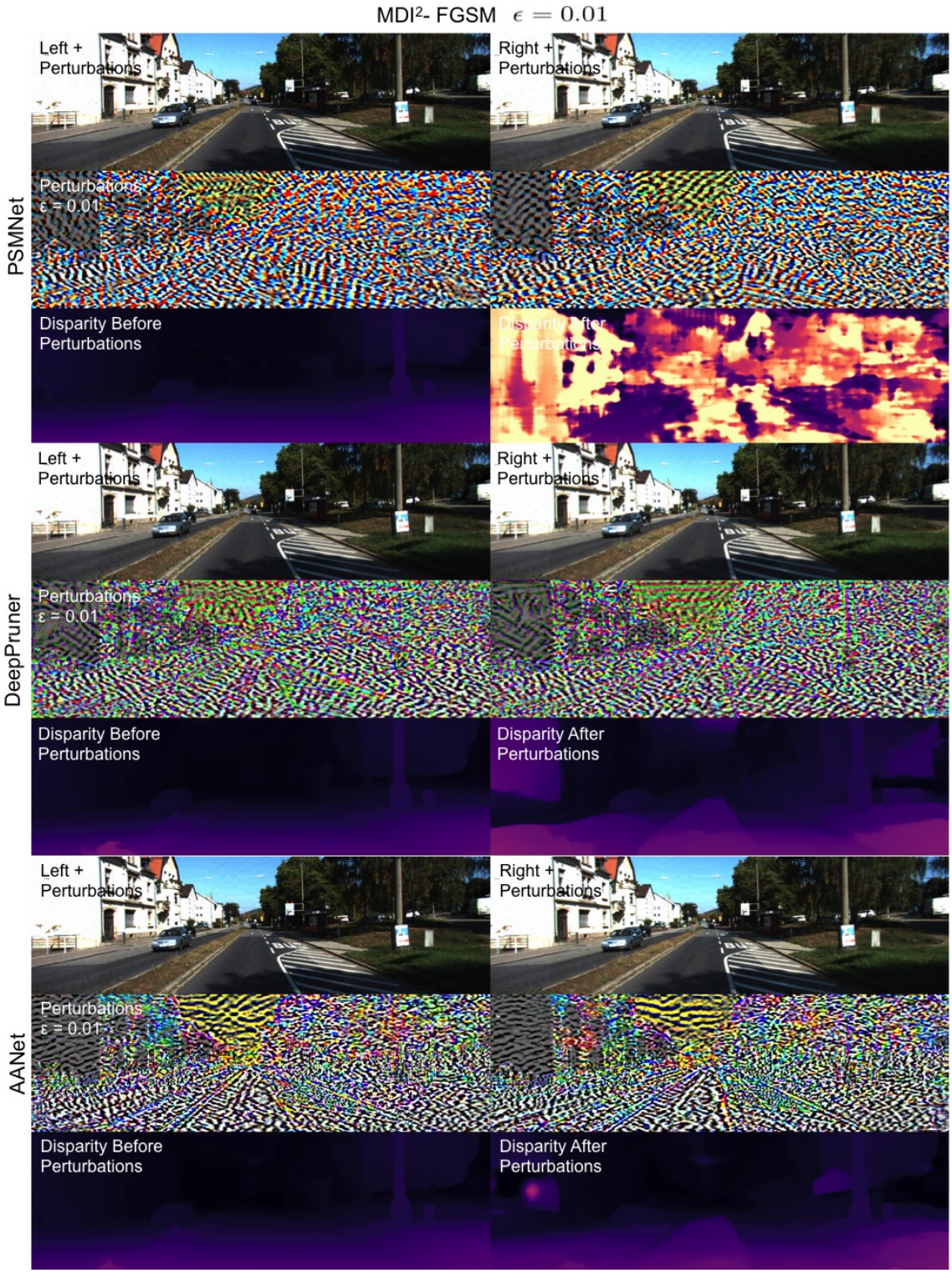


Figure A.10: Examples of MDI^2-FGSM attacks using $\epsilon = 0.01$.

REFERENCES

[1] Jia-Ren Chang and Yong-Sheng Chen. "Pyramid stereo matching network". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 5410–5418.

[2] Antonio R Damasio, Hanna Damasio, and Gary W Van Hoesen. "Prosopagnosia: anatomic basis and behavioral mechanisms". In: *Neurology* 32.4 (1982), pp. 331–331.

[3] Tom van Dijk and Guido de Croon. "How do neural networks see depth in single images?" In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 2183–2191.

[4] Yinpeng Dong et al. "Boosting adversarial attacks with momentum". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 9185–9193.

[5] Shivam Duggal et al. "Deeppruner: Learning efficient stereo matching via differentiable patchmatch". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 4384–4393.

[6] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for autonomous driving? the kitti vision benchmark suite". In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2012, pp. 3354–3361.

[7] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples". In: *arXiv preprint arXiv:1412.6572* (2014).

[8] Jan Hendrik Metzen et al. "Universal adversarial perturbations against semantic image segmentation". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 2755–2764.

[9] Heiko Hirschmuller. "Stereo Processing by Semiglobal Matching and Mutual Information". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 30.2 (Feb. 2008), pp. 328–341. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2007.1166. URL: https://doi.org/10.1109/TPAMI.2007.1166.

[10] Berthold KP Horn and Brian G Schunck. "Determining optical flow". In: *Techniques and Applications of Image Understanding*. Vol. 281. International Society for Optics and Photonics. 1981, pp. 319–331.

[11] Andrew Ilyas et al. "Adversarial examples are not bugs, they are features". In: *Advances in Neural Information Processing Systems*. 2019, pp. 125–136.

[12] Alex Kendall et al. "End-to-end learning of geometry and context for deep stereo regression". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 66–75.

[13] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. "Adversarial machine learning at scale". In: *arXiv preprint arXiv:1611.01236* (2016).

[14] David G Lowe. "Object recognition from local scale-invariant features". In: *Proceedings of the seventh IEEE international conference on computer vision*. Vol. 2. Ieee. 1999, pp. 1150–1157.

[15] Yi Ma et al. *An invitation to 3-d vision: from images to geometric models*. Vol. 26. Springer Science & Business Media, 2012.

[16] Nikolaus Mayer et al. "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 4040–4048.

[17] Moritz Menze and Andreas Geiger. "Object scene flow for autonomous vehicles". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3061–3070.

[18] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. "Deepfool: a simple and accurate method to fool deep neural networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2574–2582.

[19] Seyed-Mohsen Moosavi-Dezfooli et al. "Universal adversarial perturbations". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1765–1773.

[20] Konda Reddy Mopuri, Aditya Ganeshan, and R Venkatesh Babu. "Generalizable data-free objective for crafting universal adversarial perturbations". In: *IEEE transactions on pattern analysis and machine intelligence* 41.10 (2018), pp. 2452–2465.

[21] Muhammad Muzammal Naseer et al. "Cross-domain transferability of adversarial perturbations". In: *Advances in Neural Information Processing Systems*. 2019, pp. 12905–12915.

[22] Anh Nguyen, Jason Yosinski, and Jeff Clune. "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 427–436.

[23] Jiahao Pang et al. "Cascade residual learning: A two-stage convolutional neural network for stereo matching". In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2017, pp. 887–895.

[24] Jonathan Peck et al. "Lower bounds on the robustness to adversarial perturbations". In: *Advances in Neural Information Processing Systems*. 2017, pp. 804–813.

[25] Anurag Ranjan et al. "Attacking optical flow". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 2404–2413.

[26] Christian Szegedy et al. "Intriguing properties of neural networks". In: *arXiv preprint arXiv:1312.6199* (2013).

[27] Florian Tramèr et al. "Ensemble adversarial training: Attacks and defenses". In: *arXiv preprint arXiv:1705.07204* (2017).

[28] Alex Wong, Safa Cicek, and Stefano Soatto. "Targeted Adversarial Perturbations for Monocular Depth Prediction". In: *arXiv preprint arXiv:2006.08602* (2020).

[29] Cihang Xie et al. "Adversarial examples for semantic segmentation and object detection". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 1369–1378.

[30] Cihang Xie et al. "Improving transferability of adversarial examples with input diversity". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 2730–2739.

[31] Cihang Xie et al. “Mitigating adversarial effects through randomization”. In: *arXiv preprint arXiv:1711.01991* (2017).

[32] Haofei Xu and Juyong Zhang. “AANet: Adaptive Aggregation Network for Efficient Stereo Matching”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 1959–1968.

[33] Sergey Zagoruyko and Nikos Komodakis. “Learning to compare image patches via convolutional neural networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 4353–4361.

[34] Jure Žbontar and Yann LeCun. “Stereo matching by training a convolutional neural network to compare image patches”. In: *The journal of machine learning research* 17.1 (2016), pp. 2287–2318.