Stereoscopic Universal Perturbations across Different Architectures and Datasets
SUPPLEMENTARY MATERIALS

Zachary Berger†
UCLA Vision Lab
zackeberger@g.ucla.edu

Parth Agrawal†
UCLA Vision Lab
parthagrawal24@g.ucla.edu

Tian Yu Liu
UCLA Vision Lab
tianyu139@g.ucla.edu

Stefano Soatto
UCLA Vision Lab
soatto@cs.ucla.edu

Alex Wong
UCLA Vision Lab
alexw@cs.ucla.edu

A. Summary of Contents

We begin with a discussion on the existence of adversarial and universal perturbations for calibrated stereo in Sec. B. In Sec. C, we illustrate our training pipeline in Fig. 1. In Sec. D we discuss implementation details, hyperparameters, time and space requirements for training perturbations, fine-tuning with adversarial data augmentation, and retraining variants of AANet, DeepPruner, and PSMNet. We also describe the error metrics used throughout the paper. In Sec. E, we include results for KITTI 2012 (omitted in the main text) on the performance on each stereo network against our stereoscopic universal perturbations (SUPs). We also present additional analyses on the effect of SUPs on scene geometry, on robustness of semantic classes, and on correlation between clean and perturbed features extracted for a given image and registered points between two images. In Sec. F we provide a discussion on the formulation of deformable convolutions and their use in stereo matching network. In Sec. G, we discuss the sensitivity of the proposed SUPs to number of training samples, and in Sec. H, the limitations of the SUPs as an attack and the proposed architectural designs to increase robustness of stereo networks against them. In Sec. I, we discuss potential negative impact of our work and how we can mitigate them. Finally, in Sec. J, we conclude with qualitative comparisons between perturbations crafted for the full image and the proposed 64×64 tiles. We also include additional qualitative results on the KITTI 2012, KITTI 2015 and FlyingThings3D datasets across all norms and transferability experiments.

B. On the Existence of Adversaries for Stereo

Adversarial perturbations exist for a number of tasks from classification [9, 25], object detection [36], even single image depth prediction [29]. While it may seem that they should exist for calibrated stereo, there is a qualitative difference between stereo and other single image based tasks where adversarial perturbations have been observed – such are *purely inductive* tasks: Without training data and a strong inductive prior, a single image does not enable inference of depth or labels of objects. The likelihood is flat and adversarial perturbations have free reign to modify the outcome of inference, even to control the network to yield a desired 3D scene as outcome [29]. Not so for stereo: binocular disparity is sufficient to infer depth wherever the image gradient is non-trivial, without any need for induction from a training set. One is not free to change the outcome of inference without observable changes in the likelihood. So, the fact that adversarial perturbations exist, i.e. Stereopagnosia [33], is indeed surprising for stereo, and that they would survive architectural changes even more so.

C. Training Pipeline

In Fig. 1 we visualize the training pipeline for one iteration of our algorithm for crafting universal perturbations for stereo. Let $f_\theta(x_L, x_R) \in \mathbb{R}^{H \times W}$ be a pretrained stereo network that estimates the disparity between the left x_L and right x_R images of a stereo pair. Let $(v_L, v_R) \in \mathbb{R}^{h \times w}$ be a left and right perturbation subject to $h \mid H$ and $w \mid W$. First, we apply (v_L, v_R) to (x_L, x_R) over the entire image space, by evenly repeating the perturbation v_I across x_I with no overlap for $I \in \{L, R\}$. we then compute the gradient of the stereo network's loss $\ell(f_\theta(\cdot), y_{gt})$ with respect to each image x_I in the stereo pair:

$$g_I^{(n)} = \nabla_{x_I^{(n)}} \ell(f_\theta(\hat{x}_L^{(n)}, \hat{x}_R^{(n)}), y^{(n)}). \quad (1)$$

For a given stereo pair (x_L, x_R), we take the mean over the gradient with respect to the image g_I for all tiles:

† denotes authors with equal contributions.
Code: github.com/alexklwong/stereoscopic-universal-perturbations

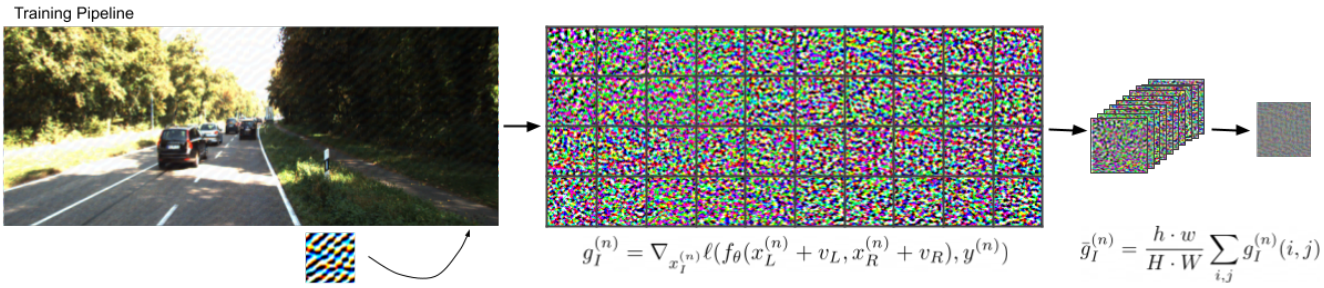


Figure 1. *Training pipeline*. Perturbations are tiled across their respective images. We take mean over the gradient with respect to each image for all tiles, which is used to update the each of the perturbations.

$$\bar{g}_I^{(n)} = \frac{h \cdot w}{H \cdot W} \sum_{i,j} g_I^{(n)}(i,j). \tag{2}$$

This aggregated result can then be used to update the tile perturbation v_I for $I \in \{L, R\}$.

D. Implementation Details

Datasets. We optimize our perturbations on the KITTI dataset [7], which contains ≈47K 376×1240 resolution stereo pairs of real-world outdoor driving scenarios. We evaluate them on the KITTI 2012, KITTI 2015 [17], and Scene Flow [16] stereo datasets using AANet [37], DeepPruner [5], and PSMNet [2]. Due to computational limitations, we resize all images to 256×640 and adjusted disparity maps accordingly. Hence, the baseline error is slightly higher than those reported by each method.

KITTI 2012 contains 194 stereo pairs with sparse ground-truth disparities. KITTI 2015 contains 200 stereo pairs with high quality ground-truth disparity maps. Images in both datasets have dimension 376×1240. Following the KITTI validation protocol, KITTI 2012 is divided into 160 for training and 34 for validation and KITTI 2015 is divided into 160 for training and 40 for validation. We do not use any samples from KITTI 2012 or KITTI 2015 to optimize our perturbations, and only evaluate on the validation sets.

We demonstrate that our perturbations can generalize across datasets by testing them on Scene Flow [16] – a synthetic dataset comprised of 35K training and 4370 testing 540×960 resolution images paired with ground-truth disparity maps. Like KITTI 2012 and 2015, we do not optimize our perturbations on Scene Flow and simply leverage their testing set, FlyingThings3D, in our evaluation.

We used PyTorch to implement our approach. We employed the publicly available code and pretrained models of AANet, DeepPruner, and PSMNet. We note that while AANet and PSMNet released separate pretrained models for KITTI 2012 and KITTI 2015, DeepPruner released one model trained on both datasets. All three stereo networks released pretrained models for Scene Flow. We note that the pretrained model on Scene Flow of PSMNet, provided by the authors, did not reproduce the results in their paper. We obtained D1-error $> 30\%$ and EPE > 4px when running on the Scene Flow test set. This is a known issue in their code repository. Hence, we fine-tuned the pretrained PSMNet model on Scene Flow, lowering the baseline D1-error to 5.57% and the EPE to 1.27px, and used this model for our experiments. We note that DeepPruner provided two model variants, DeepPruner-Best and DeepPruner-Fast. Additionally, AANet provided AANet and AANet+. We used AANet and DeepPruner-Best for all of our experiments. To analyze the robustness of semantic classes, we used the implementation and pretrained model of SDCNet [44], a segmentation network for driving scenes.

Hyper-parameters. We considered the upper norms of $\epsilon \in \{0.002, 0.005, 0.01, 0.02\}$. We searched over learning rates of $\alpha \in \{0.00005, 0.0001, 0.0002, 0.0004, 0.0008\}$. We optimized SUPs on each network using square tiles of 16, 32, 64, and 128, and the full image size 256×640. As searching the full space of tile sizes would be intractable, we chose these as representative tiles at different scales. We omitted results for the 128×128 perturbation from the main paper due to space constraints, but describe them in Sec. E.

Training perturbations. We used an Nvidia GTX 1080Ti on a standard workstation for all of our experiments. To optimize SUPs, we iterated through the KITTI training set one time. Doing so took $\approx$ 5.5hr to craft SUPs for AANet, $\approx$ 7.0hr for DeepPruner, and $\approx$ 12.0hr for PSMNet. Our procedure took $\approx$ 4.3GB of GPU memory for AANet, $\approx$ 4.8GB for DeepPruner, and $\approx$ 8.8GB for PSMNet. As the SUPs are additive, they can be applied to an image in real time. We note that Alg. 1 uses zero initialization for the perturbations. When using the estimated disparity from the clean images as pseudo ground truth, this would yield no training signal. Thus, we added zero mean Gaussian noise to the pseudo ground truth.

Fine-tuning stereo models with adversarial data augmentation. To fine-tune the stereo models with adversarial data augmentation, we used 4 Nvidia GTX 1080Ti.

The models were fine-tuned for 1000 epochs on KITTI 2015 with a batch size of 8. SUPs with upper norm $\epsilon \in \{0.002, 0.005, 0.01, 0.02\}$ were selected at random and added to the training images with 50% probability. The learning rate was initially set to 1×10^{-5}, but switched to 5×10^{-6} and 1×10^{-6} after the 250th and 500th epoch, respectively. It took $\approx$ 9hr, $\approx$ 11hr, and $\approx$ 10hr to fine-tune AANet, DeepPruner, and PSMNet, respectively. We note that on a standard workstation, this process can take up to a week to complete.

Training stereo models with deformable convolutions. We trained a version of (i) PSMNet and (ii) DeepPruner from scratch, each with 25 deformable convolutions in the encoder. We also trained a version of (iii) PSMNet with 6 deformable convolutions, and a version of (iv) AANet without deformable convolutions. Four Nvidia GTX 1080Ti GPUs were used to train each variant, which took $\approx$ 4 days per model.

Both PSMNet models were trained with a batch size of 12, while DeepPruner was trained with a batch size of 16. PSMNet was trained on Scene Flow for 20 epochs, then fine-tuned on KITTI 2015 for 500 epochs. DeepPruner was first trained on Scene Flow for 64 epochs, then fine-tuned on a mixture of KITTI 2012 and KITTI 2015 for 1040 epochs. AANet was first trained on Scene Flow with a batch size of 22 for 64 epochs, fine-tuned on a mixture of KITTI 2012 and KITTI 2015 with a batch size of 12 for 1000 epochs, then fine-tuned on KITTI 2015 with a batch size of 8 for 1000 epochs. Note that DeepPruner is equivalent to PSMNet with an explicit matching module, so we sometimes refer to "DeepPruner" as "PSMNet with explicit matching" in our experiments.

Evaluation metrics. To evaluate the robustness of each stereo network, we use the official KITTI D1-error (the average number of erroneous pixels in terms of disparity) for KITTI 2012 and KITTI 2015 experiments:

$$\delta(i,j) = |f_\theta(\cdot)(i,j) - y_{gt}(i,j)|, \tag{3}$$

$$d(i,j) = \begin{cases} 1 & \text{if } \delta(i,j) > 3, \frac{\delta(i,j)}{y_{gt}(i,j)} > 5\%, \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

$$\text{D1-error} = \frac{1}{\|\Omega_{gt}\|} \sum_{i,j \in \Omega_{gt}} d(i,j), \tag{5}$$

and the official Scene Flow end-point-error (EPE) metrics on FlyingThings3D for generalization experiments:

$$\text{EPE} = \frac{1}{\|\Omega_{gt}\|} \sum_{i,j \in \Omega_{gt}} \delta(i,j), \tag{6}$$

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

E. Experiments and Results

In the main paper, we justified our use of a 64×64 sized perturbation for our experiments. Here we expand on our search of tile sizes. We omitted results on KITTI 2012 in the main paper due to space constraints, but present them in this section. Similarly, in the main paper, we only showed results on AANet for our experiments on the effect of SUPs on scene geometry, robustness of semantic classes, and effect on the feature extractor. Here, we also present results for SUPs trained for DeepPruner and PSMNet.

Comparing tile sizes. We optimized SUPs on KITTI for AANet at each tile size and attacked AANet (Fig. 2-(a)), DeepPruner (Fig. 2-(b)), and PSMNet (Fig. 2-(c)). The 16×16 and 32×32 tiles consistently performed worst, which justifies our choice not to explore smaller tiles. The 128×128 tile performed negligibly better than the 64×64 tile. However, Fig. 2-(d) shows that the 64×64 tile generalizes better than the 128×128 perturbation across all networks on FlyingThings3D. For SUPs with $\epsilon = 0.02$, 64×64 achieves 46.14% error on AANet, 34.87% on DeepPruner and 31.93% on PSMNet, while 128×128 achieves 40.84%, 27.89%, and 29.33% respectively. As demonstrated in the main paper, the full-size perturbations do worse than both, at 36.09% error on AANet, 23.28% on DeepPruner, and 25.35% on PSMNet. We note that, in choosing the tile size, there is a clear trade-off between the performance on the model and dataset for which SUPs are optimized and the generalization to novel architectures and data distributions. For the purpose of universal perturbations that transfer across architectures and datasets, we choose the 64×64 tile size. However, given our results, we leave it up to the user to decide which tile size best suits their use case.

Generalization across architectures and data. We optimized three sets of 64×64 SUPs on KITTI for AANet, DeepPruner, and PSMNet, respectively. In Fig. 3, we attack each network with each set of SUPs on three datasets: KITTI 2012 and 2015 (real datasets of outdoor driving scenarios), and Flyingthings3D (synthetic dataset of random "flying" objects). We report D1-error for KITTI 2012, and 2015, and EPE for FlyingThings3D. In the main paper, we omitted results for KITTI 2012 due to space constraints, but present them here.

In Fig. 3-(a), we add the perturbations optimized for each network to the input stereo pairs from KITTI 2015 for all three networks. Here, KITTI 2015 is a held out data split from KITTI so the distribution of scenes follow that of the training set. When a set of perturbations are applied to the network for which they are optimized, as expected, they tend to be the most successful at corrupting the outputs of the network. We note that AANet (red lines) is consistently more robust against attacks and PSMNet (blue lines) is consistently more susceptible.

In Fig. 3-(b), we add the perturbations optimized for each

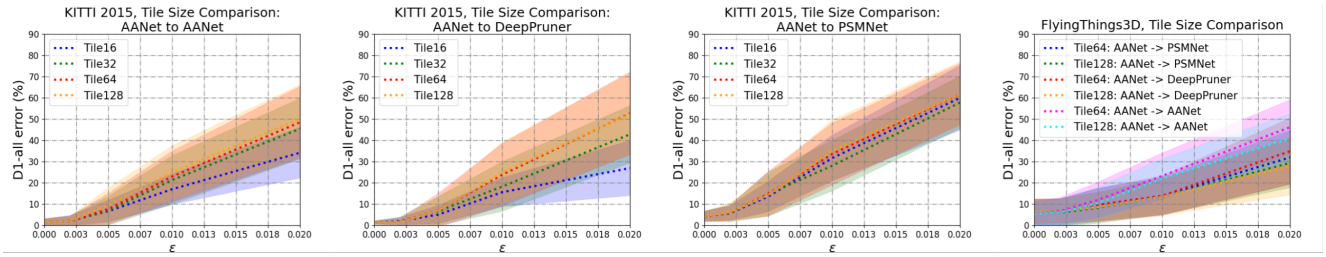


Figure 2. *Comparing tile sizes*. We optimized square tile perturbations with size 16, 32, 64, 128 on KITTI for AANet. (a, b, c) the smaller 16×16 and 32×32 tiles perform worst. (a) When applied to inputs from the dataset they are optimized on, the largest tile size 128×128 marginally outperforms 64×64. However, (d) shows that 64×64 consistently generalizes the best from KITTI to FlyingThings3D for all three architectures. There exists a trade-off between performance on the dataset on which a set of perturbations are optimized and their ability to transfer across models and datasets, where a smaller tile size e.g. 64×64 can generalize better, but a larger tile size e.g. 128×128 may deal more damage to the network and dataset for which the perturbations are optimized.

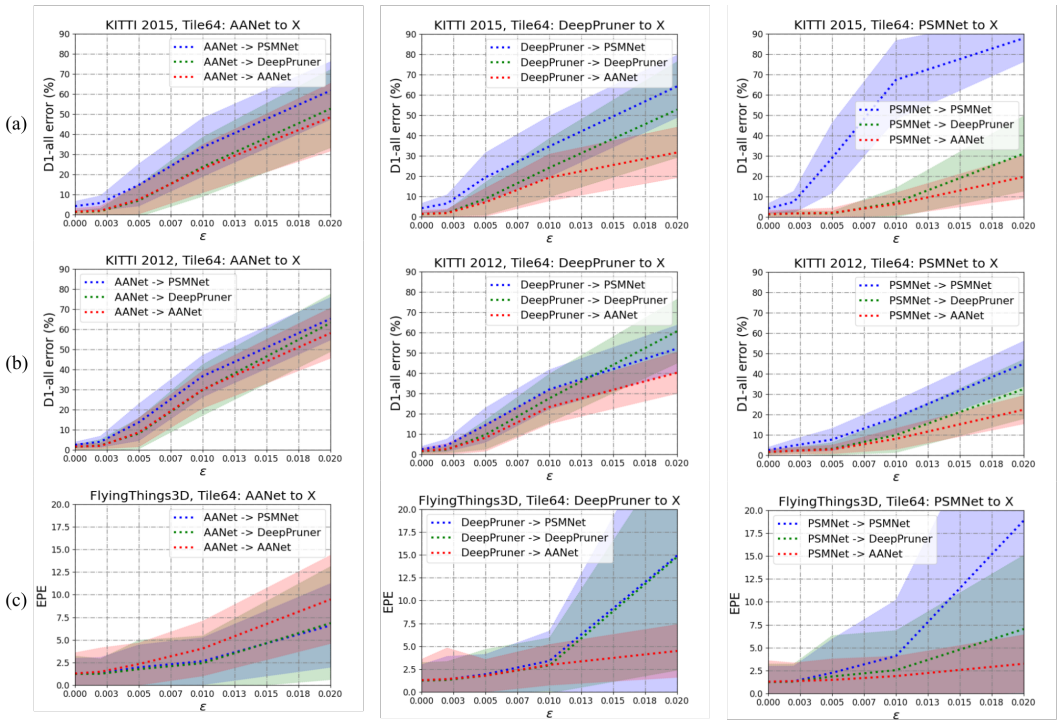
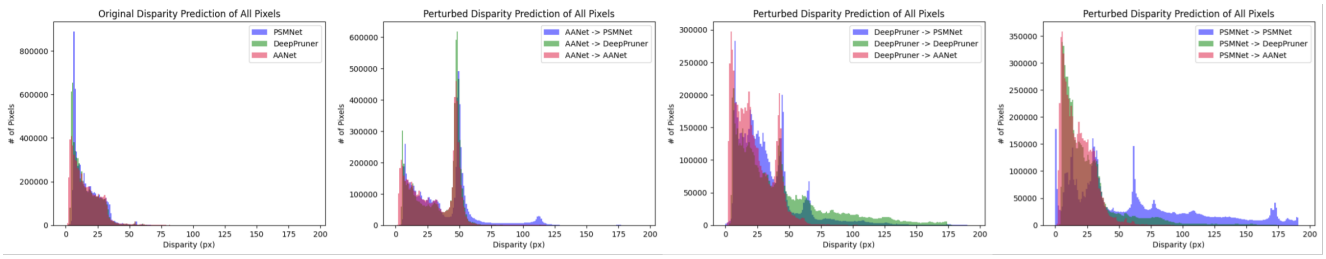


Figure 3. *Generalization across architectures and datasets*. We optimized stereoscopic universal perturbations on AANet, DeepPruner and PSMNet for KITTI (real dataset of outdoor driving scenario) and tested them on three datasets. We measure D1-error for KITTI 2012 and 2015, and EPE for FlyingThings3D. For all three networks, we add the perturbations optimized for each network to input stereo pairs from (a) KITTI 2015, (b) KITTI 2012, and (c) FlyingThings3D. The proposed universal perturbations optimized for a single network on KITTI can fool different network architectures across multiple datasets.

network to input stereo pairs from KITTI 2012 for all three networks. Like KITTI 2015, KITTI 2012 is also a held out data split from KITTI, but contains different objects populating the scene. We observe similar trends as KITTI 2015, where AANet is still the most robust and PSMNet the least robust. We note that for all KITTI 2012 and 2015 experiments, all stereo models are trained on KITTI.

In Fig. 3-(c), we test the generalization of our perturbations across different data distributions. To this end, we add the perturbations optimized for each network to input stereo pairs from FlyingThings3D for all three networks. Here, FlyingThings3D is a synthetic dataset comprised of random "flying" objects where the scene distribution is differs greatly from that of KITTI. For this experiment, all of the stereo models are trained on Scene Flow datasets, which consists of Monkaa, Driving, and FlyingThings3D. Despite being optimized for a single network on KITTI, each pair of perturbations are able fool different network architectures



(a) (b) (c) (d)

Figure 4. *Distribution of disparities before and after adding perturbations*. (a) Distribution of estimated disparities for AANet, DeepPruner and PSMNet on clean (no added perturbations) stereo pairs. Most disparities are concentrated at 2 pixels. Adding perturbations optimized for (b) AANet, (c) DeepPruner, and (d) PSMNet on KITTI to all three models. Disparities shift from 2 pixels to ≈50 pixels in (b), and to ≈40 and ≈60 pixels in (c) and (d). We note that in all cases, the disparities grow larger, meaning that the estimated depth grows smaller, so the perceived distances to objects populating the scene are closer than they should be.

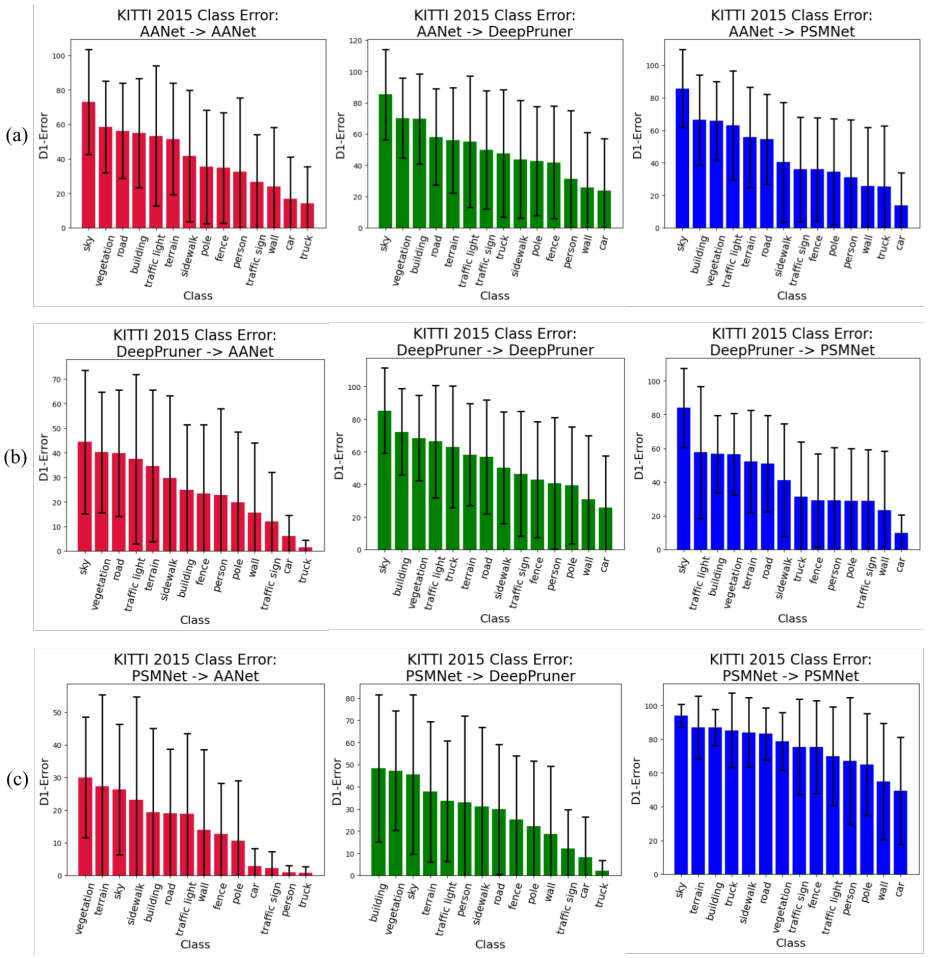


Figure 5. *Quantitative evaluation on KITTI 2015, D1-error for each semantic class*. We use SDCNet, an off-the-shelf semantic segmentation network to partition the image domain of stereo pairs from KITTI 2015 into semantic classes. We show the effect of perturbations optimized on KITTI on different classes for (a) AANet, (b) DeepPruner, and (c) PSMNet. Each semantic class exhibits a different level of robustness against adversaries. However, there are some common trends. For all perturbations and across all networks, the classes that are most susceptible are "building", "vegetation", "sky" and "road". The least susceptible are "car", "person", "pole" and "traffic sign".

trained on different datasets comprised of different scene distributions in a different domain. To the best of our knowledge, we are the first to demonstrate stereoscopic universal perturbations that generalize across architectures and data.

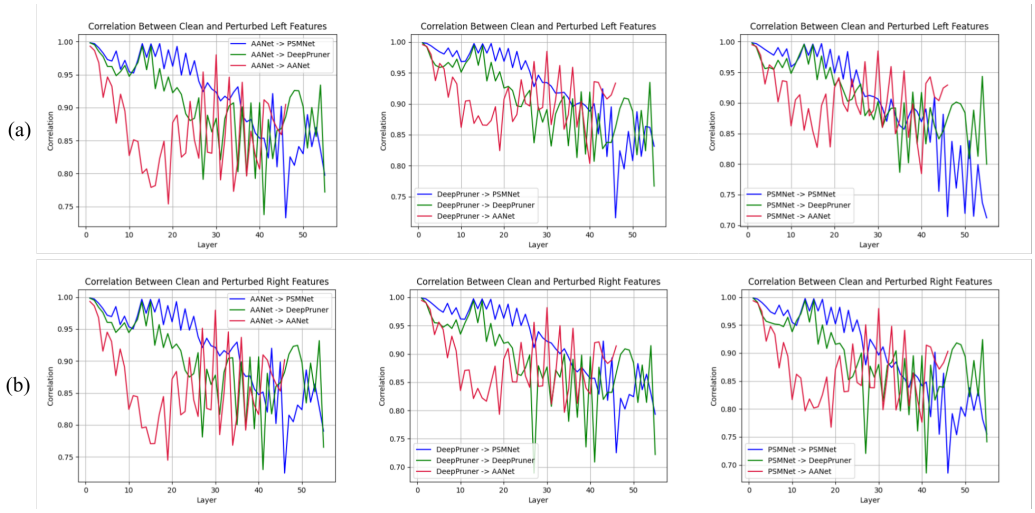


Figure 6. *Effect on clean and perturbed left and right feature maps*. Perturbations optimized for AANet, DeepPruner and PSMNet were added to KITTI 2015 stereo pairs. Correlation was computed between the (a) clean and perturbed left stereo images, and (b) clean and perturbed right stereo images. In both cases, the correlation decreased between the clean and perturbed feature maps. The perturbation signal is amplified by the encoding function as it is fed through the layers in a forward pass.

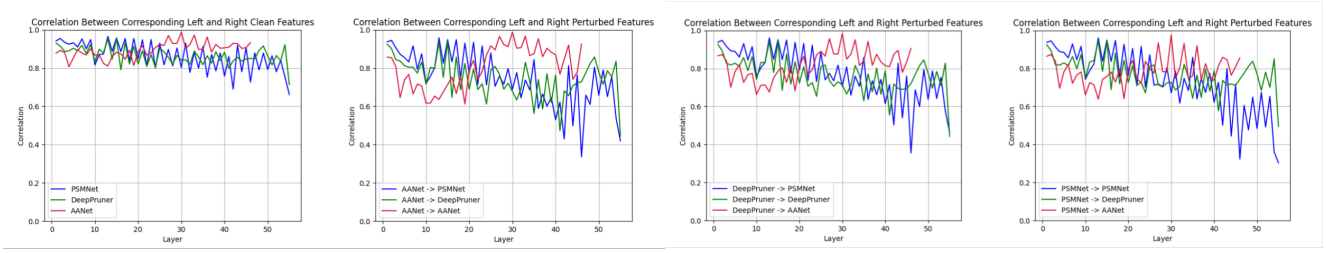


Figure 7. *Effect on corresponding left and right feature maps*. Perturbations optimized for AANet, DeepPruner and PSMNet were added to each stereo pair in KITTI 2015. (a) Correlation was computed between corresponding left and right clean features. Correlation was then computed between corresponding left and right perturbed features, using a perturbation optimized for AANet (b), DeepPruner (c), and PSMNet (d). The perturbed features become uncorrelated relative to the clean features. This suggests that universal perturbations cause similar regions in the RGB domain to be dissimilar in the embedding space, resulting in incorrect correspondences.

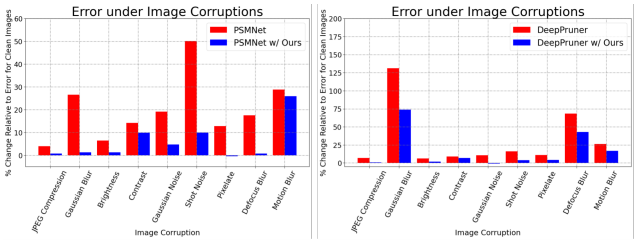
Effect on scene geometry. In the main text, we showed that the mode of the distribution of disparities shifts from ≈2 pixels to ≈50 pixels when each network is attacked by SUPs optimized for AANet. To more effectively quantify how SUPs affect the predicted scene geometry, we plotted the distributions of disparity estimates for all networks tested, where clean (no added perturbation) baseline disparities are shown in Fig. 4-(a) and disparities for perturbed stereo pairs are shown in Fig. 4-(b,c,d). Specifically, we consider the change in disparity distribution for SUPs optimized and tested on the same network and SUPs optimized on other networks. In Fig. 4-(b), we apply perturbations optimized for AANet, in Fig. 4-(c) DeepPruner, and in Fig. 4-(d) PSMNet. We note that all of these experiments were performed on the KITTI 2015 validation set.

As mentioned in the main text, there is a systemic increase in the estimated disparities when perturbations are added to the input stereo pairs; in other words, a decrease in depth, where objects are perceived as closer than they really are. While the general trend is present for all networks, the effect is varied as SUPs optimized for AANet causes a sharp mode at ≈50 pixels whereas SUPs optimized for PSMNet and DeepPruner also create modes at ≈40 pixels and ≈60 pixels. While we do not conduct additional experiments to determine the reason for this bias, we hypothesize that it is induced by the dataset that the perturbations are trained on and will leave this analysis to future works. Interestingly, the same effect transfers to other datasets as well e.g. KITTI 2012 and FlyingThings3D, where visualizations of the output disparities are consistently closer than those estimated from clean images. Indeed, this shows that there are common vulnerabilities across images not only within a dataset, but across datasets and domains. We illustrate this phenomenon in Fig. 1, Fig. 3 and Fig. 8 in the main text, where larger disparities or smaller depths are depicted as brighter (yellow) regions in the colormap. We also ob-

serve a similar phenomenon in the additional qualitative results provided below (see Fig. 12, Fig. 13, Fig. 14, Fig. 18, Fig. 19, Fig. 20).

Robustness of semantic classes. In the main text, we observed that different semantic classes exhibit different levels of robustness against adversaries. We measured the per class error of the disparities estimated by AANet when attacked by a perturbation optimized for AANet. For completeness, we show the per class errors for each network when attacked by adversaries optimized for AANet (Fig. 5-(a)) DeepPruner (Fig. 5-(b)) and PSMNet (Fig. 5-(c)). To this end, we use SDCNet [44], an off-the-shelf semantic segmentation network to partition the image domain of stereo pairs from KITTI 2015 into semantic classes. We observe that each semantic class exhibits a different level of robustness against adversaries. However, there are some common trends across all the networks. Fig. 5 shows that for all perturbations and across all networks, the classes that are most susceptible are "building", "vegetation", "sky" and "road". The least susceptible are "car", "person", "pole" and "traffic sign". We note that the most robust object classes tend to be those with rich textures. In contrast, the least robust classes are those that tend to be comprised of largely homogeneous regions e.g. "sky", "road". We hypothesize that this is due to the nature of the correspondence problem. Because stereo networks employ feature matching as a measure of data fidelity, regions with sufficiently exciting textures tend to have unique correspondences; whereas, there exists inherent ambiguity in locating correspondences in textureless, repeating patterns, and homogeneous regions. This leads to the network relying on the regularizer or the prior (learned from data) to fill in the gaps. The information of the regularizer is stored in the weights or parameters of a network via the training process. As the perturbation signal corrupts the activations of feature maps on which the weights operate on, it is thus corrupting the regularizer or prior, and so the prediction for textureless and homogeneous regions is worse.

Effect on Feature Maps. In the main text, we discussed the effect of stereoscopic universal perturbations (SUPs) on the encoder, or embedding function, for AANet. Here we show results for all three networks in Fig. 6. We added the perturbations optimized for each network into stereo pairs from KITTI 2015 and forwarded them through each network. Similarly, we also fed clean stereo pairs without any added perturbations into each network. The per layer activations of the encoder, shared between the left and right images, were extracted for clean and perturbed stereo pairs. Correlation was then computed between the clean and perturbed left feature maps, and clean and perturbed right feature maps. Fig. 6-(a) shows the correlation between clean and perturbed left image feature maps for each network, while Fig. 6-(b) shows the same for the right image. In



(a) (b)

Figure 8. *Robustness to Image Perturbations*. (a) Robustness of PSMNet, PSMNet with 25 layers of deformable convolutions, (b) DeepPruner, and DeepPruner with 25 layers of deformable convolutions against different types of image corruptions.

both cases, the correlation decreases between the clean and perturbed feature maps. This demonstrates that even though the input perturbation is constrained to be some ϵ small, the perturbation signal is amplified by the encoding function as it is fed through the layers in a forward pass. In effect, this can cause a network to output significantly different results.

However, stereo networks employ an explicit matching mechanism. Even though the perturbations may cause the clean and perturbed features to be different, as long as the true corresponding (registered) pixels between the left and right images are "close" in the embedding space, the correct disparity between the two images can be found. Fig. 7 shows that indeed the perturbations not only causes clean and perturbed feature maps to be different, but also the corresponding registered left and right features to be different. Specifically, Fig. 7-(a) shows that registered clean features are well-correlated throughout all of the features maps in the encoder. This is expected. However, Fig. 7-(b, c, d) shows that left and right registered perturbed features become increasing uncorrelated relative to the clean features as they pass through the shared encoder. This suggests that our universal perturbations cause similar regions in the RGB domain to become dissimilar in the embedding space. As registered points in the images are no longer similar, this in turn fools the explicit matching modules to find incorrect correspondences, resulting in the wrong disparity estimates. We note that in both Fig. 6 and Fig. 7, the correlation of the features of AANet increase from layers 20 to 30. This phenomenon is present for all adversaries, and coincides with the use of deformable convolution in AANet. We conjecture that deformable convolutions may be related to the robustness of AANet. Hence, we use this observation to motivate the use of deformable convolutions (and explicit matching modules like correlation or PatchMatch) as part of our proposed architectural designs to increase robustness against SUPs (see Sec. 5 in the main paper).

Robustness to Image Perturbations. In the main text, we discussed the robustness of PSMNet and PSMNet with 25 DCs against common image corruptions i.e. lossy

(JPEG) compression, Gaussian, defocus and motion blur, shot noise (Fig 10-(d)). As shown in Fig. 8-(a) and also in Fig 10-(d), our design improves over PSMNet by an average of 70% to common image perturbations. Fig. 8-(b) shows that DeepPruner (red), like PSMNet, is also susceptible to noise and blurring. In the case of Gaussian blur with a kernel size of 5 and a standard deviation of 1, the error increases by as much as 131%. Our design of DeepPruner with 25 DCs (blue) is significantly more robust across all tested common image corruptions. There is an improvement of 43% on gaussian blur and 37% on defocus blur. Moreover, our design improves by 76% on shot noise and 60% on average.

F. On Deformable Convolution

In the main paper, we found that replacing convolutional layers with deformable convolutional layers can increase the robustness of stereo networks. For the sake of completeness, in this section, we motivate the use of deformable convolutions in convolutional neural networks (CNNs) through their formulation. We then describe the use of deformable convolution in stereo matching networks.

Motivation. Due to rigid components, CNNs are limited at modeling geometric variations in object viewpoint, pose, scale, and part deformation. Conventional architectures are comprised of layers of fixed size convolutional filters over a regular grid with limited receptive field. While feature pyramids using spatial downsampling (e.g. max pooling and strided convolution) allow for modeling different scales of objects populating the scene, extensive geometric data augmentations are necessary to capture the variations listed above. Yet, it is not possible to fully capture all variations in the data. Additionally, the effective field of view is limited to a local neighborhood sampled regularly, which is ignorant of object boundaries. Hence, [4, 42] proposed deformable convolutions to learn object deformations by directly conditioning on the data. Deformable convolution adds 2D offsets, predicted based on the input feature map, to the sampling grid of standard convolution. It allows the sampling grid to deform freely and hence capture visibility phenomena such as occlusions, as visualized in Fig. 9.

Formulation. Let $\mathbf{x}$ be an input feature map and $\mathbf{y}$ be an output feature map. The standard 2D convolution is a two-step operation: 1) sample $\mathbf{x}$ over a regular grid $\mathcal{R}$; 2) compute a weighted sum of the sampled values to generate $\mathbf{y}$. Formally, for each location $\mathbf{p}_0$ in $\mathbf{y}$,

$$\mathbf{y}(\mathbf{p}_0) = \sum_{\mathbf{p}_n \in \mathcal{R}} \mathbf{w}(\mathbf{p}_n) \cdot \mathbf{x}(\mathbf{p}_0 + \mathbf{p}_n) \tag{7}$$

where $\mathbf{p}_n$ are the locations in $\mathcal{R}$.

In deformable convolution, the sampling grid $\mathcal{R}$ are augmented with offsets $\{\Delta\mathbf{p}_n | n = 1, ..., |\mathcal{R}|\}$. Eqn. 7 extends

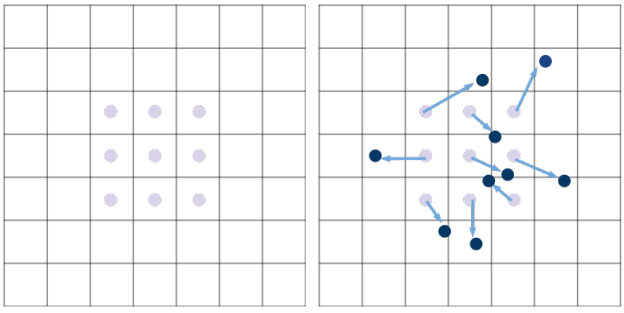


Figure 9. *Deformable convolution*. (a) The sampling grid of a standard 3×3 convolution. (b) The sampling locations of a deformable convolution (dark blue), adaptively offset from the regular sampling grid.

to

$$\mathbf{y}(\mathbf{p}_0) = \sum_{\mathbf{p}_n \in \mathcal{R}} \mathbf{w}(\mathbf{p}_n) \cdot \mathbf{x}(\mathbf{p}_0 + \mathbf{p}_n + \Delta\mathbf{p}_n). \tag{8}$$

Here, we sample at offset locations $\mathbf{p}_n + \Delta\mathbf{p}_n$. Because the offset $\mathbf{p}_n$ can be fractional, $\mathbf{x}(\mathbf{p}_0 + \mathbf{p}_n + \Delta\mathbf{p}_n)$ may not correspond to an actual element and is subject to quantization effects. Hence, $\mathbf{x}(\mathbf{p}_0 + \mathbf{p}_n + \Delta\mathbf{p}_n)$ is computed via bilinear interpolation.

Use in stereo matching. AANet [37] introduce an intra-scale aggregation (ISA) module for cost aggregation, designed to adaptively sample points from regions of similar disparity. The authors intuit that adaptive sampling prevents the edge-fattening issue at disparity discontinuities [23]. Formally, let $\mathbf{C} \in \mathbb{R}^{D \times H \times W}$ be a cost volume with maximum disparity D, height H, and width W. For K^2 sampling points, the ISA computation is

$$\widetilde{\mathbf{C}}(d, \mathbf{p}) = \sum_{k=1}^{K^2} w_k \cdot \mathbf{C}(d, \mathbf{p} + \mathbf{p}_k + \Delta\mathbf{p}_k) \tag{9}$$

where $\widetilde{\mathbf{C}}(d, \mathbf{p})$ is the aggregated cost at pixel $\mathbf{p}$ for disparity d, w_k is the weight for point k, $\mathbf{p}_k$ is a fixed offset from $\mathbf{p}$, and $\Delta\mathbf{p}_k$ is a learned offset. Since the formulations are similar, ISA is implemented with deformable convolution. We note that in addition to the ISA module, AANet also employs deformable convolutions in its encoder to reduce sampling across object boundaries, which minimizes the bleeding effect often seen in backprojected point clouds. While the proposed use case is mainly to handle object deformation and variations in view point, we found that deformable convolutions are amicable towards defending against adversaries as their formulation naturally allows for "avoiding" certain signals e.g. occlusion boundaries, part deformation, and perhaps adversarial perturbations present in the input.

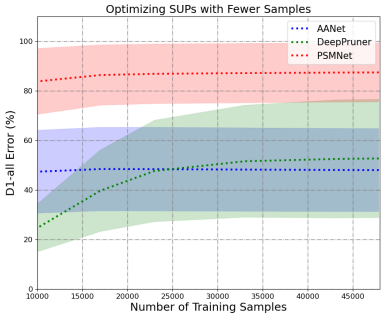


Figure 10. *Sample efficiency*. We optimize SUPs for AANet, DeepPruner and PSMNet with fewer samples and test their effectiveness on KITTI 2015. SUPs for AANet and PSMNet be still be effective even when optimized with 79% fewer samples; the effectiveness of SUPs optimized for DeepPruner start to decrease when optimized with less than 60% of the dataset.

G. On Sample Efficiency

In Fig 5, 8 of main text, SUPs were optimized using KITTI (outdoor driving, real domain) and are added to images from the FlyingThings3D test set (randomly generated scenes, synthetic), part of the Scene Flow [17] datasets. These perturbed images are fed to AANet [37], DeepPruner [5] and PSMNet [2] that were trained on Scene Flow. For this particular set of experiments, our SUPs were trained with substituted data from KITTI and zero-shot transferred to FlyingThings3D.

In this section, we study the sample efficiency of SUPs. We optimized SUPS for AANet, DeepPruner and PSMNet using the KITTI raw dataset for varying training sample sizes. Fig. 10 shows that SUPs for AANet and PSMNet are still effective when optimized with just 21% of KITTI; whereas SUPs for DeepPruner start to decrease in effectiveness when we remove 60% of the samples.

H. Limitations

While we have demonstrated stereoscopic universal perturbations (SUPs) that generalize across network architectures, datasets and domains and showed that deformable convolutions and explicit matching can help mitigate them, there exist limitations on both fronts. Despite reaching as high as 87% in D1-error error on PSMNet, SUPs has a limited effectiveness on AANet and reaches 50% D1-error – this is equivalent to half of the perceived scene being corrupted. In none of our experiments do the error reach 100%, meaning that there are still portions of the scene that are reasonably correct.

We note that such perturbations are not undefendable; we showed that fine-tuning with adversarial data augmentation does mitigate them to some extent, but not fully. For example D1-error decreased from 87.72% to 2.96% for PSMNet when attacked by the adversary it was trained on, but when attacked by new adversaries, it still has a 35.75% error. While our proposed architecture designs involving deformable convolutions and explicit matching does help, they also do not fully mitigate the attack.

Our scope is limited to stereo [2, 5, 20, 37]; however, the component proposed in our architectural design. i.e. deformable convolutions and inductive biases like explicit matching, are used in many geometric problems such as optical flow [1, 12–14, 24, 26, 38], multi-view stereo [3, 10, 27, 40, 41], monocular depth prediction [6, 8, 19, 21, 22, 28, 34], and depth completion [11, 15, 18, 30–32, 35, 39, 43]. These are problems where adversarial and universal perturbations are less studied. So there is a long road ahead, but we hope that our findings will be useful towards realizing robustness deep neural networks.

I. Discussion of Potential Negative Impact

Deep learning models have been extensively deployed for many applications. Hence, adversarial perturbations have been treated as a security concern. These concerns were initially far fetched, as adversarial perturbations were optimized per-image instance; it would be computationally infeasible to corrupt a model in real-time. Yet, the discovery of universal adversarial perturbations made the security threat more realistic.

We showed that stereoscopic universal perturbations (SUPs) that generalize across architecture and data exist. These can be applied effectively to attack models in the black-box setting, and so appear to present a more immediate security threat. Yet, we believe these perturbations will not cause damage outside of the academic setting. While our perturbations can be realized as a filter to be placed on top of a camera lens, autonomous agents typically have a myriad other sensors. An autonomous system should not fail due to a corrupted disparity map as long as it can rely on sensor measurements from other sources.

Rather, we have used stereoscopic universal perturbations to better understand the robustness of stereo networks. By identifying how SUPs can corrupt stereo networks, we were able to motivate several architectural designs (see Sec. 5, main text) that ultimately improve the robustness of stereo models. Adversarial perturbations expose inherent problems with our deep networks – yet, we view them as an opportunity to unravel our black-box models and develop more robust representations.

Nonetheless, to mitigate them, one can redesign models with the proposed architectural changes, leverage adversarial data augmentation to fine-tune existing models, or utilize techniques to denoise, purify, or rectify the input as discussed in our Related Works section (Sec. 2, main text). Additionally, we will also restrict code and perturbations

usage via our release license.

J. Additional Qualitative Results

In Fig. 11 we show SUPs optimized for AANet, DeepPruner, and PSMNet on the KITTI dataset. Each panel of two rows shows SUPs optimized over the full 256×640 image and the 64×64 sized perturbations, tiled across the image domain, for each method. The tiled perturbations are then used for each subsequent visualization. We note that for all full image size SUPs, there are structural artifacts biased by the scenes in the dataset. For instance, full image size SUPs for PSMNet shows a road pattern in the both perturbation images. As a result, these dataset specific structures limit the generalization of SUPs that were optimized over the full image. Unlike them, 64×64 tiles do not contain any of these structures as they are spatially invariant, enabling them to transfer across datasets and domains.

Next, we demonstrate attacks against each model using the SUPs trained for it. We show the perturbed stereo pair, the original disparities estimated from clean stereo pairs and the corrupted disparities estimated from the perturbed stereo pair for KITTI 2015 (Fig. 12), KITTI 2012 (Fig. 13), and FlyingThings3D (Fig. 14). To demonstrate how an attack varies for different upper norms, we attack each network for scenes from KITTI 2015 using perturbations with upper norm $\epsilon \in \{0.002, 0.005, 0.01, 0.02\}$; we look at AANet in Fig. 15, DeepPruner in Fig. 16, and PSMNet in Fig. 17. As expected, as the upper norm ϵ increases, we also observe more corruption in the disparity map. The corrupted regions are generally estimated as "closer" to the camera. We conclude by visualizing transferability of the SUPs to PSMNet across different datasets in Fig. 18, Fig. 19, and Fig. 20.

References

[1] Filippo Aleotti, Matteo Poggi, Fabio Tosi, and Stefano Mattoccia. Learning end-to-end scene flow by distilling single tasks knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10435–10442, 2020. 9

[2] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5410–5418, 2018. 2, 9

[3] Rui Chen, Songfang Han, Jing Xu, and Hao Su. Point-based multi-view stereo network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1538–1547, 2019. 9

[4] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017. 8

[5] Shivam Duggal, Shenlong Wang, Wei-Chiu Ma, Rui Hu, and Raquel Urtasun. Deeppruner: Learning efficient stereo matching via differentiable patchmatch. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4384–4393, 2019. 2, 9

[6] Xiaohan Fei, Alex Wong, and Stefano Soatto. Geo-supervised visual depth prediction. *IEEE Robotics and Automation Letters*, 4(2):1661–1668, 2019. 9

[7] 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*, pages 3354–3361. IEEE, 2012. 2

[8] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019. 9

[9] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 1

[10] Xiaodong Gu, Zhiwen Fan, Siyu Zhu, Zuozhuo Dai, Feitong Tan, and Ping Tan. Cascade cost volume for high-resolution multi-view stereo and stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2495–2504, 2020. 9

[11] Mu Hu, Shuling Wang, Bin Li, Shiyu Ning, Li Fan, and Xiaojin Gong. Penet: Towards precise and efficient image guided depth completion. *arXiv preprint arXiv:2103.00783*, 2021. 9

[12] Dong Lao and Ganesh Sundaramoorthi. Minimum delay moving object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4250–4259, 2017. 9

[13] Dong Lao and Ganesh Sundaramoorthi. Extending layered models to 3d motion. In *Proceedings of the European conference on computer vision (ECCV)*, pages 435–451, 2018. 9

[14] Dong Lao and Ganesh Sundaramoorthi. Minimum delay object detection from video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5097–5106, 2019. 9

[15] Yuankai Lin, Tao Cheng, Qi Zhong, Wending Zhou, and Hua Yang. Dynamic spatial propagation network for depth completion. *arXiv preprint arXiv:2202.09769*, 2022. 9

[16] Nikolaus Mayer, Eddy Ilg, Philip Hausser, 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4040–4048, 2016. 2

[17] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3061–3070, 2015. 2, 9

[18] Jinsun Park, Kyungdon Joo, Zhe Hu, Chi-Kuei Liu, and In-So Kweon. Non-local spatial propagation network for depth completion. In *European Conference on Computer Vision, ECCV 2020*. European Conference on Computer Vision, 2020. 9

[19] Matteo Poggi, Filippo Aleotti, Fabio Tosi, and Stefano Mattoccia. On the uncertainty of self-supervised monocular

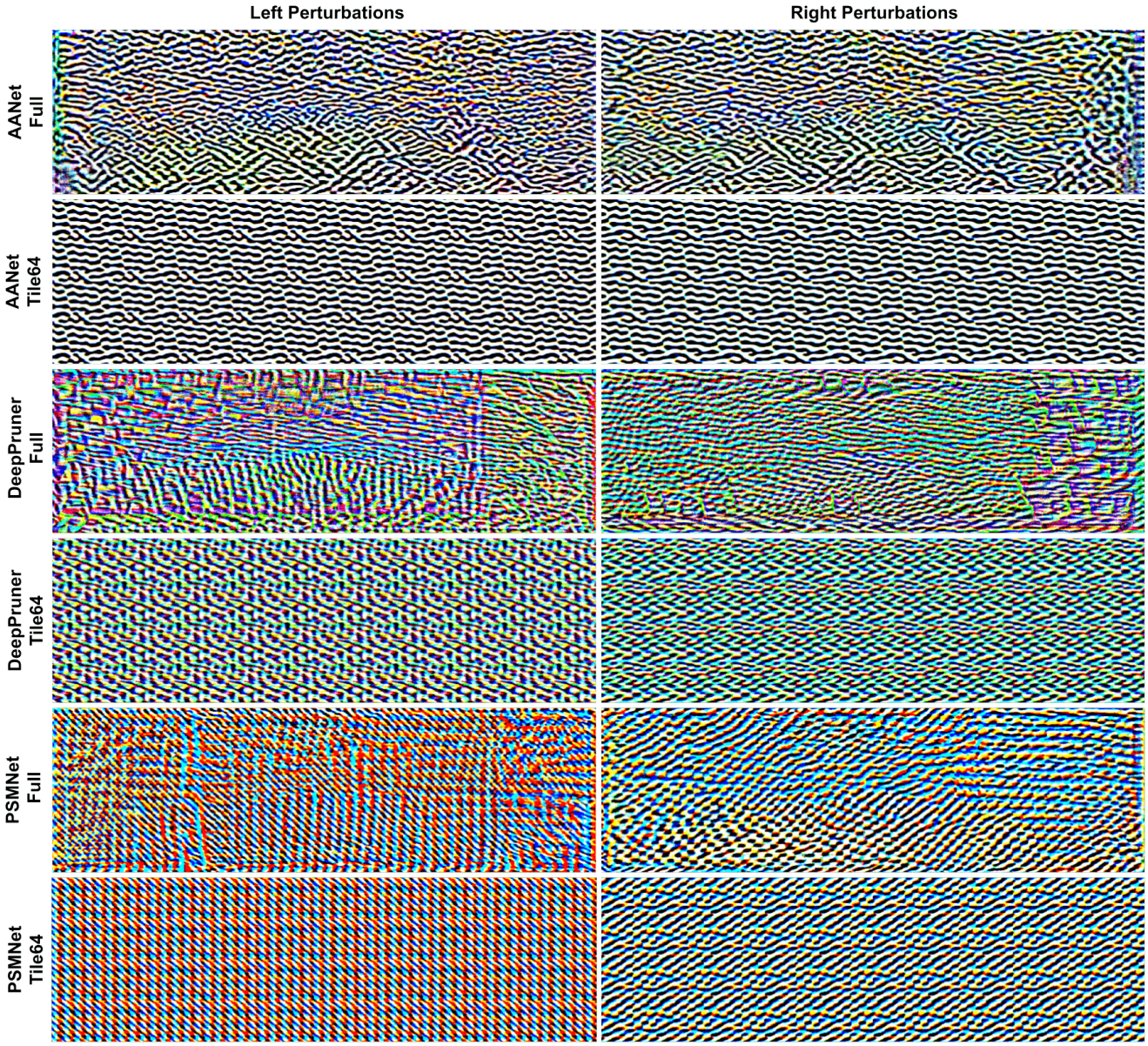


Figure 11. *Examples of Stereoscopic universal perturbations* (SUPs) optimized for AANet, DeepPruner, and PSMNet on the KITTI dataset. Each panel of two rows shows SUPs optimized over the full 256×640 image and the 64×64 sized perturbations, tiled across the image domain, for each method. We note that for all full image size SUPs, there are structural artifacts biased by the scenes in the dataset, which limits their generalization capabilities.

depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3227–3237, 2020. 9

[20] Matteo Poggi, Filippo Aleotti, Fabio Tosi, Giulio Zaccaroni, and Stefano Mattoccia. Self-adapting confidence estimation for stereo. In *European Conference on Computer Vision*, pages 715–733. Springer, 2020. 9

[21] Matteo Poggi, Fabio Tosi, Filippo Aleotti, and Stefano Mattoccia. Real-time self-supervised monocular depth estimation without gpu. *IEEE Transactions on Intelligent Transportation Systems*, 2022. 9

[22] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12179–12188, 2021. 9

[23] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47(1):7–42, 2002. 8

[24] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz.

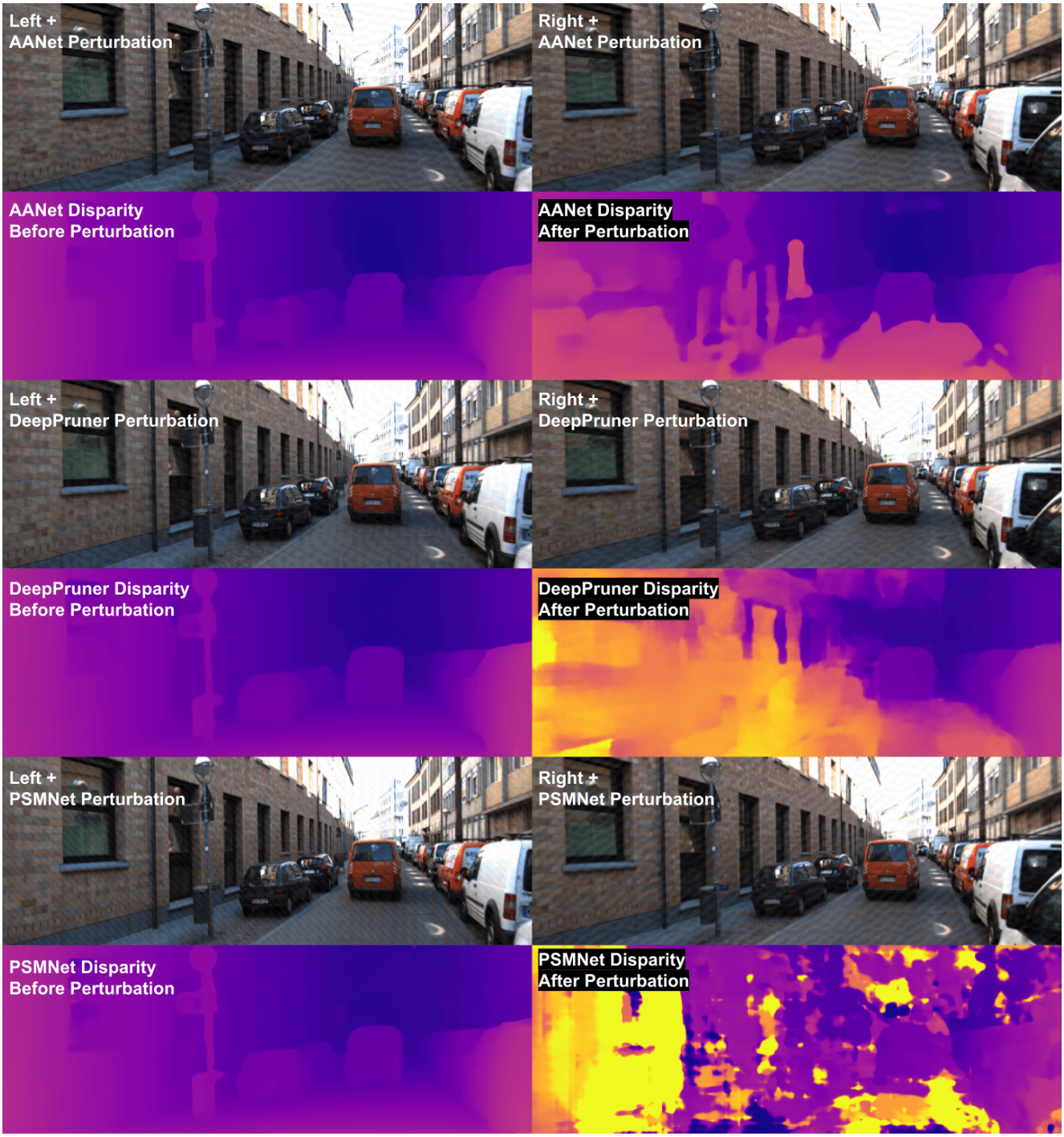


Figure 12. *Attacking AANet, DeepPruner, and PSMNet on a scene from KITTI 2015 using the SUP trained for each model.*

Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8934–8943, 2018. 9

[25] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. 1

[26] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pages 402–419. Springer, 2020. 9

[27] Fangjinhua Wang, Silvano Galliani, Christoph Vogel, Pablo

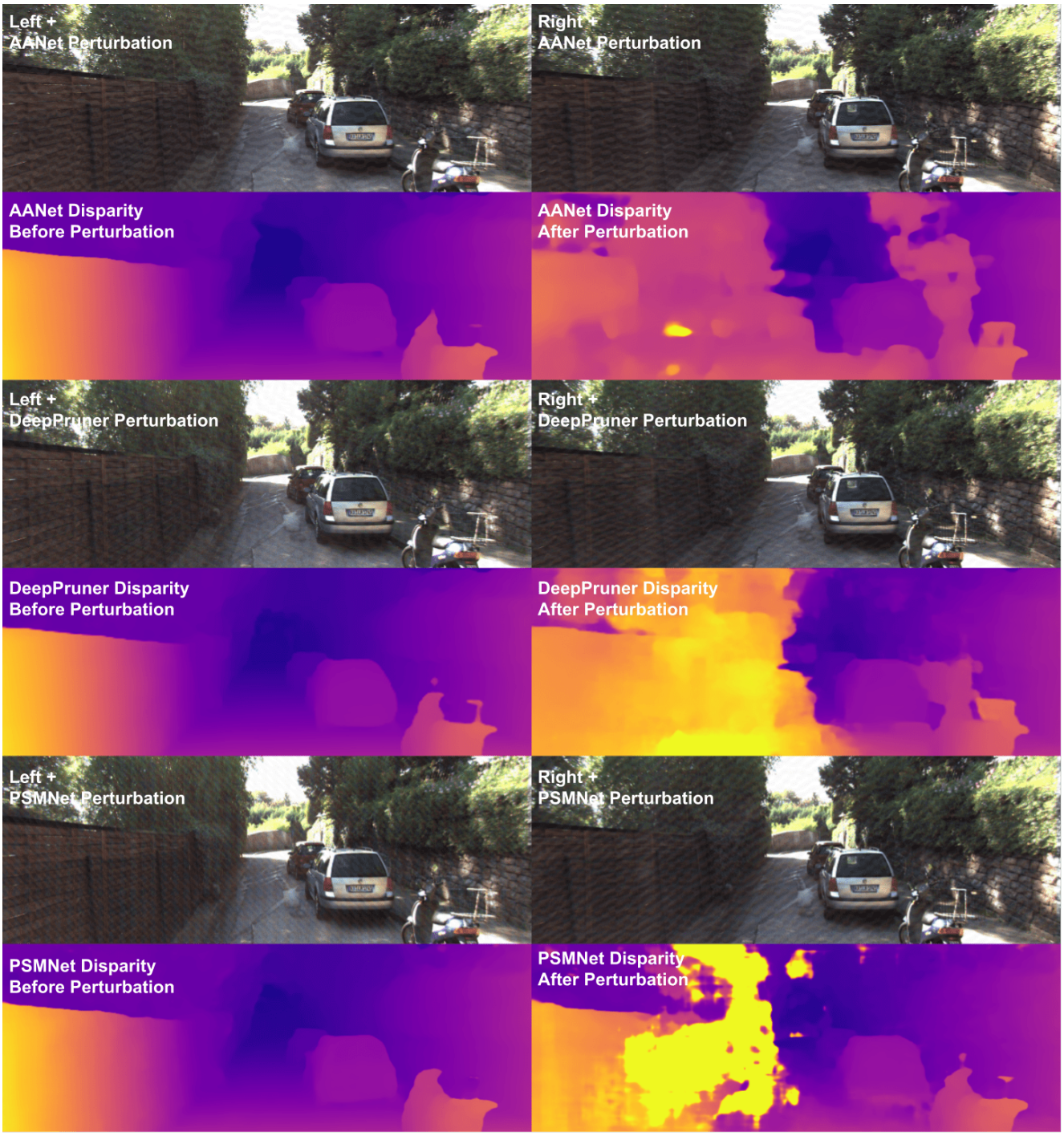


Figure 13. *Attacking AANet, DeepPruner, and PSMNet on a scene from KITTI 2012 using the SUP trained for each model.*

Speciale, and Marc Pollefeys. Patchmatchnet: Learned multi-view patchmatch stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14194–14203, 2021. 9

[28] Jamie Watson, Michael Firman, Gabriel J Brostow, and Daniyar Turmukhambetov. Self-supervised monocular depth hints. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2162–2171, 2019. 9

[29] Alex Wong, Safa Cicek, and Stefano Soatto. Targeted adversarial perturbations for monocular depth prediction. *Advances in Neural Information Processing Systems*, 33, 2020. 1

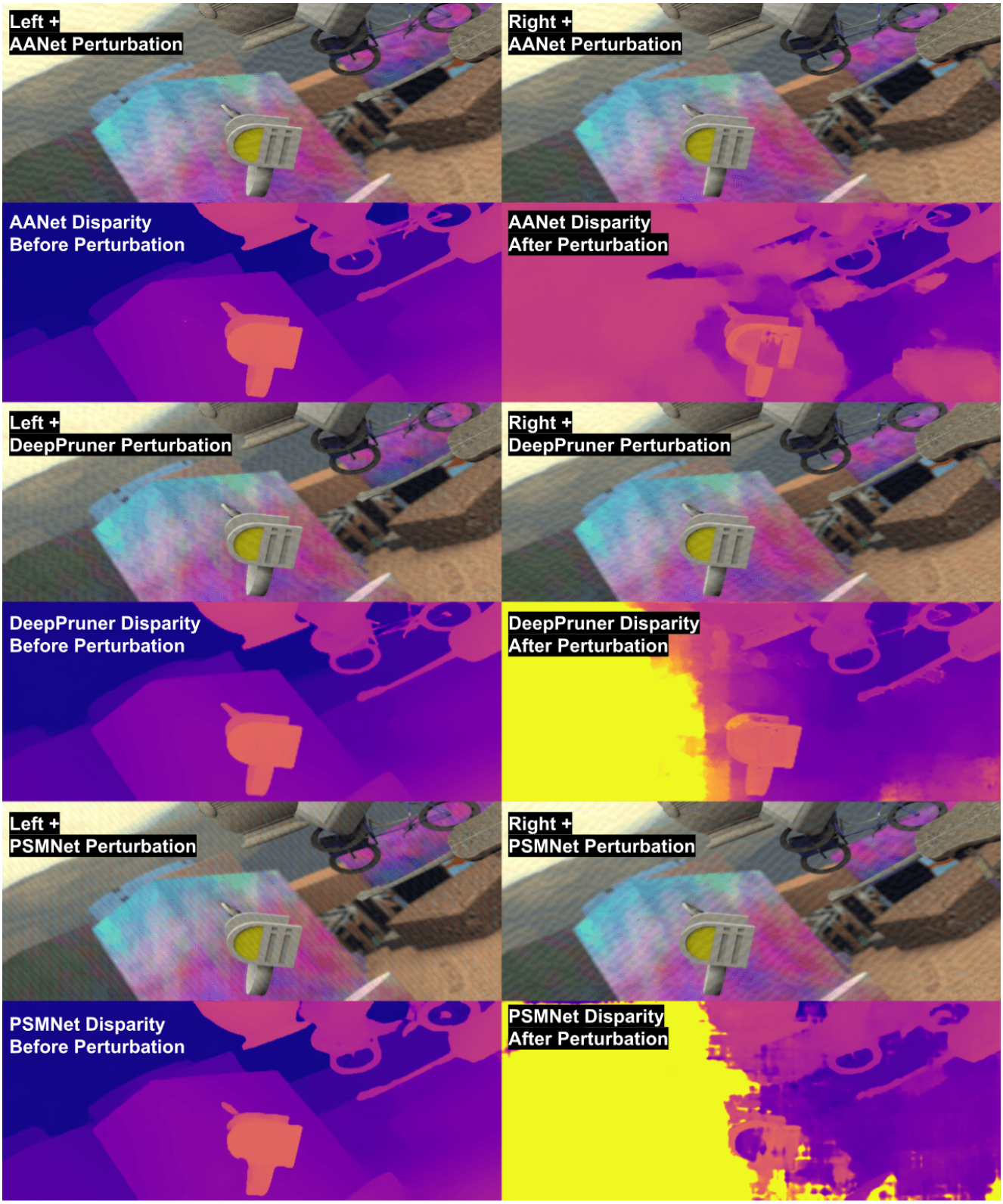


Figure 14. *Attacking AANet, DeepPruner, and PSMNet on a scene from FlyingThings3D using the SUP trained for each model.*

Figure 15. *Attacking AANet at different upper norms* $\epsilon \in \{0.002, 0.005, 0.01, 0.02\}$ *for a scene from KITTI 2015.*

[30] Alex Wong, Safa Cicek, and Stefano Soatto. Learning topology from synthetic data for unsupervised depth completion. *IEEE Robotics and Automation Letters*, 6(2):1495–1502, 2021. 9

[31] Alex Wong, Xiaohan Fei, Byung-Woo Hong, and Stefano Soatto. An adaptive framework for learning unsupervised depth completion. *IEEE Robotics and Automation Letters*, 6(2):3120–3127, 2021. 9

[32] Alex Wong, Xiaohan Fei, Stephanie Tsuei, and Stefano Soatto. Unsupervised depth completion from visual inertial odometry. *IEEE Robotics and Automation Letters*, 2020. 9

[33] Alex Wong, Mukund Mundhra, and Stefano Soatto. Stere-

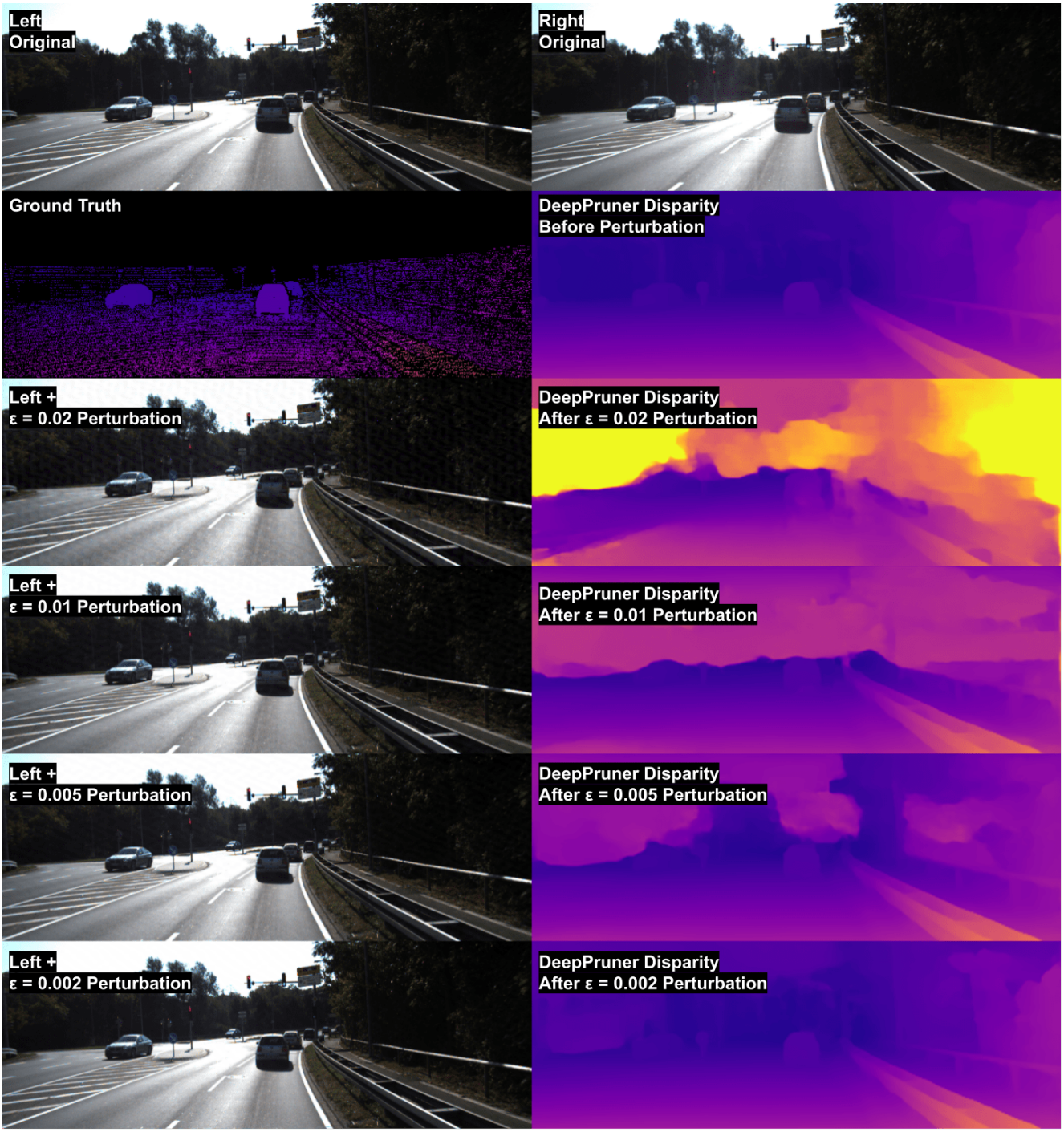


Figure 16. *Attacking DeepPruner at different upper norms* $\epsilon \in \{0.002, 0.005, 0.01, 0.02\}$ *for a scene from KITTI 2015.*

opagnosia: Fooling stereo networks with adversarial perturbations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021. 1

[34] Alex Wong and Stefano Soatto. Bilateral cyclic constraint and adaptive regularization for unsupervised monocular depth prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5644–5653, 2019. 9

[35] Alex Wong and Stefano Soatto. Unsupervised depth completion with calibrated backprojection layers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12747–12756, 2021. 9

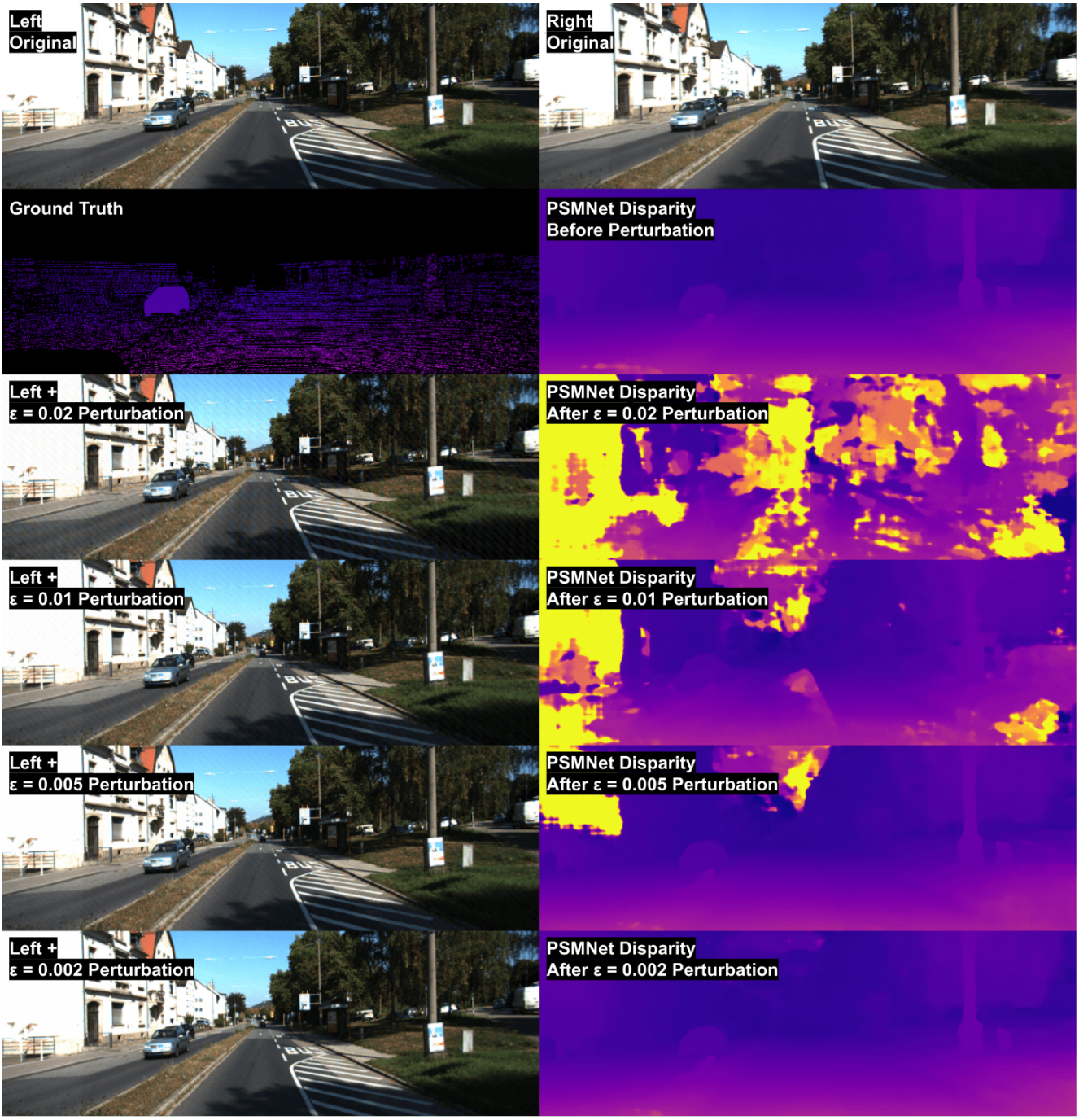


Figure 17. *Attacking PSMNet at different upper norms* $\epsilon \in \{0.002, 0.005, 0.01, 0.02\}$ *for a scene from KITTI 2015.*

[36] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. Adversarial examples for semantic segmentation and object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1369–1378, 2017. 1

[37] 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*, pages 1959–1968, 2020. 2, 8, 9

[38] Yanchao Yang and Stefano Soatto. Conditional prior networks for optical flow. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 271–287, 2018. 9

[39] Yanchao Yang, Alex Wong, and Stefano Soatto. Dense depth

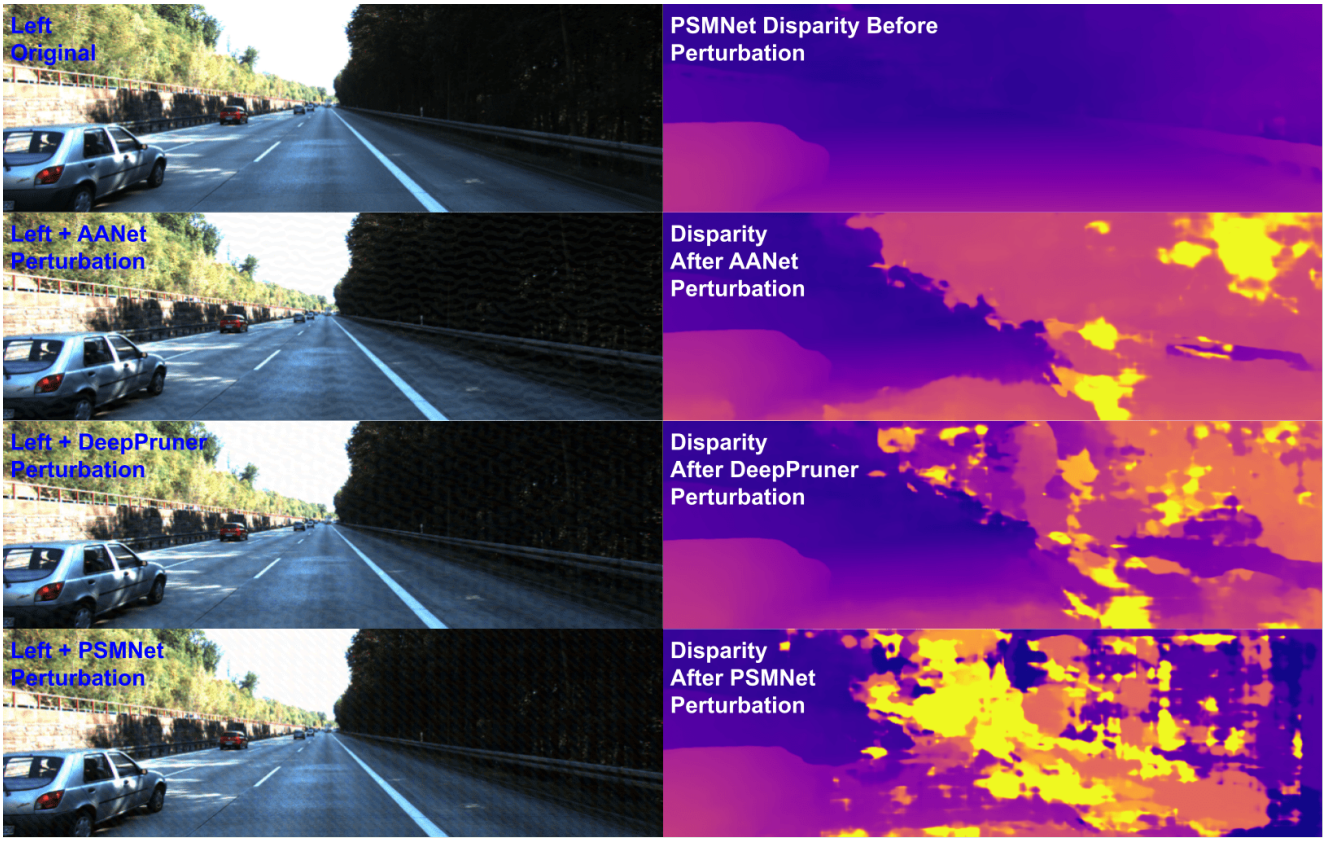


Figure 18. *Attacking PSMNet for a scene from KITTI 2015 with a stereoscopic universal perturbations optimized on KITTI for AANet, DeepPruner, and PSMNet.*

posterior (ddp) from single image and sparse range. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3353–3362, 2019. 9

[40] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 767–783, 2018. 9

[41] Yao Yao, Zixin Luo, Shiwei Li, Tianwei Shen, Tian Fang, and Long Quan. Recurrent mvsnet for high-resolution multi-view stereo depth inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5525–5534, 2019. 9

[42] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9308–9316, 2019. 8

[43] Yufan Zhu, Weisheng Dong, Leida Li, Jinjian Wu, Xin Li, and Guangming Shi. Robust depth completion with uncertainty-driven loss functions. *arXiv preprint arXiv:2112.07895*, 2021. 9

[44] Yi Zhu, Karan Sapra, Fitsum A Reda, Kevin J Shih, Shawn Newsam, Andrew Tao, and Bryan Catanzaro. Improving semantic segmentation via video propagation and label relaxation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8856–8865, 2019. 2, 7

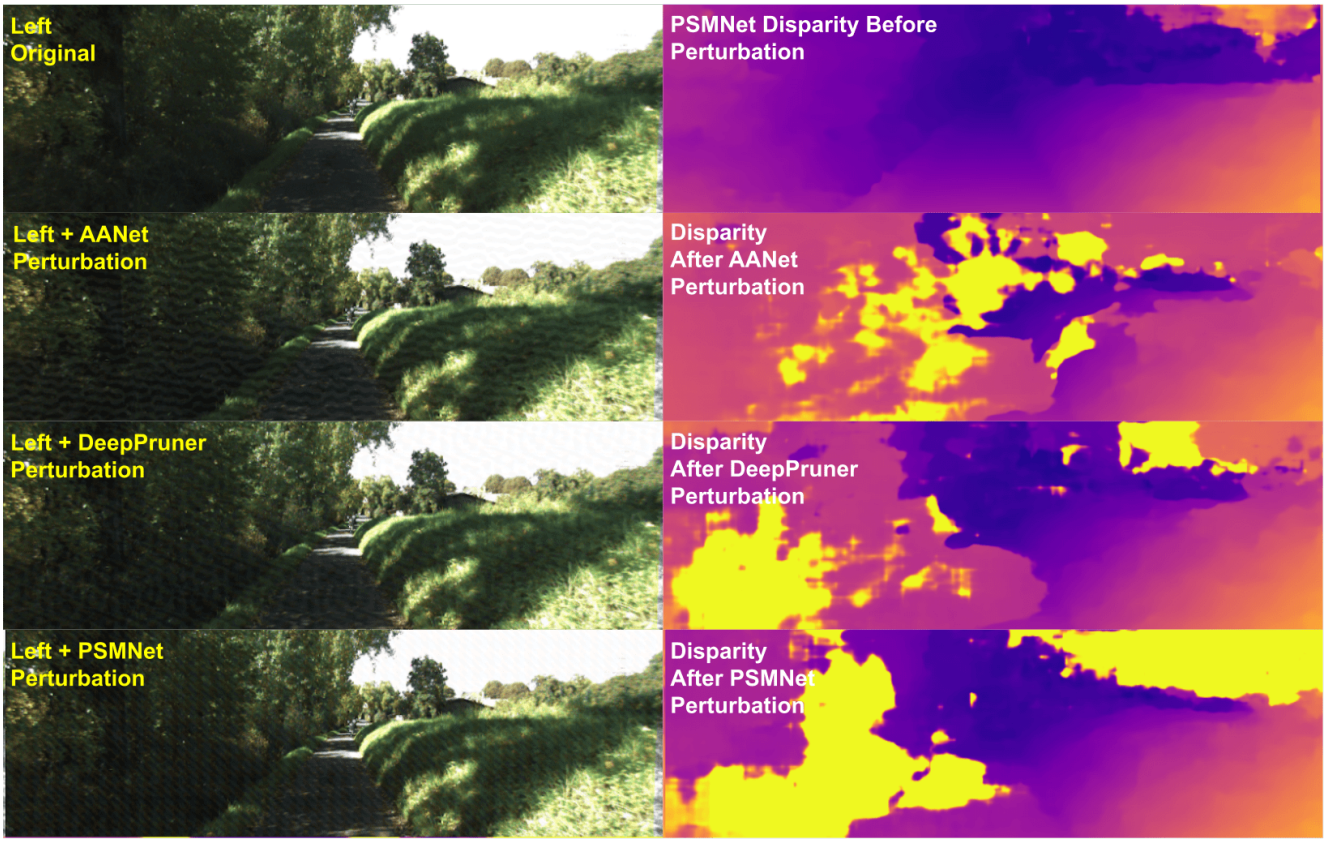


Figure 19. *Attacking PSMNet for a scene from KITTI 2012 with a stereoscopic universal perturbations optimized on KITTI for AANet, DeepPruner, and PSMNet.*

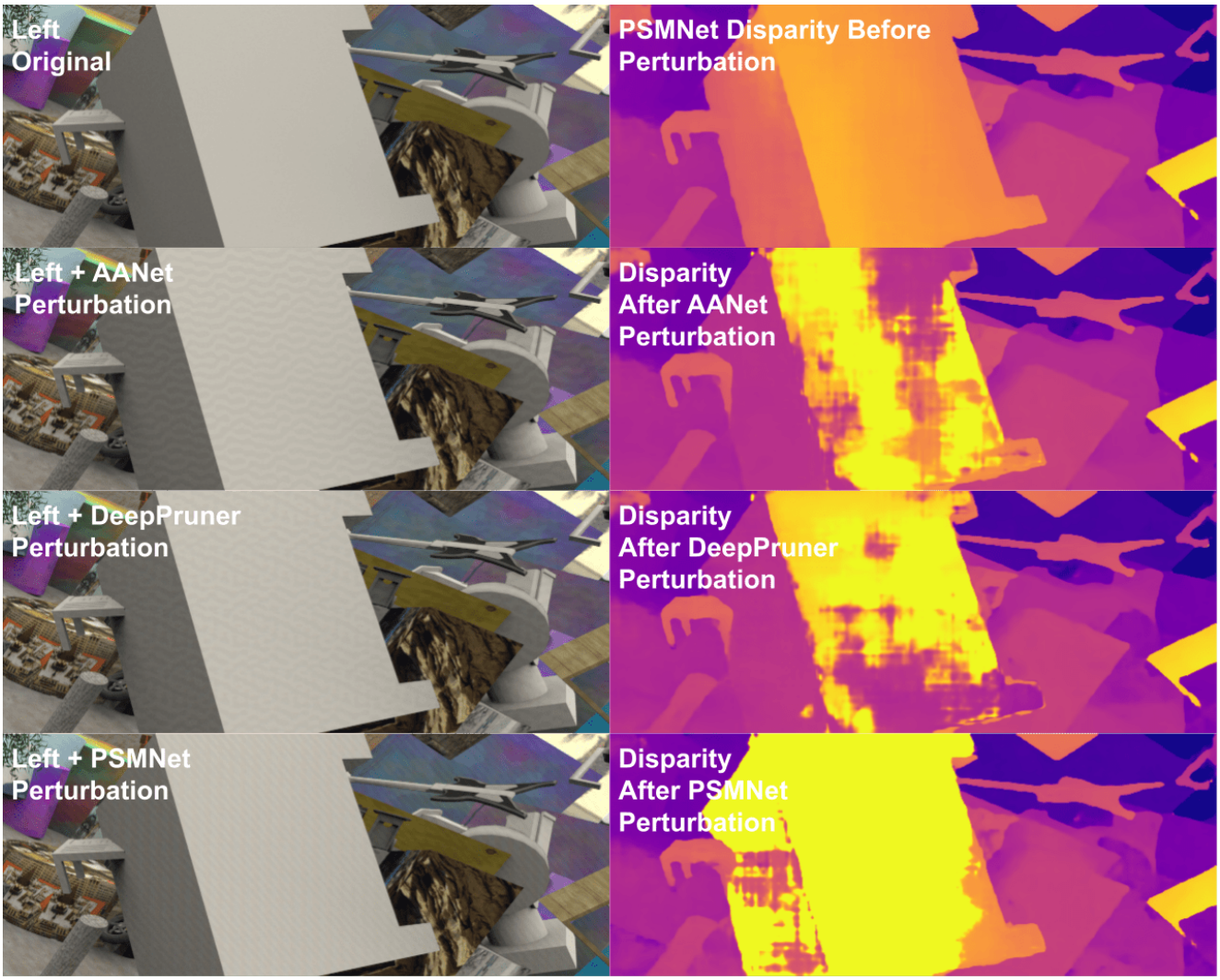


Figure 20. *Attacking PSMNet for a scene from FlyingThings3D with a stereoscopic universal perturbations optimized on KITTI for AANet, DeepPruner, and PSMNet.*