ZeroScatter: Domain Transfer for Long Distance Imaging and Vision through Scattering Media

Zheng Shi[1*] Ethan Tseng[1*] Mario Bijelic[2*] Werner Ritter[2] Felix Heide[1,3]
[1]Princeton University [2]Mercedes-Benz AG [3]Algolux

In this Supplemental Document we provide additional explanations and results in support of the main manuscript. Specifically, we provide additional details of the setup for ZeroScatter in Section 1 and we provide additional results and comparisons in Section 2.

1. Domain Transfer with ZeroScatter

In the following Section we provide additional details on the model-based synthetic supervision in Section 1.1, the multi-modal indirect supervision in Section 1.2, the temporal and stereo consistency in Section 1.3 and the generator architecture in Section 1.4.

1.1. Model-Based Synthetic Supervision

The model-based training component is facilitated by two functions, an adverse weather simulator $F_{\text{Syn}} : S^c \rightarrow S$ and a user-defined ISP processing function $F_{\text{Proc}} : X \rightarrow Y$, which we describe in the following.

Adverse Weather Simulator Existing literature [16, 10] has shown that many adverse weather patterns caused by falling particles such as rain and snow exhibit similar fog-like characteristics for far distance objects. This is because for a fixed camera resolution there is a distance beyond which individual rain and snowfall particles effectively become smaller than a single pixel. Hence, singular raindrops or snowflakes are not visible beyond that distance. We can calculate this maximal distance z by assuming a pinhole camera model:

$$z = \frac{f \cdot s}{p}, \tag{1}$$

where f is the camera focal length, s the particle size and p the pixel pitch. In our camera setup f = 8 mm and $p = 3\,\mu\text{m}$. Therefore, a rain particle with a size of 2 mm according to Garg et al. [10] is maximally visible up to a distance of 5 m and snow particles with a size of 3 mm according to Rasmussen et al. [27] up to a distance of 8.3 m.

Thus, as this fog-like characteristic is a dominant trait of adverse weather, we choose to develop our synthetic adverse weather simulator F_{Syn} using Koschmieder's scatter estimation model [21] with several key modifications. See Figure 1 for examples of our synthetic scatter generation. For a given clear raw image I, the simulated scattered raw image S is given by

$$S = It + L(1 - t), \tag{2}$$

where t is the transmission given by

$$t = e^{-\beta d}, \tag{3}$$

where d denotes the depth and β is the scattering density. We vary the value of β to produce different adverse weather patterns, see Figure 1 for examples. The depth for all methods is estimated using a PSM-Net [3] which has been finetuned on sparse lidar pointclouds. We smooth the resulting transmission maps using a guided filter [17] similar to Sakardis et al. [29].

We also extend our simulator by developing a more accurate model for spot light sources in scattering media. Previous methods assumed that the airlight was constant across the whole image. This approach has been implemented using a parameterized method similar to Sakaridis et al. [29]. For our spot light estimation we rely on techniques from traffic light

*indicates equal contribution.

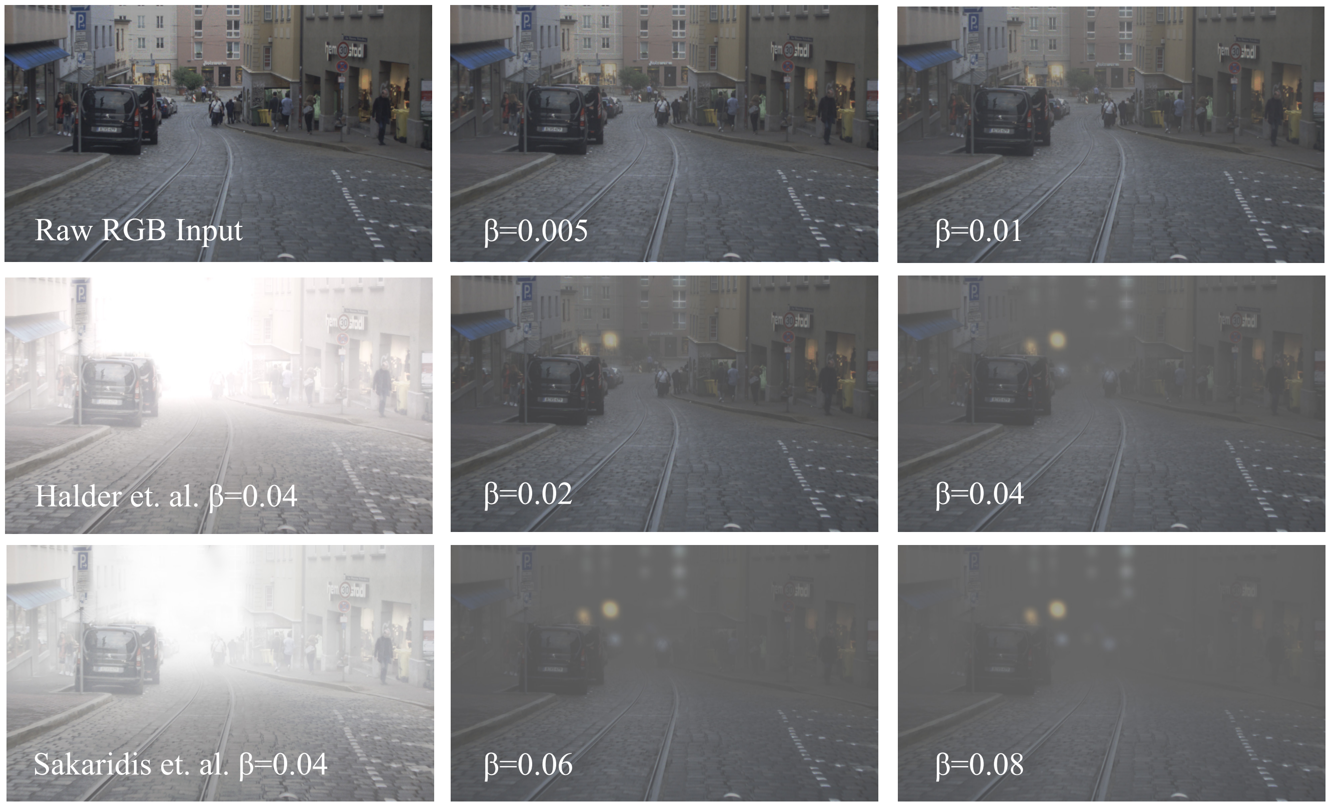

Figure 1: Our adverse weather simulator F_{Syn} produces a wide range of scattering media. We show qualitative examples for an input clear raw image and increasing fog densities of $\beta = 0.005, 0.01, 0.02, 0.04, 0.06, 0.08$. We also show a comparison to Halder et al. [16] and Sakaridis et al. [29] at fog density $\beta = 0.04$. These methods over-estimate airlights and produce unrealistically bright scatter.

detection [19, 9]. We first identify the light sources by applying a Top-hat filter on the radiance channel. Afterwards we filter all objects above the airlight target value, which is estimated using a reparameterized dark channel prior from Sakaridis et al. [29]. All selected point light sources are then overlaid on a constant airlight map. Finally the light sources are broadened to match the effect in scattering media by applying a Gaussian blur with a kernel size of 51 and a sigma value of 12.

ISP Processing Pipeline Image processing pipelines take raw image captures and output tone-mapped images that are displayed to the user. These pipelines are typically implemented on dedicated ASIC blocks and incorporate hand-crafted processing steps. We train our reconstruction network in ZeroScatter to automatically apply this image processing procedure simultaneously with descattering.

For this training purpose, we define an ISP processing function F_{Proc} which takes raw debayered automotive captures and converts them into perceptually pleasing daytime images, but note that F_{Proc} does not remove scattering media. See Figure 2 for representative examples. We implement F_{Proc} as a software ISP. Our ISP first clips the top and bottom 5% quantiles assuming top and bottom saturation. Second, we enhance the available contrast using CLAHE [34] with a kernel size of 128 and a clip limit of 0.0025. Afterwards, we apply wavelet denoising [4] to filter out high frequency noise. Finally, we apply a gamma correction curve in the HSV space with gamma factors 0.7 for lightness and 0.6 for saturation.

We emphasize that ZeroScatter is not limited to our specific ISP implementation of F_{Proc}. F_{Proc} can be adjusted towards different user subjective preferences and ZeroScatter can still be trained in the same manner as described in the main manuscript.

Multi-scale Discriminator We employ multi-scale discriminators inspired by Wang et al. [31] for both cycles. Here, the discriminator D_C is applied to the "Clear to Scatter to Clear" (C2C) direction and discriminates between tone-mapped

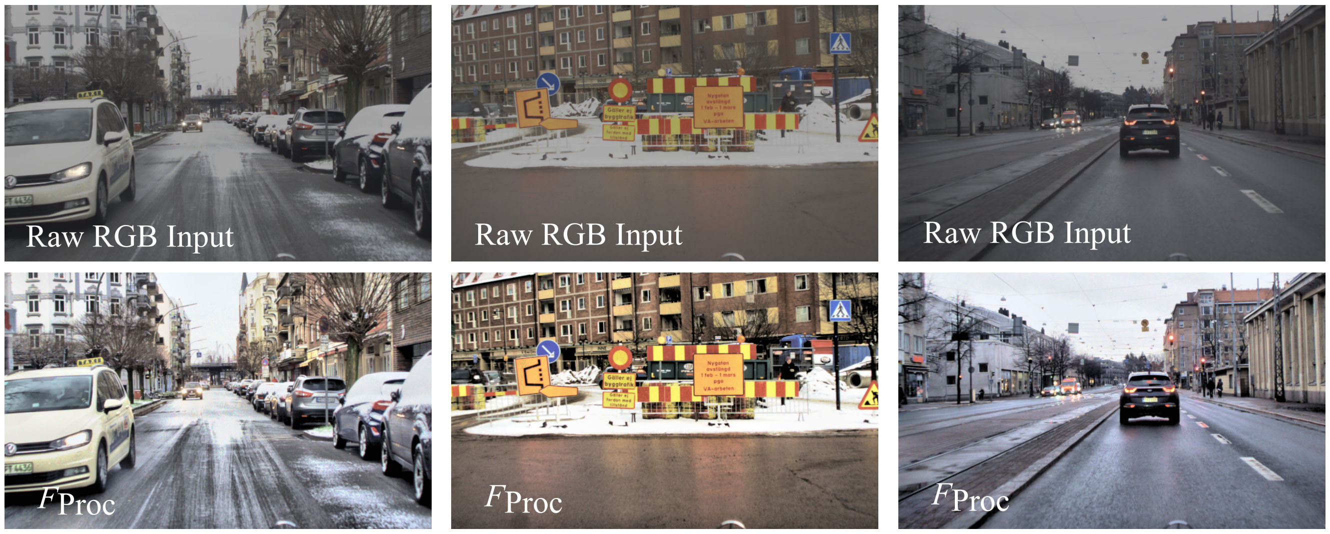

Figure 2: Our user-defined ISP processing function F_{Proc} consumes automotive raw captures and tone-maps them for the user display.

Table 1: Discriminator network architecture of d_1 and d_2. In the table, "conv-k(a)-s(b)-LRelu" represents a convolution layer with an $a \times a$ kernel window, using stride b, followed by a Leaky Relu ($\alpha = 0.2$) activation function.

Discriminator		
Name	Layer Type	Channels
Input		3
down1	conv-k4-s2-LRelu	32
down2	conv-k4-s2-LRelu	64
down3	conv-k4-s2-LRelu	128
down4	conv-k4-s2-LRelu	256
Output	conv-k4-s1	1

captures acquired under clear weather conditions and the translated outputs from our generator translation block. A separate discriminator D_S is used for the "Scatter to Clear to Scatter" (S2S) direction and judges between tone-mapped adverse weather captures and the re-scattered translated outputs. Each discriminator contains two sub-discriminators d_1 and d_2 that share the same architecture but that operate at two different resolutions: d_1 operates at the full resolution while d_2 operates at $2\times$ smaller resolution. The multi-scale discriminators offer coarse-to-fine training signals. The network architecture of d_1 and d_2 is described explicitly in Table 1.

Loss Functions Our model-based supervision aims to minimize

$$\mathcal{L}_{\text{Model}} = \mathcal{L}_{\text{C2C}} + \mathcal{L}_{\text{S2S}}. \tag{4}$$

For the C2C cycle we compute the loss using the input clear weather image $I_{\text{in}} \in X \setminus S$:

$$\mathcal{L}_{\text{C2C}} = (\mathcal{L}_1 + \mathcal{L}_{\text{perc}} + \mathcal{L}_{\text{grad}} + \mathcal{L}_{\text{adv}})(I_{\text{T}}, I_{\text{target}}), \tag{5}$$

where $I_{\text{T}} = G_{\text{T}}(F_{\text{Syn}}(I_{\text{in}}))$ and $I_{\text{target}} = F_{\text{Proc}}(I_{\text{in}})$ is the tone-mapped target image. $\mathcal{L}_1$ is the Mean Absolute Error loss

$$\mathcal{L}_1(I_{\text{T}}, I_{\text{target}}) = \left\| I_{\text{T}} - I_{\text{target}} \right\|_1 . \tag{6}$$

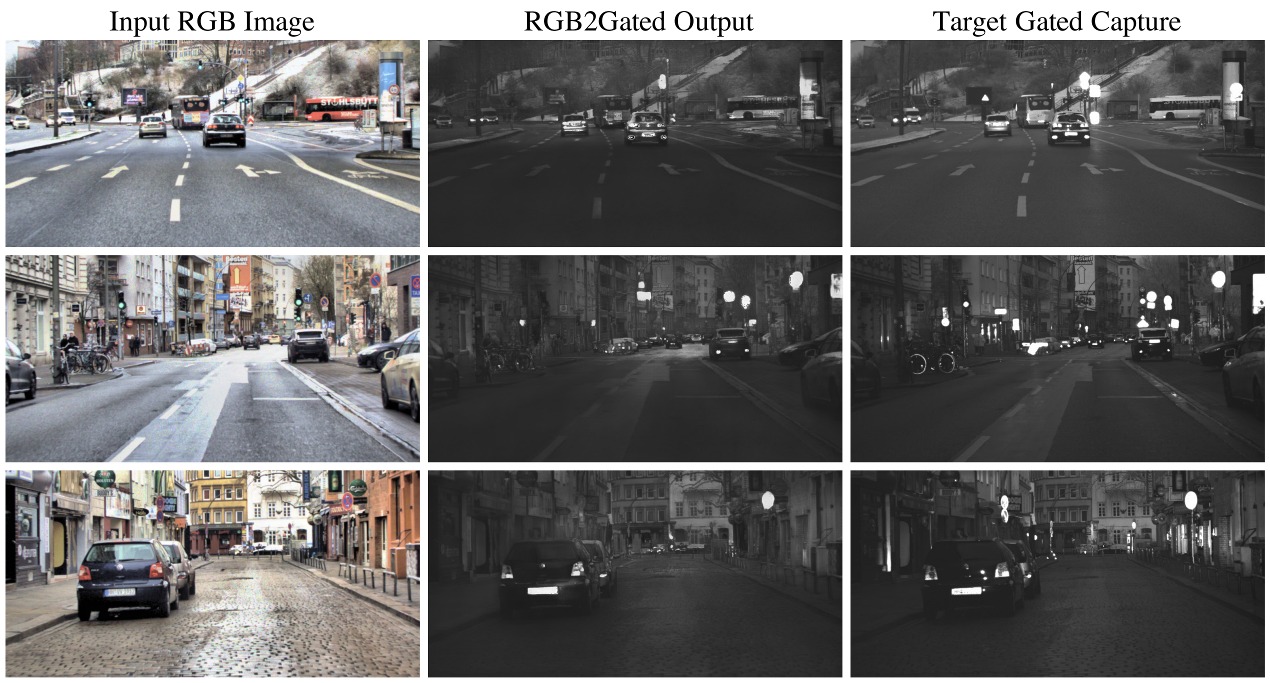

Figure 3: Gated predictions from our RGB2Gated network. Our RGB2Gated network translates tone-mapped clear day RGB captures into gated images. This allows us to train ZeroScatter using experimental gated captures under adverse weather conditions.

$\mathcal{L}_{\text{perc}}$ is a VGG-19 based perceptual loss [20]. We compute the $\mathcal{L}_1$ distance at the block2_conv1 ($\Phi_{2,1}$) and block3_conv1 ($\Phi_{3,1}$) layers of a VGG-19 network pretrained on ImageNet [7]:

$$\mathcal{L}_{\text{perc}}(I_{\text{T}}, I_{\text{target}}) = \sum_{b=2,3} (\|\Phi_{b,1}(I_{\text{T}}) - \Phi_{b,1}(I_{\text{target}})\|_1). \tag{7}$$

$\mathcal{L}_{\text{grad}}$ is an image gradient loss, where we first convolve the images with the following 4 kernels:

$$k_1 = \begin{bmatrix} 1 & -1 \end{bmatrix}, k_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, k_3 = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}, k_4 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}. \tag{8}$$

We then compute the spatial gradient loss as

$$\mathcal{L}_{\text{grad}}(I_{\text{T}}, I_{\text{target}}) = \sum_{n=1,2,3,4} (\|I_{\text{T}} * k_n - I_{\text{target}} * k_n\|_1), \tag{9}$$

where $*$ is the convolution operator.

$\mathcal{L}_{\text{adv}}$ is an adversarial loss using binary cross entropy [13] and the discriminators D_C and D_S from Section 1.1.

1.2. Multi-Modal Indirect Supervision

RGB2Gated Network Our multi-modal indirect supervision is facilitated by a pre-trained RGB2Gated network $F_{\text{RGB2Gated}}$ that predicts the gated image corresponding to a clear daytime input and allows us to compute the loss with respect to the experimentally acquired gated image. We train our RGB2Gated network on tone-mapped clear day RGB captures and their corresponding gated image captures, using the Adam optimizer with a learning rate of $1\text{e}{-4}$. The RGB2Gated network architecture is described explicitly in Table 2. Example predictions of our RGB2Gated network are shown in Figure 3.

Gated Imaging We used gated images for indirect supervision. Our gated imaging system consists of a flood-illuminator that emits a pulsed illumination and a synchronized camera that captures photons whose return travel time falls within

Table 2: RGB2Gated network architecture. In the table, "conv-k(a)-s(b)-d(c)-LRelu" represents a convolution layer with an $a \times a$ kernel window, using stride b with dilation rate c, followed by a Leaky Relu ($\alpha = 0.02$) activation function. We use convT to denote transposed convolution. Note that the final output is a single channel image.

RGB2Gated		
Name	Layer Type	Channels
Input		3
down0_1	conv-k3-s1-d1-LRelu	32
down0_2	conv-k3-s1-d1-LRelu	32
down1_1	conv-k3-s2-d1-LRelu	64
down1_2	conv-k3-s1-d1-LRelu	64
down1_3	conv-k3-s1-d1-LRelu	64
down2_1	conv-k3-s2-d1-LRelu	128
down2_2	conv-k3-s1-d1-LRelu	128
down2_3	conv-k3-s1-d1-LRelu	128
bridge_1	conv-k3-s2-d1-LRelu	256
bridge_2	conv-k3-s1-d2-LRelu	256
bridge_3	conv-k3-s1-d4-LRelu	256
bridge_4	conv-k3-s1-d8-LRelu	256
bridge_5	sum([bridge_1,bridge_2, bridge_3,bridge_4])	256
up2_1	convT-k2-s2-d1-Relu	128
up2_2	concat([up2_1, down2_3])	256
up2_3	conv-k2-s1-d1-Relu	128
up1_1	convT-k2-s2-d1-Relu	64
up1_2	concat([up1_1, down1_3])	128
up1_3	conv-k2-s1-d1-Relu	64
up0_1	convT-k2-s2-d1-Relu	32
up0_2	concat([up0_1, down0_2])	64
up0_3	conv-k2-s1-d1-Relu	32
out	conv-k3-s1-d1-Relu	1

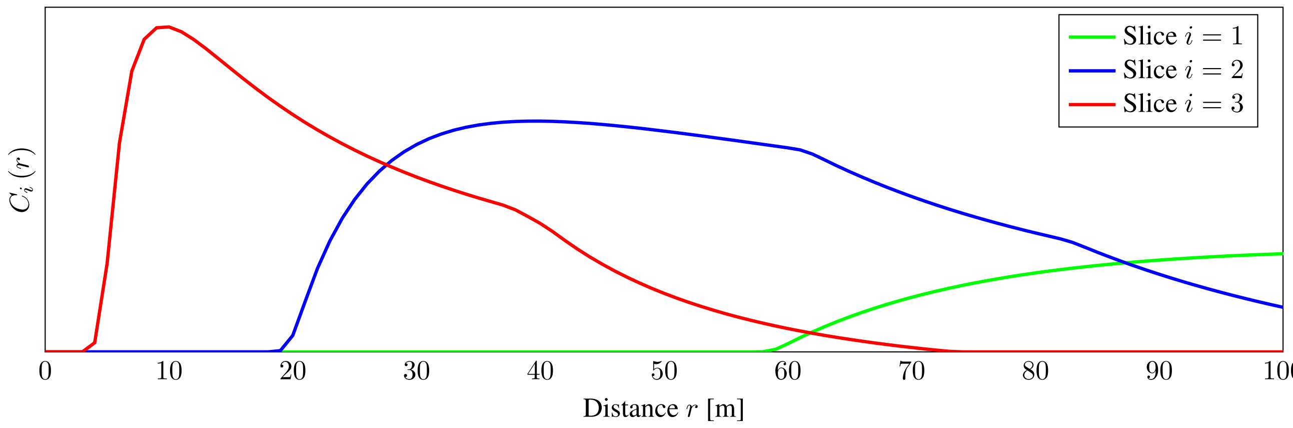

Figure 4: Range intensity profiles used to acquire gated images for multi-modal supervision. The gated system consists of a pulsed laser source and a time-synchronized imager. The range-intensity-profile $C_i(r)$ describes the distance-dependent illumination for a slice i.

select time buckets. Following Gruber et al. [15], the range-intensity profile (RIP) $C_i(r)$ describes the distance-dependent integration, which does not depend on the scene and is defined by

$$C(r) = \int_{-\infty}^{\infty} g(t-\xi)p\left(t - \frac{2r}{c}\right)\beta(r)dt, \tag{10}$$

where g is the temporally modulated camera gate, p the laser pulse profile, and $\beta(r)$ models atmospheric effects. Assuming a Lambertian scene with albedo α at distance $\tilde{r}$, the measurement for each pixel location is defined by

$$z = \alpha C(\tilde{r}) + \eta_p(\alpha C(\tilde{r})) + \eta_g, \tag{11}$$

where η_p is the Poissonian photon shot noise and η_g is the Gaussian read-out noise [8].

This coarse temporal slicing allows the gated imager to "see" through back-scatter caused by fog, rain, and snow and emphasize objects that would normally be obscured. In this work, we use the second and third slice out of the three acquired gated slices with different profiles $C_i(r)$ as shown in Figure 4. We neglect the first slice in order to focus more on objects at longer distances which are more heavily impacted by adverse weather.

Loss Functions The multi-modal supervision loss is expressed as

$$\mathcal{L}_{\text{Multi-Modal}} = \mathcal{L}_{\text{perc}}(M_{\text{ent}} \odot I_{\text{gated}}, M_{\text{ent}} \odot I'_{\text{gated}}), \tag{12}$$

where $\odot$ is point-wise multiplication and M_{ent} is a local entropy mask used to filter out areas that contain insufficient information due to extreme long distance or overly strong reflections from retro-reflectors. We compute the local entropy $H(I_{\text{gated}})$ of the target gated image by computing the local entropy of the 7×7 disk around each pixel and we mask out pixels with entropy lower than a threshold that we empirically set to be larger than 3.

$$M_{\text{ent}} = H(I_{\text{gated}}) > 3. \tag{13}$$

Lastly, we compute the perceptual loss $\mathcal{L}_{\text{perc}}$ using the outputs from block3_conv1 ($\Phi_{3,1}$), block4_conv1 ($\Phi_{4,1}$), and block5_conv1 ($\Phi_{5,1}$) of a pretrained VGG-19 following

$$\mathcal{L}_{\text{perc}}((M_{\text{ent}} \odot I_{\text{gated}}, M_{\text{ent}} \odot I'_{\text{gated}}) = \sum_{b=3,4,5} (\left\|\Phi_{b,1}(M_{\text{ent}} \odot I'_{\text{gated}}) - \Phi_{b,1}(M_{\text{ent}} \odot I_{\text{gated}})\right\|_1). \tag{14}$$

1.3. Temporal and Stereo Consistency

F_{TempWarp} denotes a bilinear image warp (implemented via TensorFlow Addons), and it warps the image based on the per-pixel flow field that we compute using the F-Net from TecoGAN [6], which we call F_{flow}. After training F_{flow} on temporal sequences of clear raw captures, we then optimize ZeroScatter for temporal consistency by penalizing inconsistencies between the processed current frame and the processing of temporally adjacent frames. This is done by warping two adjacent input frames $I_{\text{in}}^{(t+1)}$ and $I_{\text{in}}^{(t-1)}$ to the current time point t and then merging them

$$F_{\text{TempWarp}}(\{I_{\text{in}}^{(t+\epsilon)}\}) = \text{OpticalWarp}(\{I_{\text{in}}^{(t+\epsilon)}\}, F_{\text{flow}}(\text{MeanFilter}(\{I_{\text{in}}^{(t+\epsilon)}\}))) \tag{15}$$

for $\epsilon \in \{-1, 1\}$. We use two adjacent frames so that we are able to recover pixels that are invisible in one of them, such as out-of-view pixels at the image boundary and occluded pixels due to movements happened between frames. And we merge them by minimizing the warping error for each pixel

$$\begin{aligned} I'_{\text{in}}[i,j,c] = \arg\min \Bigg(& \left\| F_{\text{TempWarp}}\left(I_{\text{in}}^{(t-1)}\right)[i,j,c] - I_{\text{in}}^{t}[i,j,c] \right\|_1, \\ & \left\| F_{\text{TempWarp}}\left(I_{\text{in}}^{(t+1)}\right)[i,j,c] - I_{\text{in}}^{t}[i,j,c] \right\|_1 \Bigg). \end{aligned} \tag{16}$$

Based on the reconstructed frame I'_{in}, we compute the temporal loss using the same $\mathcal{L}_1$ loss and perceptual loss as used in the C2C cycle.

$$\mathcal{L}_{\text{Temp}} = (\mathcal{L}_1 + \mathcal{L}_{\text{perc}})(G_{\text{C}}(G_{\text{T}}(I_{\text{in}})), G_{\text{T}}(I'_{\text{in}})). \tag{17}$$

$F_{\text{StereoWarp}}$ is a disparity-based warping adapted from MonoDepth [12]. For the disparity prediction we use a pre-trained PSM-Net [3] which is finetuned on clear raw captures and sparse lidar measurements for depth supervision. To avoid penalizing warping errors caused by occlusions, we compute a visibility mask M_{Stereo} that removes these pixel mis-matches. We formulate the mask as

$$M_{\text{Stereo}} = \exp(-\alpha \mathcal{L}_1(I_{\text{in}}, I'_{\text{in}})), \tag{18}$$

where we set $\alpha = 10$. The stereo consistency loss is defined as the $\mathcal{L}_1$ distance between the masked output of the current left frame and the warped right frame.

$$\mathcal{L}_{\text{Stereo}} = \left\| M_{\text{Stereo}} \odot G_{\text{C}}(G_{\text{T}}(I_{\text{in}})), M_{\text{Stereo}} \odot G_{\text{T}}(I'_{\text{in}}) \right\|_1 . \tag{19}$$

1.4. Generator Architecture

Our ZeroScatter generator network consists of two sequential components: a translation block G_{T} and a consistency block G_{C}. Our translation block consists of two streams, one which operates at the full resolution and the other at a $2\times$ smaller resolution. We use an extended encoder with parallel feature extraction streams in both streams to expand the network's receptive field size. Our U-Net structured consistency network further removes artifacts such as snowflakes and sensor noise from the translation block output to enforces temporal consistency and stereo consistency. The detailed architecture of our generator is described explicitly in Table 3.

1.5. Training Data

We train our model using a dataset consisting of automotive captures in harsh weather scenarios from [1]. The dataset contains RGB captures under different weather types and the corresponding data acquired by lidars, a gated imager, and an FIR camera, as well as auxiliary information such as temperature and vehicle speed. 48.5% of the captures are in clear conditions, 6.9% are in dense fog conditions, 8.1% are in light fog conditions, and 36.5% are in precipitation (snow/rain fall) conditions.

2. Assessment

In the following Section we are going to provide more information about the method assessment. Section 2.1 provides more information about the baseline methods, Section 2.2 shows experiments in a controlled fog chamber, Section 2.3 and Section 2.4 provide more qualitative results with synthetically generated and real-world captured adverse weather data. Lastly, Section 2.5 adds details about the object detection evaluation and shows qualitative detection results.

2.1. Baseline methods

Implementations We compare ZeroScatter to three types of baseline methods: traditional tone-mapping methods such as F_{Proc} which we described in Section 1.1, fully supervised descattering methods, and unpaired style transfer methods. Fully supervised methods include EPDN [26], PFF-Net [24], DehazeNet [2], and Bidirectional-FCN [25]. These methods require pixel-wise correspondence between the input and target images and thus can only be trained with synthetic data. In contrast, unpaired image-to-image translation methods such as CyCADA [18], CycleGAN [33], and ForkGAN [32] do not require such direct supervision because of their cycle consistency training and are capable of learning from unpaired experimental captures under clear and adverse weather conditions. All methods are trained and tested on 12-bit input images.

For fully supervised methods, we train the models on simulated adverse weather data generated using our adverse weather simulator F_{Syn} which we described previously in Section 1.1. For EPDN and PFF-Net, we use the tone-mapped clear day images as target images. DehazeNet and Bidirectional-FCN cannot be directly trained on tone-mapped clear day images as these methods attempt to estimate the unknown parameters of a fixed adverse weather formation model, specifically airlight L and transmission t. Thus we train DehazeNet and Bidirectional-FCN using raw clear day images as the target and apply the same tone-mapping process F_{Proc} onto the model output as a post processing step. We train the unsupervised methods CyCADA, CycleGAN, and ForkGAN on experimental captures only, where the models are shown unpaired tone-mapped clear day captures and captures under adverse weather conditions. All methods were optimized following the original implementations provided by the respective authors.

Qualitative Analysis Traditional methods such as CLAHE are able to tone-map the image but cannot remove any scattering media. Unsupervised image stylization methods CyCADA, CycleGAN and ForkGAN perform better, but are still

Table 3: Generator network architecture. Our reconstruction network consists of two components: a translation block and a consistency block. In the table, "conv-k(a)-s(b)-d(c)-IN-LRelu" represents a convolution layer with an $a \times a$ kernel window, using stride b with dilation rate c, followed by instance normalization and a Leaky Relu ($\alpha = 0.02$) activation function. We use convT to denote transposed convolution and ResNet to denote ResNet blocks.

Translation Block G_{T}		
Name	Layer Type	Channels
Input		3
	Low Resolution Stream	
down1_1	avgpool-k3-s2	3
down1_2	conv-k7-s1-d1-LRelu	32
down1_glo	conv-k5-s1-d2-IN-LRelu $\times 3$	32
down1_loc	conv-k3-s1-d1-IN-LRelu $\times 3$	32
down1_3	concat[down1_glo, down1_loc]	64
down1_4	conv-k3-s1-d1-IN-LRelu	64
down2	conv-k3-s2-d1-LRelu	128
down3	conv-k3-s2-d1-LRelu	256
down4	conv-k3-s2-d1-LRelu	512
bridge	ResNet-k3-s1-d1-LRelu $\times 6$	512
up4	convT-k3-s2-d1-Relu	256
up3	convT-k3-s2-d1-Relu	128
out_lowres	convT-k3-s2-d1-Relu	64
	High Resolution Stream	
down0_1	conv-k7-s1-d1-LRelu	16
down0_glo	conv-k5-s1-d2-IN-LRelu $\times 3$	16
down0_loc	conv-k3-s1-d1-IN-LRelu $\times 3$	16
down0_2	concat[down0_glo, down0_loc]	32
down0_3	conv-k3-s1-d1-IN-LRelu	32
down1	conv-k3-s2-d1-LRelu	64
fusion	add[down1, out_lowres].	64
bridge	ResNet-k3-s1-d1-LRelu $\times 3$	64
up1	convT-k3-s2-d1-Relu	32
out	conv-k7-s1-d1-Relu	3

Consistency Block G_{C}		
Name	Layer Type	Channels
Input		3
down0_1	conv-k3-s1-d1-LRelu	32
down0_2	conv-k3-s1-d1-LRelu	32
down1_1	conv-k3-s2-d1-LRelu	64
down1_2	conv-k3-s1-d1-LRelu	64
down1_3	conv-k3-s1-d1-LRelu	64
down2_1	conv-k3-s2-d1-LRelu	128
down2_2	conv-k3-s1-d1-LRelu	128
down2_3	conv-k3-s1-d1-LRelu	128
down3_1	conv-k3-s2-d1-LRelu	256
down3_2	conv-k3-s1-d1-LRelu	256
down3_3	conv-k3-s1-d1-LRelu	256
down4_1	conv-k3-s2-d1-LRelu	512
down4_2	conv-k3-s1-d1-LRelu	512
down4_3	conv-k3-s1-d1-LRelu	512
up3_1	convT-k2-s2-d1-Relu	256
up3_2	concat([up3_1, down3_3])	512
up3_3	convT-k2-s1-d1-Relu	256
up2_1	convT-k2-s2-d1-Relu	128
up2_2	concat([up2_1, down2_3])	256
up2_3	conv-k2-s1-d1-Relu	128
up1_1	convT-k2-s2-d1-Relu	64
up1_2	concat([up1_1, down1_3])	128
up1_3	conv-k2-s1-d1-Relu	64
up0_1	convT-k2-s2-d1-Relu	32
up0_2	concat([up0_1, down0_2])	64
up0_3	conv-k2-s1-d1-Relu	32
out	conv-k3-s1-d1-Relu	3

unable to recover high-quality scatter-free images from perturbed input images. Compared to CycleGAN, CyCADA utilizes an additional training objective (object detection) to provide more training cues and is thus able to achieve better results than CycleGAN. ForkGAN also utilizes cycle consistency from CycleGAN and was previously designed for night-to-day image stylization. It uses a compact domain-invariant representation to create both day and night scenes. While it may be efficient for day-to-night translation, however, when used for adverse weather descattering the intermediate latent representation also reduces the information flow and thus makes it difficult for the model to recover fine details from the adverse weather captures. Taking advantage of pixel-wise training cues, fully supervised descattering methods EPDN, PFF-Net, DehazeNet, and Bidirectional-FCN produce better performance compared to the unsupervised methods. However, DehazeNet and Bidirectional-FCN suffer from amplified prediction error caused by their disjoint learning approach. EPDN and PFF-Net are both trained in an end-to-end fashion and demonstrate strong performance on the synthetic dataset. However, they are not capable of including experimental captures into training and consequently are not robust to out-of-distribution test inputs and fail to generalize to in-the-wild adverse weather captures.

Additional Ablations We demonstrate the importance of our generator architecture with a comparison where we train PFF-Net using our specialized loss supervision, see Figure 5. Furthermore, we demonstrate the importance of the physical

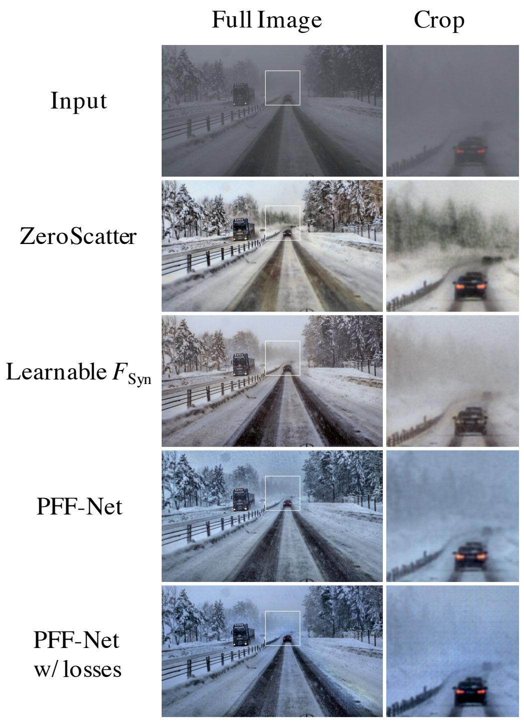

Figure 5: Additional ablation comparisons. ZeroScatter outperforms baseline methods that simply use the proposed loss functions. Replacing F_{Syn} with a learnable mapping results in worse performance.

model for F_{Syn} with a comparison where we replace F_{Syn} with a learnable U-Net as is done in CycleGAN, see Figure 5. Replacing F_{Syn} with a learnable U-Net results in drastically lower quality, we attribute this to the difficulty of learning how to apply various intensity and depth-dependent effects seen in realistic atmospheric scattering.

2.2. Controlled Fog Chamber Environment

To evaluate the method on non-synthetic experimental captures with paired pixel-wise ground truth targets, we use experimental captures from Gruber et al. [14] that were taken in a controlled fog chamber environment, see Figure 6. The dataset contains four scenes consisting of a construction area, highway, pedestrian zone, and residential area. For each scenario a target image is acquired under clear weather conditions. The adverse weather samples cover light rain, heavy rain, and span 17 different visibility levels in fog (20-100 m in 5 m increments), resulting in 1600 samples in total. Additional qualitative results of our method and all baseline methods using this dataset are presented in Figure 6.

2.3. Synthetic Dataset Evaluation

Qualitative results of ZeroScatter and all baseline methods applied to synthetically generated adverse weather conditions are shown in Figure 7 and Figure 8. Our method produces the most accurate reconstructions from the degraded input images.

2.4. In-the-Wild Experimental Evaluation

Additional qualitative results of our method and all baseline methods applied to real-world automotive captures are shown in Figures 9, 10, and 11. Our high-quality outputs demonstrate the proposed method's descattering ability under diverse conditions seen in-the-wild.

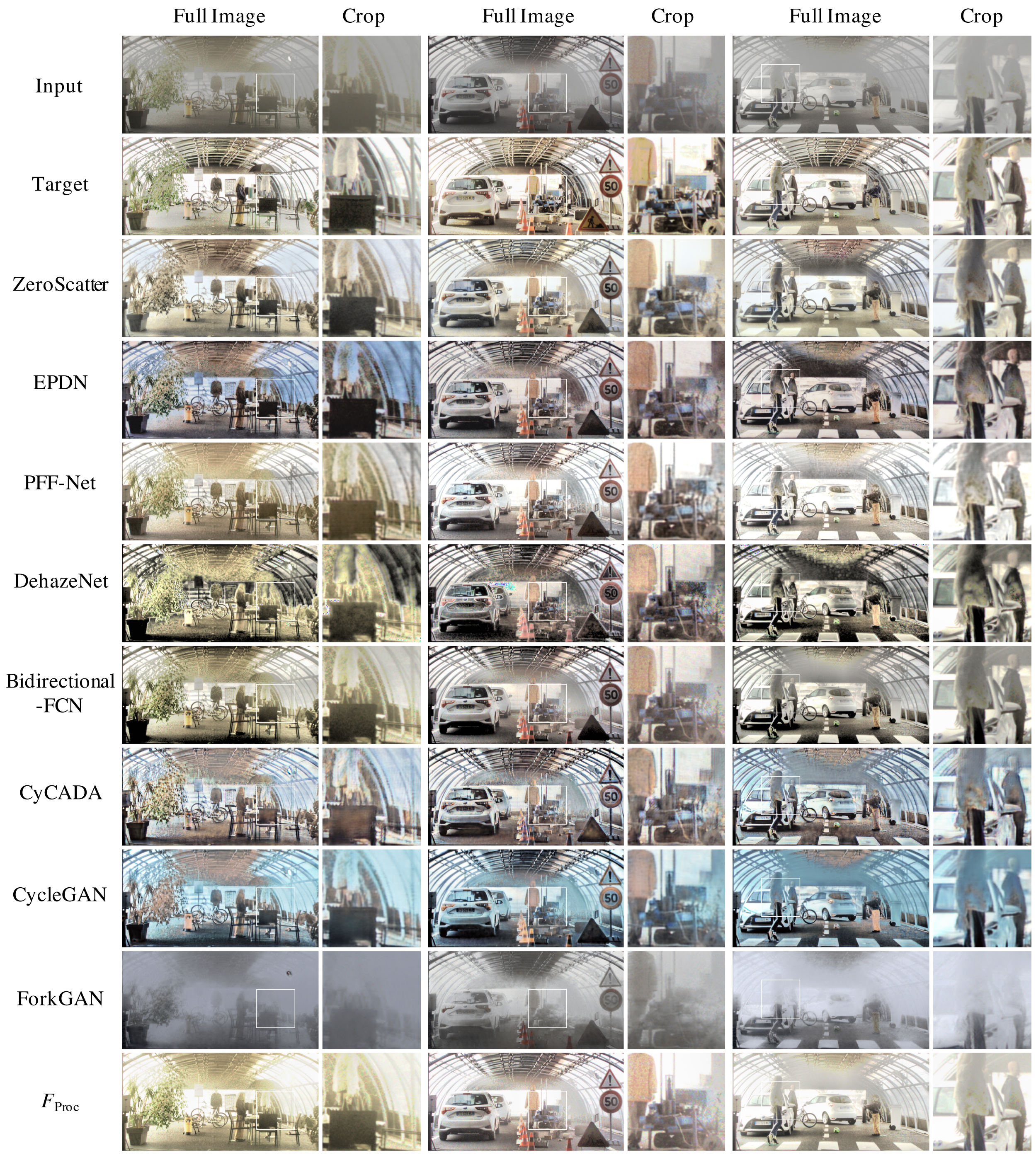

Figure 6: Additional qualitative performance comparison on controlled fog chamber measurements. Compared to the baseline methods, ZeroScatter produces more accurate reconstructions with less reconstruction noise.

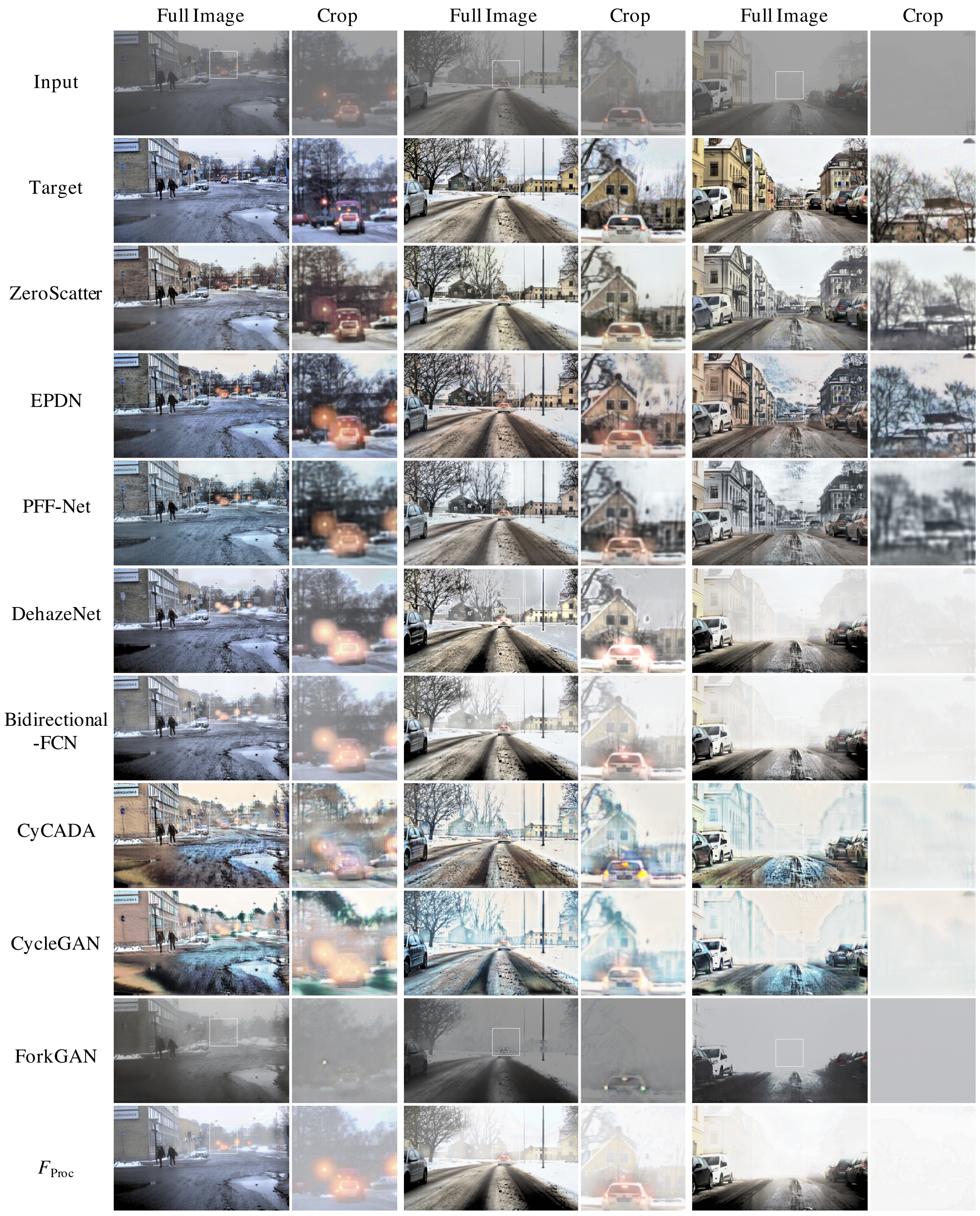

Figure 7: Qualitative performance comparison on scenes with synthetic adverse weather. Compared to the baseline methods, ZeroScatter is less affected by the halos around lights, which can be seen in the first example. ZeroScatter produces accurate, high contrast outputs without amplifying noise, as shown in the second and third examples.

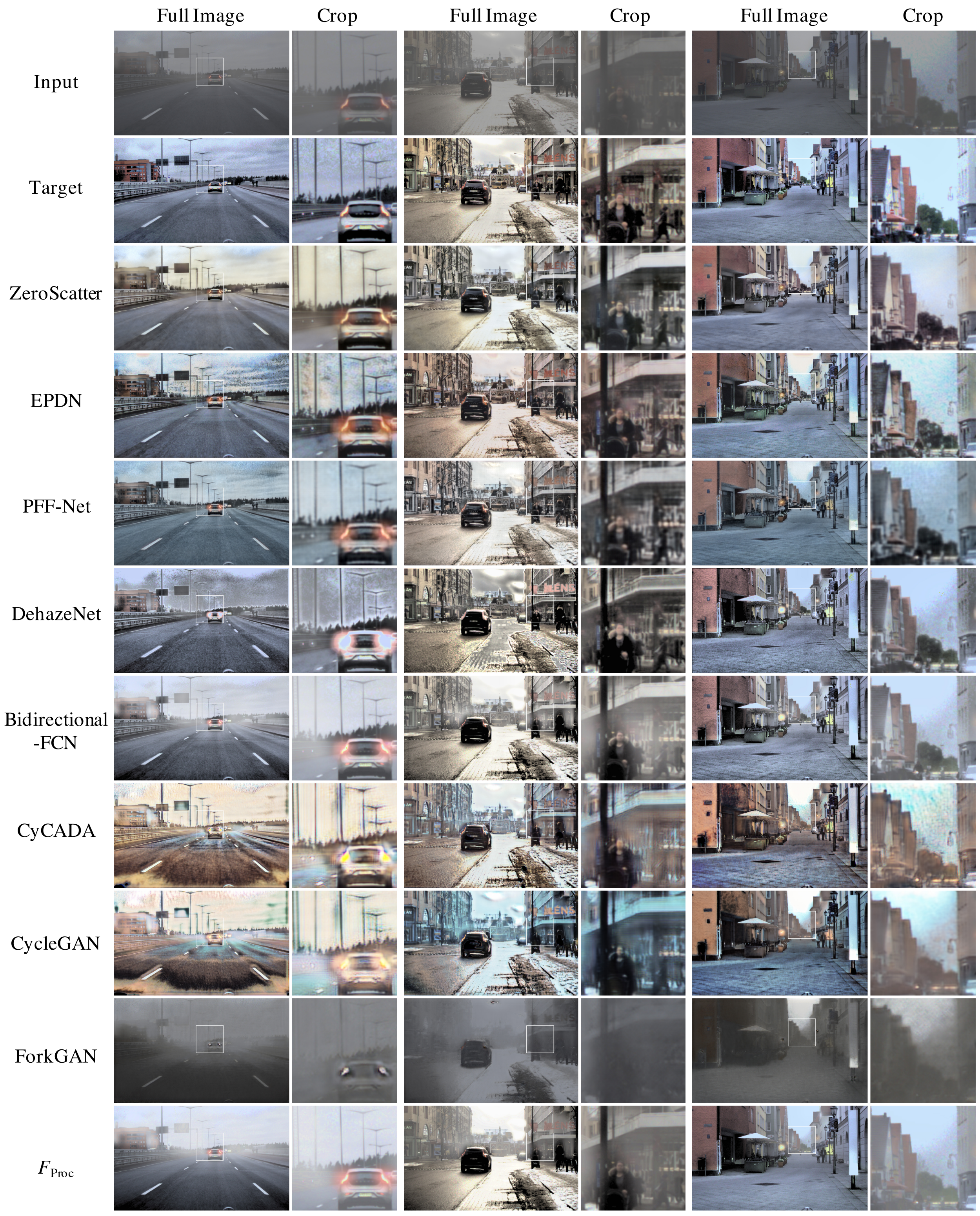

Figure 8: Qualitative performance comparison on scenes with synthetic adverse weather. Compared to the baseline methods, ZeroScatter is less affected by the halos around lights, which can be seen in the first example. ZeroScatter produces accurate, high contrast outputs without amplifying noise, as shown in all three examples.

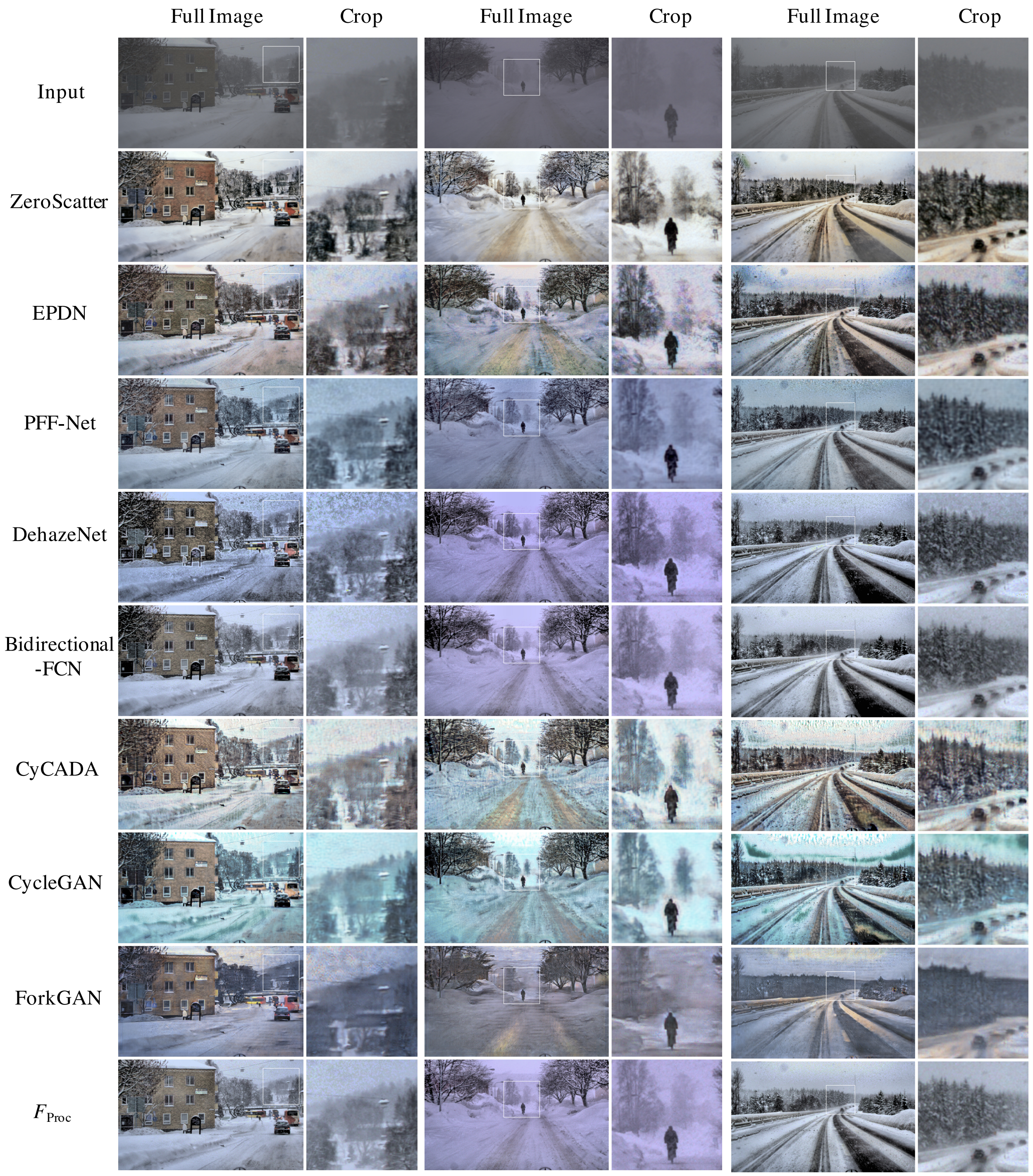

Figure 9: Qualitative performance comparison on scenes with real-world adverse weather. Our approach significantly reduces scattering media present in the scene and reveals object at long distances, such as the trees and cars in all three examples. Compared to EPDN and PFF-Net, ZeroScatter produces images with better contrast and less reconstruction noise.

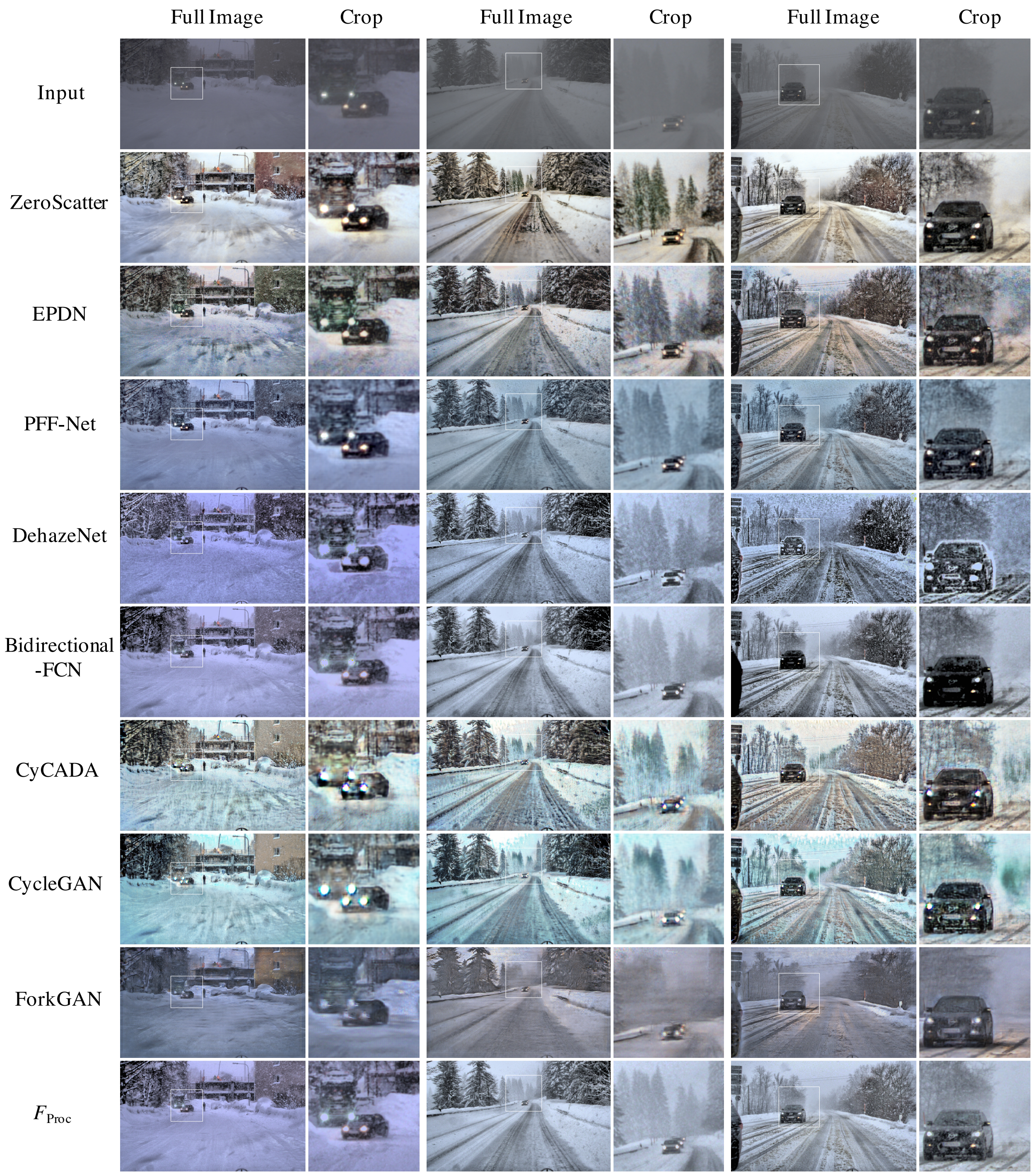

Figure 10: Qualitative performance comparison on scenes with real-world adverse weather. Our approach significantly reduces scattering media present in the scene and is able to remove snowflakes, as can be seen in the first and third examples. ZeroScatter reveals objects at long distances, such as the trees and cars in the second example, and produces images with better contrast and less reconstruction noise than EPDN and PFF-Net.

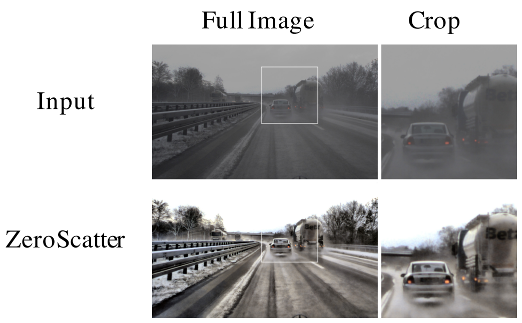

Figure 11: In addition to descattering for heavy snow and dense fog scenes, as shown in Figures 9 and 10, ZeroScatter also performs high-quality descattering for real-world heavy rain scenes.

Table 4: SSD network architecture. In the table, "conv-k(a)-s(b)-d(c)-Relu" represents a convolution layer with an $a \times a$ kernel window, using stride b with dilation rate c, followed by a Relu activation function. **Bold** feature maps indicate features used for bounding box regression and classification.

SSD		
Name	Layer Type	Channels
Input		3
conv0_1	conv-k3-s1-d1-Relu	32
conv0_2	conv-k3-s1-d1-Relu	32
maxpool0	maxpool-k2-s2-d1	32
conv1_1	conv-k3-s2-d1-Relu	64
conv1_2	conv-k3-s1-d1-Relu	64
maxpool1	maxpool-k2-s2-d1	64
conv2_1	conv-k3-s1-d1-Relu	128
conv2_2	conv-k3-s1-d1-Relu	128
conv2_3	conv-k3-s1-d1-Relu	128
conv2_3	conv-k3-s1-d1-Relu	128
maxpool2	maxpool-k2-s2-d1	128
conv3_1	conv-k3-s1-d1-Relu	256
conv3_2	conv-k3-s1-d1-Relu	256
conv3_3	conv-k3-s1-d1-Relu	256
conv3_4	conv-k3-s1-d1-Relu	256
maxpool3	maxpool-k2-s2-d1	256
conv4_1	conv-k3-s1-d1-Relu	512
conv4_2	conv-k3-s1-d1-Relu	512
conv4_3	conv-k3-s1-d1-Relu	512
conv4_4	conv-k3-s1-d1-Relu	512
maxpool3	maxpool-k3-s1-d1	512
conv5_1	conv-k3-s1-d6-Relu	1024
conv6_1	conv-k2-s1-d1-Relu	1024
conv7_1	conv-k1-s1-d1-Relu	256
conv7_2	conv-k3-s2-d1-Relu	512
conv8_1	conv-k1-s1-d1-Relu	128
conv8_2	conv-k3-s2-d1-Relu	256
conv9_1	conv-k1-s1-d1-Relu	128
conv9_2	conv-k3-s2-d1-Relu	256
conv10_1	conv-k1-s1-d1-Relu	128
conv10_2	conv-k3-s2-d1-Relu	256

2.5. Object Detection

For our object detection evaluation we rely on a SSD model [23] with a feature pyramid consisting out of 6 feature maps and a truncated VGG backbone [30] similar to AVOD [22] and MV3D [5]. The detailed network architecture can be found in Table 4.

For object localization we regress the bounding boxes using 21 default anchor boxes for cars and pedestrians. The anchors are optimized using k-nearest neighbor clustering following Redmon et al. [28]. During training the ground truth boxes are assigned to matching anchors with learned relative displacements x_i using a minimum intersection over union score of 0.5. Matching boxes are refined using a Huber loss $H(x)$ given by

$$H(x) = \begin{cases} x^2/2, & \text{if } |x| < 1 \\ |x| - 0.5 & \text{if } |x| > 1 \end{cases} . \tag{20}$$

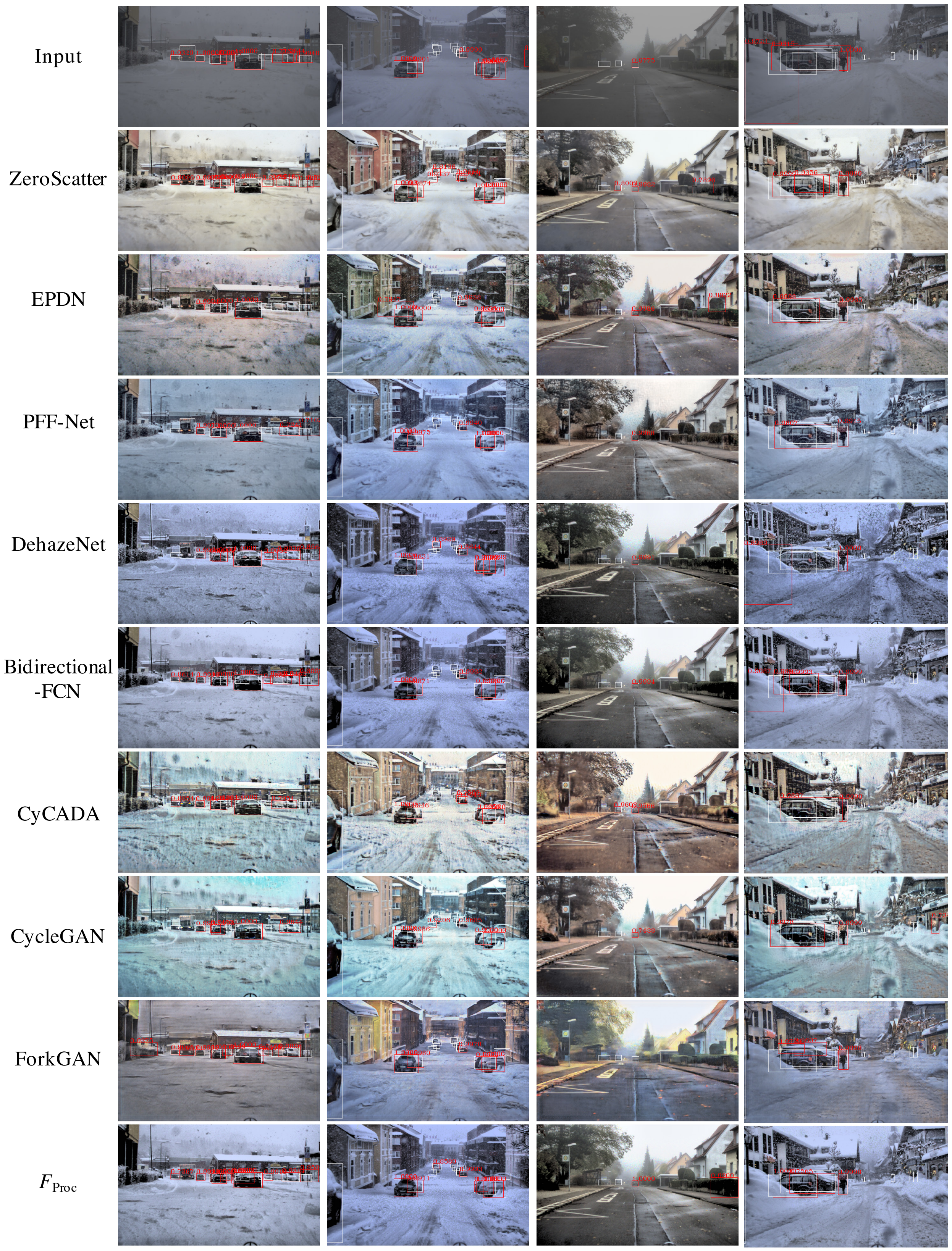

Figure 12: Object detection qualitative comparisons. Each row shows the detector's performance on each method's output. White boxes are the ground truth labels; red boxes and the number in the corner are detection box output and the corresponding confidence respectively. Compared to the baseline methods, ZeroScatter allows the detector to detect more objects at far distances, such as in the first three examples, and with higher confidence, such as in the first and fourth examples.

To assign the correct object category y_i with probability p_i we rely on the cross entropy loss with softmax, which is defined as

$$H(p) = \sum_i (y_i \log(p_i) + (1 - y_i) \log(1 - p_i)). \tag{21}$$

Negative non-matching anchors are only considered in the class assignment and restricted to $5\times$ the number of positive examples using hard example mining [23].

We first train a base object detection network on raw RGB input data for 60 epochs using the Adam optimizer with a learning rate of $5\mathrm{e}{-4}$ and applying the $\mathcal{L}_2$ loss with weight decay of $5\mathrm{e}{-4}$. We then separately finetuned the base RGB object detection network on the outputs of each method for 10 epochs. We apply a non-maximum suppression (NMS) of 0.5 to all methods. The evaluation categories are based on the easy, moderate, and hard categories defined in Geiger et al. [11].

Qualitative object detection results of ZeroScatter and all baseline methods are shown in Figure 12. ZeroScatter is able to remove scattering media and improve object detection in adverse weather through more accurate and higher confidence detections, especially for objects at long distances.

References

[1] M. Bijelic, T. Gruber, F. Mannan, F. Kraus, W. Ritter, K. Dietmayer, and F. Heide. Seeing through fog without seeing fog: Deep multimodal sensor fusion in unseen adverse weather. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 7

[2] B. Cai, X. Xu, K. Jia, C. Qing, and D. Tao. Dehazenet: An end-to-end system for single image haze removal. *IEEE Transactions on Image Processing*, 25(11):5187–5198, 2016. 7

[3] J.-R. Chang and Y.-S. Chen. Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5410–5418, 2018. 1, 7

[4] S. G. Chang, Bin Yu, and M. Vetterli. Adaptive wavelet thresholding for image denoising and compression. *IEEE Transactions on Image Processing*, 9(9):1532–1546, 2000. 2

[5] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1907–1915, 2017. 16

[6] M. Chu, Y. Xie, J. Mayer, L. Leal-Taixé, and N. Thürey. Learning temporal coherence via self-supervision for gan-based video generation. *ACM Transactions on Graphics (TOG)*, 39:75:1 – 75:13, 2020. 6

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255. Ieee, 2009. 4

[8] A. Foi, M. Trimeche, V. Katkovnik, and K. Egiazarian. Practical poissonian-gaussian noise modeling and fitting for single-image raw-data. *IEEE Transactions on Image Processing*, 17:1737–1754, 2008. 6

[9] A. Fregin, J. Müller, and K. Dietmayer. Feature detectors for traffic light recognition. pages 339–346, 2017. 2

[10] K. Garg and S. K. Nayar. Vision and rain. *International Journal of Computer Vision*, 75(1):3–27, 2007. 1

[11] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361. IEEE, 2012. 18

[12] C. Godard, O. M. Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6602–6611, 2017. 7

[13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014. 4

[14] T. Gruber, M. Bijelic, F. Heide, W. Ritter, and K. Dietmayer. Pixel-accurate depth evaluation in realistic driving scenarios. In *International Conference on 3D Vision (3DV)*, pages 95–105, Sept. 2019. 9

[15] T. Gruber, F. Julca-Aguilar, M. Bijelic, and F. Heide. Gated2depth: Real-time dense lidar from gated images. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1506–1516, 2019. 5

[16] S. S. Halder, J.-F. Lalonde, and R. de Charette. Physics-based rendering for improving robustness to rain. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019. 1, 2

[17] K. He, J. Sun, and X. Tang. Guided image filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(6):1397–1409, 2013. 1

[18] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. Efros, and T. Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International Conference on Machine Learning*, pages 1989–1998. PMLR, 2018. 7

[19] M. B. Jensen, M. P. Philipsen, A. Møgelmose, T. B. Moeslund, and M. M. Trivedi. Vision for looking at traffic lights: Issues, survey, and perspectives. *IEEE Transactions on Intelligent Transportation Systems*, 17(7):1800–1815, 2016. 2

[20] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 694–711. Springer, 2016. 4

[21] H. Koschmieder. *Theorie der horizontalen Sichtweite*. 1924. 1

[22] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander. Joint 3d proposal generation and object detection from view aggregation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8. IEEE, 2018. 16

[23] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 21–37. Springer, 2016. 16, 18

[24] K. Mei, A. Jiang, J. Li, and M. Wang. Progressive feature fusion network for realistic image dehazing. In *Asian Conference on Computer Vision (ACCV)*, 2018. 7

[25] R. Mondal, S. Santra, and B. Chanda. Image dehazing by joint estimation of transmittance and airlight using bi-directional consistency loss minimized fcn. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1033–10338, 2018. 7

[26] Y. Qu, Y. Chen, J. Huang, and Y. Xie. Enhanced pix2pix dehazing network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8152–8160, 2019. 7

[27] R. M. Rasmussen, J. Vivekanandan, J. Cole, B. Myers, and C. Masters. The estimation of snowfall rate using visibility. *Journal of Applied Meteorology*, 38(10):1542–1563, 1999. 1

[28] J. Redmon and A. Farhadi. Yolo9000: Better, faster, stronger. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6517–6525, 2017. 16

[29] C. Sakaridis, D. Dai, and L. Van Gool. Semantic foggy scene understanding with synthetic data. *International Journal of Computer Vision*, 126(9):973–992, 2018. 1, 2

[30] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015. 16

[31] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8798–8807, 2018. 2

[32] Z. Zheng, Y. Wu, X. Han, and J. Shi. Forkgan: Seeing into the rainy night. In *Proceedings of the European Conference on Computer Vision (ECCV)*, August 2020. 7

[33] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 7

[34] K. Zuiderveld. *Contrast Limited Adaptive Histogram Equalization*, page 474–485. Academic Press Professional, Inc., USA, 1994. 2