Supplementary Material of:
Robust Monocular Depth Estimation under Challenging Conditions

Stefano Gasperini[*,1,2]   Nils Morbitzer[*,1]   HyunJun Jung[1]
Nassir Navab[1]   Federico Tombari[1,3]

[1] Technical University of Munich   [2] VisualAIs   [3] Google

A. Supplementary Material

The source code of our method, the main trained models reported in the experiments, and the generated translated images are publicly available at https://md4all.github.io.

This appendix includes additional details and results. Sections A.1 and A.2 include additional information on the method and the experimental setup, while Sections A.3 and A.4 introduce more results, quantitatively and qualitatively, respectively.

In particular, this appendix is organized as follows:

- Section A.1.1 includes details about our supervised framework.

- Section A.1.2 adds details about our self-supervised baseline.

- Section A.1.3 describes the noise and time-dependent normalization used for our self-supervised models.

- Section A.2.1 further reports details about the experimental setup for image translation.

- Section A.2.2 includes details on the setup used for the nuScenes dataset.

- Section A.2.3 includes details on the setup used for the RobotCar dataset.

- Section A.2.4 adds details about the experimental setup for prior works.

- Section A.3.1 reports a detailed ablation study of our method on both nuScenes and RobotCar.

- Section A.3.2 adds preliminary results with snow and fog on the DENSE dataset.

- Section A.3.3 analyzes the effect of different data distributions among the conditions during training on RobotCar.

- Section A.3.4 compares different configurations of our supervised framework on nuScenes.

- Section A.3.5 looks into quantitative results with rain at nighttime and averages across the various conditions on nuScenes.

- Section A.3.6 compares methods on the test set of nuScenes.

- Section A.3.7 analyzes the performance of the methods at varying distances from the ego vehicle, both on nuScenes and RobotCar.

- Section A.4.1 reports qualitative results of our self-supervised method on nuScenes.

- Section A.4.2 adds qualitative results of our fully-supervised method on nuScenes.

- Section A.4.3 discusses qualitative results of our self-supervised method on RobotCar.

- Section A.4.4 looks into failure cases of our self- and fully-supervised methods, exemplified on nuScenes.

- Section A.4.5 analyzes images generated via image translations for both nuScenes and RobotCar.

- Section A.5 lists attempted and alternative approaches that did not work.

A.1. Additional Details on the Method

A.1.1   Supervised md4all

In the main paper, we mostly focused on the more complex self-supervised setting (Sections 3.1.2 and 3.1.3), and we extended our method to the supervised setup (Section 3.2),

---

* The authors contributed equally.
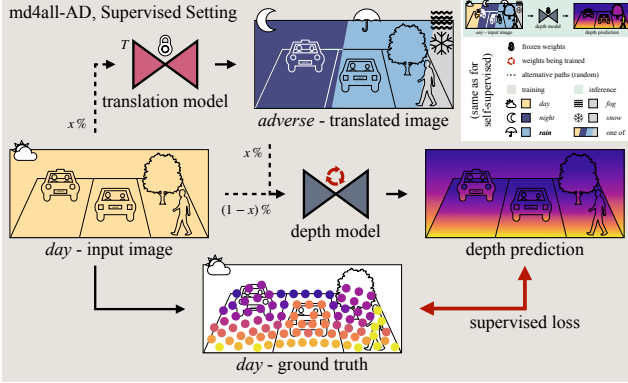Contact author: Stefano Gasperini (*stefano.gasperini@tum.de*).

Figure 7. Apart from the type of supervision, our md4all-AD supervised framework works similarly to the md4all-AD self-supervised framework described in Section 3.1.2 and Figure 4, with Always Daytime, No Bad Weather. The depth model is trained with a mix of easy and translated samples, while its supervision is obtained from the ground truth data corresponding only to the easy samples. As in the self-supervised case, inference time is unchanged and performed with a single depth model without specialized branches (top right).

making it the first depth estimation work to explore and address bad weather in supervised monocular settings.

Supervised models learn directly from the ground truth (e.g., LiDAR data). However, in adverse conditions (e.g., rain), the ground truth becomes unreliable (Figure 2). Our supervised md4all aims to eliminate the sources of unreliability in the ground truth by providing a reliable signal in all conditions. As shown in Figure 7, we achieve this with the same principles described for the self-supervised settings: having a single depth model learn robust features agnostic of the condition in input by feeding a mix of easy and hard samples, with the ground truth always corresponding to the easy samples. Therefore, we use the same image translation model to generate adverse images corresponding to the easy ones in the training data. Then, we train the depth model with a mix of original easy and generated adverse inputs. Unlike the self-supervised settings where the training signal came from a pre-trained baseline model (md4all-DD) or the photometric losses (md4all-AD), the training signal is obtained directly from the ground truth data for supervised methods. When translating an image e_i to adverse conditions h_i^c, we use as ground truth for h_i^c the LiDAR data corresponding to e_i.

We associate this supervised method with our AD configuration since no distillation from a pre-trained model occurs (unlike for DD). Moreover, the depth model is trained in the same way as its baseline, i.e., via the ground truth, in an Always Daytime, no Bad Weather fashion, similarly to our self-supervised AD model (Section 3.1.2). Furthermore, the translated images should be used only for those conditions that render the ground truth unreliable. There-

fore, for the experiments, we translated the inputs from *day-clear* to *day-rain*, since the ground truth is unreliable with rain (Figure 2). Still, we used the original *night* inputs since the ground truth is reliable at night (Figure 5).

A.1.2 Self-Supervised Baseline

In this section, we further describe the loss functions used for the baseline of Section 3.1.1. Such baseline is equivalent to Monodepth2 [7] made scale-aware through weak velocity supervision from Guizilini et al. [8].

The photometric loss is the combination of $\mathcal{L}_1$-loss and SSIM [14], as done in [6]:

$$\mathcal{L}_1(I_t, \hat{I}_t) = \left\| I_t - \hat{I}_t \right\|_1$$
$$\mathcal{L}_{\text{SSIM}} = 1 - \text{SSIM}\left(I_t, \hat{I}_t\right) \quad (1)$$
$$pe\left(I_t, \hat{I}_t\right) = (1 - \alpha)\,\mathcal{L}_1(I_t, \hat{I}_t) + \frac{\alpha}{2}\mathcal{L}_{\text{SSIM}}(I_t, \hat{I}_t)$$

where $\alpha = 0.85$ is a weight to balance between the two terms. Furthermore, similarly to [7], we account for partial occlusions by only considering the minimum reprojection error:

$$\mathcal{L}_p\left(I_t, \hat{I}_{s \to t}\right) = \min_s pe\left(I_t, \hat{I}_{s \to t}\right). \quad (2)$$

Moreover, following the so-called auto-mask from Monodepth2 [7], we automatically mask out the pixels that do not change appearance across different frames:

$$M_a = \min_s \mathcal{L}_p\left(I_t, I_s\right) > \min_s \mathcal{L}_p\left(I_t, \hat{I}_{s \to t}\right). \quad (3)$$

Therefore, the photometric loss is only computed in the areas where $M_a = 1$. Additionally, to encourage local smoothness and preserve sharp edges, we use the following term from [6]:

$$\mathcal{L}_s\left(I_t, d_t^*\right) = \frac{1}{N} \sum_{p \in N} \sum_{i \in x, y} |\partial_i d_t^*(p)|\, e^{-|\partial_i I_t|} \quad (4)$$

where $|\cdot|$ is the absolute value computed element-wise, ∂_x and ∂_y are the gradients in x and y directions, and $d_t^* = d_t^*/\overline{d_t^*}$ is the inverse of the depth prediction normalized by the mean.

As already described in Section 3.1.1, we follow [8] by using a weak velocity supervision $\mathcal{L}_v$ to achieve scale-awareness. This is defined as:

$$\mathcal{L}_v\left(\hat{\mathbf{t}}_{t \to s}, \mathbf{t}_{t \to s}\right) = \left| \|\hat{\mathbf{t}}_{t \to s}\|_2 - \|\mathbf{t}_{t \to s}\|_2 \right| \quad (5)$$

where $\hat{\mathbf{t}}_{t \to s}$ and $\mathbf{t}_{t \to s}$ are the predicted and ground truth pose translations, respectively, which can be easily obtained from the available odometry information, through the ego vehicle speed and the time interval across frames.

ID	Method	tr.data	day-clear – nuScenes			night – nuScenes			day-rain – nuScenes		
			absRel	RMSE	δ_1	absRel	RMSE	δ_1	absRel	RMSE	δ_1
A0	md2 [7], *all*	*a: dnr*	0.1477	6.771	85.25	2.3332	32.940	10.54	0.4114	9.442	60.58
A1	md2, *n* real	*dn*	0.1345	6.575	85.47	2.4536	34.295	11.71	0.1753	7.701	77.13
A2	md2, *n* transl.$_{15\%}$	*dT(n)*	0.1390	6.670	85.36	0.2655	9.892	54.44	0.1861	7.800	76.28
A3	md2, *day-c* only	*d*	0.1374	6.692	85.00	0.2828	9.729	51.83	0.1727	7.743	77.57
A4	+ v.-sup = b.line	*d*	0.1333	6.459	**85.88**	0.2419	10.922	58.17	0.1572	7.453	79.49
A5	+ noise, $\mathcal{L}$ clean	*d*	0.1428	6.609	84.43	0.2256	9.672	63.50	0.1592	7.619	78.95
A6	+ all *n* transl.	*dT(n)*	0.1624	7.042	80.50	0.2214	9.092	67.01	0.1752	8.272	76.41
A7	– pose transl.	*dT(n)*	0.1597	7.143	81.37	0.2184	8.754	66.90	0.1689	8.210	77.23
ADn	+ *day* loss only	*dT(n)*	0.1433	6.954	83.27	0.2230	9.001	68.63	0.1545	7.915	78.36
A9	– time norm.	*dT(n)*	0.1554	6.949	81.66	0.2121	8.502	67.43	0.1627	7.797	77.62
ADa	ADn + *r* transl.	*dT(nr)*	0.1523	6.853	83.11	0.2187	9.003	68.84	0.1601	7.832	78.97
A11	+ day distill.	*dT(nr)*	0.1387	6.621	84.11	0.1960	8.595	70.08	0.1444	7.355	80.20
DDn	ADn + distill.	*dT(n)*	**0.1302**	**6.373**	85.02	0.1959	**8.471**	70.14	0.1429	7.312	79.60
DDr	*r* distill.	*dT(r)*	0.1323	6.437	85.18	0.2502	11.847	57.02	**0.1364**	**7.100**	**81.37**
DDa	ADa + distill.	*dT(nr)*	0.1366	6.452	84.61	0.1921	8.507	**71.07**	0.1414	7.228	80.98
A15	– test time norm.	*dT(nr)*	0.1367	6.449	84.56	**0.1881**	8.524	70.65	0.1412	7.234	80.99

Table 4. Ablation study on our self-supervised method on the nuScenes [4] validation set. md2: Monodepth2 [7]. DD*a* is our self-supervised approach reported throughout this work for nuScenes: md4all-DD trained on *dT(nr)*. Instead, AD*a* is our self-supervised md4all-AD model. Notation reused from Table 1.

A.1.3 Noise and Normalization

We did not apply either of these two techniques in the supervised setting, i.e., where we applied our method on AdaBins [1], since the LiDAR ground truth provides a strong signal which already enables such robust feature extraction.

For the self-supervised models, we normalize the inputs at training time depending on the time of the day (i.e., day and night). Towards this end, we precompute the mean and variance of the pixel values across the two conditions throughout the dataset and normalize the inputs accordingly. In Tables 4 and 5, we show how this time-dependent normalization has a positive impact at training time, as it aligns the features in a condition-agnostic manner. Additionally, we show that this normalization can be avoided at inference time for similar results. Avoiding it ensures that the operations executed across all conditions are identical at inference time. Thus, after deployment, our method does not require any knowledge about the current weather and illumination settings, which may be hard to define and may intersect with other conditions (e.g., wet ground without rain or dusk). Nevertheless, due to the relatively small difference during inference, we used time-dependent normalization for self-supervised models unless otherwise noted.

Furthermore, in the case of camera sensors delivering significant noise levels (e.g., nuScenes [4]), we augment the inputs of self-supervised models with heavy noise. The noise is randomly applied to 50% of the inputs, regardless of their condition. This helps to learn more robust features. When the noise is used, we compute the losses on the sam-

ples without noise. Specifically, we generated the noise by adding to the image a random pattern following the uniform distribution [0.005, 0.05], then clamped the pixel values to [0, 1], thereby ensuring that the input remains within a valid range.

A.2. Additional Details on the Experimental Setup

A.2.1 Day-to-adverse Translation

For the experiments, we focused on two adverse conditions in nuScenes [4] (i.e., *rain* and *night*) and one in Robot-Car [10] (i.e., *night*), alongside the standard conditions *day-clear / day*. Towards this end, we trained two different ForkGAN [16] models for nuScenes, one for each condition, and one for RobotCar, to enable translations from *day-clear* to each challenging condition, tailored to each dataset. For the RobotCar dataset, we trained the GAN using the 34128 daytime samples from the scene 2014-12-09-13-21-02 and the 32585 nighttime samples from 2014-12-16-18-44-24. The dataset offered enough samples to train the image translation model thanks to the high frame rate. Instead, the nuScenes dataset only provides 6951 samples for *day-rain* and 4706 for *night*, which are insufficient for the GAN to learn such day-to-adverse translation. Therefore, to learn the transition from *day-clear* to *day-rain*, we additionally used all *day-rain* samples from the nuImages dataset [4] resulting in a total number of 19857 *day-rain* frames. We balanced this with the 19685 *day-clear* images of the nuScenes training set. Since the nuImages dataset does not provide any metadata about the weather condition,

Method	tr.data	day – RobotCar				night – RobotCar			
		absRel	sqRel	RMSE	δ_1	absRel	sqRel	RMSE	δ_1
Monodepth2 [7]	d	0.1196	0.670	**3.164**	86.38	0.3029	1.724	5.038	45.88
WSGD [12]	a: dn	0.1760	1.603	6.036	75.00	0.1740	1.637	6.302	75.40
[ours] baseline	d	0.1209	0.723	3.335	86.61	0.3909	3.547	8.227	22.51
[ours] md4all-AD	$dT(n)$	**0.1113**	0.707	3.248	**88.02**	0.1223	0.851	3.723	**85.77**
[ours] **md4all-DD**	$dT(n)$	0.1128	0.648	3.206	87.13	**0.1219**	0.784	3.604	84.86
[ours] md4all-DD w/o test time norm.	$dT(n)$	0.1129	**0.640**	3.190	87.02	0.1256	0.824	3.703	83.87
[ours] md4all-AD w/ LiDAR scaling	$dT(n)$	0.1192	0.747	3.184	86.81	0.1275	0.834	3.641	86.15
[ours] md4all-DD w/ LiDAR scaling	$dT(n)$	**0.1133**	**0.642**	**3.052**	**87.45**	**0.1230**	**0.739**	**3.439**	**86.41**

Table 5. Evaluation of self-supervised works on the RobotCar [10] test set up to 50 meters. Different configurations of our method are compared with Monodepth2 and WSGD. LiDAR scaling indicates the use of LiDAR data at test-time to scale the predictions (as done by Monodepth2 and WSGD), which is equivalent to * in the supervision notation of Table 1 and 2. The method highlighted in bold is reported throughout this work for RobotCar as md4all-DD trained on *dT(n)*. This Table extends Table 2.

we manually labeled all its samples with their respective weather condition. Nevertheless, *night* samples are insufficient in nuScenes and nuImages (14302) to train a GAN. For this reason, we first trained the *day-clear* to *night* translation model on BDD100K [15], which includes 36728 *day* and 27971 *night* images in its training set. Then, we fine-tuned it on the available nuScenes *night* samples from the training set.

A.2.2 nuScenes

For the depth experiments on nuScenes [4], we followed the setup of R4Dyn [5], using the official data splits and evaluating up to 80 meters comparing the predictions with a single LiDAR scan. As in R4Dyn, we discarded static frames (i.e., where the ego vehicle is stationary) for self-supervised models. While a single scan is highly sparse compared to the dense depth prediction, it limits the artifacts introduced by accumulating multiple scans over time for denser ground truth (e.g., due to moving objects and changing perspectives). We augmented the inputs with heavy noise for self-supervised models to mimic that in the night samples. For the supervised setting, learning from such a sparse signal means reducing the workload needed for producing the ground truth, albeit rendering it more challenging. As it is standard for supervised setups, the models do not learn the depth of the unreachable areas for the ground truth sensor (e.g., the sky for LiDAR). All qualitative images and quantitative results reported in the main paper and this supplementary material are from the validation set unless otherwise noted (e.g., test set in Table 14).

A.2.3 RobotCar

For the experiments on RobotCar [10], we followed the setup of WSGD [12] using the six sequences in the 2014-12-09-13-21-02 traversal as daytime samples, and the six

sequences in the 2014-12-16-18-44-24 traversal as night-time ones. Since the peculiarity of RobotCar is that it was recorded by driving over the same route multiple times over a year, a training-test split with non-overlapping drives is required to properly assess the models' generalization capabilities. Therefore, we used the split provided by WSGD.

As in [13], we used the left images of the front stereo-camera (Bumblebee XB3), of which we removed the bottom 20% (i.e., ego vehicle bonnet), and the ground truth data from the LMS front LiDAR sensor. We used the official toolbox to accumulate multiple LiDAR scans and project them to the input images. Towards this end, we used visual odometry, as recommended by the official documentation of the dataset, and a time margin of $\pm 4e6$ from the origin timestamp, as in [13]. As commonly done for self-supervised methods, we discard static frames thresholding the translation provided by the visual odometry. We did not apply heavy noise for RobotCar as the night samples did not exhibit it. Furthermore, since the RobotCar camera occasionally suffers from inconsistent illumination across neighboring frames, we discarded these too. Specifically, we removed all triplets where the keyframe's mean RGB value is ≥ 0.9, or the RGB mean value difference between two consecutive frames is > 0.05. In addition, only the images with a corresponding LiDAR ground truth could be evaluated.

For the experiment with degraded translations via random erasing (Table 3), we applied it randomly to 10% of the inputs, with a patch sized randomly between 5% and 10% of the input dimensions, placed randomly within the image, with an aspect ratio between 0.3 and 3.3. When applying random erasing, the performance of md4all marginally improved in terms of δ_1 by 0.32% on *all* (absRel slightly decreased as shown in Table 3), thanks to the augmentation and regularization effect introduced by the patches. Throughout the main paper and this supplementary mate-

Method	day-clear		fog		snow	
	absRel	δ_1	absRel	δ_1	absRel	δ_1
md2 [7]	0.1642	82.35	0.1698	81.97	0.1798	76.68
[ours]	**0.1520**	**83.54**	**0.1524**	**83.36**	**0.1788**	**77.93**

Table 6. <u>These are only preliminary results</u> (details in Section A.3.2). Evaluations with *snow* and *fog* on the DENSE dataset [2]. AbsRel and δ_1 are reported for each condition. Monodepth2 [7] (md2) trained on *day-clear* is compared with our md4all-DD trained on *day-clear* plus translated images to *fog* and *snow* ($x = 66\%$).

Method	avg/all		day		night	
	absRel	δ_1	absRel	δ_1	absRel	δ_1
md2 [7]	0.2122	65.92	0.1196	86.38	0.3029	45.88
70*d* - 30*n*	0.1189	**86.39**	0.1138	**87.80**	0.1239	**85.01**
50*d* - 50*n*	**0.1174**	85.99	**0.1128**	87.13	**0.1219**	84.86
30*d* - 70*n*	0.1221	85.86	0.1168	87.16	0.1273	84.59

Table 7. Impact of different training data distributions between *day* (*d*) and *night* (*n*) samples by varying the parameter *x*. AbsRel and δ_1 are reported on the test set of the RobotCar dataset [10]. Different distributions are shown for md4all-DD, *a*. In the rest of this work, the balanced 50*d*-50*n* configuration (i.e., $x = 50\%$) was used.

rial, all qualitative images and quantitative results are from the test set defined by WSGD.

A.2.4   Prior Works

For prior works on RobotCar, we reported the values computed by Vankadari et al. [12], who retrained RNW [13] on a non-overlapping split (which inherently reduced the scores), and also re-evaluated DeFeatNet [11] and ADIDS [9] on the same test split (again reducing the scores). Among works focusing on depth estimation in the dark, only RNW reported its results on the more challenging nuScenes dataset. However, since the authors reported their scores on a different, custom split, we retrained their model on the official split. For this reason, the results of RNW differ from those reported directly by Wang et al. in their paper [13]. Nevertheless, the difference is relatively small as RNW reported at *night*: absRel of 0.3150 (0.3333 from our experiment with RNW), sqRel of 3.793 (4.006), RMSE of 9.6408 (10.098), and δ_1 of 50.81 (43.72). While this performance gap should be attributed to the different data splits used, it does not affect the comparisons since our models performed significantly better than what Wang et al. reported in their paper [13], both quantitatively and qualitatively. Furthermore, on nuScenes, we also report the values of R4Dyn [5] and PackNet-SfM [8], as provided to us by the authors of [5]. Additional related works tackling adverse conditions exist (Section 2.2.1). Still, their lack of open-source code or their use of unconventional and unclear experimental setups prevented us from directly comparing with their methods.

A.3. Additional Quantitative Results

A.3.1   Ablation Study

In Table 4, we report an ablation study over the main components of our method.

We started from a Monodepth2 [7] trained on the entire training set of nuScenes [4] (A0), meaning all available conditions. A0 performed poorly under adverse conditions due to the difficulty of establishing pixel correspon-

dences across consecutive *night* and *rain* frames. A0 delivered scores and errors similar to those reported for Monodepth2 by prior works in their papers, such as RNW [13] and WSGD [12]. Furthermore, the outputs of A0 exhibit the same issues shown by RNW and WSGD in their qualitative comparisons (e.g., holes in the ground), which are not present and much improved when training Monodepth2 only on *day-clear*, as reported throughout this work (A3). In particular, with A0-A3, we show how the standard Monodepth2 performs substantially better than the complex RNW overall and significantly better than WSGD in the daytime (Table 5).

A1 is a Monodepth2 model trained on *day-clear* and *night* (i.e., everything excluding *day-rain*). A1 performed similarly to A0 at *night*, but significantly better with *rain*. Additionally, it can be seen how training on *day-rain* samples negatively affected the *day-clear* performance (A0) while excluding such rainy samples improved in standard conditions (A1). Then, A2 is another Monodepth2 model, trained on *day-clear* and the translated *night* samples we generated with the GAN. A2 was fed a mix of *day-clear* and generated *night* ones with $x = 15\%$, to resemble the day-night distribution of the training set (used by A1). The comparison of A1 with A2 shows mainly two aspects about the translated images (Figure 17): they are not entirely realistic, and, unlike the real ones, they do not prevent establishing the pixel correspondences. If the generated samples were completely realistic (i.e., like the real *night* ones from nuScenes), there would have been a much smaller difference between A1 and A2. In particular, the generated images do not fully resemble the real *night* ones (Figure 17), especially for the noise, which is more consistent throughout the generated frames compared to the real ones, and the darkness levels, with images that are not as black as the real *night* ones of nuScenes. This lack of realism in the generated images is the reason for the performance improvement of A2 at *night* compared to A1.

Similarly, WSGD [12] showed the importance of denoising *night* images, with noise detrimental to the models.

Method	tr.data	day-clear – nuScenes			night – nuScenes			day-rain – nuScenes		
		absRel	RMSE	δ_1	absRel	RMSE	δ_1	absRel	RMSE	δ_1
AdaBins [1]	*a*: *dnr*	0.1384	5.582	81.31	0.2296	7.344	63.95	0.1726	6.267	76.01
AdaBins [1]	*d*	0.1138	4.805	87.98	0.3336	14.002	45.77	0.1540	6.119	81.20
[ours] md4all-AD, *rain*	*dT(r)*	**0.1052**	**4.621**	**89.58**	0.2644	10.749	55.51	**0.1380**	6.030	**83.32**
[ours] **md4all-AD, *all***	*dnT(r)*	0.1206	4.806	88.03	**0.1821**	**6.372**	**75.33**	0.1562	**5.903**	82.82

Table 8. Additional evaluation of LiDAR-supervised methods on the nuScenes [4] validation set. This table adds the second and third lines compared to Table 1.

Method	avg/all – nuScenes				night-rain – nuScenes				d-clear	night	d-rain
	absRel	sqRel	RMSE	δ_1	absRel	sqRel	RMSE	δ_1	sqRel	sqRel	sqRel
md2 [7], *d*	0.1576	2.002	7.164	80.49	0.3148	3.001	9.523	46.72	1.820	2.879	2.296
R4Dyn [5], *d* (radar)	**0.1365**	1.830	6.957	**84.01**	0.2431	2.945	10.055	56.95	**1.661**	2.889	1.938
RNW [13], *dn*	0.2931	3.557	9.304	55.13	0.3400	4.783	10.189	44.68	3.433	4.066	3.796
baseline, *d*	0.1480	2.032	7.065	82.08	0.2684	3.368	10.664	53.54	1.738	2.776	2.273
md4all-AD, *dT(nr)*	0.1602	2.245	7.226	81.02	0.2470	3.442	9.153	65.17	2.141	2.991	2.259
md4all-DD, *dT(nr)*	0.1429	**1.828**	**6.782**	82.67	**0.2143**	**2.628**	**8.376**	**68.03**	1.752	**2.386**	**1.829**
AdaBins [1], *a*	0.1604	1.103	5.868	78.72	0.2343	1.704	7.088	61.62	0.980	1.773	1.249
md4all-AD, *dnT(r)*	**0.1328**	**0.952**	**5.139**	**85.92**	**0.1967**	**1.632**	**6.423**	**71.67**	**0.821**	**1.525**	**1.199**

Table 9. Evaluation of fully-supervised (based on AdaBins [1]) and self-supervised methods (based on md2: Monodepth2 [7]) on the nuScenes [4] validation set. The models are the same as in Table 1. This table complements Table 1 with an evaluation on *all* conditions combined, as well as the most challenging *night-rain*.

Since the translated images do not exhibit the same kind of noise and reflections as the real ones and are particularly unrealistic when translating a sunny sample (Figure 17), A2 was able to establish pixel correspondences across the translated samples to a certain extent. Additionally, as x randomizes the condition of each input independently, with A2, the translations also introduce a regularization effect as data augmentation. A3 was also a Monodepth2 model but trained only on the *day-clear* samples. As shown in Table 1, this improves significantly compared to training on all conditions (A0) due to the impossibility of establishing correspondences at *night* for A0.

The weak velocity supervision [8] (A4) improved significantly over A3, thanks to better pose estimates. Compared to A0-A3, which need ground truth median scaling at test time, A4 is scale-aware and does not use it. With A5, we added heavy noise (consistently throughout the triplets) but computed the losses on the clean samples (i.e., without noise). This made it worse for *day-clear* and *day-rain*, but improved for *night* compared to A4. The motivation for A5 develops from the intense noise present in the *night* samples of nuScenes (Figure 2), which may confuse the models. We did not apply this under supervised settings (i.e., our method on AdaBins). The improvement seen with adding noise while computing the loss on the inputs without noise paved the way for the concept of our AD model. With A6, we added the translated images generated with the GAN from day to night ($x = 50\%$) to the training data.

This made it worse than A0 for *day-clear*, but similarly to A2, it improved for *night* due to the lack of realism of the generated samples, which allowed to establish pixel correspondences. For A6 (and A2), perfectly realistic generated samples would have been detrimental to learning.

With A7, we did not feed the translated images to the pose model but only to the depth one. This guarantees reasonable pose estimates, which improve the task at hand under all three conditions. Then, with AD*n*, we computed the losses only on the *day-clear* e_i samples, corresponding to the translated ones given as input. This significantly improved the model performance on *day-clear*, reaching a level similar to A3 (i.e., only a marginal degradation on the standard conditions). It should be noted that if the translated images perfectly mimicked the real *night* ones, A2, A6, and A7 would have performed relatively poorly, i.e., similarly to A0 and A1 at *night*. In the case of perfect day-to-night translations, always computing the loss only on the *day-clear* e_i samples (as in AD*n*, instead of calculating it on the translated ones, as in A2, A6, and A7) would have had a significantly positive impact at *night*.

With A9, we removed the time-dependent normalization from AD*n*, which was used from A6 to AD*n*. This shows that this technique benefits both *day-clear* and *day-rain*, as it helps construct a unified representation for all conditions. With AD*a*, we incorporated day-to-rain translations to AD*n* alongside the day-to-night images ($x = 66\%$, i.e., one-third for each condition). This delivered a similar performance

Method	40m – *day-clear* – nuScenes				40m – *night* – nuScenes				40m – *day-rain* – nuScenes			
	absRel	sqRel	RMSE	δ_1	absRel	sqRel	RMSE	δ_1	absRel	sqRel	RMSE	δ_1
md2	**0.1095**	**0.796**	**3.535**	88.89	0.2401	1.640	5.842	60.40	0.1405	1.083	4.259	82.64
b.line	0.1131	0.932	3.624	**89.46**	0.2118	1.816	6.476	63.47	0.1333	1.200	4.397	83.66
AD	0.1306	1.074	3.866	86.95	0.1907	1.670	5.414	73.24	0.1329	1.083	4.332	83.54
DD	0.1173	0.877	3.592	88.22	**0.1672**	**1.322**	**5.025**	**75.50**	**0.1190**	**0.927**	**4.036**	**85.37**

Table 10. Evaluation up to 40 meters of self-supervised approaches on the validation set of nuScenes [4]. md2: Monodepth2 [7] trained on *d*. b.line: baseline trained on *d*. AD: md4all-AD trained on *dT(nr)*. DD: md4all-DD trained on *dT(nr)*. The models are the same as in Table 1.

Method	40m – *avg/all* – nuScenes				40m – *night-rain* – nuScenes			
	absRel	sqRel	RMSE	δ_1	absRel	sqRel	RMSE	δ_1
md2	0.1276	**0.926**	3.882	85.04	0.2768	2.020	6.604	53.26
b.line	0.1262	1.063	4.034	85.93	0.2466	2.301	7.440	57.13
AD	0.1369	1.135	4.096	85.03	0.2184	2.064	6.066	68.96
DD	**0.1225**	0.930	**3.807**	**86.49**	**0.1925**	**1.653**	**5.661**	**71.38**

Table 11. Evaluation up to 40 meters of self-supervised approaches on the validation set of nuScenes [4]. md2: Monodepth2 [7] trained on *d*. b.line: baseline trained on *d*. AD: md4all-AD trained on *dT(nr)*. DD: md4all-DD trained on *dT(nr)*. The models are the same as in Table 1. This table complements Table 10.

to AD*n* (e.g., improved for *night*, and improved the RMSE, with a worse absRel for *day-clear*). As for Table 1, the Li-DAR ground truth is not fully reliable for *day-rain* and also significantly sparser than for *day-clear* (Figure 2). With A11, we added the day distillation loss (Equation 1) on the translated inputs while keeping the standard losses for the *day-clear* inputs. This combination improved across the board. Then, with DD, we simplified the training process by using only the day distillation loss for all inputs (including *day-clear*). Thus, DD*n* does this for *day-clear* and *night*, DD*r* does it for *day-clear* and *day-rain*. Our day distillation provides a dense and reliable signal (from A4 inferring only on *day-clear* samples), improving the errors and metrics across the board.

Finally, with A15, we show the impact of avoiding the time-dependent normalization at test time. Compared to DD*a*, A15 does not apply such time-dependent normalization at inference time but only at training time. A15 obtains comparable results throughout the various settings. Instead, as shown with A9, the time-dependent normalization is helpful at training time. After training, our model has learned robust features agnostic to the condition, allowing it to perform similarly regardless of the image normalization applied at test time. This demonstrates how our method does not need any condition-specific setups at inference time to deliver robust predictions, clearly separating our md4all from previous works requiring custom branches for each condition. In the supervised settings (i.e., our method applied on AdaBins), we did not perform any time-dependent normalization since the strong LiDAR supervision is enough to learn depth estimation at *night*.

In Table 5, we report various configurations of our method on the RobotCar [10] dataset. As for nuScenes (A15 in Table 4), we show that not applying the time-dependent normalization at test-time (i.e., executing the same operations with the same setup across the different conditions) does not negatively affect the predictions, achieving comparable results. Furthermore, we show how the results change when applying the median scaling via LiDAR data at test time. This technique is used by Monodepth2 [7], WSGD [12], and most other methods compared in this work. Our model does not need such scaling via ground truth data, thanks to its scale awareness learned via the weak velocity supervision introduced by PackNet-SfM [8].

Nevertheless, accurate scaling can further improve the results, especially at *night*. Compared to nuScenes, the RobotCar dataset provides less precise odometry information, causing difficulties for the baseline and our models to learn the correct scaling. This can be seen by the improved scores at *night* when applying the median scaling via Li-DAR data. With reliable scaling via the ground truth data, md4all-DD outperforms md4all-AD.

AD and DD While the benefit of our DD configuration over AD is evident for nuScenes, the gap is not as significant for RobotCar, with the two delivering comparable results (Table 5). This difference can be attributed to various reasons. First of all, nuScenes is more challenging, as demonstrated by the lower scores obtained by the models across all conditions, especially at *night*. Thus, the improvements of DD over AD might be reduced for RobotCar since AD already achieves solid results. The higher amount of images

Method	60m – *day-clear* – nuScenes				60m – *night* – nuScenes				60m – *day-rain* – nuScenes			
	absRel	sqRel	RMSE	δ_1	absRel	sqRel	RMSE	δ_1	absRel	sqRel	RMSE	δ_1
md2	0.1283	**1.387**	5.447	86.15	0.2739	2.469	8.444	53.40	0.1623	1.779	6.312	79.23
b.line	**0.1279**	1.522	5.422	**86.91**	0.2348	2.779	9.502	59.28	0.1506	1.833	6.267	80.73
AD	0.1461	1.745	5.744	84.23	0.2113	2.526	7.789	69.92	0.1519	1.774	6.421	80.28
DD	0.1310	1.419	**5.364**	85.65	**0.1859**	**2.029**	**7.377**	**72.10**	**0.1347**	**1.463**	**5.938**	**82.24**

Table 12. Evaluation up to 60 meters of self-supervised approaches on the validation set of nuScenes [4]. md2: Monodepth2 [7] trained on *d*. b.line: baseline trained on *d*. AD: md4all-AD trained on *dT(nr)*. DD: md4all-DD trained on *dT(nr)*. The models are the same as in Table 1.

Method	60m – *avg/all* – nuScenes				60m – *night-rain* – nuScenes			
	absRel	sqRel	RMSE	δ_1	absRel	sqRel	RMSE	δ_1
md2	0.1483	1.558	5.886	81.76	0.3086	2.782	8.790	47.76
b.line	0.1423	1.698	5.966	83.15	0.2648	3.130	9.858	54.09
AD	0.1536	1.827	6.057	82.16	0.2402	2.983	8.233	65.79
DD	**0.1371**	**1.487**	**5.658**	**83.74**	**0.2101**	**2.368**	**7.661**	**68.58**

Table 13. Evaluation up to 60 meters of self-supervised approaches on the validation set of nuScenes [4]. md2: Monodepth2 [7] trained on *d*. b.line: baseline trained on *d*. AD: md4all-AD trained on *dT(nr)*. DD: md4all-DD trained on *dT(nr)*. The models are the same as in Table 1. This table complements Table 12.

available on RobotCar to learn the translation task led to more realistic translations than nuScenes (Section A.4.5). Then, the less precise odometry information of RobotCar impacted the performance of the baseline through weak velocity supervision. Therefore, the baseline possibly learned wrong poses. This is not the case on nuScenes (Table 4), where the baseline (A4) improved significantly over Monodepth2 (A3). This did not happen for RobotCar. Since our md4all-DD learns to mimic the baseline via knowledge distillation, our model is directly affected by the weaker baseline in RobotCar, delivering similar results to AD. Instead, in nuScenes the gap between AD and DD is substantial throughout the conditions.

As shown with Monodepth2 [7] and AdaBins [1], our method is widely flexible and applicable to different architectures and types of supervision. While being out of the scope of this work, our approach can be seamlessly applied to other self-supervised or supervised frameworks, such as PackNet-SfM [8], since we do not alter the model architecture, but only its training scheme. In particular, to apply the proposed md4all to an existing depth estimation method, no structural changes are needed, as it is sufficient to feed to the model the translated images $x\%$ of the time during training.

A.3.2 DENSE Dataset: *Snow* and *Fog*

Disclaimer: First, please consider that these are only preliminary experiments and that we have not yet explored these conditions and models to the same extent as *night* and *rain* in the rest of this work. Nevertheless, we report them here as they provide interesting insights.

In Table 6, we show a first attempt to tackle the problem of monocular depth estimation in the presence of snow or fog with the DENSE dataset [2]. While our md4all performed better than the standard Monodepth2 [7] across the board, the improvement is relatively small compared to the other datasets and conditions explored (e.g., Tables 1 and 2). There are multiple reasons for this, explained below.

An impactful aspect to be considered is related to the available data. The condition boundaries are somewhat blurry. Overcast *day-clear* samples can be similar to light *fog* or light *snow*. This is problematic for the GAN used for image translation, which cannot distinguish the distributions and adequately learn the translation task.

Furthermore, the term snow is generic and includes various scenarios, such as light snow, heavy snow, blizzard, partly covered ground, fully covered ground, piles of snow, or wet ground with light snow falling. These settings differ substantially, but all belong to the same *snow* condition. This high variability is problematic for the translation task. While this issue can occur similarly with *night* and *rain* too, it is not as severe, and the diversity is more limited.

Another significant issue is the amount of usable image data for these conditions, which is insufficient to properly learn the translation task with ForkGAN [16]. As we did for *night* for nuScenes (Section 4.1), also for DENSE, we had to supplement with extra snow images taken from another dataset: Boreas [3]. We trained the snow ForkGAN with 17591 *day-clear* and 8443 *snow* samples from DENSE, plus 25036 *day-clear* and 26437 *snow* samples from Boreas for the pre-training. While supplementing with data from Boreas helped, the number of images from DENSE was relatively low compared to nuScenes and RobotCar, prevent-

Method	test – 40m – nuScenes				test – 60m – nuScenes				test – 80m – nuScenes			
	absRel	sqRel	RMSE	δ_1	absRel	sqRel	RMSE	δ_1	absRel	sqRel	RMSE	δ_1
md2	0.1162	0.811	3.701	87.59	0.1376	1.364	5.650	84.08	0.1465	1.755	6.941	82.69
RNW	0.2500	2.237	6.114	62.63	0.2781	3.420	9.222	57.50	0.2900	4.169	11.289	55.56
b.line	0.1126	0.840	3.827	87.17	0.1275	1.364	5.747	84.33	0.1332	1.679	6.938	83.22
AD	0.1214	0.881	3.851	86.99	0.1353	1.387	5.731	84.14	0.1409	1.691	6.915	83.00
DD	**0.1090**	**0.757**	**3.606**	**88.29**	**0.1221**	**1.204**	**5.418**	**85.57**	**0.1277**	**1.503**	**6.607**	**84.46**
AdaBins	0.1434	**0.617**	3.233	83.09	0.1494	0.852	4.689	80.67	0.1532	1.055	5.849	79.62
AD sup.	**0.1182**	0.641	3.279	**89.40**	**0.1221**	0.785	4.293	**87.98**	**0.1240**	0.887	5.021	**87.33**

Table 14. Evaluation up to 40, up to 60, and up to 80 meters of self-supervised approaches on the test set of nuScenes [4]. All conditions are evaluated here (*avg/all*). AD sup.: our md4all-AD trained on *dnT(r)* applied on AdaBins [1]. md2: Monodepth2 [7] trained on *d*. b.line: baseline trained on *d*. AD: md4all-AD trained on *dT(nr)*. DD: md4all-DD trained on *dT(nr)*. Time-dependent normalization was not applied to obtain these results as nuScenes provides no condition annotations for the test set. The models are the same as in Table 1.

ing effective translations.

Training data for the depth models was 7947 *day-clear* keyframes for DENSE. These keyframes were relatively few (15129 were used for nuScenes and 17790 for RobotCar), and they were extracted from short sequences, so they did not exhibit high variability. This reduced the depth estimation performance of the models. The validation set was also small with only 289 for *day-clear*, 1281 for *snow*, and 543 for *fog*. DENSE contains more images, but those were not usable due to various reasons, e.g., they were captured by different sensors.

Furthermore, as with *rain* (Figure 2), the LiDAR is not reliable in the presence of *snow* or *fog*, as it often captures snowflakes, fog particles, or is even obstructed by the snow accumulated on the sensor itself. We mitigated this problem by filtering the erroneous LiDAR points via clustering, but we could not eliminate all problematic measurements. As seen for *rain* on nuScenes, in adverse conditions the LiDAR sensor is unable to collect measurements at further distances (e.g., Figure 5 *rain* vs. *night* ground truth depth). Therefore, we could only evaluate a limited set of points at a closer distance. We used a single LiDAR scan as ground truth. Additionally, among the snow data, many samples were recorded in remote areas with relatively flat surroundings. Considering the limited distance and the flat surroundings, a model overfitting on flat ground may seem erroneously adequate by obtaining good quantitative results.

Additionally, for the weak velocity supervision of our baseline, we exploited the information from the CAN bus, as provided by the authors of DENSE. We used the vehicle speed and the frame rate to compute the camera translation between the frames. However, since the vehicle speed is provided as single value for each short sequence, the camera poses could only be coarsely approximated. This likely affected the performance of the baseline, hence that of our md4all-DD too. Furthermore, we used the CAN data to discard static inputs (i.e., stationary ego vehicle) and those where the ego vehicle is turning. We filtered the latter when

the steering wheel angle exceeded 20°. This filtering led to the numbers indicated above.

All these points should be considered when evaluating these preliminary results on DENSE.

First of all, regarding Table 6, it can be seen how the *day-clear* results are not as good as those seen for nuScenes (Table 1) or RobotCar (Table 2). This could be attributed to DENSE containing more challenging data. More likely, it is due to the inability of the models to properly generalize on DENSE due to the relatively low diversity in the training data and the limited amount of training samples. Therefore, our md4all-DD learned from a weak baseline which could not correctly estimate depth in standard conditions. Nevertheless, our md4all-DD outperformed Monodepth2 in standard settings, thanks to the regularization effect of our translations.

In the table, we report light-fog for *fog* and full-coverage or currently snowing for *snow*. We opted for light-fog since dense-fog exhibited too few LiDAR points for the evaluation, all at relatively close distances (easier). Instead, light-fog allowed for a more thorough assessment at further distances. For reference, all results were better with dense-fog than light-fog. For *snow*, we selected those with full-coverage or weather metadata snow. This is because, among the annotated conditions, they had the most precise boundaries with other conditions.

Moreover, both models perform similarly with *fog* as in ideal settings (i.e., *day-clear*). While this hints that fog is not as challenging as rain or night (Tables 1 and 2), the values are also affected by the limited distance of the ground truth used for the evaluation. Therefore, the performance may degrade significantly at further distances due to the fog preventing seeing the details, but that cannot be evaluated. Nevertheless, already at the available ground truth distances, our model outperformed Monodepth2, on which ours is based.

Despite the limited distance of the ground truth, *snow* appears more challenging than *fog*, causing a significant

drop in performance compared to the ideal settings (i.e., *day-clear*). With snow, the limited data available to learn proper translations substantially impacted our method's performance, which obtained only slightly better scores than Monodepth2.

Stronger condition boundaries (e.g., more precise annotations) and more training data would significantly improve the translations and our method's outcomes. Furthermore, depth ground truth reaching further distances without any artifacts would allow us to assess the actual performance of the models. While these factors would contribute to a more considerable gap between the proposed md4all and Monodetph2, the issues with the translations also highlight the limitations of our approach: the difficulty in collecting adverse data that would lead to solid results (e.g., Table 2 with RobotCar [10]).

Due to the substantial limitations encountered with this data, the DENSE dataset is unsuitable for depth estimation. However, we used it to provide these preliminary results with snow and fog. New real data with artifact-free long-distance ground truth is needed to properly explore monocular depth estimation in these conditions.

A.3.3 Different Distributions of Conditions

In Table 7, we explore the effect of different data distributions among the conditions during training. We vary this via the parameter x. In the rest of this work, x was selected to equally distribute the inputs among the conditions. So for RobotCar 50% for half for *day* and half for *night* (i.e., 50d - 50n in the table); for nuScenes 66% corresponding to one third for each of *day-clear*, *night*, and *rain*; one third each also for the DENSE dataset.

While intuitively increasing the amount of *day* images could improve the performance on *day*, this is not the case by randomizing via x at each training sample independently. This is because with enough epochs, our model sees all images in all conditions, so the training data remains unchanged, causing only minor differences as the model might be fed more or fewer translations (Table 7). For *day*, beyond the observed regularization effect (e.g., Table 2), there is little room for gains as long as the baseline model B to distill from remains the same. Instead, if x affected which portion of the training data is translated, it would have a more significant impact than shown in Table 7. In that case, seeing too many or too few translated images may impair the performance as the model does not experience enough of the ideal settings or not enough adverse conditions to tackle them properly.

A.3.4 Supervised Configuration Comparisons

Table 8 reports a comparison of different supervised configurations of AdaBins [1] and our md4all-AD applied on

AdaBins. Specifically, AdaBins trained only on *day-clear* resulted in a significant improvement on *day-clear* and *day-rain* compared to the AdaBins trained in all conditions (i.e., *a*). Analogously, our model trained on *day-clear* and translated *day-rain* samples performed better than ours trained on all but substantially worse at *night*. As seen in the self-supervised case, our model trained in all conditions outperformed the baseline across the board (i.e., AdaBins trained on all), thereby not introducing any trade-off while improving in adverse conditions over the model it is based on.

A.3.5 nuScenes *Night-rain* and Average

In Table 9, we report results on more conditions of nuScenes [4], such as the most difficult *night-rain* and an average over *all*, alongside the sqRel errors not fitting in Table 1 (due to the limited space available). *All* is not computed as an average on the various conditions but rather as an average of the performance on each sample (i.e., *night* counts marginally, accounting for only 10% of the images). Our model outperforms the baseline AdaBins across the board for the supervised case. Similarly, in the self-supervised setting, our md4all improved significantly over the baseline and Monodepth2 [7], second only in ideal conditions (*day-clear*) to the radar-based R4Dyn [5].

A.3.6 nuScenes Test Set

Table 14 reports errors and metrics on the official test set of nuScenes [4]. Since no metadata about the weather and illumination conditions is available for the test set, the values are to be considered for *all* conditions combined. As for the validation set, results were computed with the available ground truth data (i.e., LiDAR). The test set of nuScenes is more accessible than its validation set, with the models achieving better performance in the former across all three distance ranges. For these reasons, we focused on the validation set, and we report the test set results here for completeness. Nevertheless, our supervised and self-supervised models performed better than the models they are based on (i.e., AdaBins and Monodepth2, respectively) across the various conditions and depths.

A.3.7 Evaluation over Different Distances

For completeness, we also include the results for self-supervised models computed up to different distance ranges, namely up to 40, 60, and 80 meters. This complements the values reported in the main paper (up to 80 meters for nuScenes, and up to 50 for RobotCar) and should ease comparisons with future works. These results are reported across the following tables:

- Tables 10, 11, and 14 for nuScenes up to 40 meters.

Method	tr.data	40m – *day* – RobotCar				40m – *night* – RobotCar			
		absRel	sqRel	RMSE	δ_1	absRel	sqRel	RMSE	δ_1
Monodepth2 [7]	*d*	0.1181	**0.614**	**3.034**	86.51	0.3022	1.702	4.984	45.97
[ours] baseline	*d*	0.1198	0.678	3.229	86.69	0.3908	3.541	8.206	22.52
[ours] md4all-AD	*dT(n)*	**0.1099**	0.650	3.130	**88.10**	0.1203	0.762	3.531	**85.87**
[ours] **md4all-DD**	*dT(n)*	0.1120	0.618	3.125	87.18	0.1206	**0.723**	**3.479**	84.92

Table 15. Evaluation of self-supervised works on the RobotCar [10] test set up to 40 meters. The models are the same as in Table 2.

Method	tr.data	60m – *day* – RobotCar				60m – *night* – RobotCar			
		absRel	sqRel	RMSE	δ_1	absRel	sqRel	RMSE	δ_1
Monodepth2 [7]	*d*	0.1201	0.698	**3.215**	86.38	0.3029	1.728	5.045	45.88
[ours] baseline	*d*	0.1213	0.746	3.382	86.61	0.3909	3.548	8.228	22.51
[ours] md4all-AD	*dT(n)*	**0.1116**	0.731	3.291	**88.02**	0.1231	0.903	3.812	**85.76**
[ours] **md4all-DD**	*dT(n)*	0.1130	**0.661**	3.234	87.13	**0.1225**	**0.824**	**3.664**	84.86

Table 16. Evaluation of self-supervised works on the RobotCar [10] test set up to 60 meters. The models are the same as in Table 2.

Method	tr.data	80m – *day* – RobotCar				80m – *night* – RobotCar			
		absRel	sqRel	RMSE	δ_1	absRel	sqRel	RMSE	δ_1
Monodepth2 [7]	*d*	0.1203	0.718	3.245	86.38	0.3030	1.729	5.046	45.88
[ours] baseline	*d*	0.1214	0.759	3.404	86.61	0.3909	3.548	8.228	22.51
[ours] md4all-AD	*dT(n)*	**0.1118**	0.742	3.308	**88.02**	0.1236	0.952	3.880	**85.76**
[ours] **md4all-DD**	*dT(n)*	0.1131	**0.666**	**3.243**	87.13	**0.1229**	**0.865**	**3.713**	84.86

Table 17. Evaluation of self-supervised works on the RobotCar [10] test set up to 80 meters. The models are the same as in Table 2.

- Tables 12, 13, and 14 for nuScenes up to 60 meters.

- Tables 1, 8, 9, and 14 for nuScenes up to 80 meters.

- Table 15 for RobotCar up to 40 meters.

- Table 2 for RobotCar up to 50 meters.

- Table 16 for RobotCar up to 60 meters.

- Table 17 for RobotCar up to 80 meters.

The scores improve at lower distances as the problem becomes more effortless. Changes are limited due to the sparsity of the LiDAR at further distances, especially for Robot-Car. While evaluating using more LiDAR frames would have allowed us to evaluate at greater distances, we avoided it as it would have introduced artifacts in the ground truth for dynamic objects. Nevertheless, our models consistently improved across the various distances and conditions tested. As described in Section 4.1, the results we reported are to be considered up to 80 meters for nuScenes, and up to 50 meters for RobotCar, unless otherwise noted.

A.4. Additional Qualitative Results

For all qualitative results reported in this work (also Figures 1, 5, and 6 in the main paper), the predictions of our self-supervised model were all performed by our md4all-DD trained on all conditions (i.e., *dT(nr)* for nuScenes and *dT(n)* for RobotCar) and based on Monodepth2, while the supervised ones were all performed by our md4all-AD trained on *dnT(r)* and based on AdaBins. All qualitative images of the standard Monodepth2 were produced by the model trained only on *day-clear* (*day* for RobotCar). In contrast, the ones for the standard AdaBins were made by training it on all conditions, meaning that the best scoring models of each type produced them.

The following sections introduce new qualitative results on the various settings and conditions, such as nuScenes with self-supervised models (Section A.4.1), nuScenes with supervised methods (Section A.4.2), RobotCar with self-supervised approaches (Section A.4.3), failure cases (Section A.4.4), and translated images (Section A.4.5).

A.4.1 nuScenes – Self-Supervised

***Night* – nuScenes** In Figure 8, we compare our md4all-DD to Monodepth2 [7] on particularly challenging *night* samples due to the extreme darkness levels, as well as the high amounts of reflections (also due to the wet ground, in *night-rain* conditions). Instead, in Figure 9, we compare the same models on brighter, i.e., easier samples. It can be seen that the standard Monodepth2 trained on *day-clear* had marked

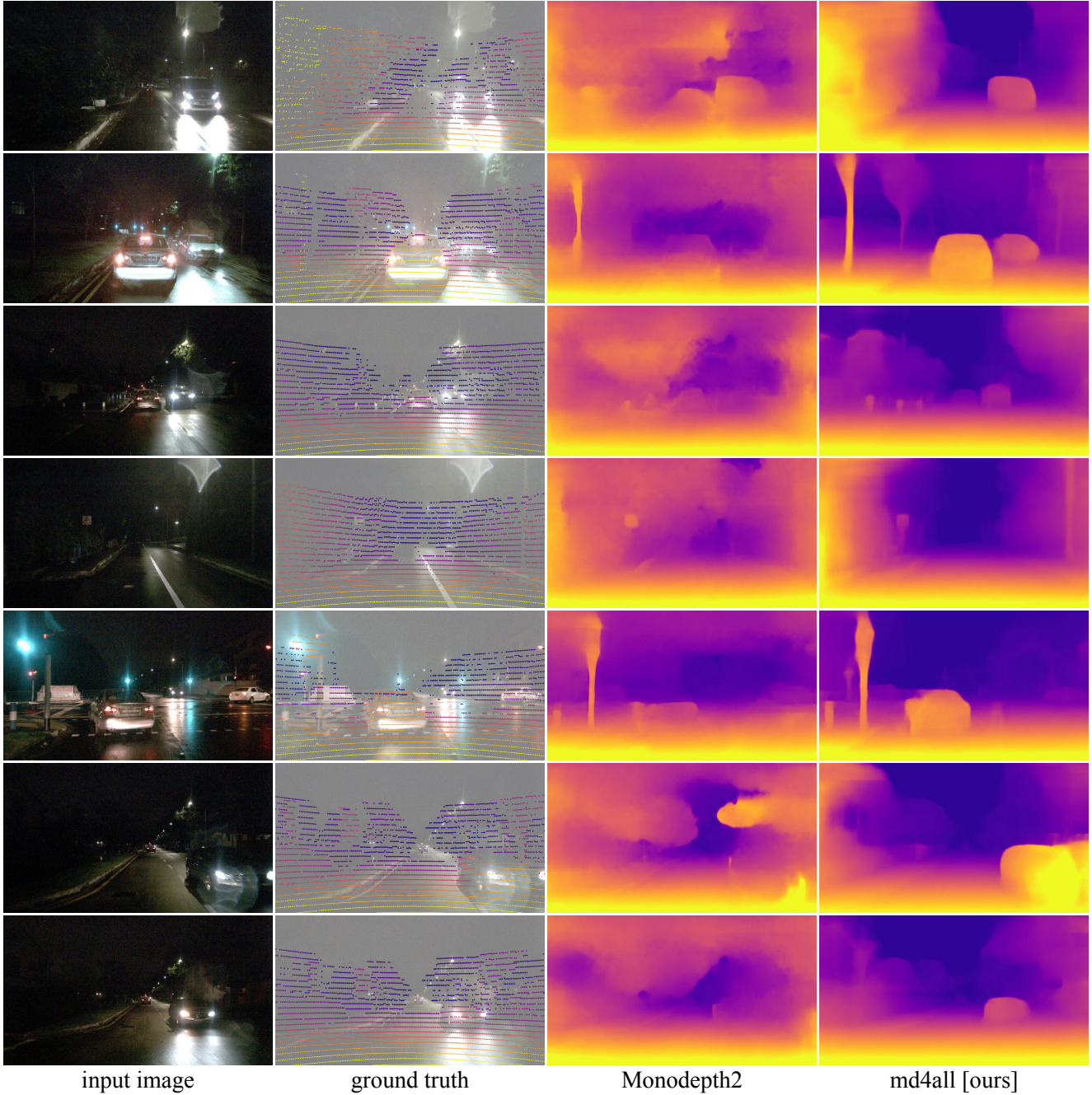| input image | ground truth | Monodepth2 | md4all [ours] |

Figure 8. Comparison of self-supervised models on nuScenes [4] *night* samples. The standard Monodepth2 [7] is compared to our md4all-DD applied to Monodepth2. This set of samples is particularly challenging for the standard Monodepth2 due to the overall darkness and reflections.

difficulties with darker scenes (Figure 8), while it delivered satisfactory results in brighter settings (Figure 9). This significant difference can be attributed to the fact that Monodepth2 was trained only on *day-clear* (the model trained on *all* conditions performed significantly worse, especially at *night*, as shown in Table 1) and that the details are easier to grasp in brighter *night* inputs compared to very dark

ones. In brighter images, the road, vehicles, curb, grass, and trees on the sides can be seen relatively well thanks to the light emitted by the streetlights, allowing the standard Monodepth2 to estimate reasonable depth maps. Conversely, with darker scenes, the depth cues are highly challenging to extract due to the noise and general blackness. Nevertheless, Monodepth2 tended to estimate the ground well also
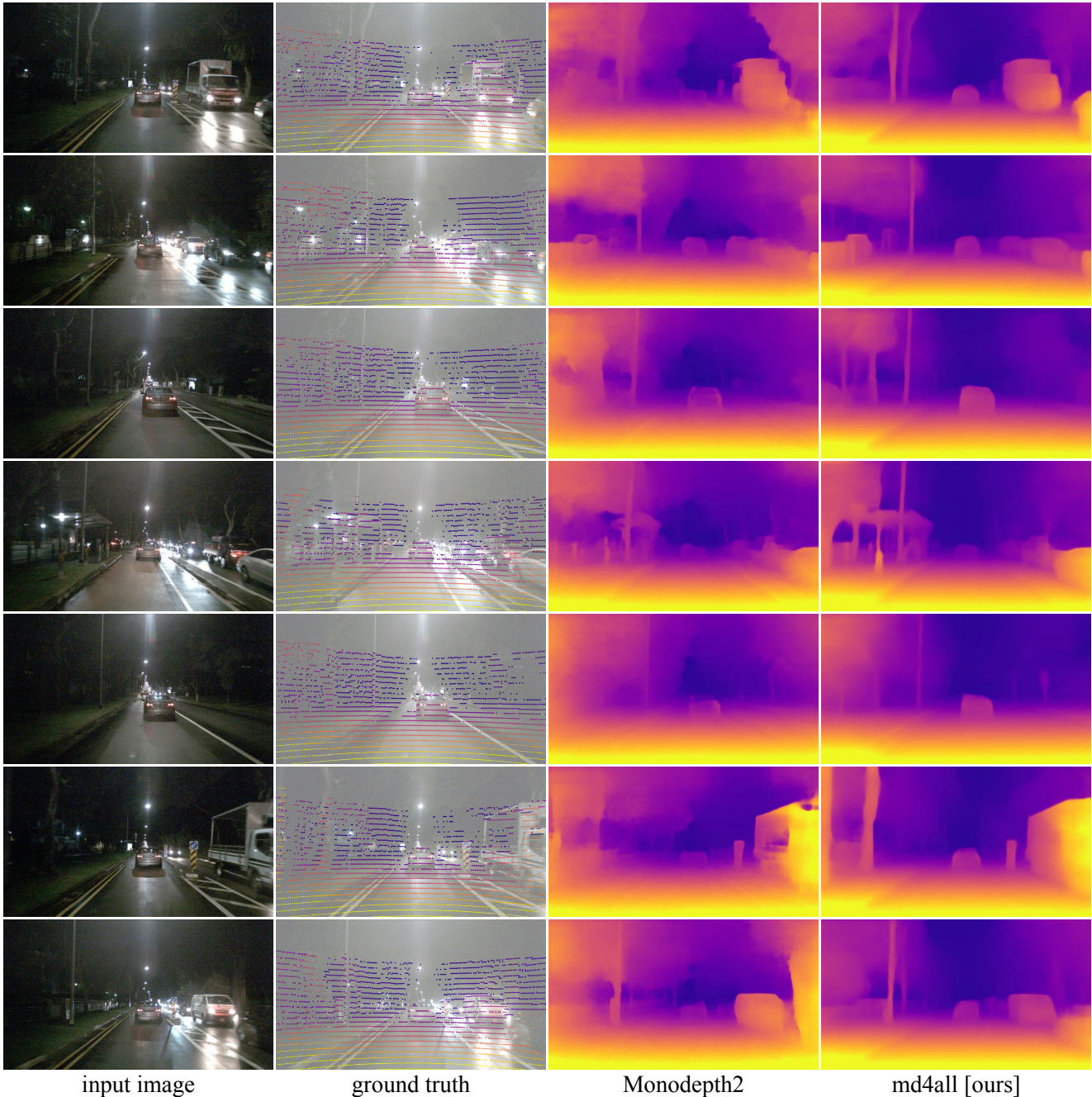
Figure 9. Comparison of self-supervised models on relatively bright nuScenes [4] *night* samples. The standard Monodepth2 [7] is compared to our md4all-DD applied to Monodepth2. This set of samples could be handled reasonably by the standard Monodepth2, thanks to the high brightness of the scenes.

in darker settings. This can be due to the model being biased toward scenes with flat and regular roads in front of the ego vehicle and the road being the most visible part of the images. Instead, Monodepth2 had severe issues with the sky, which is not evaluated since the ground truth is unavailable. Nevertheless, the proposed md4all delivered good and sharp estimates in dark and bright environments.

***Rain* – nuScenes** In Figure 10, we compare our md4all-DD to Monodepth2 [7] on *rainy* samples of nuScenes. While the standard Monodepth2 trained on *day-clear* delivered reasonable estimates for the most part, it was severely affected by the reflections on the wet ground (e.g., first row). Also, the blurriness due to water drops affected Monodepth2, triggering overly smooth estimates. The figure
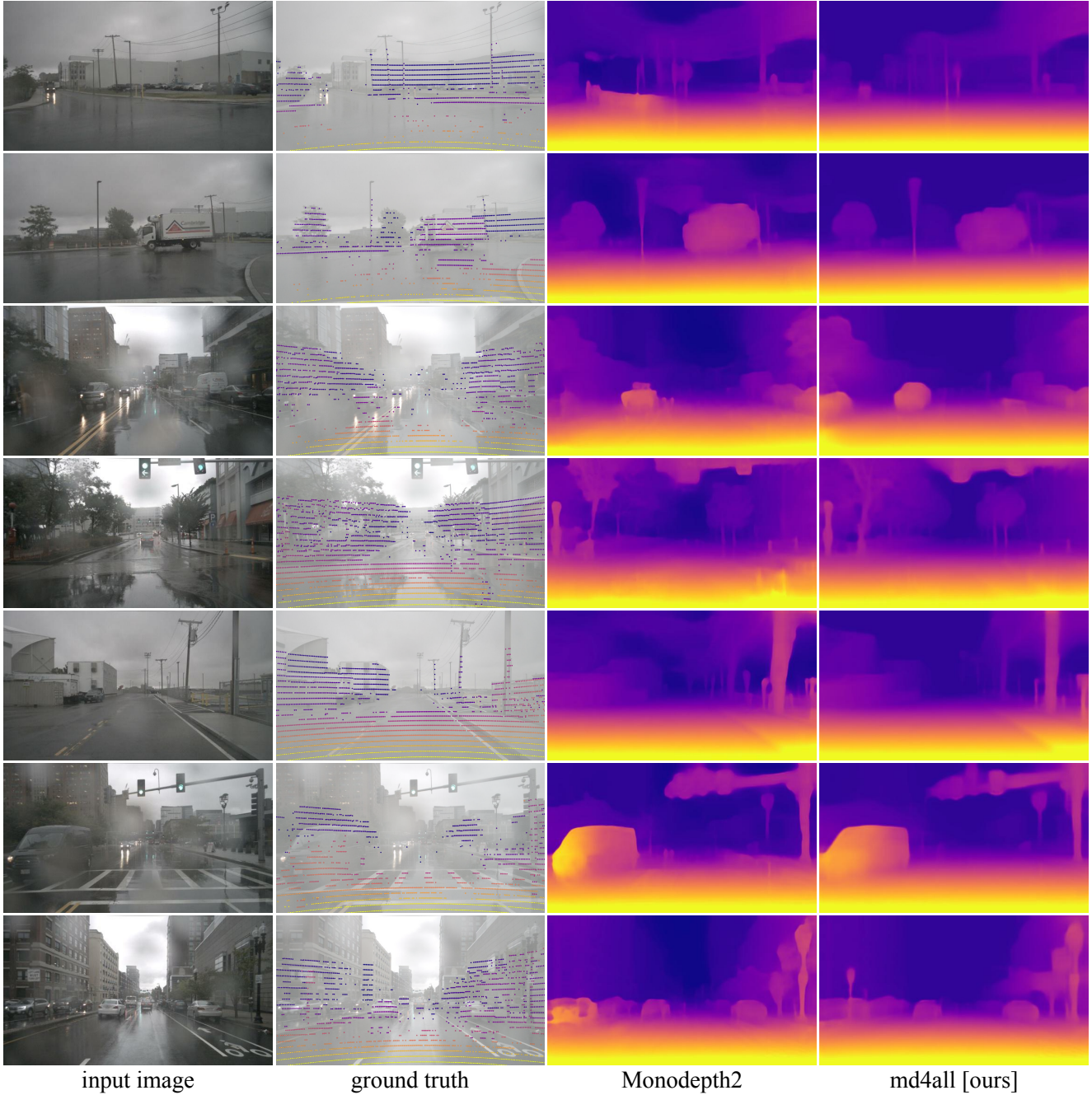
| input image | ground truth | Monodepth2 | md4all [ours] |

Figure 10. Comparison of self-supervised models on nuScenes [4] *rain* samples. The standard Monodepth2 [7] is compared to our md4all-DD applied to Monodepth2.

shows also Monodepth2 having issues estimating the depth of the sky (second and fifth row). Since none of these problems occurs without rain (i.e., *day-clear*, shown in Figure 11), they can be attributed to the challenging weather conditions. Instead, our md4all delivered reasonable depth estimates regardless of the adverse settings, because its supervision signal was always from ideal settings.

***Day-clear* – nuScenes** In the lower half of Figure 11,

we compare the performance at daytime. The *day-clear* scores of Monodepth2 (*d*) and our md4all-DD *a* are relatively close to one another (Table 1). Based on the same model (i.e., Monodepth2), the depth maps of the two are similar, with ours delivering smoother surfaces (e.g., vehicles), with fewer artifacts around the objects, such as the black car on the left of the bottom input. Our extra smoothness is due to the day distillation scheme, propagating dense
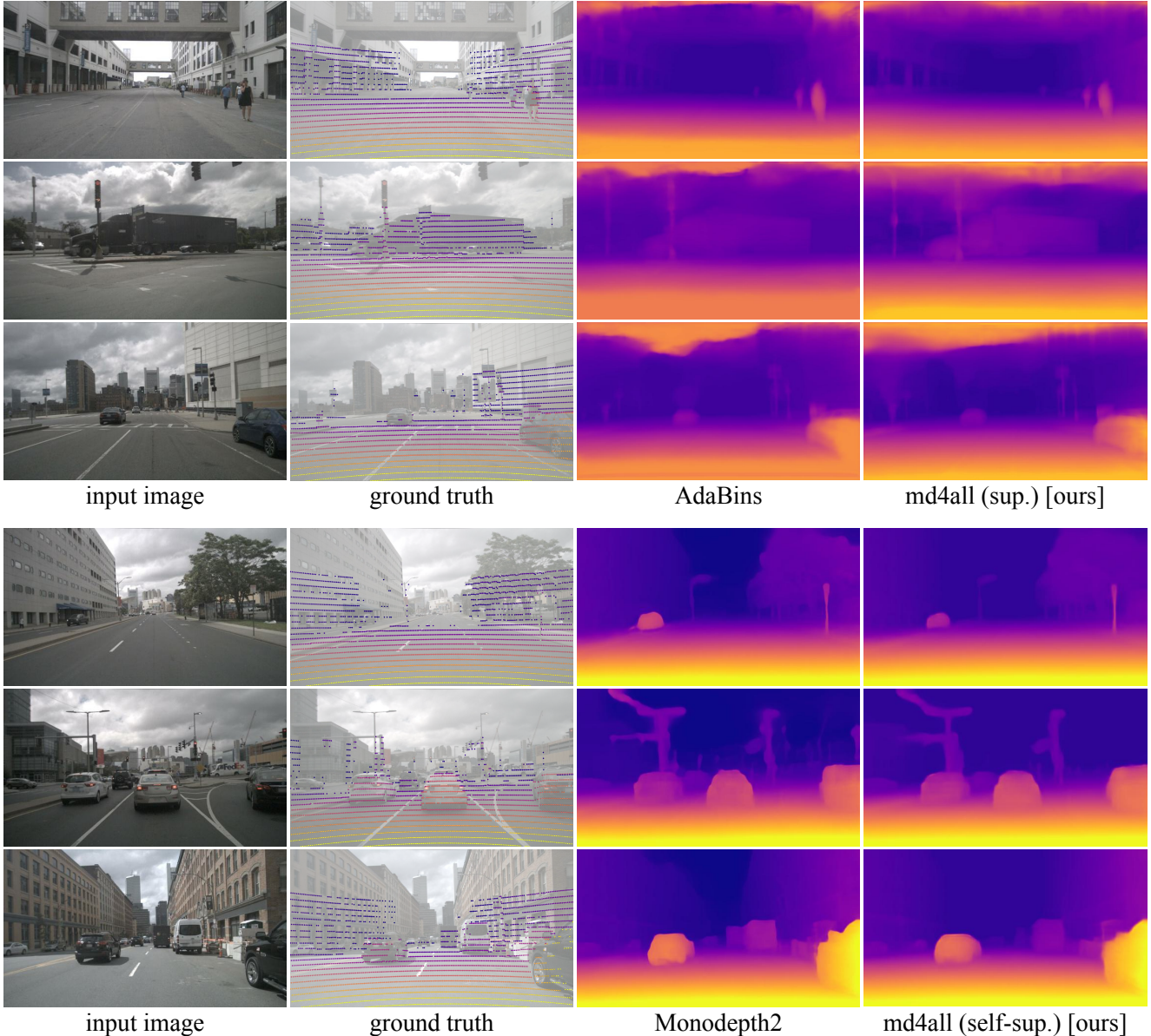
Figure 11. Comparison of models on nuScenes [4] *day-clear* samples. In the upper half, the standard AdaBins [1] is compared to our md4all-AD applied to AdaBins. In the lower half, the standard Monodepth2 [7] is compared to our md4all-DD applied to Monodepth2.

estimates from the baseline. Since our same model predicts significantly better depth maps in adverse conditions than Monodepth2, the figure confirms that our md4all does not degrade in ideal conditions. This proves the effectiveness of our techniques, as they do not introduce any performance trade-off in standard settings while enabling significantly better outcomes in challenging conditions.

A.4.2   nuScenes – Supervised

***Night* – nuScenes** In Figure 12, we compare our md4all-AD to AdaBins [1] on particularly challenging *night* sam-

ples due to the extreme darkness levels and the high amount of reflections (same samples as in Figure 8). Our md4all-AD delivered overall sharper and more accurate estimates, as can be seen for the thin structures (e.g., poles and tree trunks), as well as the boundaries of the vehicles in the bottom two rows. Although all depth maps are displayed with the same colormap, the standard AdaBins never estimated depths that triggered the use of the bright yellow color (i.e., closest distance). This problem is less pronounced during daytime (Figure 11). This can be attributed to the standard AdaBins suffering from overfitting on the sparse ground truth data of nuScenes. Towards this end, we had to ap-

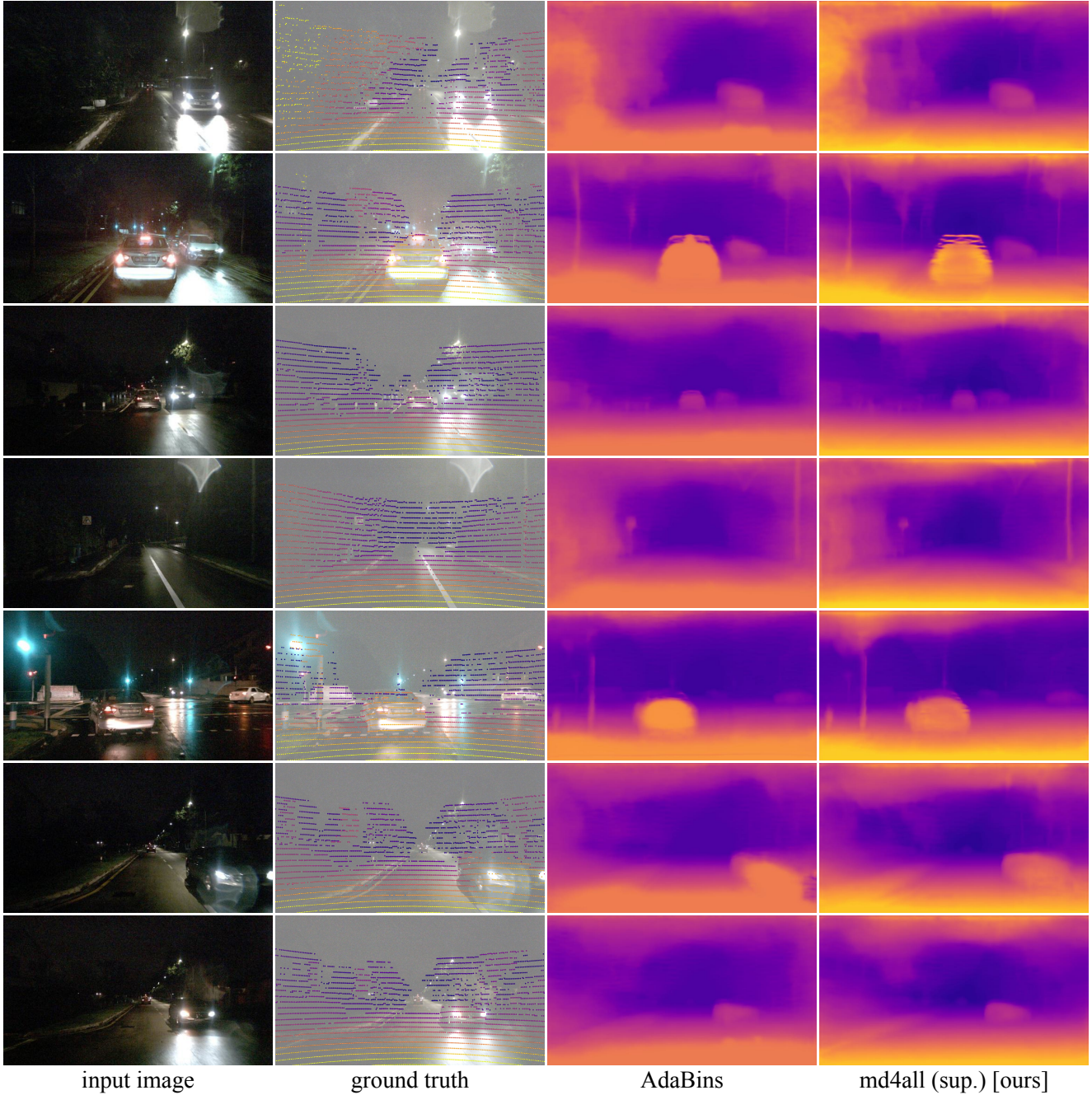| input image | ground truth | AdaBins | md4all (sup.) [ours] |

Figure 12. Comparison of supervised models on nuScenes [4] *night* samples. The standard AdaBins [1] is compared to our md4all-AD applied to AdaBins. This set of samples is particularly challenging due to the overall darkness and reflections.

ply early stopping to prevent severe horizontal artifacts from appearing on the outputs following the LiDAR detections. Applying our md4all on AdaBins introduced challenging augmentations during training, reducing overfitting and allowing it to train longer. This led to a model that estimated depth throughout a more extensive range without the horizontal artifacts. The qualitative results align with the scores gap in Table 1. Moreover, some artifacts are noticeable for

both models around the roof of the vehicles. These are due to the ground truth LiDAR data reporting further estimates in those areas (e.g., fifth row). This is caused by the relative position of the LiDAR sensor to the camera, with the former being at a higher location, thereby seeing beyond objects compared to the camera's perspective.

***Rain* – nuScenes** In Figure 13, we compare our md4all-AD to AdaBins [1] on *rain* samples of nuScenes. As

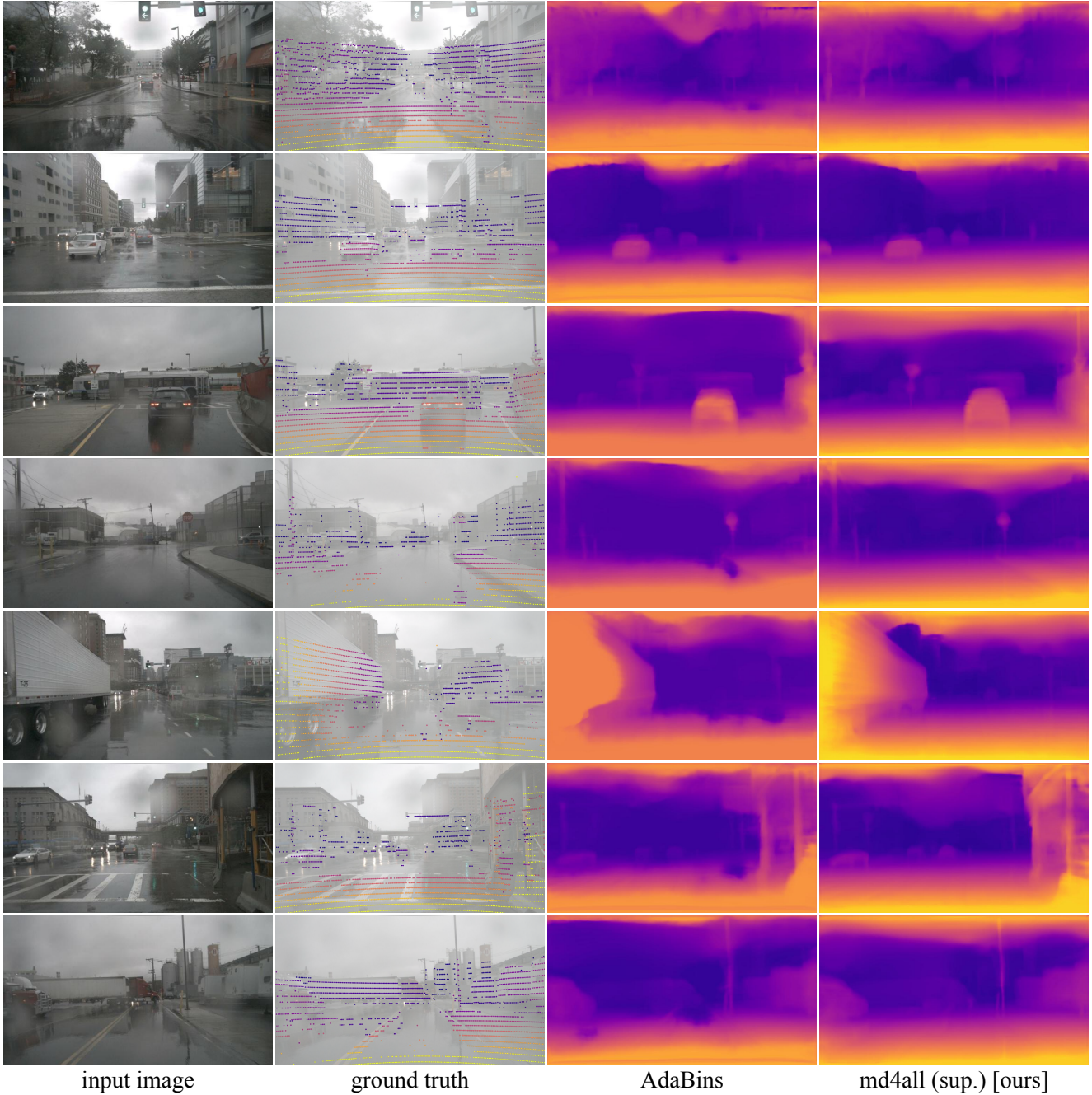| input image | ground truth | AdaBins | md4all (sup.) [ours] |

Figure 13. Comparison of supervised models on nuScenes [4] *rain* samples. The standard AdaBins [1] is compared to our md4all-AD applied to AdaBins.

the standard AdaBins learned from the erroneous measurements of the LiDAR ground truth (e.g., Figure 2), it systematically estimated artifacts on the ground, resulting in holes in its depth maps. The erroneous LiDAR measurements can also be seen from the ground truth reported for each image. Such wrong measurements are caused systematically by each highly reflective object (e.g., traffic sign) reflected on the wet ground. Therefore, training on such a

wrong signal causes the standard AdaBins to replicate the artifacts in its output (e.g., in front of the stop sign in the fourth row). Nevertheless, thanks to the reliable training signal from ideal conditions, our md4all-AD delivered good estimates without such artifacts.

***Day-clear* – nuScenes** In the upper half of Figure 11, we compare the performance at daytime. The scores between AdaBins and our md4all-AD trained on all conditions favor

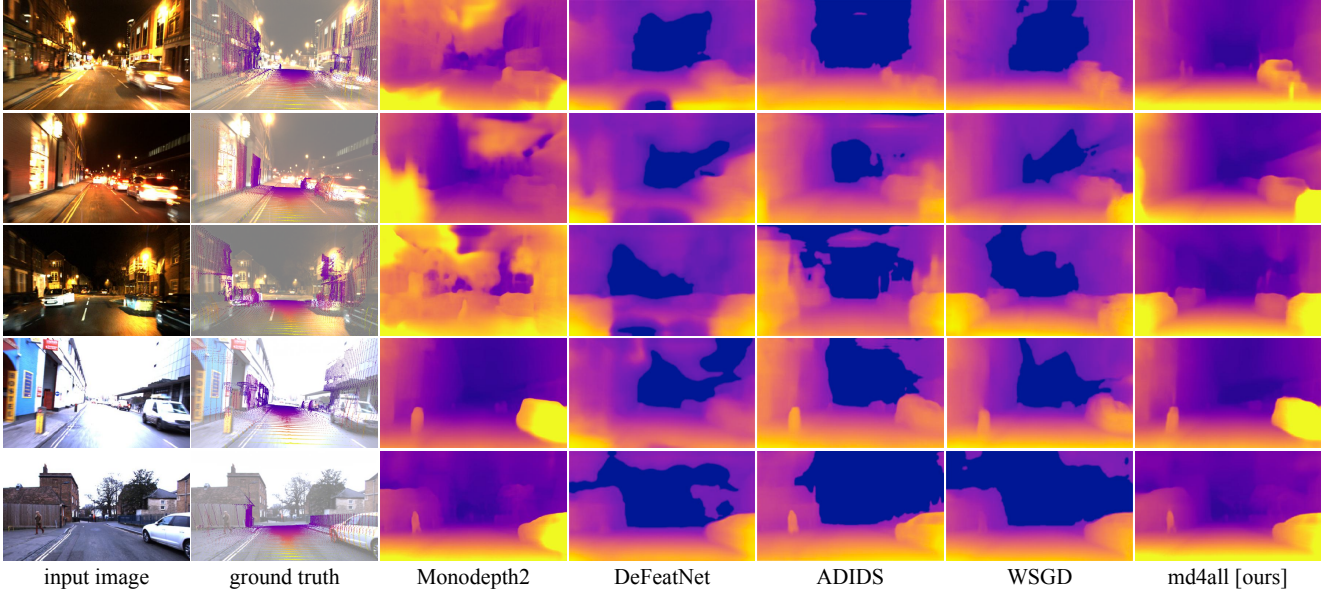| input image | ground truth | Monodepth2 | DeFeatNet | ADIDS | WSGD | md4all [ours] |

Figure 14. Comparison of self-supervised models on RobotCar [10] *night* and *day* samples. The samples are exactly the ones reported by Vankadari et al. in WSGD [12], from which we took directly the predictions of DeFeatNet [11], ADIDS [9], and WSGD [12]. We compare these with Monodepth2 [7] and our md4all-DD.

ours (Table 1). This can be seen from the depth maps, with ours delivering smoother estimates for the road and sharper details throughout. Ours separated the truck in the second frame from the background. As discussed above, this can be attributed to the strong tendency of the standard AdaBins to overfit. Instead, by feeding a mix of original and translated images, our method acts as data augmentation, mitigating the problem. Nevertheless, the limitations of such a sparse ground truth signal are evident, with smooth edges due to its sparsity and see-through effects caused by the displacement between the LiDAR and the camera. For these reasons, self-supervised outputs look better overall, as the models are also less prone to overfitting.

A.4.3 RobotCar – Self-Supervised

Comparison with other *night* methods In Figure 14, we compare our md4all-DD with various other methods targeting depth estimation in *night* conditions, namely DeFeatNet [11], ADIDS [9], and WSGD [12], as well as the standard Monodepth2 [7] designed for ideal conditions. The samples shown are exactly the ones displayed by WSGD [12] in their paper, from which we took the outputs of [11, 9, 12]. However, unlike theirs [12], for ours and Monodepth2, we do not manually threshold the maximum depth, showing the entire depth estimation, including further distances. This is possibly the reason for theirs being artificially dark in the background. Remarkably, as seen already in Table 2 and Figure 6, our md4all-DD delivered more accurate and sharper estimates in both conditions, and especially at *night*, thanks to its robust feature extraction

suitable for both *day* and *night* inputs.

***Night* – RobotCar** Figure 15 shows additional outputs of the standard Monodepth2 [7] compared to our md4all-DD applied on Monodepth2. Although the *night* scenes are not as dark and not as noisy as those from nuScenes [4] (e.g., Figure 8), Monodepth2 had major issues estimating the depth of the image. Darkness and streetlights were detrimental for the standard Monodepth2 (e.g., in the second row). Moreover, compared to nuScenes, the images from RobotCar are often blurry at night, especially those from turns (first row). Nevertheless, the proposed md4all estimated reasonable depth maps regardless of these issues in the input. Due to the textureless pure white sky in the *day* samples of RobotCar (last two rows), the models had issues capturing its depth during training. The same occurred for the top of buildings, which are often too bright and indistinguishable from the sky in the images (e.g., fourth input in Figure 14). This problem is inherent to the data itself and causes the sky to always be predicted to be relatively close (brighter color). This did not happen for nuScenes.

***Day* – RobotCar** Figure 15 also shows additional predictions during daytime (bottom rows). As seen in Table 2, while performing significantly better at *night*, the performance of the proposed method does not degrade during the *day* compared to the standard Monodepth2 [7]. Due to the knowledge distillation from the baseline to our md4all-DD model, surfaces result smoother, but edges remain sharp. The bottom input is particularly challenging as it features a turn. Both models correctly estimated the scene's depth, with Monodepth2 delivering more details on the tree in the

| input image | ground truth | Monodepth2 | md4all [ours] |

Figure 15. Comparison of self-supervised models on RobotCar [10] *night* and *day* samples. The standard Monodepth2 [7] is compared to our md4all-DD applied to Monodepth2.

foreground but less on the tree in the background (left). As discussed for the *night* scenes, the textureless white sky of RobotCar prevents the models from learning its depth correctly, which causes erroneous estimations at test time. Instead, nuScenes [4] includes different sky conditions (e.g., sunny and cloudy), allowing the models to learn its depth.

A.4.4   Failure Cases

While our techniques bring significant improvements across various conditions, there is still room for improvement. Figure 16 reports failure cases of our models. Inherited from Monodepth2 [7], our self-supervised model has issues with

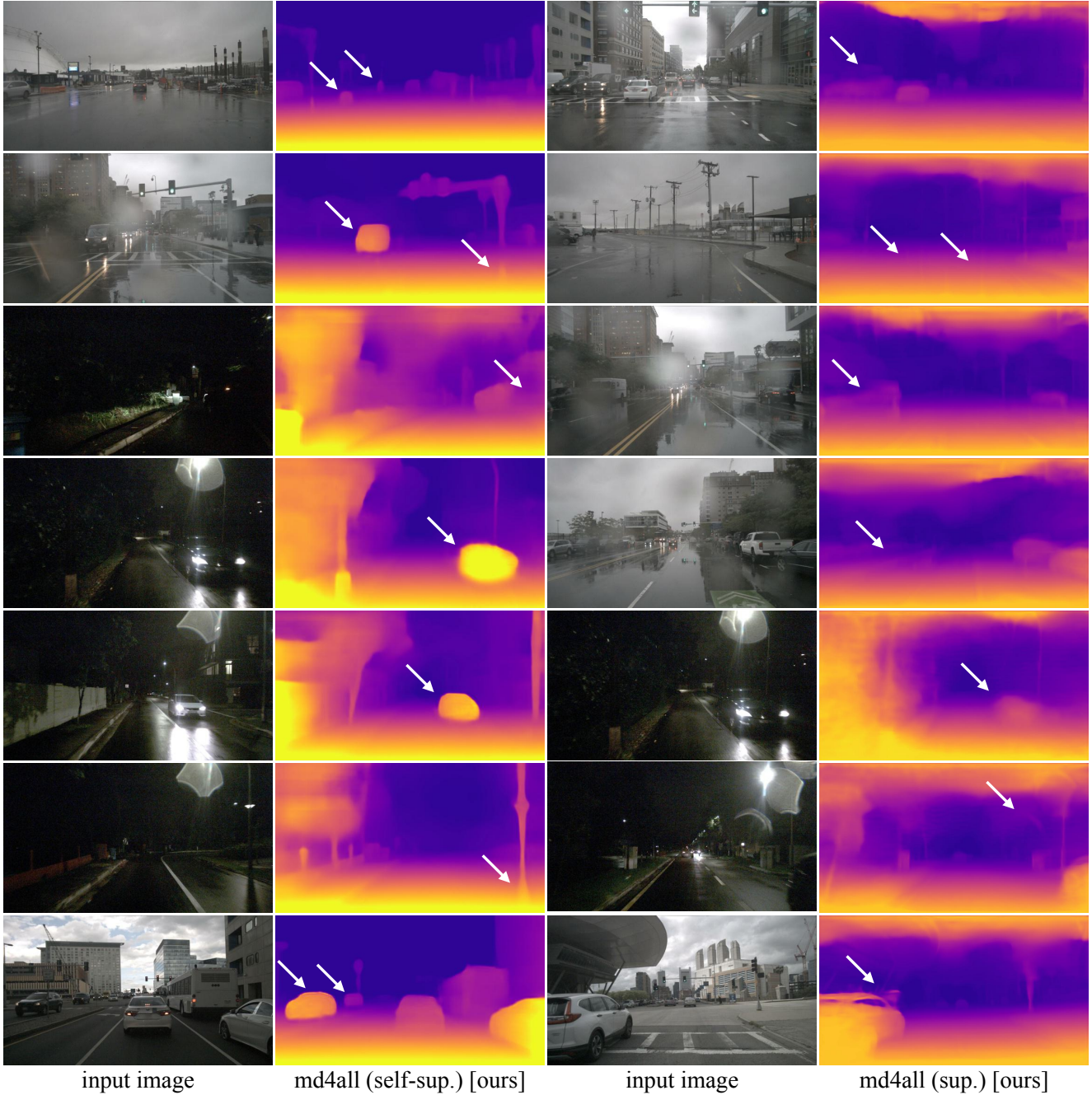|			
input image	md4all (self-sup.) [ours]	input image	md4all (sup.) [ours]

Figure 16. Failure cases of our self-supervised (md4all-DD, Monodepth2-based) and supervised (md4all-AD, AdaBins-based) models on samples from nuScenes [4]. White arrows mark issues in the predictions.

dynamic objects, especially oncoming traffic, whose distance is wrongly estimated due to the violations of the moving camera in a static world assumption [5]. This occurs in all three conditions. Furthermore, the model is occasionally fooled by reflections on the ground or misleading shadows which look like objects (first and sixth row). Additionally, highly dark scenes are also challenging (third row) due to the lack of information across nearly black pixels.

els. These issues could be mitigated by integrating the data from the cost-effective radar, as in R4Dyn [5], which is robust against adverse weather and challenging illumination conditions. Our method is not bound to a specific architecture or pipeline, so it could be applied to R4Dyn directly. Another drawback our md4all inherited from Monodepth2, which we did not address, is the lack of temporal consistency in the depth predictions. Instead, for the supervised

| original *rain* | day-to-rain translation | *day-clear* input | day-to-night translation | original *night* |

Figure 17. Example of day-to-adverse image translations on nuScenes [4]. Training samples are translated from *day-clear* to both *rain* and *night*. For reference, in the first and last columns, we included real *rain* and *night* samples from the validation set. We share publicly the translated *night* and *rain* images for the entire *day-clear* training set.

setting, we focused on eliminating the artifacts due to the erroneous ground truth measurements. While such issues are appropriately addressed, other problems persist, such as with particular reflections and the blur caused by water drops (third row). Radar may help here too.

A.4.5 Day-to-Adverse Translation

This section shows samples of the translated images we generated with ForkGAN [16]. We publicly share all translated images corresponding to the ideal settings from the training sets of nuScenes [4] and RobotCar [10].

nuScenes In Figure 17, we show examples of the translated images used to train our models on nuScenes. The GAN [16] added plausible reflections and lights simulating wet ground or streetlights. While these additions are not al-

ways realistic, they bring challenging variations to the training set, resembling the adverse conditions in the first and last columns. The standard *day-clear* training set includes a mix of sunny and cloudy scenes. For cloudy ones, shadows are limited, with less contrast overall, making it easier for the translation model. Instead, the inputs with sunny conditions are particularly challenging for the GAN, resulting in less convincing outputs (e.g., in the last row). The GAN also learned to add water drops blurring certain areas for *rain*, and intense noise for *night*. The results of our models would directly benefit from improvements in the translations towards greater realism. With perfect translations, our md4all would further reduce the gap between ideal and challenging conditions.

RobotCar In Figure 18, we report examples of the translated images used on RobotCar. In this case, the GAN was

| *day* input | day-to-night translation | original *night* | day-to-night translation | *day* input |

Figure 18. Example of day-to-adverse image translations on RobotCar [10]. Training samples are translated from *day* to *night*. For reference, we included real *night* samples from the test set in the central column. We share publicly the translated *night* images for the entire *day* training set.

trained entirely on RobotCar, thanks to the high amount of *night* samples available. Since the weather in the *day* samples tends to be always cloudy, with a consistently white sky, the GAN did not exhibit the issues seen for nuScenes with sunny inputs (Figure 17). Instead, the GAN delivered highly plausible samples with a higher degree of realism than for nuScenes, enabling our model to narrow the margin between daytime and nighttime performances. As seen for nuScenes, the GAN added streetlights resembling the ones seen throughout the dataset. Furthermore, it can be seen clearly that the GAN learned the headlights of the ego vehicle, which it added to illuminate the road ahead.

For both datasets, while being somewhat plausible, the translated samples are not perfect. For nuScenes, for example, noise patterns in the *night* translations are repeated similarly over the lower left corner of the images. The noise is more unstructured for the real *night* images. Additionally, the GAN did not learn to turn on the lights of other cars, which are often a source of issues for the models.

A.5. Attempted Approaches That Did Not Work

We explored several alternative solutions for this challenging problem, and we intend to mention them in this section to help future researchers who target the same issues. As written in Section 4.1, we experimented with diffusion models to learn our day-to-adverse image translation. However, the lack of paired images for *day* and *night* made it unfeasible. Therefore, we opted for GANs, which do not need

paired inputs. Furthermore, we experimented with the following methods on the *night* samples of nuScenes, seeking improvements over the standard Monodepth2. Spatial and temporal attention could have allowed the model to focus on valuable information. Still, it did not bring an improvement, possibly due to the large amount of noise varying across different *night* images, which could have prevented to attend on the helpful information. Additionally, we experimented with incorporating geometrical priors (e.g., from edge detection) into the losses, but they were similarly not beneficial. The simple solution presented in this work was the most effective to tackle this complex problem.

References

[1] Shariq Farooq Bhat, Ibraheem Alhashim, and Peter Wonka. AdaBins: Depth estimation using adaptive bins. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4009–4018, 2021. 3, 6, 8, 9, 10, 15, 16, 17

[2] Mario Bijelic, Tobias Gruber, Fahim Mannan, Florian Kraus, Werner Ritter, Klaus Dietmayer, and Felix Heide. Seeing through fog without seeing fog: Deep multimodal sensor fusion in unseen adverse weather. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11682–11692, 2020. 5, 8

[3] Keenan Burnett, David J Yoon, Yuchen Wu, Andrew Z Li, Haowei Zhang, Shichen Lu, Jingxing Qian, Wei-Kang Tseng, Andrew Lambert, Keith YK Leung, Angela P Schoellig, and Timothy D Barfoot. Boreas: A multi-season au-

tonomous driving dataset. *The International Journal of Robotics Research*, 42(1-2):33–42, 2023. 8

[4] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11621–11631, 2020. 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21

[5] Stefano Gasperini, Patrick Koch, Vinzenz Dallabetta, Nassir Navab, Benjamin Busam, and Federico Tombari. R4Dyn: Exploring radar for self-supervised monocular depth estimation of dynamic scenes. In *Proceedings of the International Conference on 3D Vision (3DV)*, pages 751–760. IEEE, 2021. 4, 5, 6, 10, 20

[6] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 270–279, 2017. 2

[7] 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*, pages 3828–3838, 2019. 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 18, 19

[8] Vitor Guizilini, Rares Ambrus, Sudeep Pillai, Allan Raventos, and Adrien Gaidon. 3D packing for self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2485–2494, 2020. 2, 5, 6, 7, 8

[9] Lina Liu, Xibin Song, Mengmeng Wang, Yong Liu, and Liangjun Zhang. Self-supervised monocular depth estimation for all day images using domain separation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12737–12746, 2021. 5, 18

[10] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 year, 1000 km: The Oxford RobotCar dataset. *The International Journal of Robotics Research*, 36(1):3–15, 2017. 3, 4, 5, 7, 10, 11, 18, 19, 21, 22

[11] Jaime Spencer, Richard Bowden, and Simon Hadfield. DeFeat-Net: General monocular depth via simultaneous unsupervised representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14402–14413, 2020. 5, 18

[12] Madhu Vankadari, Stuart Golodetz, Sourav Garg, Sangyun Shin, Andrew Markham, and Niki Trigoni. When the Sun Goes Down: Repairing photometric losses for all-day depth estimation. In *Proceedings of the Conference on Robot Learning*, pages 1992–2003. PMLR, 2023. 4, 5, 7, 18

[13] Kun Wang, Zhenyu Zhang, Zhiqiang Yan, Xiang Li, Baobei Xu, Jun Li, and Jian Yang. Regularizing Nighttime Weirdness: Efficient self-supervised monocular depth estimation in the dark. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16055–16064, 2021. 4, 5, 6

[14] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 2

[15] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. BDD100K: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2636–2645, 2020. 4

[16] Ziqiang Zheng, Yang Wu, Xinran Han, and Jianbo Shi. ForkGAN: Seeing into the rainy night. In *Proceedings of the European Conference on Computer Vision*, pages 155–170. Springer, 2020. 3, 8, 21