Supplementary Material
Guided Depth Super-Resolution by Deep Anisotropic Diffusion

Nando Metzger* Rodrigo Caye Daudt* Konrad Schindler
Photogrammetry and Remote Sensing, ETH Zurich
{metzgern,rcayedaudt,schindler}@ethz.ch

A. Videos

Videos depicting the diffusing process (Y_t and residuals) are included with the supplementary materials. The examples are from the Middlebury dataset, with a scaling factor of $\times 32$.

B. Training set up

We train our method with an Adam optimizer [5] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, a learning of 10^{-3}, and weight decay parameter set to 10^{-5}. Furthermore, we clip gradients to a maximum norm of 0.01 for stability during training. We find that DADA reaches convergence after 4500, 550, and 300 epochs for the datasets of Middlebury, NYUv2, and DIML respectively. During training, we apply data augmentation, which includes horizontal flips, random cropping, and rotating the samples by up to 15 degrees. We normalize all depth maps with their respective dataset-specific standard deviation. The Mean is not subtracted since the adjustment step assumes positive values.

The feature extractor is a U-Net [6] with ResNet-50 [3] backbone that operates on a $\times 2$ upsampled guide. Subsequently, the produced feature maps are downsampled again to the original spatial resolution. We ablate this design choice in Appendix G.

C. Experimental set up

Our experiments and baselines closely follow the setup described in [1] (supplementary material) and compare with mostly the same baselines. All methods are trained with the default settings of the Adam optimizer and a learning rate of 10^{-4} (except PMBA: 10^{-3}, FDSR: 5×10^{-4}). We train all learned models until convergence, which is reached after 2500, 250, and 150 epochs on the datasets of Middlebury, NYUv2, and DIML respectively. We reduce the learning rate by a factor of 0.9 every 100, 10 and 6 epochs for Middlebury, NYUv2, and DIML, respectively. In order to limit the GPU memory consumption to a manageable level, we had to reduce the patch size for some baselines from 256^2 to 128^2 for PMBA x4 even to 64^2. For the guided filter, we use a radius of 8, and for the SD Filter, we use the following hyperparameter configuration: $\lambda = 0.1$, $\sigma_g = 60$, $\sigma_u = 30$. For Pixtransform, we use the standard hyperparameters reported in their paper [2].

*Equal contribution.

Scale	x4	x8	x16	x32
MSE [cm^2]	2.52±0.04	5.63±0.09	15.6±0.10	47.6±0.50
MAE [cm]	0.11±0.00	0.20±0.00	0.47±0.00	1.35±0.01
MAPE [%]	0.04±0.00	0.07±0.00	0.17±0.00	0.46±0.00
VV [%]	0.06±0.00	0.12±0.00	0.30±0.00	0.91±0.01
EE [%]	0.82±0.02	1.43±0.01	2.92±0.02	8.09±0.03
MSE [cm^2]	4.87±0.03	17.1±0.30	59.2±0.60	223±3.00
MAE [cm]	0.64±0.00	1.33±0.01	2.65±0.03	5.76±0.02
MAPE [%]	0.17±0.00	0.36±0.00	0.73±0.01	1.62±0.01
VV [%]	0.05±0.00	0.18±0.00	0.66±0.01	2.40±0.03
EE [%]	3.68±0.06	9.90±0.20	23.2±0.30	44.1±0.08
MSE [cm^2]	1.30±0.02	2.87±0.06	7.75±0.12	38.6±0.80
MAE [cm]	0.17±0.00	0.27±0.00	0.60±0.01	1.90±0.02
MAPE [%]	0.06±0.00	0.10±0.00	0.21±0.00	0.68±0.01
VV [%]	0.03±0.00	0.07±0.00	0.18±0.00	0.85±0.01
EE [%]	2.44±0.01	3.75±0.03	7.39±0.08	19.5±0.19

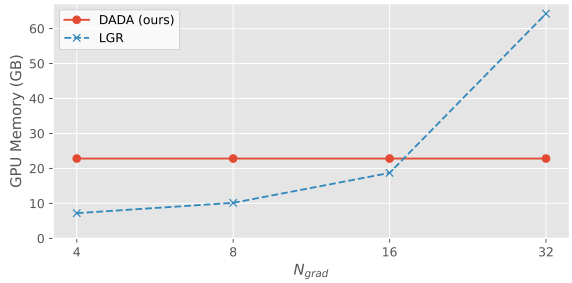
Table A1. Variability of metrics on the Middlebury, NYUv2, and DIML datasets in the top, middle, and bottom sections, respectively.

D. Stability & Statistics

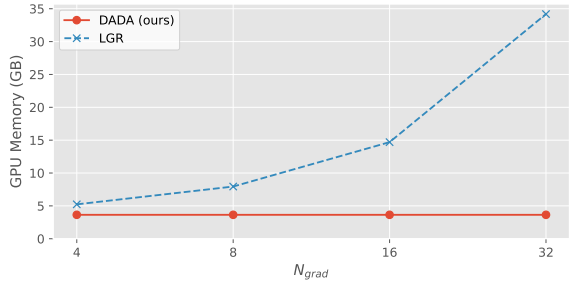
We also repeated the training using 4 additional (5 total) different random seeds to determine the epistemic uncertainty/stability of our method. We observe little variability across runs that consistently surpass previous results, see Tab. A1.

E. GPU memory consumption

In Figure A1 we show the GPU memory consumption of our method at training time (a) and at inference (b) and contrast them to the LGR approach, which is our closest competitor and is also a hybrid method. DADA requires $\sim$23 GB at training time and $\sim$4 GB at inference time, regard-



(a) Memory requirement during training



(b) Memory requirement during testing

Figure A1. Memory requirements for a batch of 8 samples for DADA and LGR. Constant memory requirements are an advantage at higher scaling factors.

less of the upsampling factor, while LGR uses up to ∼64 GB for training and ∼35 GB for testing.

F. Inference time

We measure a mean inference time for the proposed method of ∼100 ms per sample when using a batch size of 32 on an NVIDIA GeForce RTX 3090, with all other parameters equal to the canonical ones previously described. We note that the inference time is invariant w.r.t. the upsampling factor. For LGR we found it to increase with the scaling factor: 50 ms, 84 ms, 240 ms, and 910 ms for ×4, ×8, ×16, and ×32, respectively. In contrast, the feedforward methods are a lot faster – most of them are below 20 ms.

G. Further ablations

Upsampling of the guide. We explore two different ways of employing a U-Net feature extractor: Our default case where we upsample the guide by ×2 and we downsample the obtained feature maps afterward and a case without up- and downsampling modifications. Tab. A3, shows that this modification brings an improvement in the settings of ×4 to ×16, while a slight weakening effect is observed for ×32. Our intuition is that upsampling helps to produce sharp and accurately localized edges, which is especially beneficial for smaller scaling factors. Inversely, we speculate that for ×32 global context plays a bigger role, and hence, upsampling the guide decreases the receptive field of the U-Net, which in turn deteriorates the method's performance.

	U-Net RN18	U-Net RN34	**U-Net RN50**	U-Net ENB0	U-Net ENB1	U-Net ENB2	FPN RN50	DL3+ RN50
MSE	5.69	5.57	5.63	6.85	6.77	6.68	7.09	7.77
MAE	0.20	0.20	0.20	0.24	0.24	0.24	0.27	0.30

Table A2. Middlebury, scale x8. DADA is invariant to the encoder depth and fairly invariant to the choice of model architecture.

	×4 MSE / MAE	×8 MSE / MAE	×16 MSE / MAE	×32 MSE / MAE
Middlebury				
Base	**2.58 / 0.11**	**5.63 / 0.20**	**16.3 / 0.48**	50.6 / 1.38
No up-/down	2.88 / 0.12	5.92 / 0.22	18.3 / 0.55	**49.9 / 1.36**
NYUv2				
Base	**4.83 / 0.64**	**16.6 / 1.30**	**59.0 / 2.64**	228 / 5.81
No up-/down	6.12 / 0.72	18.7 / 1.40	60.7 / 2.72	**207 / 5.56**
DIML				
DADA	**1.33 / 0.17**	**2.93 / 0.28**	**7.61 / 0.59**	39.8 / 1.92
No up-/down	1.59 / 0.18	3.28 / 0.30	8.56 / 0.64	**36.8 / 1.82**

Table A3. Ablation: method as proposed *vs*. method which skips upsampling before and downsampling after the feature extractor. Errors are in cm^2 (MSE) and in cm (MAE). Up-/downsampling appears to have a positive effect for lower scales and a neutral or slightly negative effect for very large scales.

Randomization of iterations. In Fig. A2 we show the effect of randomizing the number of iterations without gradient at training time (as proposed) against choosing it to be a constant $N_{\text{grad}} = 8000$. At test time, it seems that both strategies eventually converge to an equally performant solution. However, we note that our proposed way of training leads to faster convergence in the diffusion-adjustment iterations.

Feature Extractor. We explored additional feature extractors by replacing the base case (U-Net with ResNet-50) with two more different ResNet depths (18 and 34) and EfficientNets (B0, B1, B2). We also tested different feature extractors (FPN, DeepLabv3+) with the ResNet-50 backbone. We show the results in Tab. A2. DADA is robust to changes in the exact architecture of the feature extractor.

H. Additional metrics

We concur that other metrics can provide additional insight into the method's performance and facilitate future comparisons. We computed the mean absolute percentage error (MAPE), value errors (VE), and edge errors (EE) as described in [4]. The advantage of DADA w.r.t. current methods remains clear, see Tab. A4.

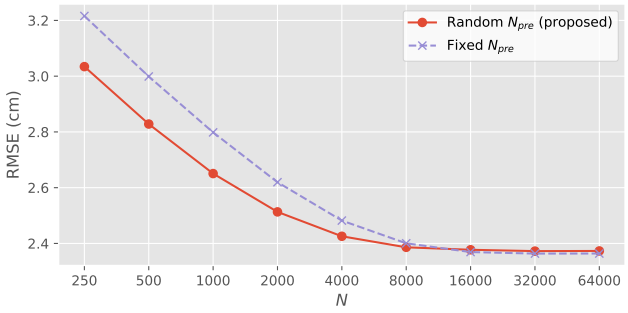


Figure A2. Randomizing the number of iterations without gradient during training speeds up convergence at inference time.

in [%]	**Middlebury** MAPE/VE/EE	**NYUv2** MAPE/VE/EE	**DIML** MAPE/VE/EE
MSG	0.8 / 1.7 / 14.7	1.9 / 3.0 / 49.8	0.9 / 1.2 / 24.8
DKN	1.2 / 2.7 / 21.9	2.4 / 4.4 / 55.8	1.4 / 2.2 / 32.3
FDKN	1.2 / 2.6 / 21.7	2.4 / 4.5 / 55.7	1.4 / 2.2 / 34.3
PMBA	1.9 / 4.1 / 35.7	2.7 / 5.2 / 57.4	0.8 / 1.2 / 25.0
FDSR	1.6 / 3.7 / 21.6	3.4 / 7.2 / 63.4	1.6 / 2.5 / 36.3
LGR	0.6 / 1.2 / 12.8	1.8 / 2.7 / 49.0	0.9 / 1.3 / 26.7
DADA	**0.5 / 0.9 / 8.1**	**1.6 / 2.4 / 44.3**	**0.7 / 0.9 / 19.7**

Table A4. Middlebury, scale x32, all numbers are in [%].

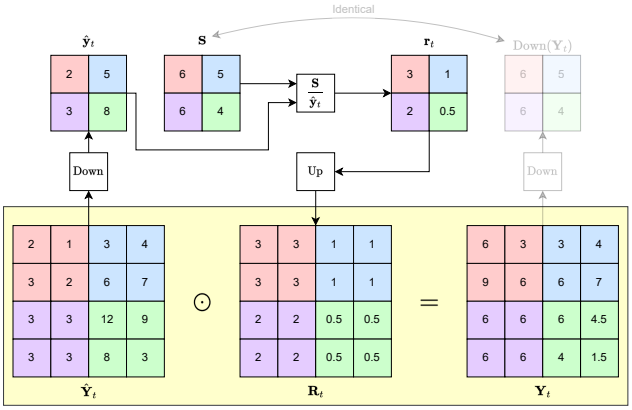


Figure A3. Numerical example of the adjustment step.

I. Main results as curves

For improved interpretation, the RMSE numbers from Tables 1 and 2 from the main paper are here also displayed as curves. These plots can be seen in Figure A4.

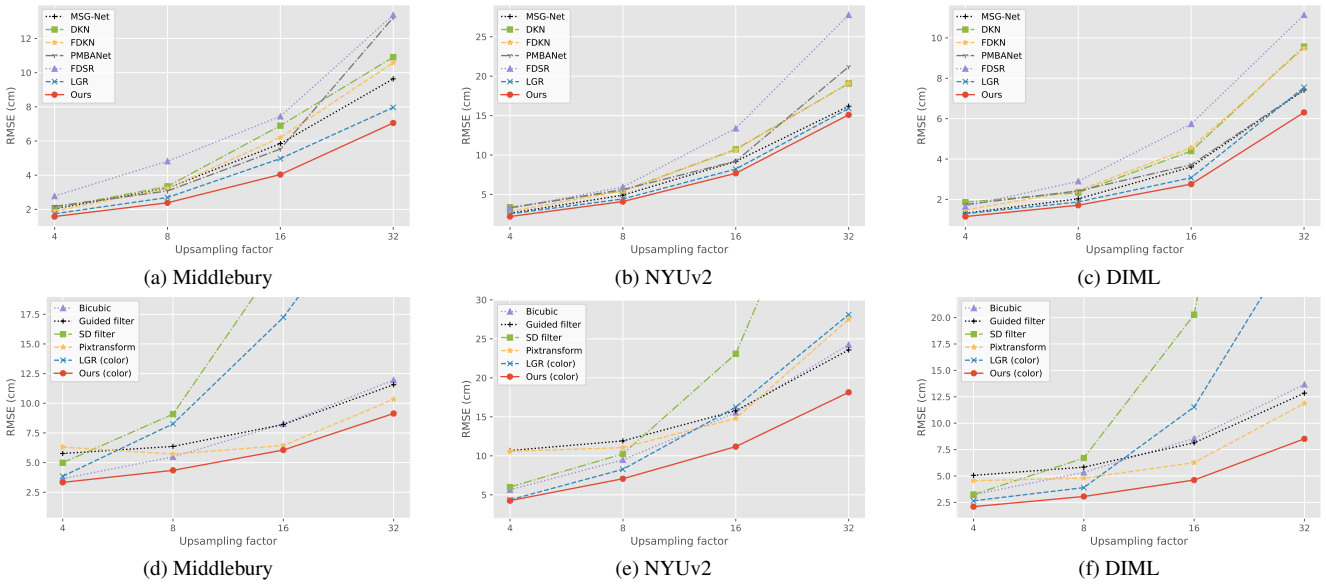
J. Intuition behind the adjustment step

The values of $\mathbf{r}_t$ and $\mathbf{R}_t$ are computed precisely so that down($\mathbf{Y}_t$) matches $\mathbf{S}$. The values in $\mathbf{r}_t$ are per-patch adjustment coefficients. A numerical example of these operations is shown in Fig. A3 to help convey the intuition behind this operation.

K. Qualitative results

We provide additional results from for all learned methods, the color version of LGR, and Pixtransform – the strongest unsupervised competitor – for all three datasets and all four scaling factors in Table A5, Table A6, and Table A7.

The plots how that for smaller upsampling factors, all learned methods perform relatively well, while for larger upsampling factors, the advantages of DADA become more apparent: Shapes are represented more accurately and edges are sharper.



(a) Middlebury (b) NYUv2 (c) DIML

(d) Middlebury (e) NYUv2 (f) DIML

Figure A4. Curves with numbers from Tables 1 and 2 in main paper for better interpretability of results. RMSE is used instead of MSE for improved visualization across scales.

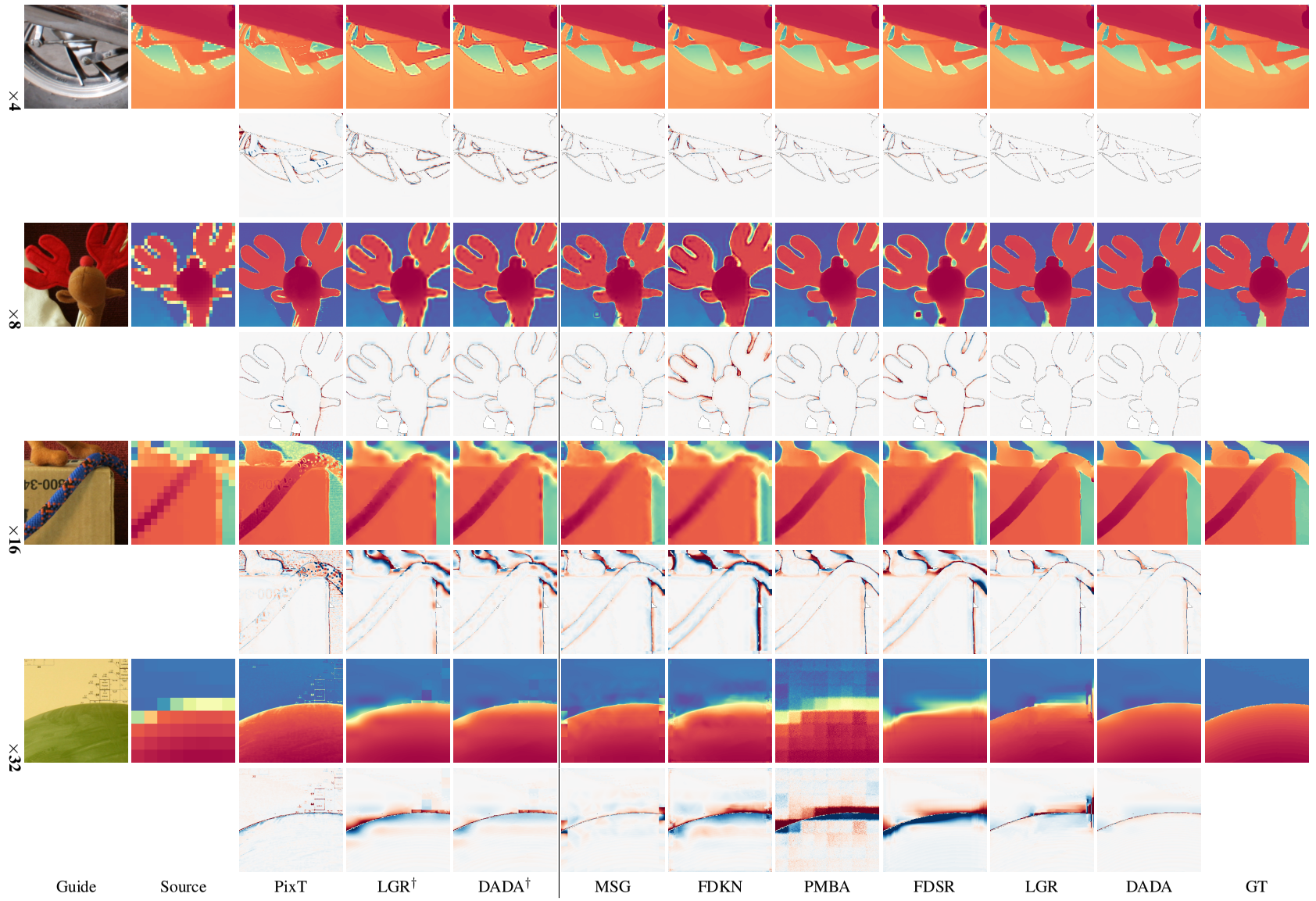


Table A5. Predictions and error maps for different guided super-resolution methods on the Middlebury dataset. Blue denotes under-estimated depth, red denotes over-estimation. All plots in a row have the same color scale. The last two columns juxtapose our predictions and the ground truth.

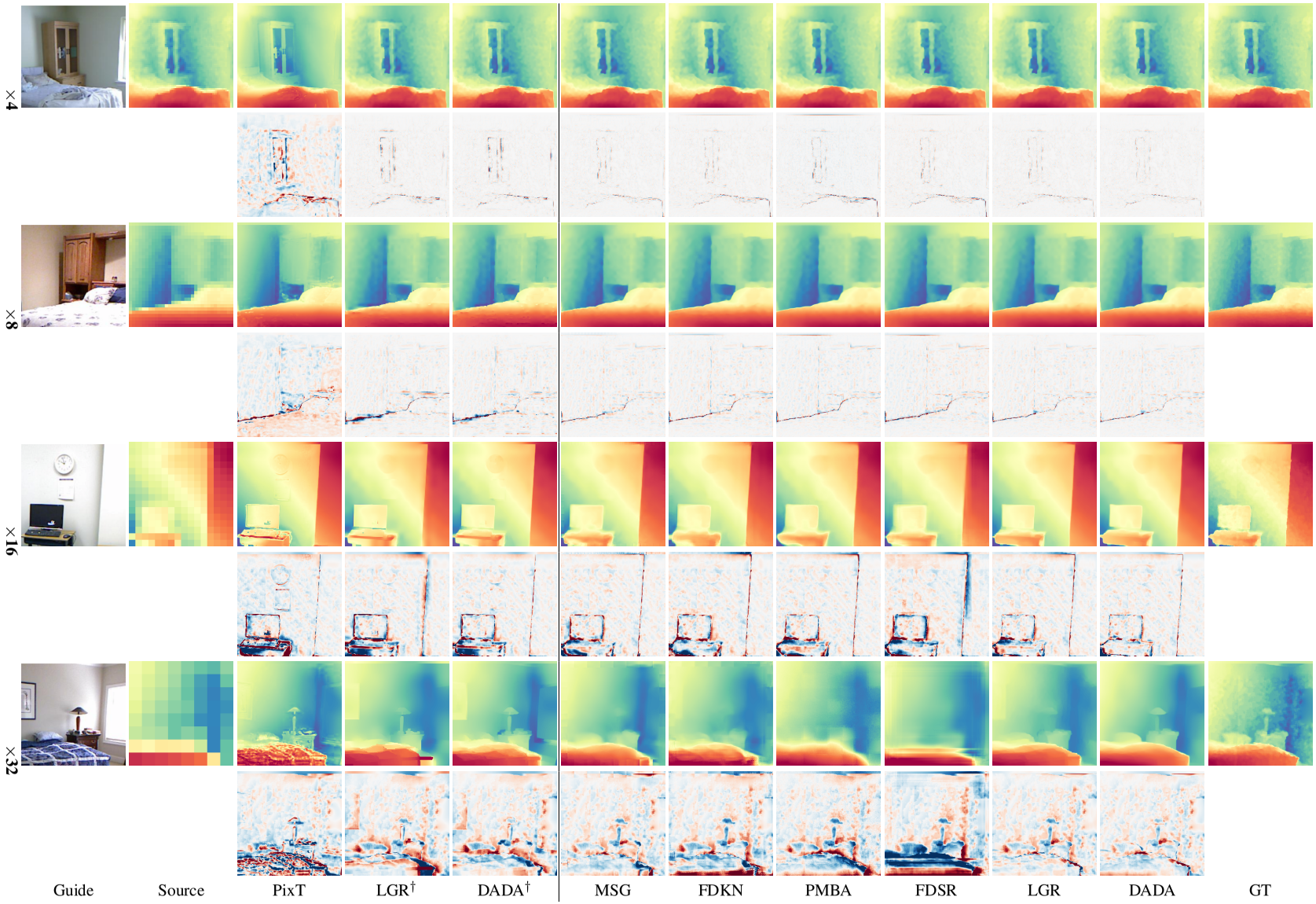


Table A6. Predictions and error maps for different guided super-resolution methods on the NYUv2 dataset. Blue denotes under-estimated depth, red denotes over-estimation. All plots in a row have the same color scale. The last two columns juxtapose our predictions and the ground truth.

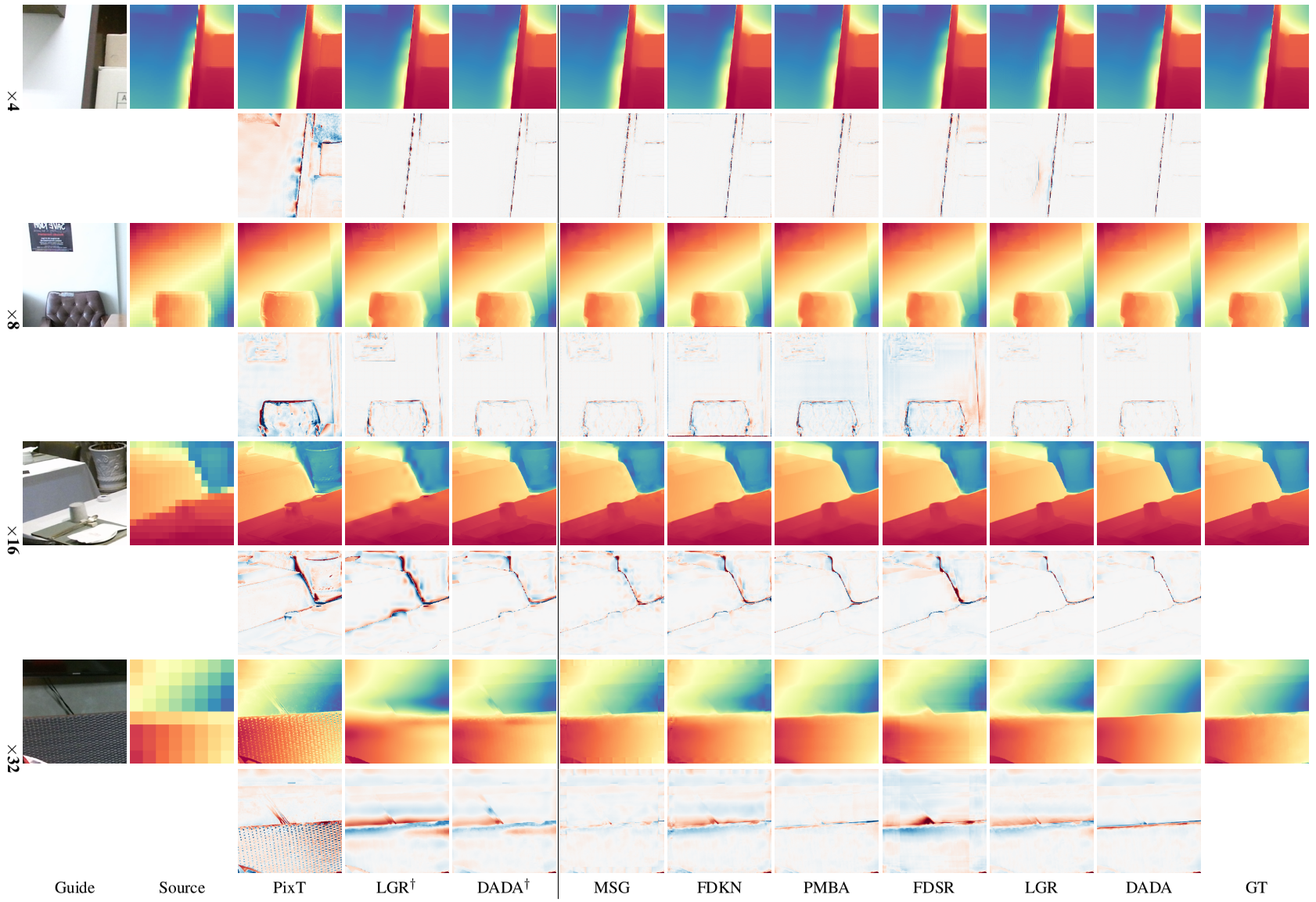


Table A7. Predictions and error maps for different guided super-resolution methods on the DIML dataset. Blue denotes under-estimated depth, red denotes over-estimation. All plots in a row have the same color scale. The last two columns juxtapose our predictions and the ground truth.

References

[1] Riccardo de Lutio, Alexander Becker, Stefano D'Aronco, Stefania Russo, Jan D Wegner, and Konrad Schindler. Learning graph regularisation for guided super-resolution. *CVPR*, 2022. 1

[2] Riccardo de Lutio, Stefano D'Aronco, Jan Dirk Wegner, and Konrad Schindler. Guided super-resolution as pixel-to-pixel transformation. *ICCV*, 2019. 1

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, 2016. 1

[4] Lingzhi He, Hongguang Zhu, Feng Li, Huihui Bai, Runmin Cong, Chunjie Zhang, Chunyu Lin, Meiqin Liu, and Yao Zhao. Towards fast and accurate real-world depth super-resolution: Benchmark dataset and baseline. *CVPR*, 2021. 2

[5] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

[6] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *MICCAI*, 2015. 1