Supplementary Material

A. Experimental Details

A.1. Training Details

In this section, we describe the details of the training environments we used. We implemented our methods based on PyTorch [12] 1.3.0, and the entire models can be trained with 2 Titan XP gpus with 12GB memory, respectively. As mentioned in Sec. 4.2, we resized original image into a resolution of 192×640 and used a batch size of 12 for all experiments. Also, the local patch size, K, is set to five, and the margin, m, is set to 0.3 for the semantics-guided triplet loss. For training with high resolution images in Table 1b and Table 2 in the main paper, we resized original image into 320×1024 and the local patch size, K, was set to seven as the image resolution increases.

The pseudo-labels for training semantic segmentation on the KITTI Eigen split are generated using the off-the-shelf segmentation network proposed in [16]. This network has been trained on Mapillary Vistas [11], Cityscapes [2], and fine-tuned on 200 images of the KITTI 2015 [10] training set.

A.2. Network Architecture

The encoder architecture is based on a ResNet-18 [7], removing the average pooling layer and the classifier layer. Table 1 shows the detailed architecture of the depth decoder. As mentioned in the main paper, we apply the CMA modules (cma0, cama1, cma2) to three layers of the decoder ($l = 0, 1, 2$) taking different scales of the feature maps from the previous layers. There is only one difference between the depth and the segmentation decoder. The latter replaces the disp layers of the depth decoder with semantic layers, which has channel dimensions of 19, the same as the number of the classes.

Decoder Architecture				
l	**layers**	**chns**	**scale**	**input**
0	upconv5	256	32	econv5
	iconv5A	256	16	[↑upconv5, econv4]
	cma0	256	16	K,V: iconv5A, Q: iconv5B
1	upconv4	128	16	cma0
	iconv4A	128	8	[↑upconv4, econv3]
	cma1	128	8	K,V: iconv4A, Q: iconv4B
	disp4	1	1	cma1
2	upconv3	64	8	cma1
	iconv3A	64	4	[↑upconv3, econv2]
	cma2	64	4	K,V: iconv3A, Q: iconv3B
	disp3	1	1	cma2
3	upconv2	32	4	cma2
	iconv2	32	2	[↑upconv2, econv1]
	disp2	1	1	iconv2
4	upconv1	16	2	iconv2
	iconv1	16	1	↑upconv1
	disp1	1	1	iconv1

Table 1: Detailed architecture of the depth decoder. ↑ denotes upsamping with nearest neighbor interpolation, and iconvB is the feature map from the segmentation decoder at the same scale. econv is skip-connections from the encoder.

Method	Training time	Memory	Inference time
Monodepth2 [12]	14h 40m	8.6GB	4.9ms
w/ semantics	16h 20m	10.5GB	7ms
w/ CMA	16h 10m	7.2GB x 2	9.7ms
w/o CMA, w/ SGT	15h 50m	7.9GB x 2	7ms
full	18h 20m	9.4GB x 2	9.7ms

Table 2: Comparison of training time, inference time and GPU memory requirements.

A.3. Training Time and GPU Memory Requirements

In Table 2, we present training time, inference time and GPU memory requirements compared with Monodepth2 [5]. All models are trained for 20 epochs with a batch size of 12 and input images are resized into a resolution of 192×640. Since both the CMA and the semantics-guided triplet loss (SGT) require additional memory, we need two GPUs for training. The inference time is estimated per each image with a resolution of 192×640. We also present various configurations that can be useful for comparison.

B. Additional Comparisons with Recent Works

In this section, we compare the proposed method with recent state-of-the-art results in a different way from that of the main paper. In Table 1 (main paper), all the results

Method	Backbone	Size	Sem	Lower is better				Higher is better		
				AbsRel	SqRel	RMS	RMSlog	< 1.25	$< 1.25^2$	$< 1.25^3$
Monodepth2 [5]	ResNet18	320×1024		0.115	0.882	4.701	0.190	0.879	0.961	0.982
PackNet-Sfm [6]	PackNet	384×1280		0.107	0.802	4.538	0.186	0.889	0.962	0.981
SGDepth [8]	ResNet18	384×1280	✓	0.107	0.768	4.468	0.186	0.891	0.963	0.982
SAFENet [1]	ResNet18	320×1024	✓	0.106	0.743	4.489	0.181	0.884	0.965	0.984
HR-Depth [9]	ResNet18	320×1024		0.106	0.755	4.472	0.181	0.892	0.966	0.984
HR-Depth [9]	ResNet18	384×1280		0.104	0.727	4.410	0.179	0.894	0.966	0.984
Ours	ResNet18	320×1024	✓	**0.102**	**0.687**	**4.366**	**0.178**	**0.895**	**0.967**	**0.984**

Table 3: Depth prediction results on the KITTI Egien test split. All methods are trained with high-resolution monocular images in the KITTI Eigen training split.

Method	Backbone	Train	Sem	Lower is better				Higher is better		
				AbsRel	SqRel	RMS	RMSlog	< 1.25	$< 1.25^2$	$< 1.25^3$
Monodepth2 [5]	ResNet18	K		0.090	0.545	3.942	0.137	0.914	0.983	0.995
Poggi *et al* [13]	ResNet18	K		0.087	0.514	3.827	0.133	0.920	0.983	0.995
PackNet-Sfm [6]	PackNet	K		0.086	0.505	3.746	0.132	0.919	0.983	0.995
SGDepth [8]	ResNet18	K	✓	0.085	0.487	3.757	0.130	0.921	0.984	0.996
HR-Depth [9]	ResNet18	K		0.085	0.472	3.768	0.130	0.920	0.985	0.996
Ours	ResNet18	K	✓	**0.083**	**0.439**	**3.627**	**0.126**	**0.924**	**0.986**	**0.996**

Table 4: Depth prediction results on 652 images of the KITTI Eigen test split. The results are evaluated with the KITTI improved ground-truth [15]

are from the models trained with the images of which the resolution is resized into 192×640, for fair comparisons. To further demonstrate that ours shows the best performance even with high-resolution images, we compare the results in Table 3. Please note that our result in Table 3 is the same as that of the last row in Table 1b (main paper).

Additionally, we report the results on the KITTI Eigen test split evaluated with improved ground-truth in Sec. B.2. We use the same model without re-training.

B.1. Comparison with High-resolution Images

As shown in Table 3, our proposed method trained with 320×1024 images shows the best result comparing with others, even including those trained with images of higher resolution (384×1280). Our method outperforms previous works in every metric and it demonstrates that it performs well in various resolutions.

B.2. Evaluation with KITTI Improved Ground-truth

To further demonstrate the performance of our work, we evaluate our model with a set of high quality depth map proposed by Uhrig *et al.* [15] without re-training our network, following evaluation protocol suggested in [5]. This ground-truth contains 652 of 697 test frames in KITTI Eigen test split [3], and it resolved occlusions and moving objects caused by re-projecting LIDAR points [5] appearing in KITTI raw dataset [4]. In Table 4, we compare ours with recent works on 652 images of the KITTI Eigen test split evaluated with improved ground-truth. For evaluation of SGDepth [8], PackNet-Sfm [6] and HR-Depth [9], we use the authors' publicly available off-the-shelf networks. We follow the evaluation process in Monodepth2 [5]; thus, we do not perform center crop in this case. Therefore, the result of PackNet-Sfm in Table 4 is different from that in the original paper [6]. Ours shows the state-of-the-art result compared with previous works as shown in Table 4.

C. Ablation Study

C.1. Performance on Different Semantic Labels

The original off-the-shelf segmentation network [16], used for generating semantic labels, has been fine-tuned with 200 images of KITTI dataset. This training process requires only 200 segmentation ground-truth for our target domain, the KITTI dataset, and this network can be used to generate semantic labels for 39,910 training set and 4,424 validation set of KITTI Eigen split. In other words, only tiny portion of ground-truth can significantly boost the depth prediction performance on the target dataset. To further verify the generalization-ability of the proposed method, we also train the network with semantic labels generated from another segmentation network [14]. This off-the-shelf network has been trained without any of images contained in the KITTI dataset. It has been pre-trained on Mapillary Vistas [11] and fine-tuned on Cityscapes [2] ground-truth. Table 5 compares depth prediction results of using different semantic labels generated from two differ-

Backbone	Seg. label	Lower is better				Higher is better		
		AbsRel	SqRel	RMS	RMSlog	< 1.25	$< 1.25^2$	$< 1.25^3$
ResNet18	[16]	0.105	0.722	4.547	0.182	0.886	0.964	0.984
ResNet18	[14]	0.106	0.717	4.529	0.182	0.886	0.964	0.983
ResNet50	[16]	0.102	0.675	4.393	0.178	0.893	0.966	0.984
ResNet50	[14]	0.103	0.688	4.449	0.179	0.892	0.966	0.984

Table 5: Depth prediction results on KITTI Eigen test split. The two variants are trained with different semantic labels generated from the off-the-shelf segmentation networks [16] and [14]. The former is used for our original implementation and it has been fine-tuned with 200 images of KITTI 2015. The latter has been trained without images from KITTI dataset.

ent networks. There is not much difference between those two cases, and the result for training segmentation with labels from [14] still shows the state-of-the-art performance. It demonstrates the proposed method can perform well even with no requirements of segmentation ground-truth for the target domain.

C.2. Class-specific Evaluation

We evaluate the class-specific performance of our proposed methods in Fig. 1. Our proposed methods show consistent improvements in every class compared with the baseline [5], and naive joint training (L_{CE}). In particular, the *full* model shows remarkable improvements especially on difficult the classes, e.g., pole, traffic light, traffic sign, rider, motorcycle, and person. These classes tend to be thin, small and distant, or they have complex shapes. This verifies the proposed method can detect precise and detailed object boundaries even for difficult cases, leading to more accurate depth predictions.

C.3. Ablations

In this section, we report the results of different configurations for the CMA module and the semantics-guided triplet loss. In Table 6, we compare the different layer configurations to which the CMA modules are applied. We chose $l = 0, 1, 2$ as it shows the best performance. Applying the CMA modules up to $l = 3$ and $l = 4$ degrades the performance. They have relatively lower channel dimensions; hence, the cross-task similarity cannot be computed accurately.

In Table 7, we compare the results of the different margins, m, for applying the semantics-guided triplet loss. We chose $m = 0.3$ for all experiments, as it shows the best performance.

In Fig. 2, we qualitatively show the improvements of each method we proposed. As shown in the 2^{nd} row, our baseline model produces poorly aligned depth predictions with the objects in the scenes, e.g., traffic light and road sign. After sharing the encoder and joint training with semantics (+L_{CE}), the predictions show improvements but they are still not that accurate. Each method we proposed, the semantics-guided triplet loss (L_{SGT}) and the CMA module, shows more improved depth predictions as shown in the 4^{th} and 5^{th} rows. They make each prediction more consistent with semantic labels. This verifies that the proposed methods can fully utilize semantic information for accurate depth predictions. Finally, our full method shows the best result, producing clearly consistent predictions with semantics.

l	Abs	Sq	< 1.25
0,1	0.109	0.776	0.882
0,1,2	0.107	0.741	0.884
0,1,2,3	0.110	0.744	0.880
0,1,2,3,4	0.110	0.718	0.877

Table 6: Depth prediction performance of the CMA module with different layer configurations.

m	Abs	Sq	< 1.25
0.1	0.110	0.783	0.880
0.2	0.109	0.772	0.881
0.3	0.108	0.755	0.882
0.4	0.108	0.739	0.881

Table 7: Depth prediction performance of semantics-guided triplet loss with different margins.

D. Additional Qualitative Evaluations

In Fig. 3 and Fig. 4, we additionally compare the visualizations of depth predictions performance with recent state-of-the-art methods, PackNet-Sfm [6], SGDepth [8] and HR-Depth [9]. All methods are trained with monocular images of size 192×640. We used authors' publicly available pre-trained network. As shown in the figures, ours shows outstanding performance especially on object boundaries, leading to more accurate depth predictions. Owing to the utilization of implicit semantic region information for depth predictions, our proposed method can produce consistent depth maps even in the weak-texture regions as shown in upper-right column of Fig. 3.

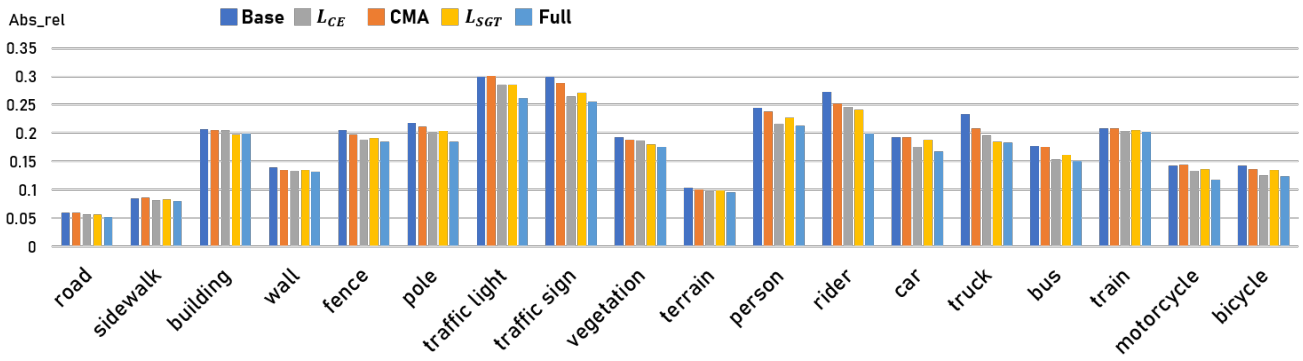

Figure 1: Class-specific performance of depth predictions after each method is applied.

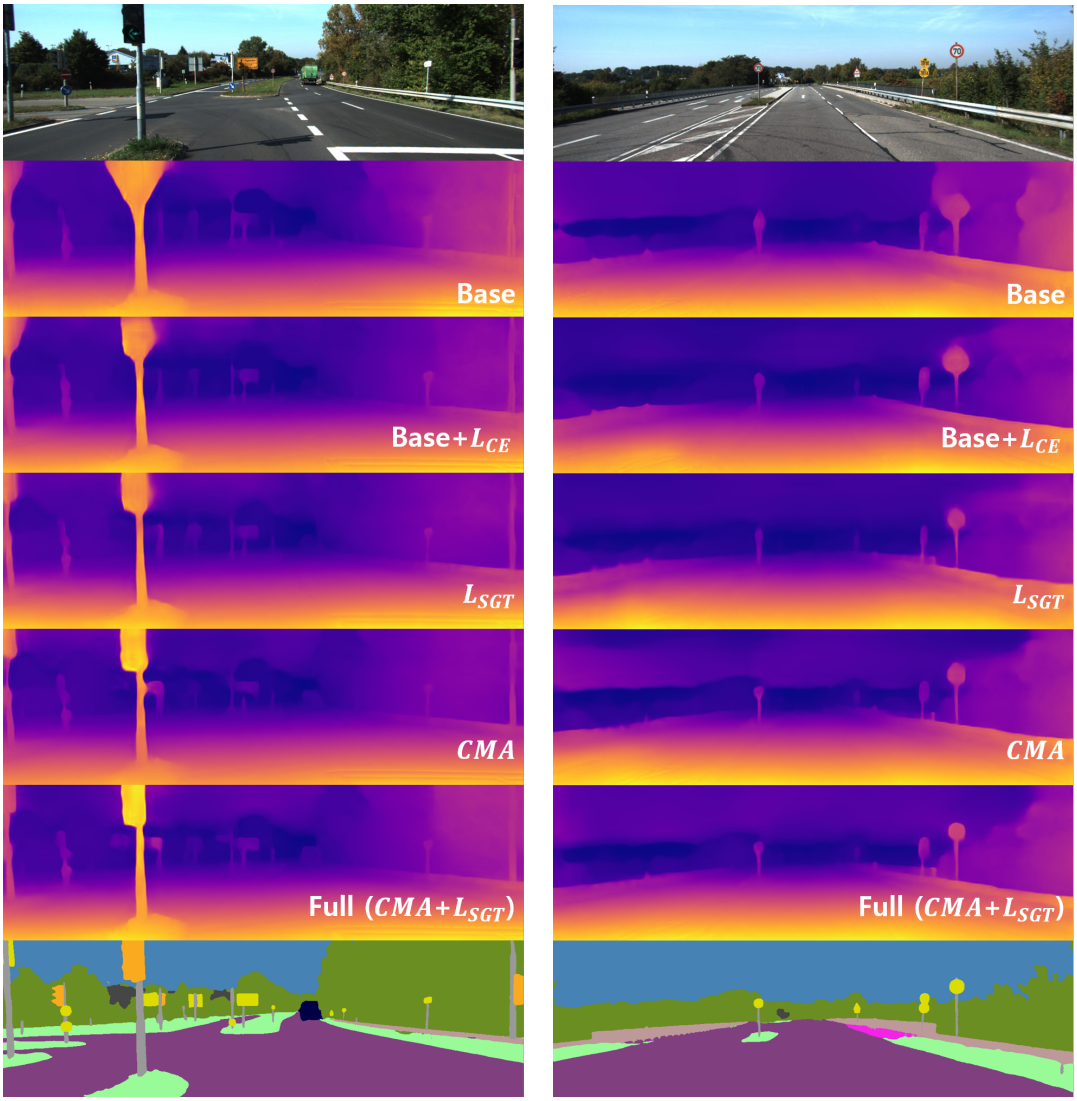

Figure 2: Qualitative results of depth predictions after each method is applied.

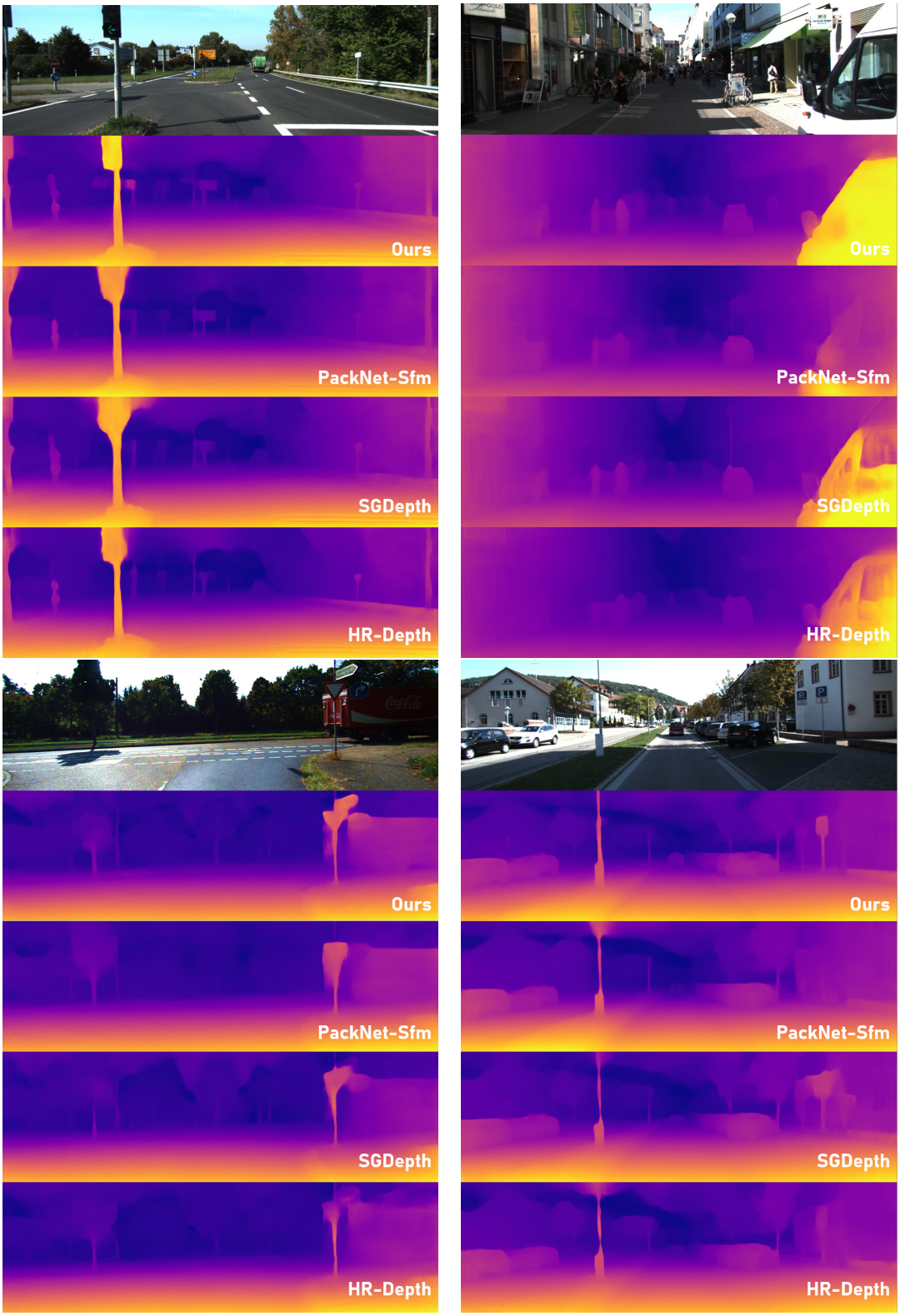

Figure 3: Qualitative comparison of the depth predictions with recent works, PackNet-Sfm [6], HR-Depth [9], and SGDepth [8].

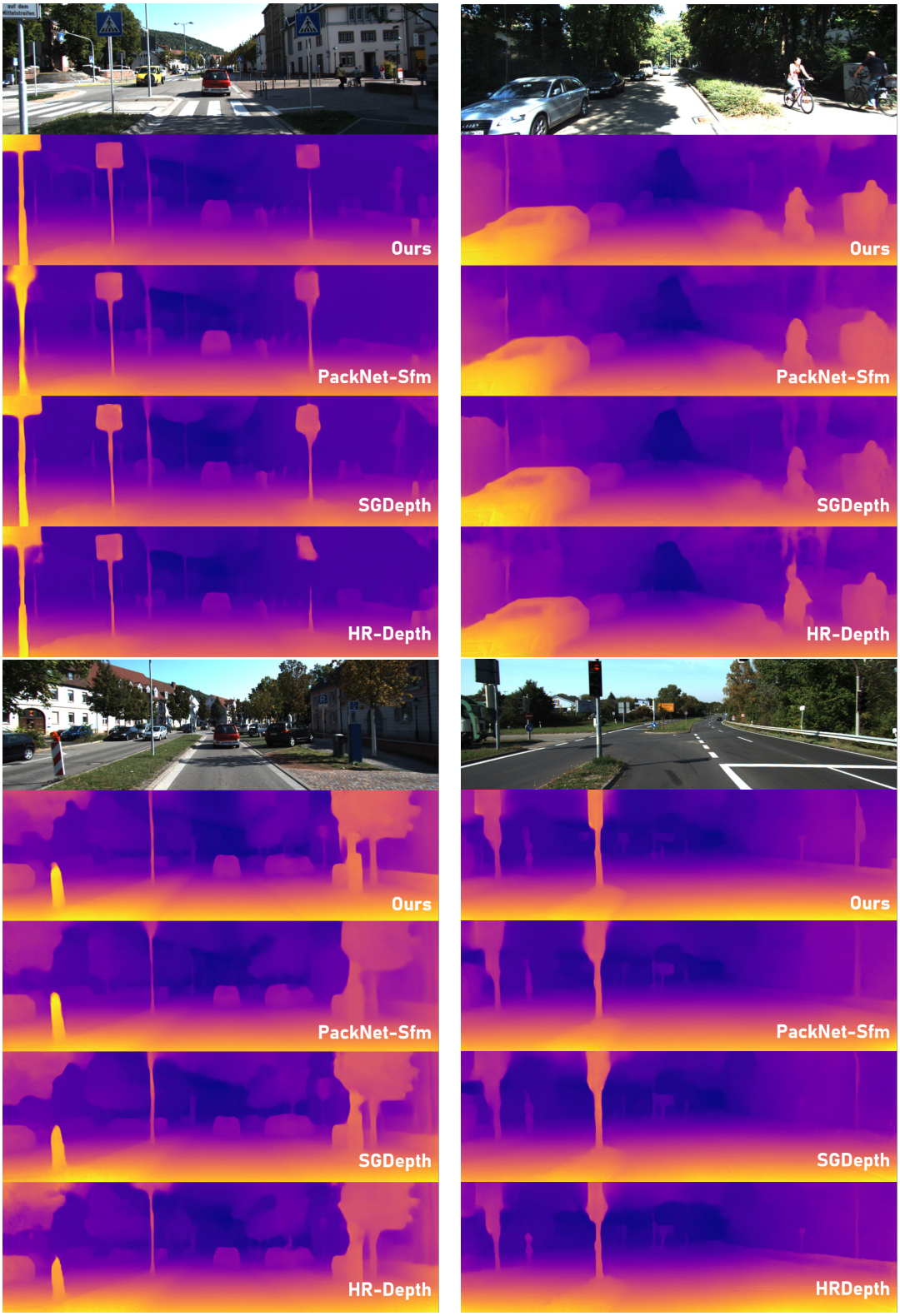

Figure 4: Qualitative comparison of the depth predictions with recent works, PackNet-Sfm [6], HR-Depth [9], and SGDepth [8].

References

[1] Jaehoon Choi, Dongki Jung, Donghwan Lee, and Changick Kim. Safenet: Self-supervised monocular depth estimation with semantic-aware feature extraction. *Neural Information Processing Systems Workshops (NeurIPSW)*, 2020.

[2] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[3] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *Neural Information Processing Systems (NeurIPS)*, 2014.

[4] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. *Computer Vision and Pattern Recognition (CVPR)*, 2012.

[5] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. *International Conference on Computer Vision (ICCV)*, 2019.

[6] V. Guizilini, Rares Ambrus, S. Pillai, and Adrien Gaidon. 3d packing for self-supervised monocular depth estimation. *Computer Vision and Pattern Recognition (CVPR)*, pages 2482–2491, 2020.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[8] Marvin Klingner, Jan-Aike Termöhlen, Jonas Mikolajczyk, and T. Fingscheidt. Self-supervised monocular depth estimation: Solving the dynamic object problem by semantic guidance. *European Conference on Computer Vision (ECCV)*, 2020.

[9] Xiaoyang Lyu, Liang Liu, Mengmeng Wang, Xin Kong, Lina Liu, Yong Liu, Xinxin Chen, and Yi Yuan. Hr-depth: High resolution self-supervised monocular depth estimation. *Artificial Intelligence (AAAI)*, 2021.

[10] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[11] G. Neuhold, T. Ollmann, S. R. Bulò, and P. Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. *International Conference on Computer Vision (ICCV)*, 2017.

[12] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[13] Matteo Poggi, Filippo Aleotti, Fabio Tosi, and Stefano Mattoccia. On the uncertainty of self-supervised monocular depth estimation. *Computer Vision and Pattern Recognition (CVPR)*, 2020.

[14] Andrew Tao, Karan Sapra, and Bryan Catanzaro. Hierarchical multi-scale attention for semantic segmentation. *arXiv:2005.10821*, 2020.

[15] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. *International Conference on 3D Vision (3DV)*, 2017.

[16] Yi Zhu, Karan Sapra, Fitsum A. Reda, Kevin J. Shih, Shawn D. Newsam, Andrew Tao, and Bryan Catanzaro. Improving semantic segmentation via video propagation and label relaxation. *Computer Vision and Pattern Recognition (CVPR)*, 2019.