RendBEV: Semantic Novel View Synthesis for Self-Supervised Bird's Eye View Segmentation

Henrique Piñeiro Monteagudo[1,2] Leonardo Taccari[1] Aurel Pjetri[1,3] Francesco Sambo[1] Samuele Salti[2]

[1]Verizon Connect, Italy [2] University of Bologna, Italy [3] University of Florence, Italy

https://henriquepm.github.io/RendBEV/

- Supplementary Material -

In this supplementary material, we present additional explanations, experiments and results to complement the main paper.

1. SkyEye's Methodology

In this section we present a brief overview of SkyEye's work and its key differences with our proposal. We would like to distinguish between their proposed network architecture (which we use in our experiments) and their proposed training framework (which we compare against). For a more detailed report, we refer the interested reader to the original paper by Gosala et al. [1].

1.1. Network Architecture

SkyEye's model is composed of four main items. a) A 2D image encoder that extracts features from the input images. Chosen to be Efficient-D3's backbone in SkyEye's main experiments (and ours). b) A lifting module which populates a 3D voxel grid with features. c) A frontal view semantic head used in their implicit supervision. This module is not used in our experiments. d) A BEV semantic segmentation head. Additionally, an independent depth network is used to generate the the pseudolabels for their explicit supervision. This depth network is not used in our experiments.

1.2. Training Framework

SkyEye's proposed training framework comprises two stages: a pretraining referred to as implicit supervision and a final stage in which the actual BEV semantic segmentation head is trained: the explicit supervision. We give an overview of these two types of supervision and then contrast with our proposed training framework, highlighting differences and similarities.

Implicit Supervision. In the implicit supervision stage, the supervision signal provided by static elements in the scene is exploited by enforcing consistency. This is done by predicting the semantic segmentation of future timestamps in frontal view using only the 3D features computed from the initial frame. A cross-entropy loss is computed between target frontal view semantic segmentation labels and predicted values. A weight factor modulates the contribution of each frame from the sequence, linearly decaying from 1 to 0.2. **Explicit Supervision.** During the implicit supervision stage, the BEV segmentation head is not trained. In order to circumvent the necessity of training the network with GT BEV annotations, SkyEye's authors propose a pseudolabel generation pipeline. This pipeline is based on an independent depth-from-mono network, a DBSCAN-based instance generation module and a densification module based on morphological operations.

Comparison with RendBEV. Similarly to SkyEye, RendBEV also tres to exploit spatiotemporal consistency with a static scene assumption to train a BEV segmentation network. However, two main differences with our method exist, one conceptual and another methodological. At the conceptual level, our method can be run without the explicit supervision stage, avoiding the usage of BEV labels or pseudolabels. This is achieved with a methodological difference in the way of providing "self-supervision" to the network. Instead of *predicting* the semantic segmentation of future timesteps with features generated from the initial timestamp (which enables SkyEye's pretraining method to supervise part of their network but not their BEV semantic segmentation head), we *render* the semantic segmentation of future timestemps, using class probability values sampled from the output of the BEV network. This lets gradient flow through the BEV semantic segmentation head already at this stage.

This difference grants the possibility to train models in a setting where no BEV supervision in any form is available and gives a good starting point for training if some GT BEV labels are available. We hypothesize that the performance gains (especially at low annotation regimes) when fine-tuning on GT BEV labels, are thanks to the capacity of our method to already provide supervision in the previous step to the semantic segmentation head and thus having a more advantageous starting point with respect to training it from scratch.

2. Experiments with Simple-BEV

We execute a supplementary set of experiments to validate RendBEV with a different architecture for the BEV semantic segmentation network. To this end, we modify Simple-BEV [2] and adapt it to our setting, by making it work with monocular frontal images only, increasing the number of classes in its semantic segmentation head and removing the auxiliary task heads. We train the network using the RendBEV method and then fine-tune the model at different percentage splits of the dataset. To provide a baseline comparison, we train the same model from scratch on the same splits. We present the results obtained in these experiments in Tab. S1. The performance we reach while using RendBEV as a standalone training is slightly inferior to the one obtained with SkyEye's architecture as BEV semantic segmentation network, but still competitive. When fine-tuning on available ground truth, RendBEV proves to be useful as pretraining in the lower annotation regimes. When the amount of ground truth data is high (in the 50% and 100% splits) the pretrained models obtain overall performances almost equal to the ones trained from scratch in terms of mIoU.

3. Pretraining with GT FV SS

We perform additional experiments by fine-tuning the model obtained with RendBEV using ground truth semantic segmentation labels instead of model predictions on 0.1%, 1%, 10%, 50% and 100% of the training data. We compare the performance of the same architecture pretrained with SkyEye's method and trained from scratch. We report the results in Tab. S2. We observe that in this setting the model pretrained with RendBEV performs similarly as the one pretrained using model predictions as targets, while SkyEye's results improve by a higher margin. Even with SkyEye's improvement with the usage of GT labels, our method continues to provide better results in low-data regimes, while the difference dissipates in models fine-tuned on 10% of the data (in the order of 2000 images) and the models pretrained with SkyEye's methodology perform slightly better on higher BEV GT data regimes.

4. Additional Qualitative Results

We present additional qualitative results from our experiments. In Fig. S1 we provide a comparison of the results obtained with the network from SkyEye and Simple-BEV training in a self-supervised way following our method.

5. Experimental Details

In this section we provide further details on the experiment configurations and the hardware used to run those experiments.

We feed the BEV network with frames of resolution 1408×384, while for Behind the Scenes we resize the images to a resolution of 640×192 used in the original paper [5]. We use a BEV resolution of 768×704, which corresponds to a real world area of $56.83m \times 52.096m$ in front of the vehicle.

In our experiments, when using a class-weighted cross entropy loss, we use the class weights proposed in [1]. When sampling 3D points along rays, we sample in a total of $m = 64$ points on each ray with $z_{near} = 3\text{m}$ and $z_{far} = 80\text{m}$.

For our self-supervised training, we use a batch of 5 sequences. For each sequence, we sample a total of 192 patches of 16×16 pixels randomly distributed across 7 other frames of the sequence, with timestamps $T = \{r-1, r+1, r+o_1, \dots, r+o_5\}$, where each temporal offset o_k is selected in a random uniform way from ranges of length 7 starting from $r+5$. The goal of this selection is to provide a good coverage in different regions of the BEV, as discussed at the end of Sec. 3 and shown in Fig. 3 of the main paper. We train for 20 epochs and use SGD as optimizer with Nesterov momentum 0.9, weight decay 0.00001 and learning rate 0.005.

In terms of hardware, we perform most of our experiments in a machine equipped with a NVidia V100 GPU with 32GB of VRAM. The self-supervised training experiments with 196 patches per sequence take approximately 8 days to complete in a single machine. The neural network architecture proposed in SkyEye [1], which we use in our main experiments has 14.6 million parameters and a runtime of 77.84 ms for a forward pass in inference.

6. Ethical considerations

In this section we address potential ethical implications of our work. We would like to focus on two main topics: data and possible misuse.

In terms of data, we don't introduce any new dataset and use for our experiments two publicly available datasets:

Table S1. Study of the performance of our method with Simple-BEV as BEV semantic segmentation network at different annotated data regimes. All scores are reported in the KITTI-360 dataset.

BEV (%)	Pretraining	Road	Sidewalk	Building	Terrain	Person	2-Wheeler	Car	Truck	mIoU
0.0	RendBEV	65.46	30.30	29.49	38.46	1.94	2.49	30.92	7.17	25.78
0.1	–	45.78	14.01	11.35	4.22	0.12	0.25	5.87	4.60	10.26
	RendBEV	**67.19**	**32.60**	**32.39**	**39.11**	**1.92**	**2.69**	**32.30**	**7.60**	**26.98**
1	–	57.45	23.16	19.34	21.37	0.06	0.11	18.20	1.52	17.65
	RendBEV	**68.84**	**34.73**	**32.76**	**38.66**	**2.18**	**3.07**	**34.27**	**5.18**	**27.46**
10	–	70.42	34.37	30.28	35.36	0.3	0.84	34.43	**10.03**	27.00
	RendBEV	**70.66**	**36.13**	**36.34**	**40.02**	**1.66**	**4.91**	**35.80**	5.74	**28.90**
50	–	**72.05**	35.51	34.92	37.36	1.01	1.51	**38.59**	**11.64**	29.07
	RendBEV	70.70	**36.00**	**36.73**	**40.38**	**1.72**	**5.17**	36.63	6.07	**29.18**
100	–	**70.66**	35.50	34.67	**41.18**	1.04	2.11	**38.27**	**12.42**	**29.48**
	RendBEV	70.40	**36.18**	**36.73**	41.17	**1.64**	**5.43**	36.65	6.32	29.32

Table S2. Impact of the pretraining (with GT PV) on BEV semantic segmentation performance using the network proposed in SkyEye on different data regimes. SkyEye results from [1], RendBEV and no pretraining run by us on same splits. All scores are reported on the KITTI-360 dataset.

BEV GT (%)	Pretraining	Road	Sidewalk	Building	Terrain	Person	2-Wheeler	Car	Truck	mIoU
0.1	SkyEye	68.78	28.20	35.56	26.08	0.00	0.00	21.61	0.00	22.53
	RendBEV	**72.15**	**37.81**	**36.70**	**46.65**	**2.62**	**3.99**	**34.56**	**6.03**	**30.07**
	–	56.43	19.95	23.64	7.17	0.00	0.00	12.59	0.00	14.97
1	SkyEye	72.56	34.33	36.70	41.66	0.00	0.16	33.85	**10.29**	28.71
	RendBEV	**75.33**	**39.29**	**38.44**	**46.74**	**3.03**	**3.95**	**38.93**	8.91	**31.82**
	–	61.01	22.68	27.81	23.69	0.00	0.00	31.31	6.32	21.60
10	SkyEye	**76.07**	40.30	40.30	45.33	3.75	**8.15**	42.64	10.73	**33.41**
	RendBEV	75.90	**40.88**	**41.06**	**47.03**	2.44	6.79	43.24	8.40	33.22
	–	73.39	37.49	35.87	40.30	**4.72**	7.44	**44.64**	**12.23**	32.01
50	SkyEye	**76.43**	39.89	**45.22**	46.64	**5.10**	**7.93**	42.43	12.30	**34.49**
	RendBEV	74.69	40.15	42.16	**47.22**	3.30	6.78	44.88	9.77	33.61
	–	75.30	**40.61**	41.79	45.34	2.88	6.64	**45.52**	**13.46**	**33.94**
100	SkyEye	**75.99**	**41.35**	**44.26**	45.91	4.08	9.53	44.13	**12.68**	**34.74**
	RendBEV	75.11	40.32	42.25	**47.55**	2.91	6.89	44.19	8.51	33.47
	–	73.01	37.78	39.15	43.68	**5.44**	**10.76**	**45.41**	12.25	33.72

the KITTI-360 dataset [3] and the Waymo dataset [4] as well as their BEV derivations provided by the authors of SkyEye [1]. The KITTI-360 dataset is shared under a CC BY-NC-SA 3.0 License, while the Waymo dataset is shared under the Waymo Dataset License Agreement for Non-Commercial Use. The BEV version of these datasets are licensed under a non-commercial RL License Agreement. We credit the original authors for the creation of these datasets. For these datasets, appropriate measures (e.g. the blurring of faces and license plates) have been taken in order to respect individual privacy rights: the KITTI-360 dataset is GDPR-compliant and thus provides extensive privacy protection and the Waymo dataset as per their original authors, was modified to protect individuals' privacy.

In terms of potential misuse, we note that the methodology and models described in this work are research artifacts, not intended for their deployment as-is in safety critical applications like autonomous driving given the limitations described in the main paper.

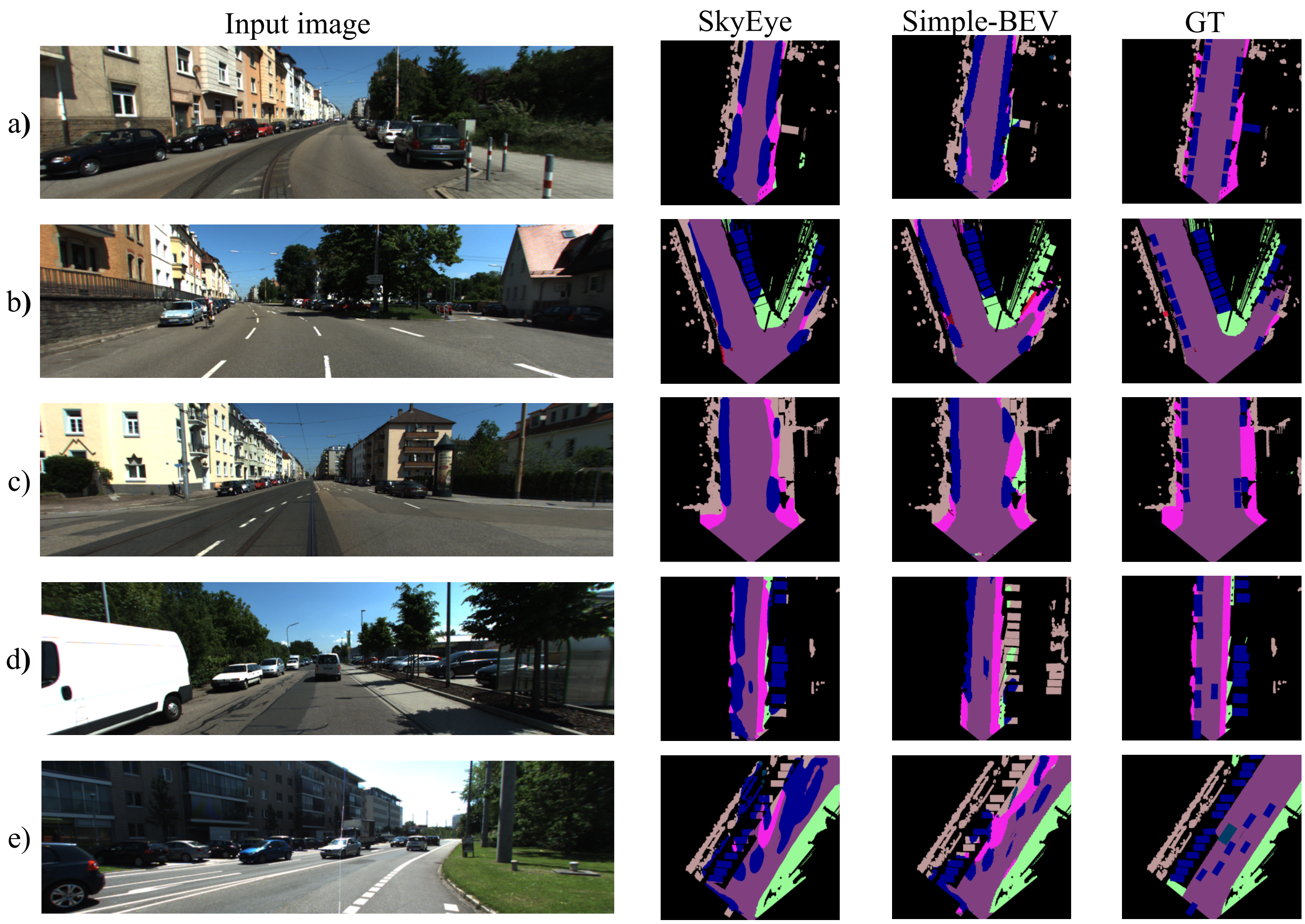

Figure S1. Qualitative comparison of the results obtained with RendBEVusing the architectures from SkyEye and Simple-BEV

References

[1] Nikhil Gosala, Kürsat Petek, Paulo L. J. Drews-Jr, Wolfram Burgard, and Abhinav Valada. SkyEye: Self-Supervised Bird's-Eye-View Semantic Mapping Using Monocular Frontal View Images. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14901–14910, Vancouver, BC, Canada, June 2023. IEEE. 1, 2, 3

[2] Adam W. Harley, Zhaoyuan Fang, Jie Li, Rares Ambrus, and Katerina Fragkiadaki. Simple-BEV: What Really Matters for Multi-Sensor BEV Perception? In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2759–2765, May 2023. 2

[3] Yiyi Liao, Jun Xie, and Andreas Geiger. KITTI-360: A Novel Dataset and Benchmarks for Urban Scene Understanding in 2D and 3D. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3292–3310, Mar. 2023. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence. 3

[4] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 3

[5] Felix Wimbauer, Nan Yang, Christian Rupprecht, and Daniel Cremers. Behind the Scenes: Density Fields for Single View Reconstruction . In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9076–9086, Los Alamitos, CA, USA, June 2023. IEEE Computer Society. 2