Supplmentary Material for
BEV-Guided Multi-Modality Fusion for Driving Perception

Yunze Man
UIUC
yunzem2@illinois.edu

Liang-Yan Gui
UIUC
lgui@illinois.edu

Yu-Xiong Wang
UIUC
yxw@illinois.edu

Appendix

1. Dataset and Implementation Details

Following [11], we use EfficientNet [14] pretrained on ImageNet [1] as our image encoder backbone. Two heads are applied to estimate pixel features and pixel-wise depth distribution from the $8\times$ downsampled feature map. The 3D feature maps are projected to the bird's-eye-view frame using mean pooling. For the bird's-eye-view decoder we use ResNet-18 [3] as backbone, and upsample the features learned from the first three meta-layers of ResNet to the final BEV output. The D_1 and D_2 domain discriminators are applied to the output feature layers of EfficientNet and ResNet backbone, respectively. We use a light weight discriminator architecture, which is composed of a global averaging pooling layer, followed by two fully connected layers, and outputs the domain label. For input, we resize and crop input images to size 128×352. For output, we consider a 100 meters $\times$ 100 meters range centered at the ego-vehicle, with the grid size set to be 0.5 meters $\times$ 0.5 meters. The depth bin is set to be 1.0 meter between 4.0 meters and 45.0 meters range. The whole model is trained end-to-end, with $\lambda_{\mathrm{T}} = 1.0, \lambda_{\mathrm{dp}} = 0.05, \lambda_{\mathrm{D}_1} = 0.1, \lambda_{\mathrm{D}_2} = 0.01$. We train CroMA using the Adam [5] optimizer with learning rate 0.001 and weight decay $1e$-7 for 50K steps for the teacher model, and 200K for the student model. We use horizontal flipping, random cropping, rotation, and color jittering augmentation during training. The whole model is implemented using the PyTorch framework [10].

Following [11, 15], we use EfficientNet [14] pretrained on ImageNet [1] as our image backbone encoder. We use pointpillars as our Lidar backbone [6], and use the projection based Radar backbone as described in Sec. 3.1. We downsample the camera images to 28×60, 1/8 of the input size. The Lidar and Radar feature embeddings are both interpolated to 200×200 size in the BEV frame. We use 4-head attention blocks with embedding of 64 channels. The decoder is composed of three $2\times$ bilinear-upsample layers, each followed by a convolution layer to obtain the final output map of the desired size.

Table A. (1) BEVGuide achieves best performance on 3D detection; (2) Performance improvement comes from both our BEV-guided multi-sensor fusion strategy and leveraging the overlooked Radar sensor; (3) Stronger backbone leads to better performance.

Method	Modality	Backbone	mAP↑	NDS↑	mAVE↓
FUTR3D	C+R	ResNet-101	35.0	45.9	0.56
BEVGuide			**42.1**	**53.7**	**0.39**
BEVFusion	C+L	Swin-T	68.5	71.4	-
BEVGuide			**68.9**	71.4	**0.25**
BEVGuide	C+R+L	EfficientNet	67.9	70.0	0.24
		ResNet-101	69.0	**71.6**	0.22
		Swin-T	**69.3**	71.5	**0.21**

We train our model with a combination of focal loss [7] for semantic segmentation and a ℓ_2 loss for velocity estimation task. We optimize the model with AdamW [9], learning rate 4e-3, and weight decay 1e-7. The model is trained on a 8-V100 machine with batch size 4 for 40 epochs.

2. Additional Results

Results on 3D Object Detection. In addition to 3D BEV semantic segmentation, we also conduct experiments on 3D object detection in Table A, which further demonstrate the effectiveness and generality of our method. Here we show comparison against strongest prior work [8] with mean Average Precision (mAP), Nuscenes Detection Score (NDS), and mean Average Velocity Error (mAVE) metrics, on the nuScenes validation set *without* test time augmentation (TTA). Results show that BEVGuide achieves **leading results** on detection. Our method also significantly improves the velocity metric mAVE, validating we exploit and benefit from the additional Radar sensor.

Meanwhile, we have chosen to conduct experiments on BEV scene segmentation task following existing work OFT [12], Lift-Splat [11], FIERY [4], and CVT [15]. This task requires a more comprehensive understanding of the surrounding environment, encompassing not only vehicles, but roads, lanes, and other possible elements. Consequently, we believe that combining BEV scene segmentation and 3D detection tasks further showcases the perceptual capabilities of our model.

Results with Different Backbones. We also show in Table A results with different backbones. From the last three rows, BEVGuide benefits from a stronger backbone, which extracts better sensory features for our sensor fusion. *Even with a more lightweight backbone (EfficientNet-b4)*, BEVGuide outperforms existing methods with stronger backbones *(Swin-T and ResNet-101)*. Note that, we keep the input size the same; on the other hand, the pretrained models and training schedules have to vary so to be optimized for different backbones. The result validates (1) the consistent effectiveness of our method on various perception tasks; (2) our improvement is due to our proposed method (e.g., attention module); and (3) our method can further benefit from stronger backbones.

3. Inference Time

BEVGuide (Camera+Lidar) with Swin-T backbone runs 114.5ms per sample (8.7 FPS), which is on par with BEVFusion running 119.2ms per sample. Our best-performing model taking all three types of sensors as input runs 143.1ms per sample (7 FPS). The efficiency of our model comes from the small number of transformer layers and the relatively small feature maps that are passed into the transformer layers.

4. More Discussion on Model Design Choices

Integrating Multiple Frames. Following LiRaNet [13], and Simple-BEV [2], we aggregate multiple frames of Radar and Lidar data by calibrating and combining multiple frames of their point clouds into a joint one as the model input.

Calibration Assumption. BEVGuide can be easily extended to a multi-frame setting if the temporal calibration assumption holds, as we can concatenate multiple sensory frames as multiple channels for the sensor-specific feature map. However, if the calibration is not guaranteed, we need to design a temporal matching and/or interpolation module before fusion. This is a challenging setting for all existing methods, and we leave it as future work.

Sensor Failure Simulation. To simulate the sensor malfunction, we can either set the sensory input to zero or set the sensory feature map to zero. We believe setting "sensor specific features" to zeros is the valid setting and the right solution. Setting input to zero will cause the model to generate random noisy feature map, which can harm the feature fusion process. Practically, we can design the system such that when a sensor is offline, the hardware sends a signal to our model which asks the feature maps to be changed to zeros.

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 1

[2] Adam W Harley, Zhaoyuan Fang, Jie Li, Rares Ambrus, and Katerina Fragkiadaki. Simple-bev: What really matters for multi-sensor bev perception? *arXiv*, 2022. 2

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

[4] Anthony Hu, Zak Murez, Nikhil Mohan, Sofía Dudas, Jeffrey Hawke, Vijay Badrinarayanan, Roberto Cipolla, and Alex Kendall. Fiery: Future instance prediction in bird's-eye view from surround monocular cameras. In *ICCV*, 2021. 1

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

[6] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *CVPR*, 2019. 1

[7] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 1

[8] Zhijian Liu, Haotian Tang, Alexander Amini, Xingyu Yang, Huizi Mao, Daniela Rus, and Song Han. Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. *arXiv*, 2022. 1

[9] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *ICLR*, 2017. 1

[10] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 1

[11] Jonah Philion and Sanja Fidler. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *ECCV*, 2020. 1

[12] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3d object detection. *arXiv preprint arXiv:1811.08188*, 2018. 1

[13] Meet Shah, Zhiling Huang, Ankit Laddha, Matthew Langford, Blake Barber, sida zhang, Carlos Vallespi-Gonzalez, and Raquel Urtasun. Liranet: End-to-end trajectory prediction using spatio-temporal radar fusion. In *CoRL*, 2021. 2

[14] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019. 1

[15] Brady Zhou and Philipp Krähenbühl. Cross-view transformers for real-time map-view semantic segmentation. In *CVPR*, 2022. 1