CoBEVT: Cooperative Bird's Eye View Semantic Segmentation with Sparse Transformers (Supplementary Materials)

Runsheng Xu[1]*, **Zhengzhong Tu**[2]*, **Hao Xiang**[1], **Wei Shao**[3], **Bolei Zhou**[1], **Jiaqi Ma**[1]†
[1] University of California, Los Angeles, [2] University of Texas at Austin
[3] University of California, Davis

Abstract: In this supplementary material, we will first provide more details about the camera track of the OPV2V dataset (Sec. 1). Afterward, the model details of the proposed FAX attention and implementation details of our CoBEVT models on different datasets will be illustrated in Sec. 2 and Sec. 3. Finally, we show more qualitative results for all three tasks tested in the main paper in Sec. 4.

1 The Camera Track of OPV2V dataset

Sensor Configuration. In OPV2V, every AV is equipped with 4 cameras toward different directions to cover 360° surroundings as Fig. 1 shows. Each camera has an 800×600 spatial resolution and 110° FOV, which introduces a 10° view overlap between any neighboring pair.

Groundtruth. The BEV semantic segmentation groundtruth mask has a pixel resolution of 256×256 and covers a 100×100 m area around the ego vehicle, which represents a map sampling resolution of 0.39 $m/pixel$. The authors also provide corresponding visible masks, where all dynamic objects that can be seen by any AV's camera rigs are marked as visible, and vice versa for the invisible. Similar to previous works [1, 2], we only consider objects that are visible during both training and testing.

2 Model Details

We give more details about the proposed 3D fused axial attention (FAX) below.

3D Relative Attention. The vanilla attention mechanism defined in [3] is a global mixing operator based on the weighted sum of all the spatial locations, whereas the weights are calculated by normalized pairwise similarity. Formally, the attention operator can be defined as

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}})\mathbf{V} \tag{1}$$

where the $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ are the query, key, and value matrices projected from the input tensor. Multi-head attention is an extension of (1) in which we split the channels into multiple "heads", in parallel, and run attention on each head separately. Here for simplicity, we only use a single-head equation, but we always use multi-head variants in the actual implementations.

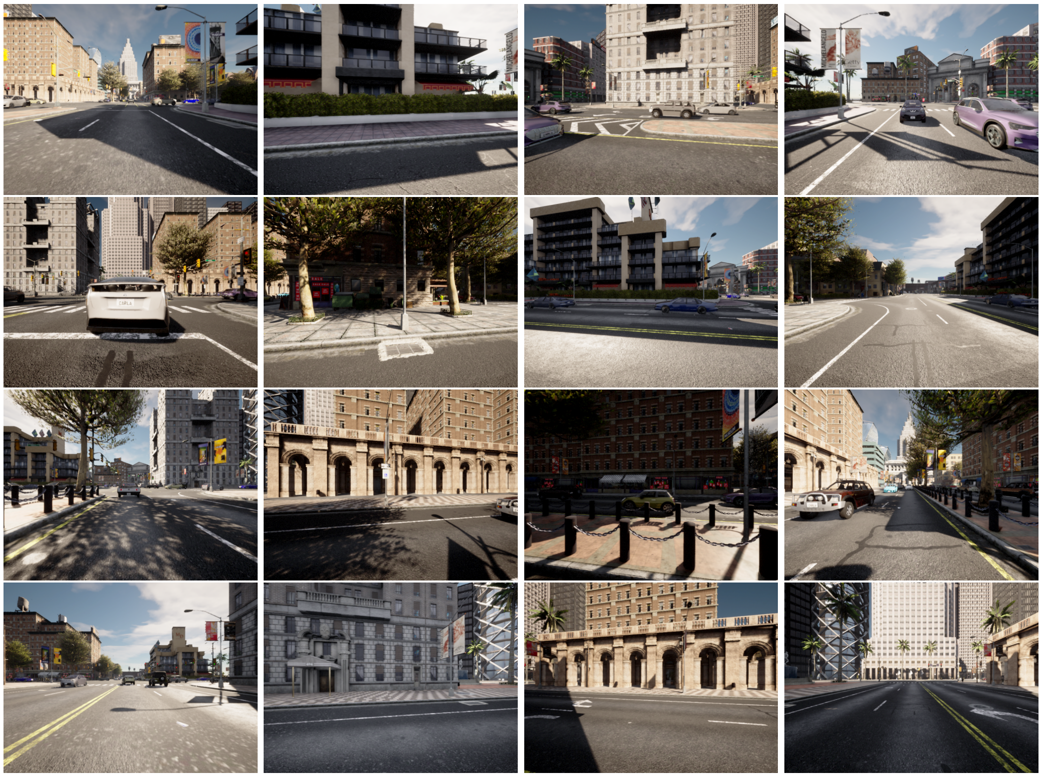
The 3D relative attention we adopt in CoBEVT is an improved attention with the relative positional encoding added in the 3D space. Given a 3D input tensor $\mathbf{z} \in \mathbb{R}^{(N \times H \times W) \times C}$, the 3D relative attention can be expressed as:

$$\text{3D-Rel-Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}} + \mathbf{B})\mathbf{V}, \tag{2}$$

where $\mathbf{B}$ is the relative position bias, whose values are taken from $\hat{\mathbf{B}} \in \mathbb{R}^{(2N-1) \times (2H-1) \times (2W-1)}$ with learnable parameters [4, 5].

*Equal contribution. †Corresponding author: `jiaqima@ucla.edu`.

6th Conference on Robot Learning (CoRL 2022), Auckland, New Zealand.



(a) front (b) right (c) left (d) back

Figure 1: **An example of the four cameras of different AVs in the same intersection.** Each row represents the full views of an AV. From left to right: (a) front camera, (b) right camera, (c) left camera, (d) back camera.

3D FAX Attention. We assume that the above defined 3D-Rel-Attention in Eq. (2) follows the convention of 1D input sequence, *i.e.*, always regard the second last dimension of an input as the "spatial axis". The proposed FAX attention can be implemented without modifications to the attention operator. We first define the Fused-Block$(\cdot)$ operator with parameter P as partitioning the input 3D feature $\mathbf{x} \in \mathbb{R}^{N\times H\times W\times C}$ into non-overlapping 3D windows each having window size $N \times P \times P$. Note that after window partitioning, we gather all the spatial dimensions in the so-called "spatial axis":

$$\textsf{Fused-Block}: (N, H, W, C) \rightarrow (N, \frac{H}{P} \times P, \frac{W}{P} \times P, C) \rightarrow (\frac{HW}{P^2}, \underbrace{N \times P^2}_{\text{“spatial axis”}}, C). \tag{3}$$

We then denote the Fused-Unblock$(\cdot)$ operation as the reverse of the above 3D window partition procedure. Likewise, for the global attention branch, we define another 3D grid partitioning operator as Fused-Grid with the grid parameter G, representing dividing the input feature using a uniform 3D grid of size $N \times G \times G$. Note that unlike Eq. (3), we need to apply an extra Transpose to place the grid dimension in the assumed "spatial axis":

$$\textsf{Fused-Grid}: (N, H, W, C) \rightarrow (N, G \times \frac{H}{G}, G \times \frac{W}{G}, C) \rightarrow \underbrace{(N \times G^2, \frac{HW}{G^2}, C) \rightarrow (\frac{HW}{G^2}, N \times G^2, C)}_{\text{swapaxes(axis1=-2,axis2=-3)}} \tag{4}$$

with its inverse operator Fused-Ungrid that reverses the 3D-gridded input back to the original tensor shape.

Now we are ready to present the whole 3D FAX attention module. The 3D local block attention can be expressed as:

$$\begin{aligned} \mathbf{x} &\leftarrow \mathbf{x} + \mathsf{Fused\text{-}Unblock}(\mathsf{3D\text{-}Rel\text{-}Attention}(\mathsf{Fused\text{-}Block}(\mathsf{LN}(\mathbf{x})))) \\ \mathbf{x} &\leftarrow \mathbf{x} + \mathsf{MLP}(\mathsf{LN}(\mathbf{x})) \end{aligned} \tag{5}$$

while the sparse global 3D Attention can be expressed as:

$$\begin{aligned} \mathbf{x} &\leftarrow \mathbf{x} + \mathsf{Fused\text{-}Ungrid}(\mathsf{3D\text{-}Rel\text{-}Attention}(\mathsf{Fused\text{-}Grid}(\mathsf{LN}(\mathbf{x})))) \\ \mathbf{x} &\leftarrow \mathbf{x} + \mathsf{MLP}(\mathsf{LN}(\mathbf{x})) \end{aligned} \tag{6}$$

where the $\mathbf{QKV}$ matrices in Eq. (2) are linearly projected from input $\mathbf{x}$ and are omitted for simplicity. LN denotes the Layer Normalization [6], where MLP is a standard MLP network [7, 4] consisting of two linear layers applied on the channel: $\mathbf{x} \leftarrow W_2\mathrm{GELU}(W_1\mathbf{x})$.

3 Implementation Details

In the following, we show the detailed architectures for the three experiments, respectively.

3.1 OPV2V Camera Track

We illustrate the architectural specifications of CoBEVT in Table A2. Further illustrations are presented below.

Table A2: Detailed architectural specifications of CoBEVT for OPV2V camera track. M represents the number of cameras and N is the number of agents.

	Output size	CoBEVT framework
ResNet34 Encoder	$N \times M \times 64 \times 64 \times 128$	[ResNet34-layer1]
	$N \times M \times 32 \times 32 \times 256$	[ResNet34-layer2]
	$N \times M \times 16 \times 16 \times 512$	[ResNet34-layer3]
SinBEVT Backbone	$N \times 128 \times 128 \times 128$	[FAX-CA, dim 128, head 4, bev win. sz.$\{16 \times 16\}$ feat win. sz.$\{8 \times 8\}$ MLP, dim 256 Res-Bottleneck-block ×2] × 1
	$N \times 64 \times 64 \times 128$	[FAX-CA, dim 128, head 4, bev win. sz.$\{16 \times 16\}$ feat win. sz.$\{8 \times 8\}$ MLP, dim 256 Res-Bottleneck-block ×2] × 1
	$N \times 32 \times 32 \times 128$	[FAX-CA, dim 128, head 4, bev win. sz.$\{32 \times 32\}$ feat win. sz.$\{16 \times 16\}$ MLP, dim 256 Res-Bottleneck-block ×2] × 1
FuseBEVT Backbone	$N \times 32 \times 32 \times 128$	[FAX-SA, dim 128, head 4, win. sz.$\{8 \times 8\}$ MLP, dim 256] × 3
Decoder	$64 \times 64 \times 128$	[Bilinear-upsample, Conv3x3, BN]
	$128 \times 128 \times 64$	[Bilinear-upsample, Conv3x3, BN]
	$256 \times 256 \times 32$	[Bilinear-upsample, Conv3x3, BN]
	$256 \times 256 \times k$	Dyna. Obj. head: [Conv1x1, 2, stride 1] Stat. Obj. head: [Conv1x1, 3, stride 1]

Model Separation. Same as [1, 2, 8, 9], we have separate models for dynamic objects and static layout BEV semantic segmentation. Both models have the same configurations except for the last layer in the network.

Image Encoder. We first resize the input images to 512×512 and utilize ResNet34 [10] to extract image features. We then take the outputs $I_0 \in \mathbb{R}^{4\times64\times64\times128}$, $I_1 \in \mathbb{R}^{4\times32\times32\times256}$, and $I_2 \in$

$\mathbb{R}^{4\times16\times16\times512}$ from the *layer1*, *layer2*, and *layer3* to interact with the BEV query, where 4 is the number of cameras..

SinBEVT. The BEV query $Q_0 \in \mathbb{R}^{H\times W\times C}$ is a learnable embedding, where $H, W, C = 128$. Q_0 is fed into our FAX-CA block as query whereas I_0 is regarded as key and value to project image features into the BEV space. We set the window/grid size of I_0 as (8, 8) and that of the B_0 as (16, 16). Afterwards, Q_0 is downsampled and refined by two standard residual blocks to obtain $Q_1 \in \mathbb{R}^{64\times64\times128}$. The BEV query will perform the same operations with I_1 and I_2 sequentially to obtain the final BEV feature Q_2 in $\mathbb{R}^{32\times32\times128}$.

FuseBEVT. The BEV features from N agents will be stacked together as $h \in \mathbb{R}^{N\times32\times32\times128}$ and fed into three sequential FAX-SA blocks to gain the fused feature $H \in \mathbb{R}^{32\times32\times128}$. The window/grid size is set as 8 for all FAX-SA blocks.

Decoder. H will be upsampled by 3× [bilinear interpolation, conv3x3, BN] to retrieve the final segmentation mask $M \in \mathbb{R}^{256\times256\times k}$, where $k = 2$ for dynamic objects and $k = 3$ for static layout.

3.2 nuScenes

To make a fair comparison, we strictly follow the same experiment setting as CVT [1] **Image Encoder.** We follow CVT [1] and Fiery [2] to use EfficientNet B-4 [11] as image feature extractor. We compute features at three scales - (56, 120), (28, 60), and (14, 30).

SinBEVT. The BEV query starts with a size of $100 \times 100 \times 32$ and ends with a size of $25 \times 25 \times 128$. We set the window/grid size of image features and BEV query for the three FAX-CA blocks as (6, 12), (6, 12), (14, 30) and (10, 10), (10, 10), (25, 25) respectively. Main architecture is the same to the SinBEVT specifications shown in Table A2.

Decoder The decoder structure is the same as CVT. The decoder consists of three (bilinear upsample + conv) layers to upsample the BEV feature to the final output size (200 × 200).

Training. We train our models with focal loss and a batch size of 4 per GPU for 30 epochs. We employ AdamW optimizer with the one-cycle learning rate scheduler. The whole training process is around 8 hours on 4 RTX3090 gpus.

Evaluation. We evaluate the 100m×100m area around the vehicle with a 50cm sampling resolution. We use the Intersection-over-Union (IoU) score between the model predictions and the ground-truth segmentation mask.

3.3 OPV2V LiDAR Track

All the comparison methods have the same configurations except for the fusion component.

Point Cloud Encoder. We select PointPillar [12] as the point cloud feature extractor and set the voxel resolution as (0.4, 0.4, 4) on x, y, and z axis. The architecture settings are the same as [12]. The extracted feature has a final resolution of 176 × 48 × 256.

FuseBEVT. The configurations of FuseBEVT is the same as the ones in OPV2V camera track.

Detection Head and Training. We simply apply two 3×3 convolution layers for classification and regression respectively. We train the models using Adaw [13] optimizer with multi-step learning rate scheduler. The learning rate starts with 0.001 and decay 10 times for every 10 epochs.

4 More Qualitative Results

OPV2V camera track. Fig. 2 and Fig. 3 show the visial comparisons between our CoBEVT and others on OPV2V camera track. Our method significantly outperforms others both on dynamic objects prediction and road topology segmentation in most of the scenarios.

OPV2V LiDAR track. We demonstrate detection visualization results in OPV2V LiDAR track in 4 different busy intersections in Fig. 4 and Fig. 5. Compare to other state-of-the-art fusion methods in including AttFuse [14], F-Cooper [15], V2VNet [16], and DiscoNet [17], our CoBEVT achieves more robust performance in general.We carefully examined the detection visualization comparisons between our method and the previous SOTA method DiscoNet. As shown in Fig. 4 and Fig. 5, we use red circles to highlight the objects that have obviously different detection results among these two methods. It is obvious that our results have fewer undetected objects and fewer displacements.

nuScenes. Fig. 6 depicts the qualitative results of our SinBEVT on nuScenes under different road typologies, traffic situations, and light conditions. Our method can recognize most of the objects and robustly estimate the complicated road layout, demonstrating strong generalization ability of the proposed FAX attention for various autonomous driving tasks.

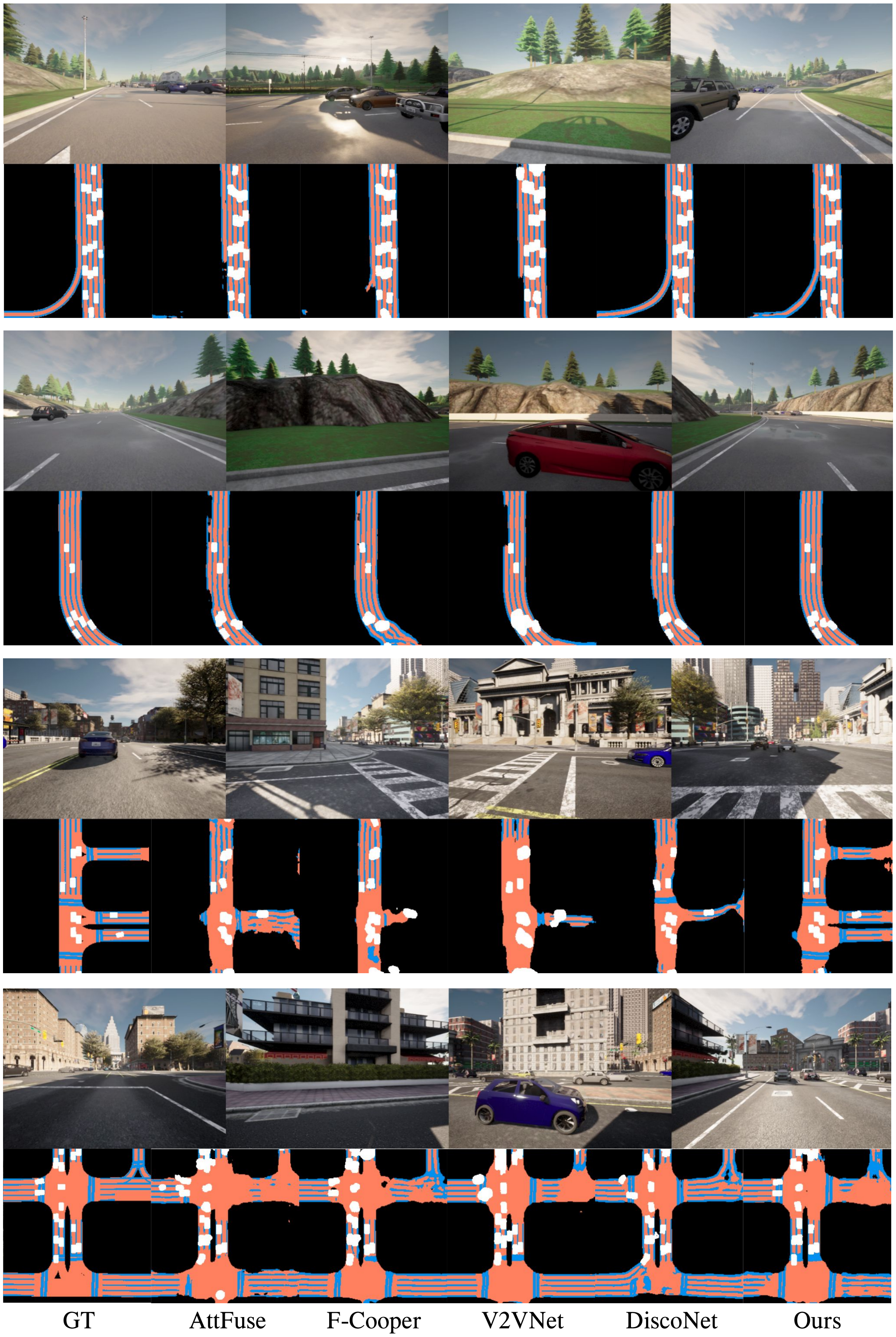


Figure 2: **More qualitative results for OPV2V camera track**. We show the four cameras of ego vehicle in the first row and all comparison methods along with groundtruth in the second row for each group.

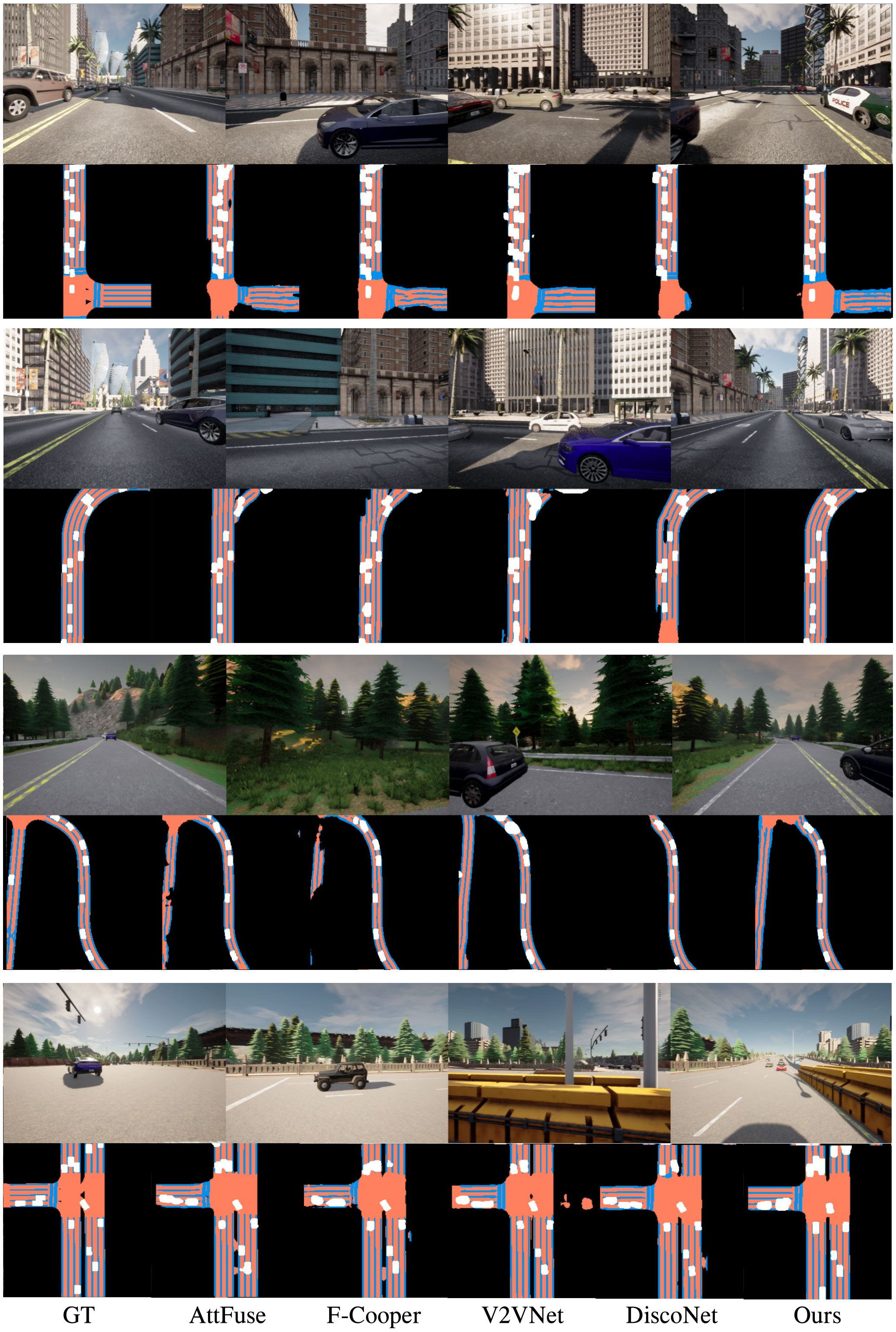


Figure 3: **More qualitative results for OPV2V camera track**. We show the four cameras of ego vehicle in the first row and all comparison methods along with groundtruth in the second row for each group.

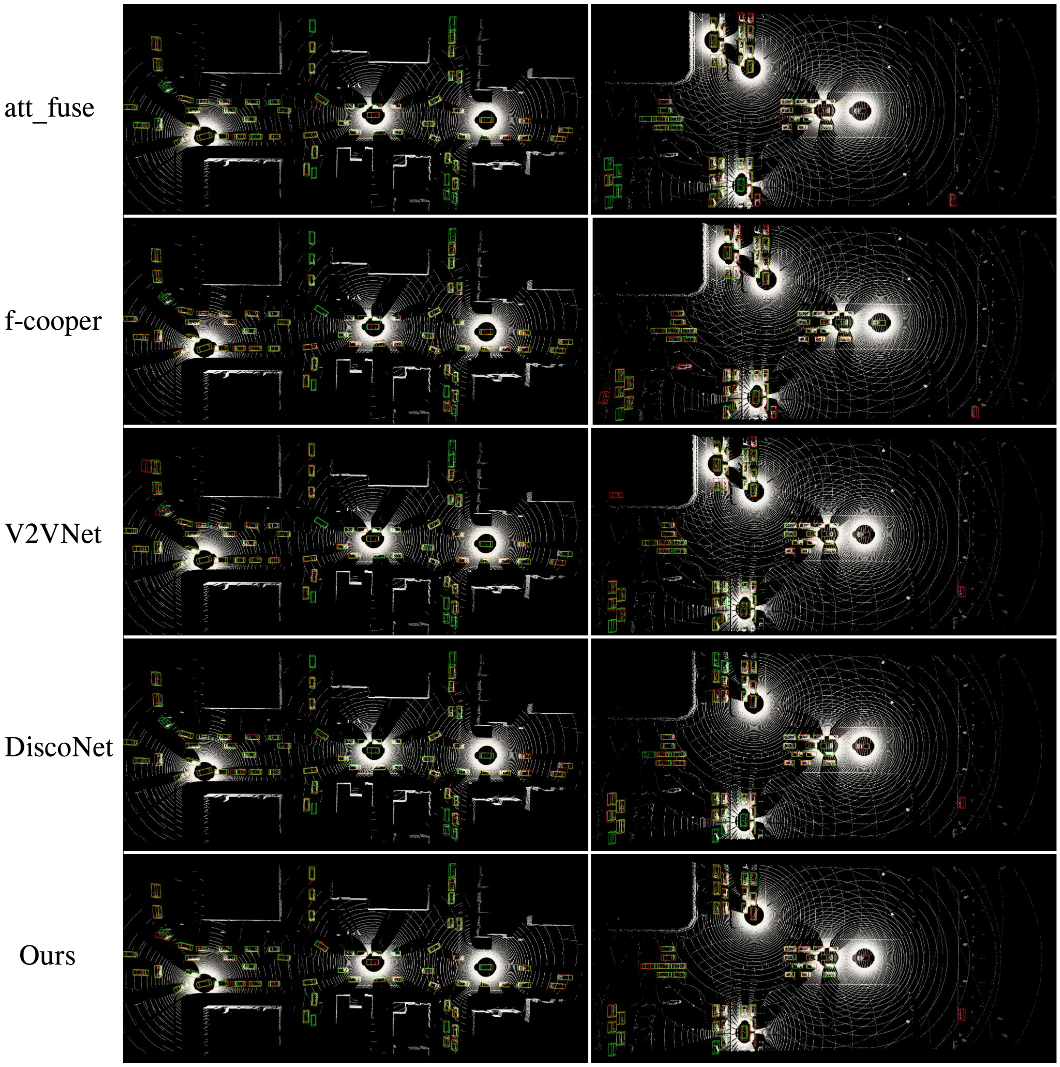


Figure 4: **Qualitative results for OPV2V LiDAR track**. We compared our predictions against other state-of-the-art methods on 2 challenging scenes.

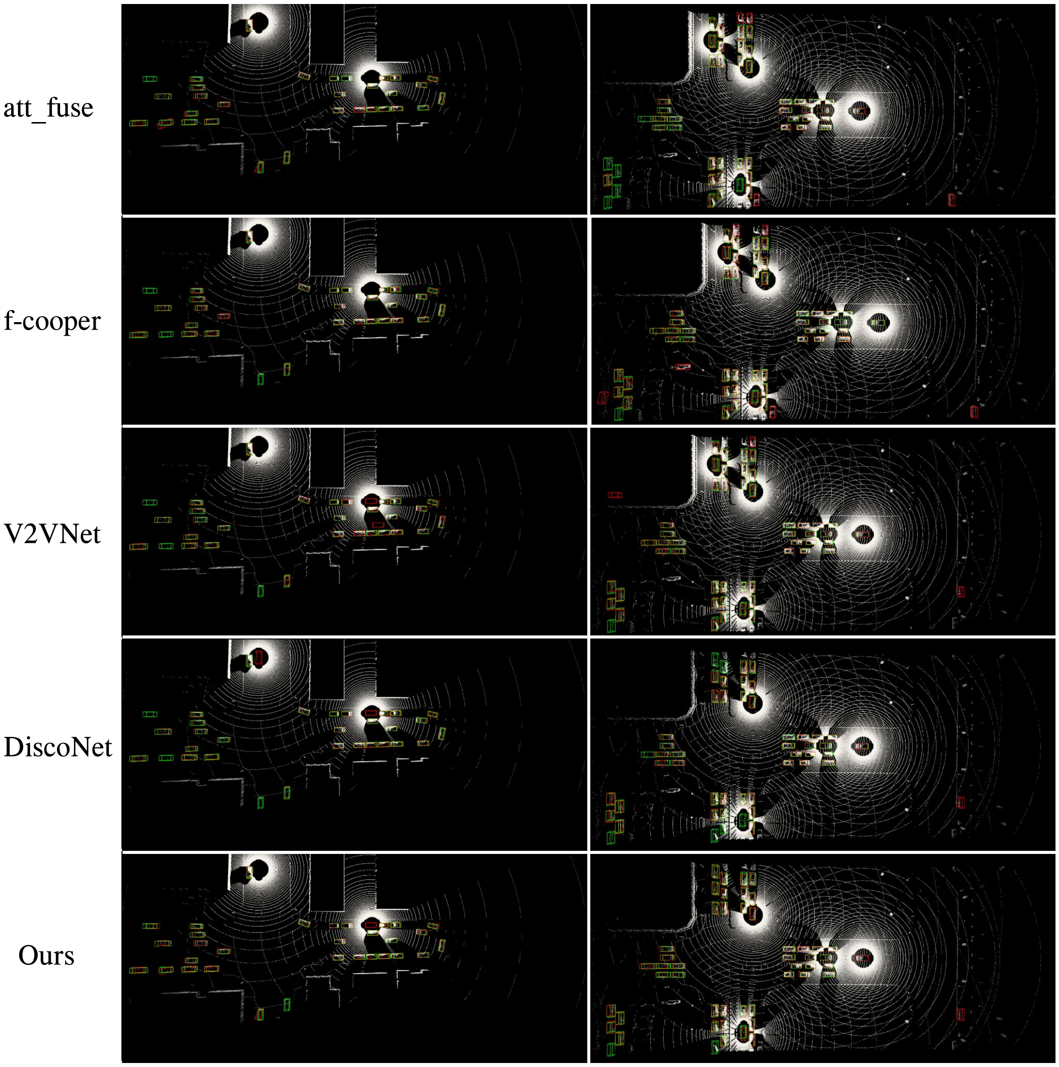
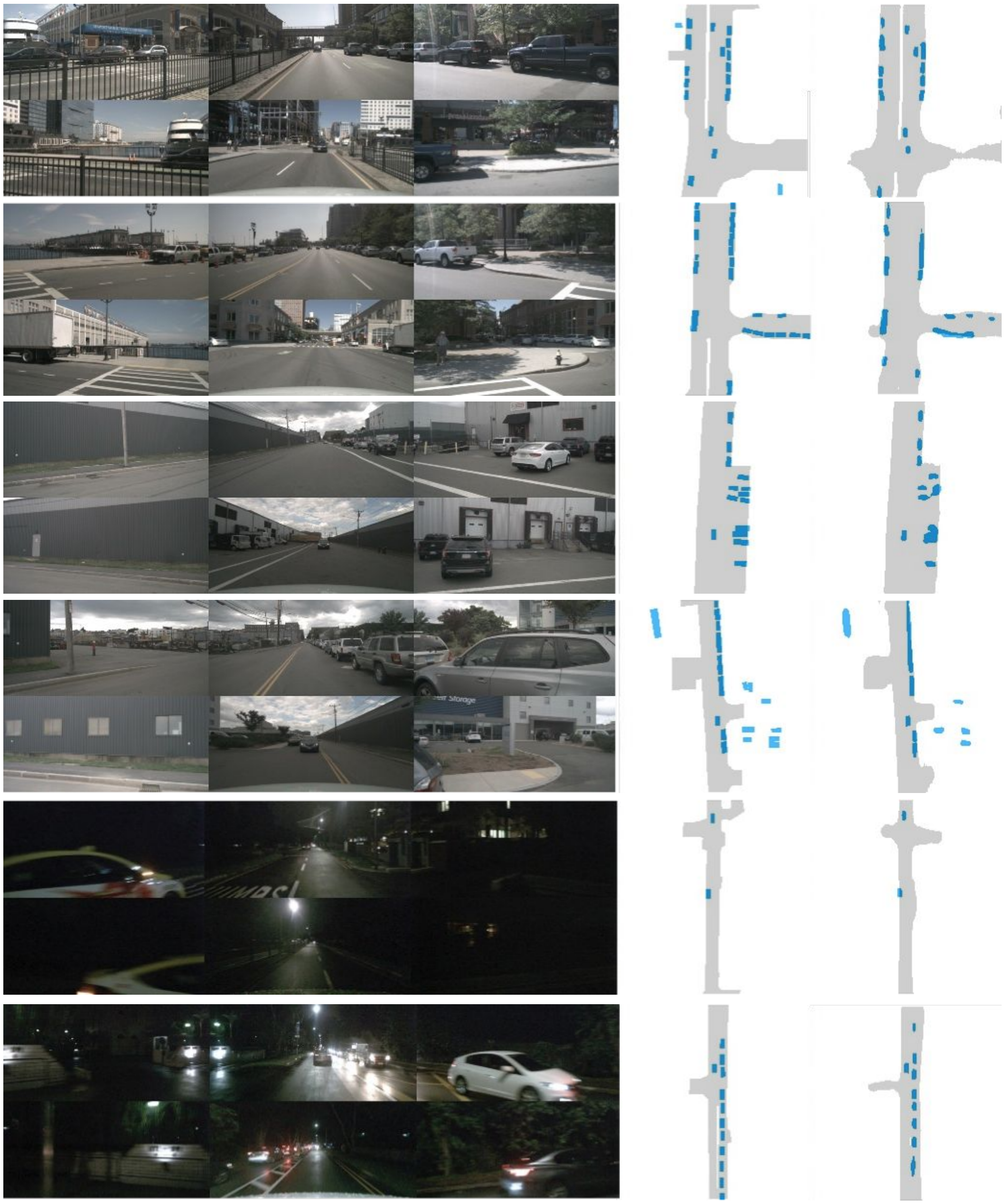


Figure 5: **More qualitative results for OPV2V LiDAR track**. We compared our predictions against other state-of-the-art methods on 2 more scenes.



(a) Multi-view camera images (b) GT (c) Our prediction

Figure 6: **Qualitative results on the nuScenes dataset for various occlusions and light conditions**. We show the (a) six camera-view images on the left group of pictures, and the (b) ground truth segmentation reference, (c) our SinBEVT predictions on the most right. The ego-vehicle is located at the center of the segmentation map.

References

[1] B. Zhou and P. Krähenbühl. Cross-view transformers for real-time map-view semantic segmentation. *arXiv preprint arXiv:2205.02833*, 2022.

[2] A. Hu, Z. Murez, N. Mohan, S. Dudas, J. Hawke, V. Badrinarayanan, R. Cipolla, and A. Kendall. Fiery: Future instance prediction in bird's-eye view from surround monocular cameras. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15273–15282, 2021.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[4] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

[5] Z. Liu, J. Ning, Y. Cao, Y. Wei, Z. Zhang, S. Lin, and H. Hu. Video swin transformer. *arXiv preprint arXiv:2106.13230*, 2021.

[6] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[8] K. Mani, S. Daga, S. Garg, S. S. Narasimhan, M. Krishna, and K. M. Jatavallabhula. Monolayout: Amodal scene layout from a single image. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1689–1697, 2020.

[9] W. Yang, Q. Li, W. Liu, Y. Yu, Y. Ma, S. He, and J. Pan. Projecting your view attentively: Monocular road scene layout estimation via cross-view transformation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15536–15545, 2021.

[10] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[11] M. Tan and Q. Le. Efficientnetv2: Smaller models and faster training. In *International Conference on Machine Learning*, pages 10096–10106. PMLR, 2021.

[12] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12697–12705, 2019.

[13] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[14] R. Xu, H. Xiang, X. Xia, X. Han, J. Liu, and J. Ma. Opv2v: An open benchmark dataset and fusion pipeline for perception with vehicle-to-vehicle communication. *arXiv preprint arXiv:2109.07644*, 2021.

[15] Q. Chen, X. Ma, S. Tang, J. Guo, Q. Yang, and S. Fu. F-cooper: Feature based cooperative perception for autonomous vehicle edge computing system using 3d point clouds. In *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*, pages 88–100, 2019.

[16] T.-H. Wang, S. Manivasagam, M. Liang, B. Yang, W. Zeng, and R. Urtasun. V2vnet: Vehicle-to-vehicle communication for joint perception and prediction. In *European Conference on Computer Vision*, pages 605–621. Springer, 2020.

[17] Y. Li, S. Ren, P. Wu, S. Chen, C. Feng, and W. Zhang. Learning distilled collaboration graph for multi-agent perception. *Advances in Neural Information Processing Systems*, 34, 2021.