V2V4Real: A Real-world Large-scale Dataset for Vehicle-to-Vehicle Cooperative Perception – Supplementary Material

Runsheng Xu[1], Xin Xia[1*], Jinlong Li[2], Hanzhao Li[1], Shuo Zhang[1], Zhengzhong Tu[3], Zonglin Meng[1], Hao Xiang[1], Xiaoyu Dong[4], Rui Song[5,6], Hongkai Yu[2], Bolei Zhou[1], Jiaqi Ma[1]

University of California, Los Angeles[1] Cleveland State University[2] University of Texas at Austin[3]
Northwestern University[4] Technical University of Munich[5] Fraunhofer Institute[6]

1. Overview

This supplementary document is organized as follows:

- We present additional information and visualization of the V2V4Real dataset in Sec. 2.
- Implementation details of the evaluated models are covered in Sec. 3.
- More ablation studies are covered in Sec. 4.
- More qualitative 3D object detection results are shown in Sec. 5.
- More qualitative Sim2Real domain adaptation results are shown in Sec. 6

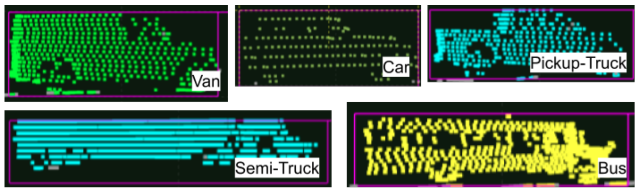


Figure 1. The LiDAR appearance for each object class in V2V4Real.

2. Dataset Visualization

We demonstrate what each object class looks like in the LiDAR data in Fig. 1. We show more visualizations of the proposed V2V4Real dataset in Fig. 2. 3 different scenes are presented: a cityroad, a highway, and another cityroad example. For each figure, we demonstrate four images. Upper left: the aggregated 3D LiDAR points; Upper right: the annotated HDMap; Bottom row: the front camera view of the two vehicles, with the green and red LiDAR corresponding to lower left and lower right images. The 3D bounding boxes drawn on the images are projected from the labels annotated in LiDAR frames using extrinsics and intrinsics.

*Corresponding Author, email address: x35xia@ucla.edu

3. Implementation Details

We provide additional details on the implemented baseline methods in our experiments.

3.1. Cooperative 3D Object Detection

PointPillars backbone. For all the experiments, we set the PointPillars backbone [4] to have a voxel resolution of 0.4 meters on x and y direction. We set the maximum points per voxel as 32 and the maximum voxel numbers as 32000.

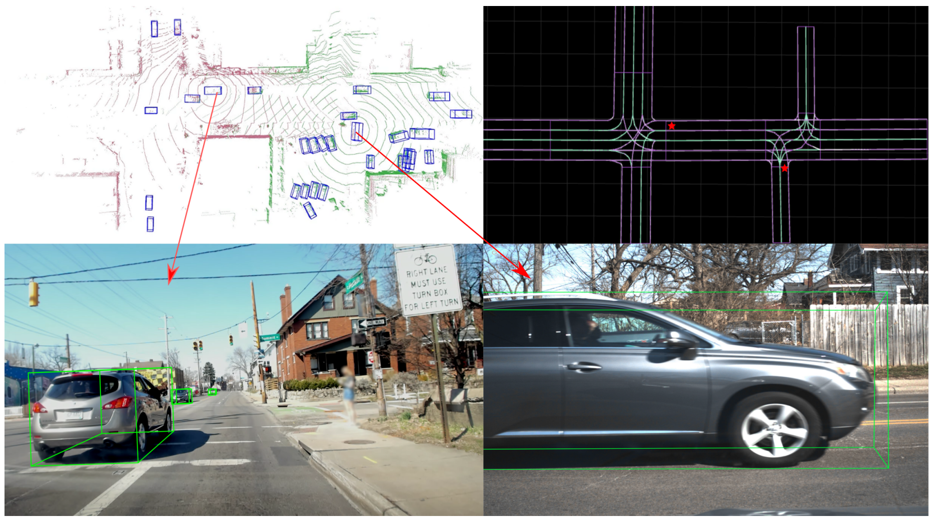
Fusion models. We have implemented five different fusion methods including F-Cooper [2], AttFuse [8], V2VNet [6], V2X-ViT [5], and CoBEVT [7]. We mainly follow the implementation and configurations from the original authors, except for V2X-ViT, wherein we regard the two vehicles as the same object type (i.e., vehicle) since there is no infrastructure in V2V4Real.

Detection head. For 3D object detection, we apply two channel-wise convolution 1×1 layers on top of the fused feature maps to obtain two heads for box regression and classification, respectively. The regression head yields $(x, y, z, w, l, h, \theta)$, denoting the position x, y, z, size w, l, and yaw angle θ of the predefined anchor boxes. The classification head outputs the confidence score of being an object or background for each anchor box, respectively. We employ the smoothed ℓ_1 loss and a focal loss for regression and classification heads.

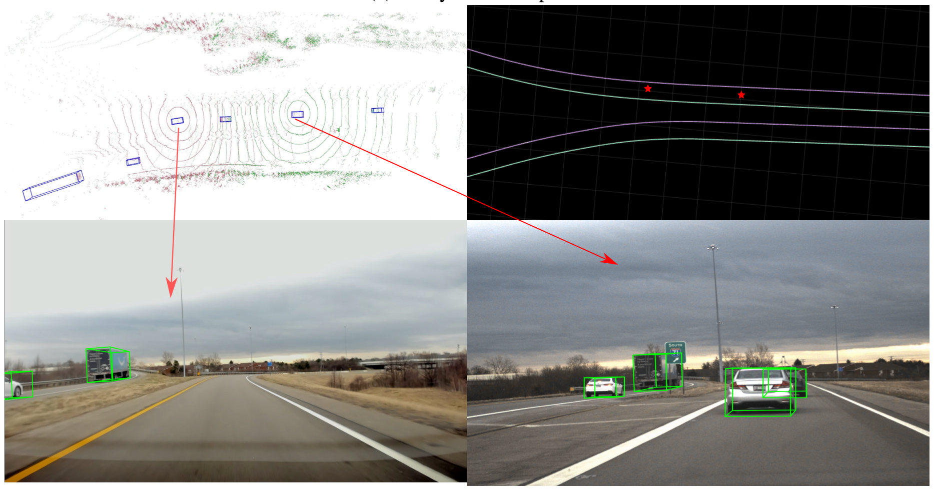
3.2. Cooperative Tracking

The proposed cooperative tracking framework in our benchmark follows the widely adopted tracking-by-detection paradigm but differs from the existing object tracking methods: the detection results are gained from shared visual information instead of individuals.

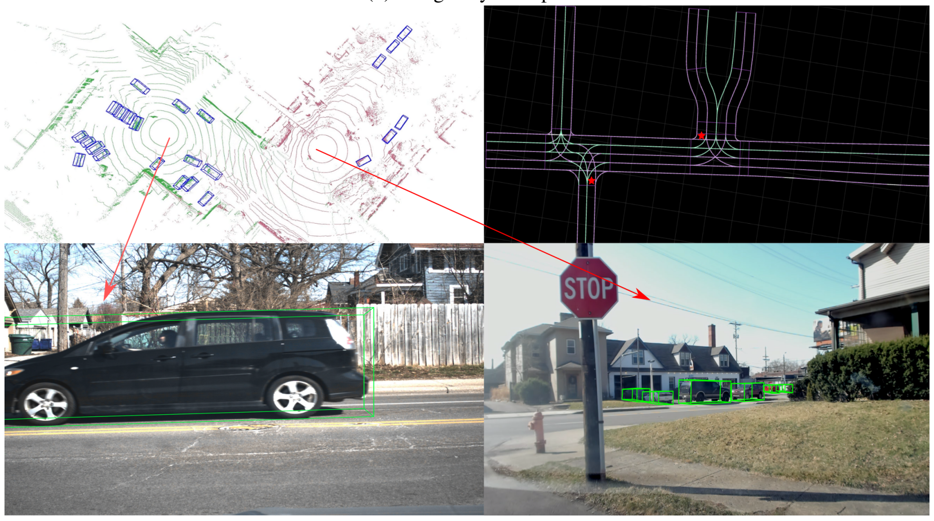
Problem definition: Assume there are N_t bounding boxes $D(t) = \{D_t^i\}_{i=1}^{N_t}$ from our cooperative detection algorithm at current frame t, where N_t represent the number of cooperatively detected



(a) A cityroad example



(b) A highway example



(b) A cityroad example

Figure 2. **3 different scenarios in V2V4Real.** *Left Up*: The aggregated 3D LiDAR. *Right Up*: The annotated HDMap, where the stars indicate the position of the two collection vehicles. *Bottom Row*: The front cameras of the two vehicles.

objects and $D_t^i = (x_t^i, y_t^i, z_t^i, \theta_t^i, w_t^i, h_t^i, l_t^i, s_t^i)$. There are M_{t-1} previous associated trajectories $T(t-1) = \{T_{t-1}^j\}_{j=1}^{M_{t-1}}$ at frame $t-1$, where M_{t-1} represent the number of previous trajectories and $T_{t-1}^j = (x_{t-1}^j, y_{t-1}^j, z_{t-1}^j, \theta_{t-1}^j, w_{t-1}^j, h_{t-1}^j, l_{t-1}^j, s_{t-1}^j, {v_x}_{t-1}^j, {v_y}_{t-1}^j, {v_z}_{t-1}^j)$. We aim to map each detected object D_t^i to their corresponding trajectory M_{t-1}^j. The (x, y, z) corresponds to the object center, and (w, h, l) represents object size in point cloud space. θ is the heading angle of the object. s is the detection confidence score, which depends on the cooperative detection network. The additional variables (v_x, v_y, v_z) in trajectories represent the object velocity in x, y, and z directions.

Detection results: The inputs of the tracking system are the detected bounding boxes, which are obtained from the cooperative 3D object detection task described in the previous section.

Trajectory prediction and association: With the detected bounding boxes from the cooperative detection module, the goal of object tracking is to find all valid matches between detected bounding boxes $D(t)$ and trajectory $T(t-1)$. A Kalman filter is applied to predict the trajectories $T(t-1)$ of objects based on a constant velocity kinematic vehicle model. These predicted spatial information of trajectories combined with the information of detected objects would be used to calculate the affinity matrix in the Hungarian to determine whether currently detected objects in $D(t)$ can be matched to trajectories in $T(t-1)$. Specifically, given a sequence of trajectories

$$T(t-1) = \{T_{t-1}^j\}_{j=1}^{M_{t-1}} \tag{1}$$

at frame $t-1$, a constant velocity kinematic vehicle model in the Kalman filter is used to predict the position of the object in each trajectory in $T(t-1)$ as follows:

$$x_{t,pred}^j = x_{t-1}^j + {v_x}_{t-1}^j \tag{2}$$

$$y_{t,pred}^j = y_{t-1}^j + {v_y}_{t-1}^j \tag{3}$$

$$z_{t,pred}^j = z_{t-1}^j + {v_z}_{t-1}^j \tag{4}$$

Therefore, the final predicted trajectory is

$$T_{t,pred}^j = (x_{t,pred}^j, y_{t,pred}^j, z_{t,pred}^j, \theta_{t-1}^j, w_{t-1}^j, \tag{5}$$

$$h_{t-1}^j, l_{t-1}^j, s_{t-1}^j, {v_x}_{t-1}^j, {v_y}_{t-1}^j, {v_z}_{t-1}^j) \tag{6}$$

After predicting the set of trajectories $T(t)_{pred}$, the 3D Intersection of Union(IoU) is used to compute the data affinity matrix $A \in \mathbf{R}^{M_{t-1} \times N_t}$ to determine the similarity between predicted trajectories and detected bounding boxes $D(t)$, where each element $A_{i,j}$ is the 3D IoU for the predicted trajectory i and the 3D bounding box j at frame t. The affinity matrix will be solved by the Hungarian algorithm, which considers the association as a bipartite matching problem, to solve the association problem.

State update and trajectory management: After gaining the predicted trajectories and association results from the Hungarian algorithm, the state update is to make the trajectory more accurate by considering the current detection results. The Kalman Filter is used to update the state of the predicted trajectory by considering the current detection information and accounting for uncertainties from the detection errors. Accordingly, we have:

$$T_t^m = KF(T_t^m, D_t^k) \tag{7}$$

,where $D_t^k \in D(t)$ and $T_t^m \in T(t)$ are the associated pair obtained from Hungarian algorithm, $k \in \{1, 2, ..., N_t\}$, $m \in \{1, 2, ..., Mt\}$. The updated state of the corresponding predicted trajectory $T_t^m \in T(t)$ is a weighted average between the related $D_t^k \in D(t)$ and $T_t^m \in T(t)$.

Trajectory management is organizing new and old trajectories. When an object starts to appear at frame t, it could either be a false positive due to the detector or it naturally enters the field of view. Similarly, when an object starts to disappear at frame t, it could either be a miss or it naturally leaves the LiDAR range. Both scenarios are handled by tracking objects in additional frames. Specifically, when D_t^l is an unmatched object entering the field of view, we will treat it as a new trajectory if D_t^l can be matched in the next few frames to prevent adding false positive detection as a new trajectory. When T_t^p is an unmatched trajectory leaving the field of view, we will treat it as a dead trajectory if T_t^p cannot be matched with any detected bounding boxes in the next couple of frames to prevent removing the true positive trajectory.

3.3. Domain Adaption

Feature-level domain discriminator: The feature-level domain discriminator will take the fused features after the fusion modules as input and classify whether the feature belongs to target domain (V2V4Real) or source domain (OPV2V). The discriminator consists of two convolution layers with 3×3 kernel size, and the second convolution will map the feature channel number to 1.

Objec-level domain discriminator: The object-level domain clarifier will take the score map obtained from the detection classification head as the input. It includes three linear projection layers with ReLU activation functions.

Loss: Both discriminators employ binary cross-entropy to compute the loss and use gradient reverse layer (GRL) [3] to backpropagate the gradients.

4. Ablation Studies

Effects of Data Augmentation. Data augmentation has been shown to be highly effective in single-vehicle per-

Method	**Sync** (AP@IoU=0.5)				**Async** (AP@IoU=0.5)				**AM**
	Overall	0-30m	30-50m	50-100m	Overall	0-30m	30-50m	50-100m	(MB)
No Fusion	28.7(-11.1)	50.0	22.9	4.9	28.7(-11.1)	50.0	22.9	4.9	0
Late Fusion	43.0(-12.0)	55.1	34.4	31.9	40.9(-9.3)	54.6	33.5	30.9	0.003
Early Fusion	48.2(-11.5)	64.3	33.2	34.0	41.0 (-11.1)	62.5	27.3	18.1	0.96
F-Cooper [1]	45.6(-15.1)	65.3	35.3	25.9	37.6(-16.0)	62.1	28.6	13.4	0.20
V2VNet [6]	49.0(-15.5)	69.2	35.0	30.6	41.5(-14.9)	65.5	32.4	13.7	0.20
AttFuse [8]	47.9(-16.8)	67.9	34.6	26.8	40.8(-16.9)	65.8	28.6	13.9	0.20
V2X-ViT [5]	48.9(-16.0)	66.0	38.1	30.0	41.6(-14.3)	62.8	32.9	17.5	0.20
CoBEVT [7]	51.1(-15.4)	69.3	40.0	32.4	44.9 (-13.7)	65.2	35.6	19.8	0.20

Table 1. Cooperative 3D object detection benchmark **without data augmentation** . The numbers in the bracket indicate the performance drop compared to the same model with data augmentation.

ception tasks, such as 3D object detection using pointclouds [4, 9]. In this work, we evaluate the impact of data augmentation on cooperative perception by conducting an ablation study that removes pointcloud rotation, flipping, and scaling augmentations. Our evaluation is performed on 3D object detection. As depicted in Table 1, all methods show a significant decrease in performance without data augmentation, such as 15.6% for CoBEVT and 11.5% for Early fusion. Additionally, the intermediate fusion methods show more benefits from data augmentation, which is possibly due to their more complex models and requirement for more data.

5. Detection Results

We demonstrate more qualitative results of the 3D detection comparisons in Figs. 3 and 4 under the *Sync* setting. As shown in the urban scene in Fig. 3, where traffic is heavier and crowded vehicles are causing severe occlusions, cooperative solutions yield significantly better detection results than No Fusion. Intermediate fusion methods also generate more accurate detection than early or late fusion within the medium radius (< 50 m). It is obvious that among all the compared approaches, CoBEVT's prediction best aligns with the ground truth bounding boxes, which is consistent with the numerical results provided in the main paper. As for the highway scene in Fig. 4 where vehicles drive at higher speed but less crowdedness, all the cooperative methods can successfully predict the surrounding vehicles' bounding boxes, with some approaches (V2X-ViT, CoBEVT) slightly more accurate than others (V2VNet, Late Fusion, Earl Fusion).

6. Domain Adaptation Results

Figs. 5 and 6 show the qualitative results of the cooperative domain adaptation. It may be observed in the highway scenario (Fig. 5) that all the models benefited from applying domain adaptation strategies, with AttFuse and F-Cooper gaining the most, observing from the huge performance difference between the detection results without and without domain adaptation. In a more crowded intersection scene (Fig. 6), we may see that F-Cooper, V2X-ViT, and CoBEVT are top performers among the compared methods. However, F-Cooper has been observed to produce more false positives while less so for V2X-ViT, after domain adaptation is applied.

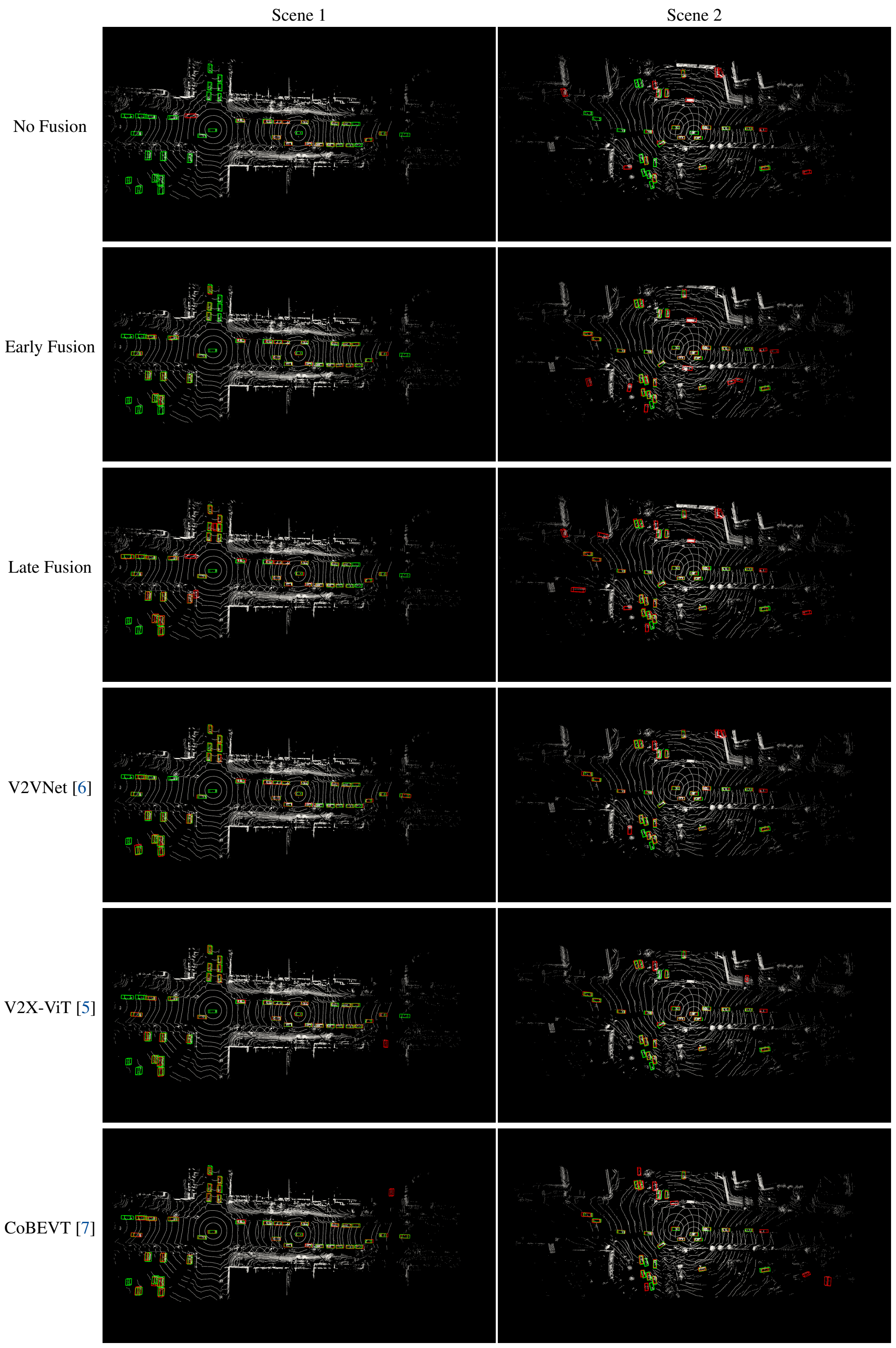


Figure 3. **Qualitative results of cooperative 3d object detection in two urban scenarios.** Green and red 3D bounding boxes represent the groundtruth and prediction, respectively.

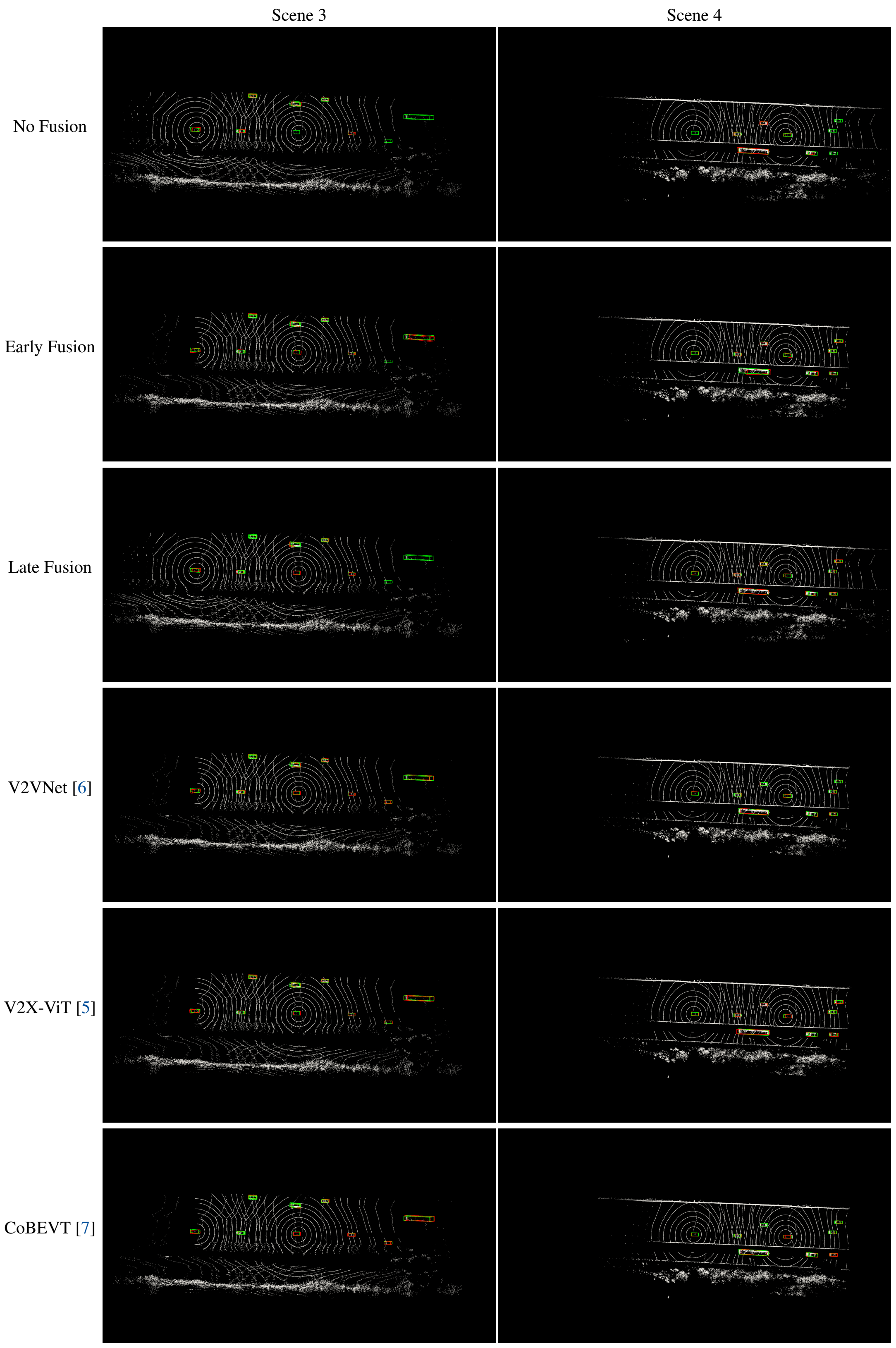


Figure 4. **Qualitative results of cooperative 3d object detection in two highway scenarios.** Green and red 3D bounding boxes represent the groundtruth and prediction, respectively.

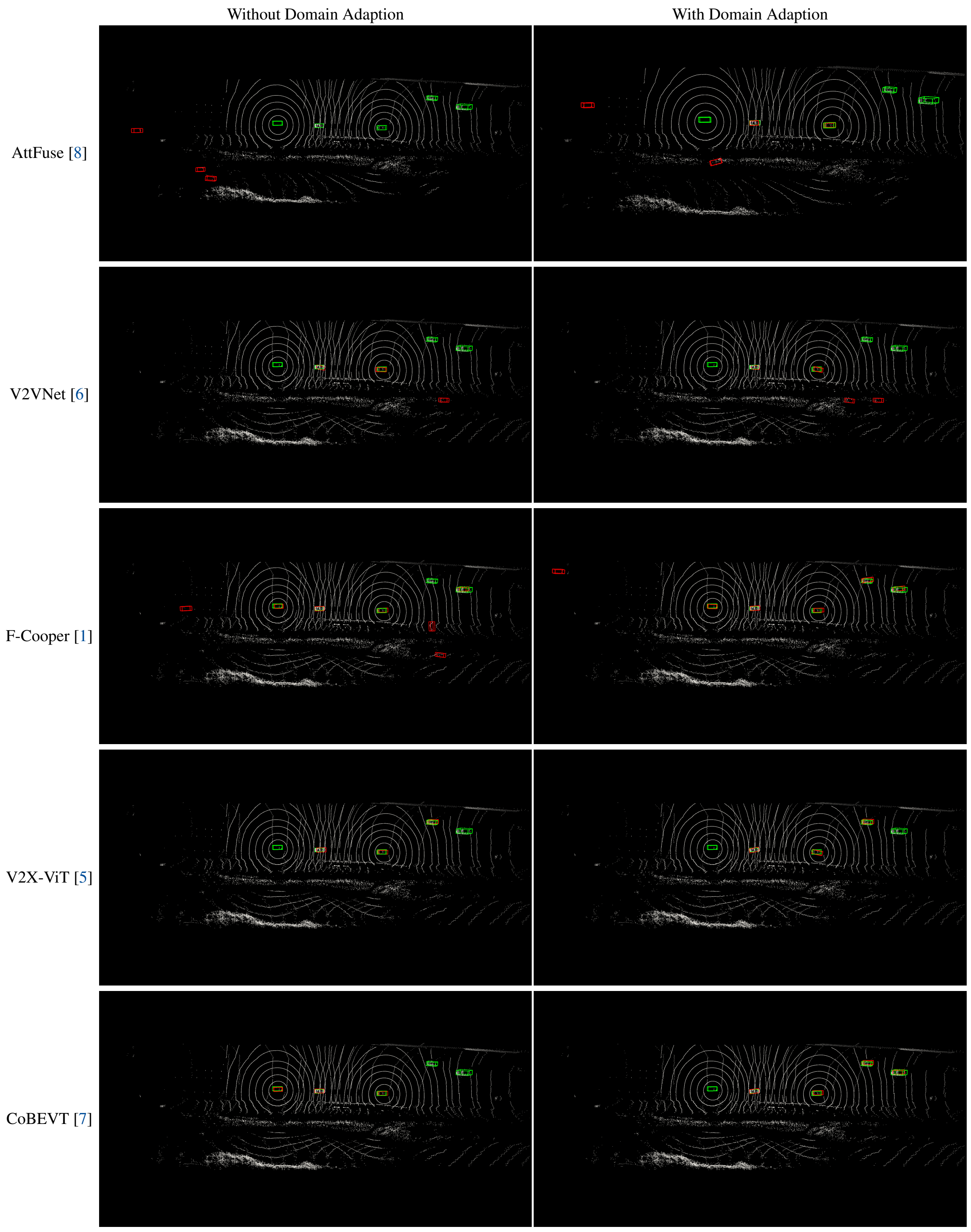


Figure 5. **Qualitative results of domain adaption in a highway scenario.** Green and red 3D bounding boxes represent the groundtruth and prediction, respectively.

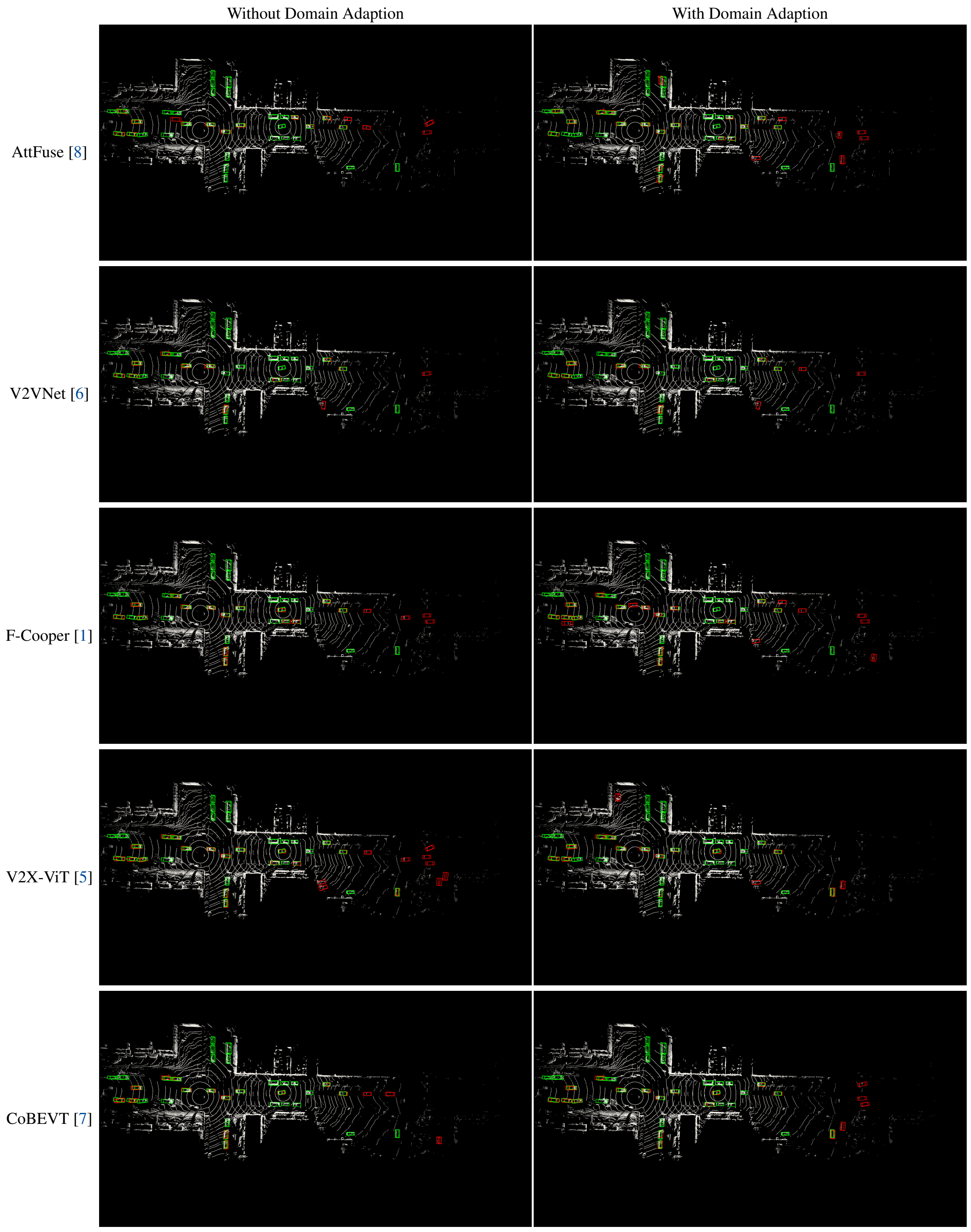


Figure 6. **Qualitative results of domain adaption in an intersection scenario.** Green and red 3D bounding boxes represent the groundtruth and prediction, respectively.

References

[1] Qi Chen, Xu Ma, Sihai Tang, Jingda Guo, Qing Yang, and Song 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. 4, 7, 8

[2] Qi Chen, Sihai Tang, Qing Yang, and Song Fu. Cooper: Cooperative perception for connected autonomous vehicles based on 3d point clouds. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pages 514–524. IEEE, 2019. 1

[3] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR, 2015. 3

[4] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar 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. 1, 4

[5] Zhengzhong Tu Xin Xia Ming-Hsuan Yang Jiaqi Ma Runsheng Xu, Hao Xiang. V2x-vit: Vehicle-to-everything cooperative perception with vision transformer. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 1, 4, 5, 6, 7, 8

[6] Tsun-Hsuan Wang, Sivabalan Manivasagam, Ming Liang, Bin Yang, Wenyuan Zeng, and Raquel Urtasun. V2vnet: Vehicle-to-vehicle communication for joint perception and prediction. In *European Conference on Computer Vision*, pages 605–621. Springer, 2020. 1, 4, 5, 6, 7, 8

[7] Runsheng Xu, Zhengzhong Tu, Hao Xiang, Wei Shao, Bolei Zhou, and Jiaqi Ma. Cobevt: Cooperative bird's eye view semantic segmentation with sparse transformers. *arXiv preprint arXiv:2207.02202*, 2022. 1, 4, 5, 6, 7, 8

[8] Runsheng Xu, Hao Xiang, Xin Xia, Xu Han, Jinlong Li, and Jiaqi Ma. Opv2v: An open benchmark dataset and fusion pipeline for perception with vehicle-to-vehicle communication. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2583–2589. IEEE, 2022. 1, 4, 7, 8

[9] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4490–4499, 2018. 4