V2X-ViT: Vehicle-to-Everything Cooperative Perception with Vision Transformer (Supplementary Material)

Runsheng Xu[1⋆], Hao Xiang[1⋆], Zhengzhong Tu[2⋆], Xin Xia[1], Ming-Hsuan Yang[3,4], and Jiaqi Ma[1†]

[1] University of California, Los Angeles
[2] University of Texas at Austin
[3] Google Research
[4] University of California, Merced

In this supplementary material, we first discuss the design choice and scalability issue of our method (Sec. 1). Then, we provide more model details and analysis (Sec. 2) in regards to the proposed architecture, including the mathematical details of the spatial-temporal correction module (Sec. 2.1), details of the proposed MSwin (Sec. 2.2), and the overall architectural specifications (Sec. 2.3). Afterwords, additional information and visualizations of the proposed V2XSet dataset are shown in Sec. 3. In the end, we present more quantitative experiments, qualitative detection results, attention map visualizations, and details about the effects of the transmission size experiment in Sec. 4.

1 Discussion of design choice

Scalability of ego vehicles. Our approach can be scalable in two ways: 1) Decentralized: the ablation studies conducted in this paper (Fig. 5a) and OPV2V [15] indicate that, when the number of collaborators is larger than 4, the performance gain becomes marginal while the computation still increases linearly. In practice, each agent only needs to share features with a limited number of agents. For example, Who2Com [10] studies which agent to request/transmit data, largely reducing computation. Moreover, the computation of selected PointPillar backbone is efficient, e.g., around 4 ms for 4 agents with full parallelization and 16 ms in sequence computing on RTX3090. 2) Centralized: Within a certain communication range, only one ego agent is selected to aggregate all the features from neighbors to predict bounding boxes and share the results with other agents. This solution requires only one computation node for a group of agents, thus being scalable.

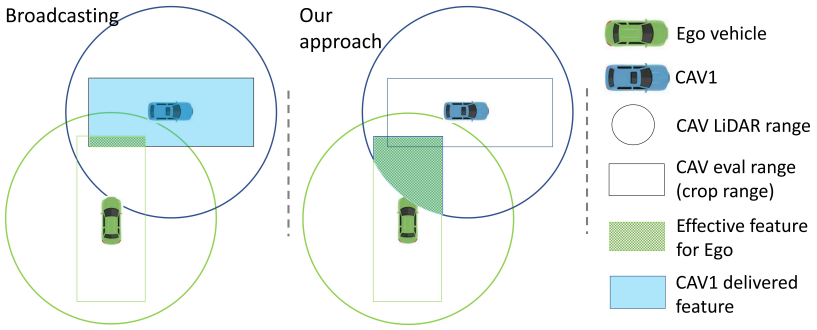
Design choices for communication. Compared to the broadcasting approach (*i.e.*, compute the features in each cav's own space and transform the feature maps directly on the ego side), our approach has more advantages in terms of detection accuracy. Most LiDAR detection methods often largely crop the LiDAR range based on the evaluation range to reduce computation. As the

⋆ Equal contribution. † Corresponding author: jiaqima@ucla.edu

Table T0: Comparison between our design choice and broadcasting approach.

	DiscoNet (broad. / ours)	V2X-ViT (broad. / ours)
AP@0.7 (perfect)	0.610 / 0.695	0.623 / 0.712

figure below shows, the CAVs crop the LiDAR data based on their own evaluation range in the broadcasting method, which leads to redundant data. Our approach, on the contrary, always does cropping based on the ego's evaluation range, thus guaranteeing more effective feature transmission. We further validate this by comparing our framework with the broadcasting approach. The Tab. T0 below shows that our design outperforms broadcasting by 8.5% and 8.9% for DiscoNet and V2X-ViT.



2 Model Details and Analysis

2.1 Spatial-Temporal Correction Module

During the early stage of collaboration, when each connected agent i receives ego vehicle's pose at time t_i, the observed point clouds of agent i will be projected to ego vehicle's pose $x_e^{t_i}$ at time t_i before feature extraction. However, due to the time delay, the ego vehicle observes the data at a different time t_e. Thus, the received features from connected agents are centered around a delayed ego vehicle's pose (*i.e.*, $x_e^{t_i}$) while the ego vehicle's features are centered around current pose (*i.e.*, $x_e^{t_e}$), leading to a delay-induced spatial misalignment. To correct this misalignment between the received features and ego-vehicle's features, a global transformation $\xi_{x_e^{t_i}, x_e^{t_e}} \in \mathfrak{se}(3)$ from ego vehicle's past pose $x_e^{t_i}$ to its current pose $x_e^{t_e}$ is required. To this end, we employ a differential 2D transformation $\Gamma_\xi(\cdot)$ to warp the intermediate features spatially [7]. To be more specific, we will

transform features' positions by using affine transformation:

$$\begin{bmatrix} x_s \\ y_s \end{bmatrix} = \Gamma_\xi(\begin{bmatrix} x_t \\ y_t \\ 1 \end{bmatrix}) = \begin{bmatrix} R_{11} & R_{12} & \delta x \\ R_{21} & R_{22} & \delta y \end{bmatrix} \begin{bmatrix} x_t \\ y_t \\ 1 \end{bmatrix} \tag{1}$$

where (x_s, y_s) and (x_s, y_s) are the source and target coordinates. As the calculated coordinates may not be integers, we use bilinear interpolation to sample input feature vectors. An ROI mask is also calculated to prevent the network from paying attention to the padded zeros caused by the spatial warp. This mask will be used in heterogeneous multi-agent self-attention to mask padded values' attention weights as zeros.

2.2 Multi-Scale Window Attention (MSwin)

Detailed formulation. Let $\mathbf{H} \in \mathbb{R}^{H \times W \times C}$ be an input feature of a single agent. Let h_j be the number of attention heads used in branch j (*i.e.* head dimension $d_{h_j} = C/h_j$), applying self-attention within each non-overlapping window $P_j \times P_j$ for branch j out of k branches on feature $\mathbf{H}$ can be formulated as:

$$\mathbf{H} = [\mathbf{H}^1, \mathbf{H}^2, ..., \mathbf{H}^{HW/(P_j)^2}], \qquad \text{for branch } j \tag{2}$$

$$\hat{\mathbf{H}}^i_m = \mathsf{Attention}(\mathbf{H}^i\mathbf{W}^Q_m, \mathbf{H}^i\mathbf{W}^K_m, \mathbf{H}^i\mathbf{W}^V_m), \qquad i = 1, ..., HW/(P_j)^2 \tag{3}$$

$$\mathbf{Y}_m = [\hat{\mathbf{H}}^1_m, \hat{\mathbf{H}}^2_m, ..., \hat{\mathbf{H}}^{HW/(P_j)^2}_m], \qquad m = 1, ..., h_j \tag{4}$$

$$\mathbf{Y}^j = [\mathbf{Y}_1, \mathbf{Y}_2, ..., \mathbf{Y}_{h_j}], \tag{5}$$

where $\hat{\mathbf{H}}^i_m \in \mathbb{R}^{P_j^2 \times d_{h_j}}$ and $\mathbf{W}^Q_m, \mathbf{W}^K_m, \mathbf{W}^V_m$ represent the query, key, and value projection matrices. $\mathbf{Y}_m$ is the output of the m-th head for branch j. Afterwards, the outputs for all heads $1, 2, ..., h_j$ are concatenated to obtain the final output $\mathbf{Y}^j$. Here the $\mathsf{Attention}$ operation denotes the relative self-attention, similar to the usage in Swin [11]:

$$\mathsf{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathsf{softmax}((\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}} + \mathbf{B})\mathbf{V}) \tag{6}$$

where $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{P_j^2 \times d}$ denote the query, key, and value matrices. d is the dimension of query/key, while P_j^2 denotes the window size for branch j. Following [11,6], we also consider an additional relative positional encoding $\mathbf{B}$ that acts as a bias term added to the attention map. As the relative position along each axis lies in the range $[-P_j + 1, P_j - 1]$, we take $\mathbf{B}$ from a parameterized matrix $\hat{\mathbf{B}} \in \mathbb{R}^{(2P_j-1)\times(2P_j-1)}$. To adaptively fuse features from all the k branches, we adopt the split-attention module [16] for the parallel feature aggregation:

$$\mathbf{Y} = \mathsf{SplitAttention}(\mathbf{Y}^1, \mathbf{Y}^2, ...\mathbf{Y}^k), \tag{7}$$

Time complexity. As mentioned in the paper, we have k parallel branches. Each branch has $P_j \times P_j$ window size and h_j heads where $P_j = jP$ and $h_j = h/j$.

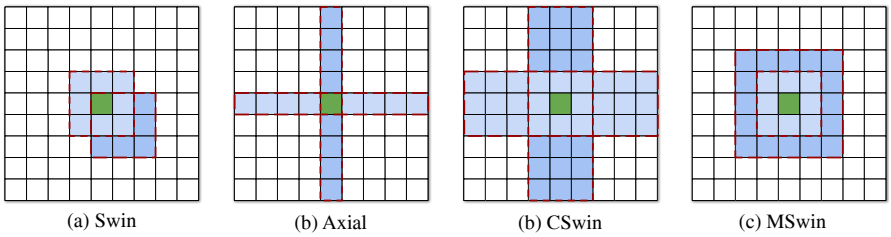


Fig. 1: Visualizations of approximated receptive fields (blue shaded pixels) for the green pixel for (a) Swin [11] (b) Axial [13], (c) CSwin [3] and (d) MSwin attention. MSwin obtains multi-scale long-range interactions at linear complexity.

After partitioning, the input tensor $\mathbf{H} \in \mathbb{R}^{H\times W\times C}$ is split into h_j features with shape $(\frac{H}{P_j} \times \frac{W}{P_j}, P_j \times P_j, C/h_j)$. Following [11], we focus on the computation for vector-matrix multiplication and attention weight calculation. Thus, the complexity of MSwin can be written as:

$$\begin{aligned}
&\mathcal{O}(\sum_{j=1}^{k} \frac{HW}{P_j^2} \times \frac{C}{h_j} \times (P_j \times P_j)^2 \times h_j + 4\frac{HW}{P_j^2} \times P_j^2 \times (\frac{C}{h_j})^2 \times h_j) \\
&= \mathcal{O}(\sum_{j=1}^{k} P_j^2 HWC + \frac{4HWC^2}{h_j}) = \mathcal{O}(\sum_{j=1}^{k} j^2 P^2 HWC + \frac{4HWC^2 j}{h}) \\
&= \mathcal{O}(\frac{1}{3}k^3 P^2 HWC + \frac{2HWC^2 k^2}{h})
\end{aligned} \tag{8}$$

where the first term corresponds to attention weight calculation, the second term is associated with vector-matrix multiplication, and the last equality is due to the fact that $\sum_{j=1}^{k} j^2 = \mathcal{O}(\frac{k^3}{3})$ and $\sum_{j=1}^{k} j = \mathcal{O}(\frac{k^2}{2})$. Thus the overall complexity is

$$\text{FLOPs}(\mathsf{MSwin}) = \mathcal{O}((\frac{k^3 P^2 C}{3} + \frac{2k^2 C^2}{h})HW) \sim \mathcal{O}(HW), \tag{9}$$

which is linear with respect to the image size. The comparison of time complexity of different types of transformers is shown in Tab. T1 where N denotes the number of input pixels, or (here $N = HW$). Our MSwin obtains multi-scale spatial interactions with a linear complexity with respect to N, while other long-range attention mechanisms like ViT [4], Axial [13], and CSwin [3] requires more than linear complexity, which are not scalable to high-resolution dense prediction tasks such as object detection and segmentation.

Effective receptive field. The comparisons of receptive fields between different transformers are shown in Fig. 1. Swin [11] enlarge the receptive fields by using shifted window but it requires sequential blocks to accumulate. Axial Transformer [13] conducts attention on both row-wise and column-wise directions. Similarly, CSwin [3] proposes to perform attention on horizontal and vertical stripes with asymmetrical receptive range in different directions, but requires

Table T1: Computational complexity comparisons of our proposed MSwin attention with (a) full attention in ViT [4], (b) Axial [13], (c) Swin [11], (d) CSwin [3].

Attention Models	Complexity
ViT [4]	$\mathcal{O}(4HWC^2 + 2(HW)^2C) \sim \mathcal{O}(N^2)$
Axial [13]	$\mathcal{O}(HWC(4C + H + W)) \sim \mathcal{O}(N\sqrt{N})$
Swin [11]	$\mathcal{O}(4HWC^2 + 2P^2HWC) \sim \mathcal{O}(N)$
CSwin [3]	$\mathcal{O}(HWC(4C + sH + sW)) \sim \mathcal{O}(N\sqrt{N})$
MSwin (ours)	$\mathcal{O}(\frac{1}{3}k^3P^2HWC + \frac{2HWC^2k^2}{h}) \sim \mathcal{O}(N)$

polynomial time complexity–$\mathcal{O}(N^{1.5})$. In contrast, our proposed MSwin can aggregate features from multi-scale branches to increase fields in parallel, which has more symmetrical receptive fields and linear complexity with respect to N.

2.3 Architectural Configurations

Given all these definitions, the entire V2X-ViT model can be formulated as:

$$z_i = \text{PointPillar}(x_i), \qquad x_i \in \mathbb{R}^{P\times 4}, z_i \in \mathbb{R}^{H\times W\times C} \qquad \text{for agent } i \quad (10)$$

$$\mathbf{z}_0 = \text{STCM}([z_0, ..., z_M]) + \text{DPE}([\Delta t_0, ..., \Delta t_M]), \qquad \text{for ego AV} \quad (11)$$

$$\mathbf{z}'_\ell = \mathbf{z}_{\ell-1} + \text{MSwin}(\text{HSMA}(\text{LN}(\mathbf{z}_0))), \quad \mathbf{z}_0 \in \mathbb{R}^{M\times H\times W\times C} \qquad \ell = 1, ..., L \quad (12)$$

$$\mathbf{z}_\ell = \mathbf{z}'_\ell + \text{MLP}(\text{LN}(\mathbf{z}'_\ell)), \qquad \ell = 1, ..., L \quad (13)$$

$$\mathbf{y} = \text{Head}(\mathbf{z}_L), \qquad (14)$$

where the input x_i denotes the raw LiDAR point clouds captured on each agent, which are fed into the PointPillar Encoder [8], yielding visually informative 2D features z_i for each agent i. These tensors are then compressed, shared, decompressed, and further fed into the spatial-temporal correction module (STCM) to spatially warp the features. Then, we add the delay-aware positional encoded (DPE) features conditioned on each agent's time delay Δt_i to the output of STCM. Afterwords, the gathered features from M agents are processed using our proposed V2X-ViT, which consists of L layers of V2X-ViT blocks. Each V2X-ViT block contains a HSMA, a MSwin, and a standard MLP network [4]. Following [4,11], we use the Layer Normalization [1] before feeding into the Transformer/MLP module. We show the detailed specifications of V2X-ViT architecture in Table T2.

3 V2XSet Dataset

Statistics We gather 55 representative scenes covering 5 different roadway types and 8 towns in CARLA. Each scene is limited to 25 seconds, and in each scene, there are at least 2 and at most 7 intelligent agents that can communicate with each other. Each agent is equipped with 32-channel LiDAR and has 120 meters

Table T2: **Detailed architectural specifications for V2X-ViT.**

	Output size	V2X-ViT framework
PointPillar Encoder	$M \times 352 \times 96 \times 256$	[Voxel samp. reso. 0.4m, Scatter, 64] [Conv3x3, 64, stride 2, BN, ReLU] $\times$ 3 [Conv3x3, 128, stride 2, BN, ReLU] $\times$ 5 [Conv3x3, 256, stride 2, BN, ReLU] $\times$ 8 [ConvT3x3, 128, stride 1, BN, ReLU] $\times$ 1 [ConvT3x3, 128, stride 2, BN, ReLU] $\times$ 1 [ConvT3x3, 128, stride 4, BN, ReLU] $\times$ 1
	$M \times 176 \times 48 \times 256$	[Concat3, 384] [Conv3x3, 256, stride 2, ReLU Conv3x3, 256, stride 1, ReLU] $\times$ 1
Delay-aware Pos. Encoding	$M \times 176 \times 48 \times 256$	[sin-cos pos. encoding] [Linear, 256] $\times$ 1
Transformer Backbone	$M \times 176 \times 48 \times 256$	[HSMA, dim 256, head 8 MSwin, dim 256, head {16, 8, 4}, ws. $\{4 \times 4, 8 \times 8, 16 \times 16\}$ MLP, dim 256] $\times$ 3
Detection Head	$176 \times 48 \times 16$	Cls. head: [Conv1x1, 2, stride 1] Regr. head: [Conv1x1, 14, stride 1]

data range. We mount sensors on top of each AV while we only deploy infrastructure sensors in the intersection, mid-block, and entrance ramp at the height of 14 feet since these scenarios are typically more congested and challenging [5]. We record LiDAR point clouds at 10 Hz and save the corresponding positional data and timestamp.

Infrastructure deployment. The infrastructure sensors are installed on the traffic light poles or steet light poles at the intersection, mid-block, and entrance ramp at the height of 14 feet. For road type like rural curvy road, there is no infrastructure installed and only V2V collaboration exists.

Dataset visualization. As Fig. 2 displays, there are 5 different roadway types in V2XSet dataset (*i.e.*, straight segment, curvy segment, midblock, entrance ramp, and intersection), covering the most common driving scenarios in real life. We collect more intersection scenes than other types as it is usually more challenging due to the high traffic volume and severe occlusions. Data samples from different roadway types can be found in Fig. 3. From the figure, we can observe that the infrastructure sensors at the entrance ramp and intersection have different measurement patterns especially near its installation position compared with vehicle sensors. This is caused by the different installation heights between vehicle and infrastructure sensors. Such observation again shows the necessity of capturing the heterogeneity nature of V2X system.

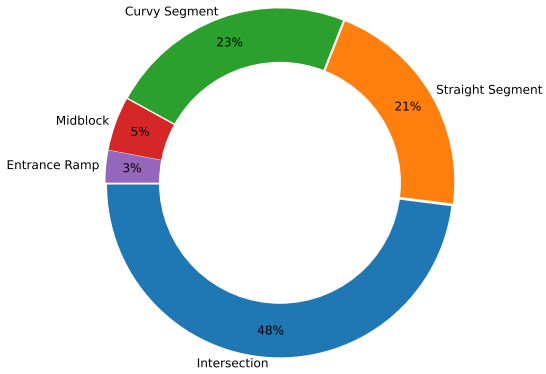


Fig. 2: Data distribution of 5 roadway types in the proposed dataset.

4 More Experimental Results

4.1 Performance for identifying dynamic objects

We group the test set based on object speeds v (km/h) and compare AP@IoU=0.7 under the noisy setting for all intermediate fusion models. As shown in Tab. T3, V2X-ViT outperforms all other intermediate fusion methods under various speed range. It is noticeable that the objects with higher speed range generally have lower AP scores as the same time delay can produce more positional misalignments for the high-speed vehicles.

Table T3: Perception performance for objects with different speed (km/h), measured in AP@0.7 under noisy setting.

Model	$v < 20$	$20 \leq v \leq 40$	$v > 40$
F-Cooper	0.539	0.487	0.354
OPV2V	0.552	0.498	0.346
V2VNet	0.598	0.518	0.406
DiscoNet	0.639	0.580	0.420
V2X-ViT	**0.693**	**0.634**	**0.488**

4.2 Performance for different road types

We also group the test scenes based on their road types and calculate the AP@IoU=0.7 scores under the noisy setting. As shown in Tab. T4, V2X-ViT ranks the first for all 5 road categories, demonstrating its detection robustness on different scenes.

Table T4: Perception performance for different road types, measured in AP@0.7 under noisy setting.

Model	Straight	Curvy	Intersection	Midblock	Entrance
F-Cooper	0.483	0.558	0.458	0.431	0.375
OPV2V	0.478	0.604	0.492	0.460	0.380
V2VNet	0.496	0.556	0.517	0.489	0.360
DiscoNet	0.519	0.594	0.572	0.472	0.440
V2X-ViT	**0.645**	**0.686**	**0.615**	**0.530**	**0.487**

4.3 Qualitative results

Figs. 4 to 6 demonstrate more detection visualizations of V2VNet[14], OPV2V [15], F-Cooper [2], DiscoNet [9], and our V2X-ViT in different scenarios under *Noisy Setting*. V2X-ViT yields more robust performance in general with fewer regression displacements and fewer undetected objects. When the scenario is challenging with high-density traffic flow and more occlusions (*e.g.*, Scene 7 in Fig. 6), our model can still identify most of the objects accurately.

4.4 Attention visualization

Fig. 7 shows more attention map visualizations of V2X-ViT under *noisy setting*. The LiDAR points of ego vehicle, the other connected autonomous vehicle (cav), and infrastructure are plotted in blue, green, and red respectively. The brighter color in the attention map means more attention ego vehicle pays. Generally, the color of infrastructure attention maps is brighter than others, especially for the occluded regions of other agents, indicating the more importance ego vehicle assigns to the infrastructure. This observation agrees with our intuition that the sensor observation of infrastructure has fewer occlusions, which leads to better feature representations.

4.5 Explanation on effects of transmission size

Here we provide more explanations of the data transmission size experiment in our paper. Different fusion strategies usually have distinct bandwidth requirements *e.g.*, early fusion requires large bandwidth to transmit raw data, whereas late fusion only delivers minimal size of data. This communication volume will significantly influence the time delay, thus we need to simulate a more realistic time delay setting to study the effects of transmission size.

Following [12], we decompose the total time delay into two parts: i) the data transmission time t_c during broadcasting, ii) the idle time t_i caused by the lack of synchronization between the perception system and communication system. The total time delay is calculated as

$$t_{total} = t_c + t_i \tag{15}$$

As mentioned in the paper, the data transmission time has

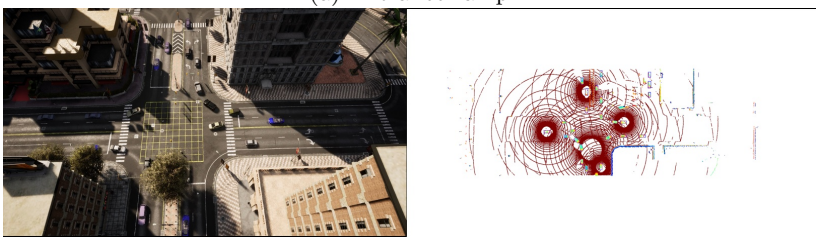
$$t_c = f_s/v \tag{16}$$

where f_s is the data size and v is the transmission rate. Idle time t_i can be further decoupled into the idle time on the sender side and the time on the receiver side *i.e.*, $t_i = t_{i,1} + t_{i,2}$. For $t_{i,1}$, the worst case in terms of delay happens when the communication system just misses a perception cycle and needs to wait for the next round. Similarly, for $t_{i,2}$, the worst case occurs when new data is received just after a new cycle of the perception system has started. Assume both perception system and communication system have the same rate of $10Hz$, then $0\ ms < t_i < 200\ ms$. We employ a uniform distribution $\mathcal{U}(0, 200)$ to model this uncertainty. In summary, we use the following equation to mimic the real-world time delay.

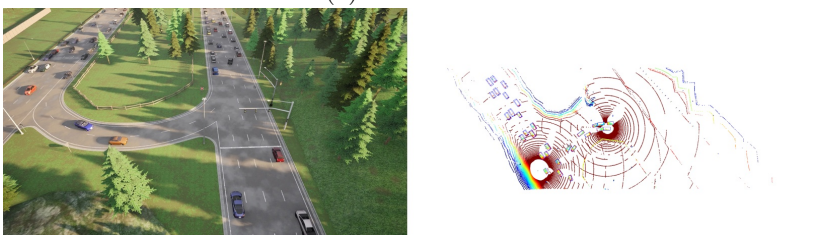
$$t_c = f_s/v + \mathcal{U}(0, 200) \tag{17}$$

which captures the influence of transmission size and asynchrony-caused uncertainty. In practice, we sample the time delay according to Eq. 16 and discretize it to the observed timestamps, which are discrete in a 10Hz update system.

(a) Entrance ramp



(b) Intersaction



(c) Mid-block

(d) Rural curvy road

(e) Urban curvy street

Fig. 3: **Data samples of 5 different roadway types.** Left is the snapshot of simulation and right is the corresponding aggregated LiDAR point clouds from multiple agents.

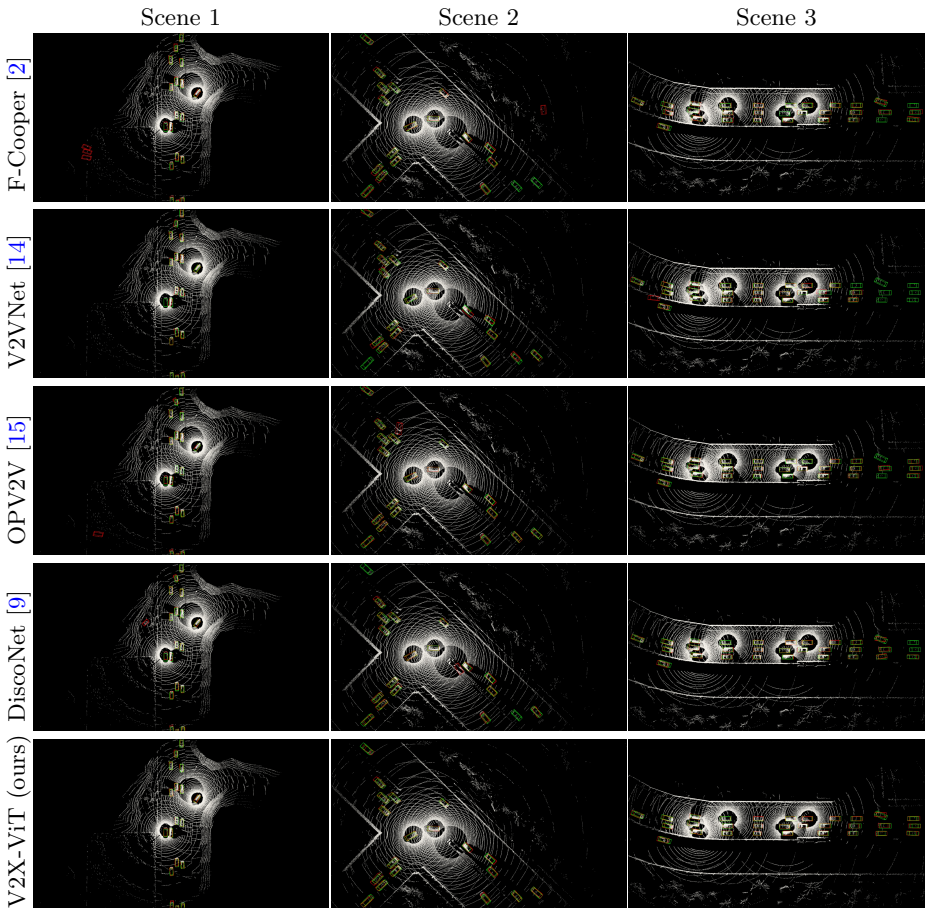


Fig. 4: **Qualitative comparison on scenarios 1-3.** Green and red 3D bounding boxes represent the groun truth and prediction respectively. Our method yields more accurate detection results.

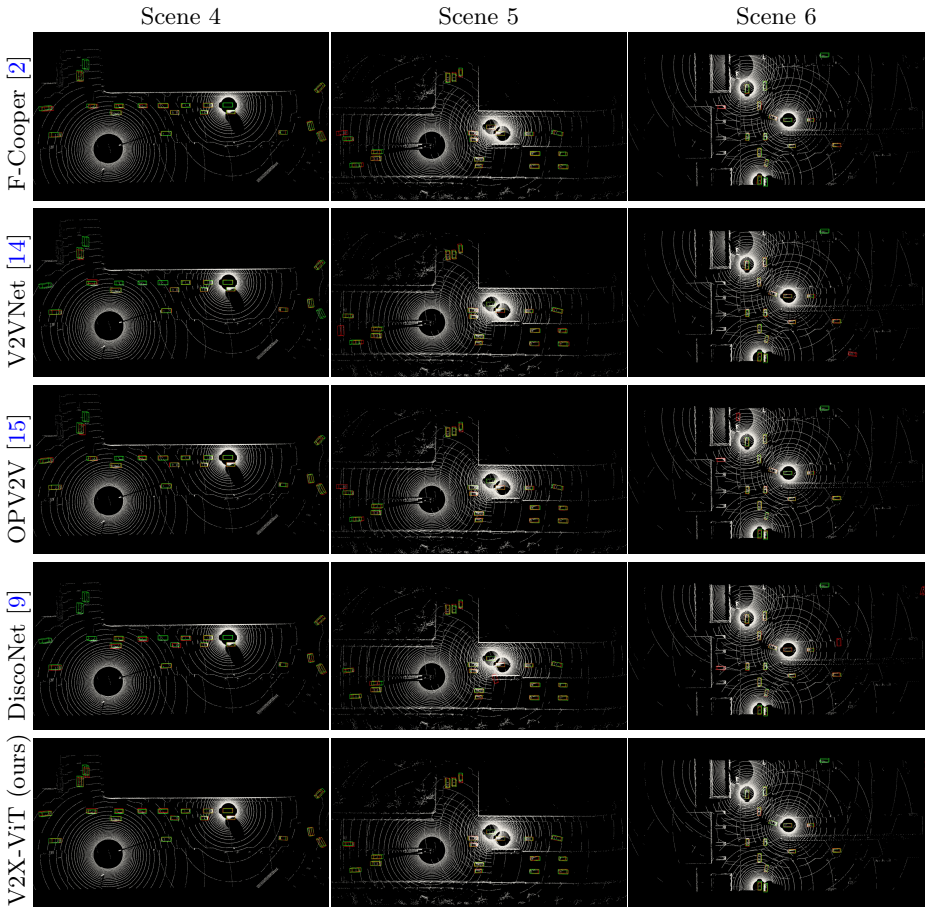


Fig. 5: **Qualitative comparison on scenarios 4-6.** Green and red 3D bounding boxes represent the groun truth and prediction respectively.

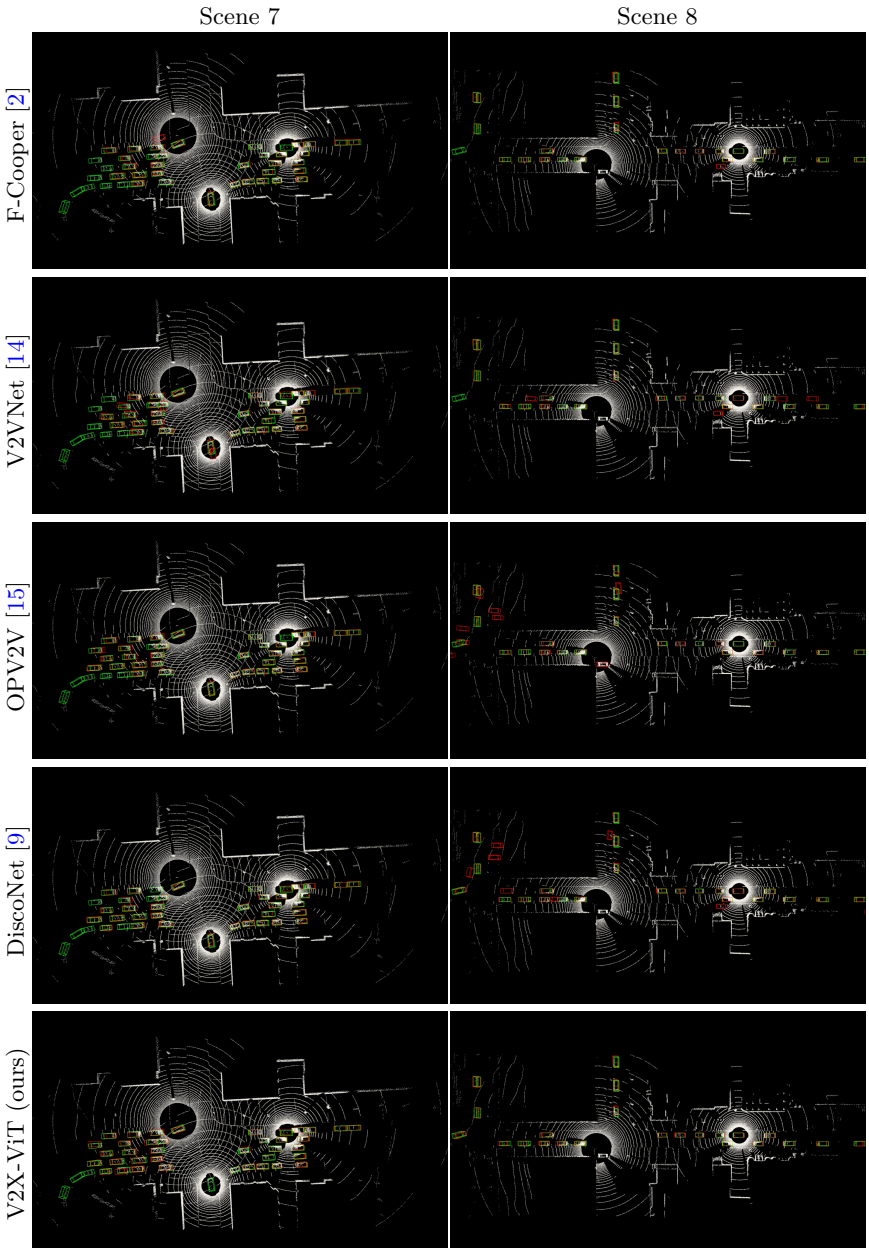
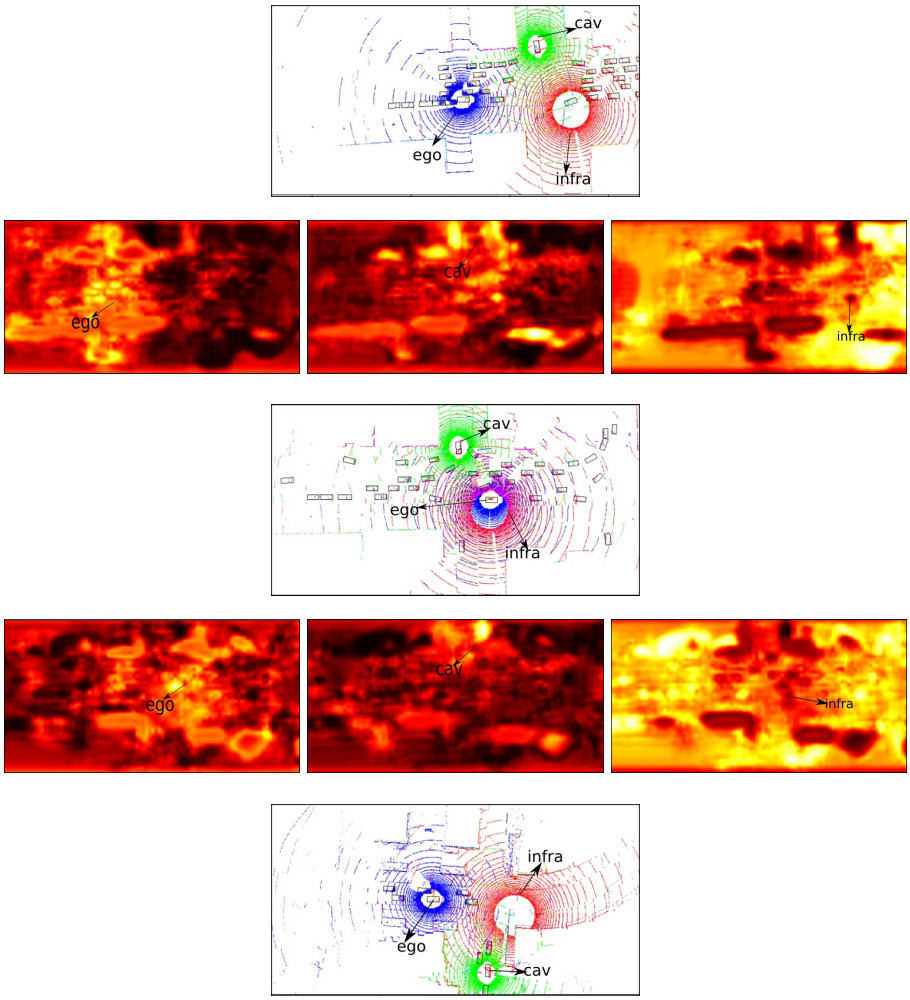
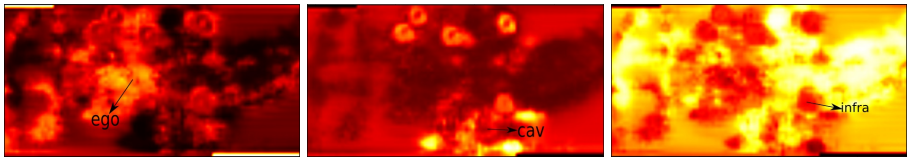


Fig. 6: **Qualitative comparison on scenarios 7-8.** Green and red 3D bounding boxes represent the groun truth and prediction respectively.



(a) LiDAR point clouds



(b) Attention ego→ego (c) Attention ego→cav (d) Attention ego→infra

Fig. 7: **Additional attention map visualizations on 3 different scenes.** V2X-ViT learned to pay more attention to infra features on occluded areas from AV's perspectives, thus yielding more robust detection under occlusions.

References

1. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint arXiv:1607.06450 (2016) 5
2. Chen, Q., Ma, X., Tang, S., Guo, J., Yang, Q., Fu, S.: 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. pp. 88–100 (2019) 8, 11, 12, 13
3. Dong, X., Bao, J., Chen, D., Zhang, W., Yu, N., Yuan, L., Chen, D., Guo, B.: Cswin transformer: A general vision transformer backbone with cross-shaped windows. arXiv preprint arXiv:2107.00652 (2021) 4, 5
4. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020) 4, 5
5. Guo, Y., Ma, J., Leslie, E., Huang, Z.: Evaluating the effectiveness of integrated connected automated vehicle applications applied to freeway managed lanes. IEEE Transactions on Intelligent Transportation Systems (2020) 6
6. Hu, H., Zhang, Z., Xie, Z., Lin, S.: Local relation networks for image recognition. In: ICCV. pp. 3464–3473 (2019) 3
7. Jaderberg, M., Simonyan, K., Zisserman, A., et al.: Spatial transformer networks. In: NeurIPS (2015) 2
8. Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: Pointpillars: Fast encoders for object detection from point clouds. In: CVPR. pp. 12697–12705 (2019) 5
9. Li, Y., Ren, S., Wu, P., Chen, S., Feng, C., Zhang, W.: Learning distilled collaboration graph for multi-agent perception. NeurIPS **34** (2021) 8, 11, 12, 13
10. Liu, Y.C., Tian, J., Ma, C.Y., Glaser, N., Kuo, C.W., Kira, Z.: Who2com: Collaborative perception via learnable handshake communication. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). pp. 6876–6883. IEEE (2020) 1
11. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. arXiv preprint arXiv:2103.14030 (2021) 3, 4, 5
12. Rauch, A., Klanner, F., Dietmayer, K.: Analysis of v2x communication parameters for the development of a fusion architecture for cooperative perception systems. In: 2011 IEEE Intelligent Vehicles Symposium (IV). pp. 685–690. OPTorganization (2011) 8
13. Wang, H., Zhu, Y., Green, B., Adam, H., Yuille, A., Chen, L.C.: Axial-deeplab: Stand-alone axial-attention for panoptic segmentation. In: European Conference on Computer Vision. pp. 108–126. Springer (2020) 4, 5
14. Wang, T.H., Manivasagam, S., Liang, M., Yang, B., Zeng, W., Urtasun, R.: V2vnet: Vehicle-to-vehicle communication for joint perception and prediction. In: ECCV. pp. 605–621. Springer (2020) 8, 11, 12, 13
15. Xu, R., Xiang, H., Xia, X., Han, X., Liu, J., Ma, J.: Opv2v: An open benchmark dataset and fusion pipeline for perception with vehicle-to-vehicle communication. arXiv preprint arXiv:2109.07644 (2021) 1, 8, 11, 12, 13
16. Zhang, H., Wu, C., Zhang, Z., Zhu, Y., Lin, H., Zhang, Z., Sun, Y., He, T., Mueller, J., Manmatha, R., et al.: Resnest: Split-attention networks. arXiv preprint arXiv:2004.08955 (2020) 3