V2XPnP: Vehicle-to-Everything Spatio-Temporal Fusion for Multi-Agent Perception and Prediction

Supplementary Material

Contents

A. Implementation Details

In this section, we provide detailed configurations for cooperative perception and prediction tasks, including the baseline models used in our experiments and the proposed V2XPnP framework.

A.1. Benchmark Model Details

PointPillar Backbone. For all experiments, we employ the anchor-based PointPillar model [8] as the LiDAR Feature Extraction backbone. The voxel resolution is set to 0.4 meters in both the x and y directions, with a maximum of 32 points per voxel and a total of 32,000 voxels. Additionally, we set the number of anchors per grid cell to 2.

Intermediate Fusion Methods. We implement several state-of-the-art single-frame intermediate fusion methods, including *V2VNet* [13], *F-Cooper* [2], *DiscoNet* [10],*CoBEVFlow* [14], *FFNet* [21], *V2X-ViT* [16], and our proposed V2XPnP model, integrating them with our end-to-end model to replace the spatio-temporal fusion module. The model settings and configurations for the fusion module adhere to the original implementations.

Map Feature Extraction. HD maps are represented as sets of polylines, with each polyline comprising 10 points. Because the map is projected onto the BEV space, each grid only contains the five nearest polylines. Each waypoint in a polyline contains seven attributes: $(x, y, d_x, d_y, type, x_{pre}, y_{pre})$, representing position, direction, lane type, and previous position. These attributes are encoded using MLP layers into a 256 hidden dimension feature, followed by 1 or 2 Transformer layers with two attention heads to model interactions among map elements.

Decoupled Attention Predictor. For the decoupled perception and prediction pipeline, we implement an attention-based predictor for trajectory-level prediction tasks. This predictor utilizes a 1D Convolution + LSTM Network [4] to encode temporal historical trajectories and a Transformer layer to capture the interaction among objects and the map, then an LSTM-based decoder generates the future predicted trajectories. All trajectory data, including historical and predicted trajectories, are represented in the local coordinate frame of each object.

A.2. V2XPnP Model Details

Temporal Attention. To capture the temporal dependence, we initialize the historical timestamp sequence using Sinusoidal positional encodings conditioned on time and further process these encodings through a Linear layer. The temporal attention block in the multi-frame temporal fusion module has four attention heads. To enhance the inter-frame feature representation, we stack three temporal fusion modules with the temporal attention block.

Self-spatial Attention. This block is applied following either the temporal attention or the multi-agent spatial attention. In self-spatial attention, the feature map is partitioned into patches using common window sizes of $(2, 4, 8)$. Given the complexity of spatio-temporal fusion across multiple agents, the self-spatial attention module employs a higher number of attention heads $(16, 8, 4)$ after multi-agent spatial fusion, compared to the heads $(8, 4, 2)$ used following temporal attention.

Multi-agent Spatial Attention. Our dataset categorizes agents as infrastructure agents, denoted by negative labels (*i.e.*, -1 and -2), or connected automated vehicles (CAV)

agents, denoted by positive labels (*i.e.*, 1 and 2). To capture the heterogeneous dependencies among these agents, we construct a heterogeneous graph and employ distinct attention fusion parameters for each agent type. The multi-agent spatial attention utilizes eight attention heads, and we stack three multi-agent spatial fusion modules with the multi-agent spatial attention to capture the inter-agent relationships.

A.3. Loss Function

This section provides the loss function employed in our multi-task model. The initial weights of regression loss $\mathcal{L}_{\text{reg}}$, classification loss $\mathcal{L}_{\text{cla}}$ and prediction loss $\mathcal{L}_{\text{pred}}$ are set as $w_{\text{reg}}, w_{\text{cla}}, w_{\text{pred}} = 2.0, 1.0, 2.0$. For single-task learning, the same loss function is used but weights exclusively on the components relevant to that task.

Perception Loss. The perception task loss combines classification and regression components, designed to align predicted anchor boxes with ground truth labels. For classification, which involves identifying objects and background elements, we employ Focal Loss [12] to address the imbalance between foreground and background samples. The Focal Loss is expressed as:

$$\mathcal{L}_{\text{cla}} = -\alpha(1-p_t)^{\gamma}\log(p_t), \tag{S1}$$

where p_t is the predicted probability for the target anchor box, and α and γ are balancing and focusing factors. Anchor-wise weights are applied to further enhance the balance between positive and negative samples.

For the regression component, we employ Smooth ℓ_1-Loss to optimize the predicted bounding boxes to match the ground truth labels in terms of position and orientation, and a sine-cosine encoding is employed to handle rotational ambiguities. The Smooth ℓ_1-Loss is defined as:

$$\mathcal{L}_{\text{reg}} = \begin{cases} 0.5 \cdot \frac{\Delta^2}{\beta}, & \text{if} |\Delta| < \beta, \\ |\Delta| - 0.5 \cdot \beta, & \text{otherwise}, \end{cases} \tag{S2}$$

where $\Delta = \text{prediction} - \text{target}$, and β is a hyper-parameter controlling the transition between ℓ_1 and ℓ_2 loss.

Prediction Loss. We adopt the ℓ_2-loss function to minimize the discrepancy between the predicted trajectory and the ground truth.

$$\mathcal{L}_{\text{pred}} = \frac{1}{N_{\text{det}}}\frac{1}{T_{\text{valid}}}\sum_{i=1}^{N_{\text{det}}}\sum_{t=1}^{T_{\text{valid}}}\|\boldsymbol{\mu}_t^i - \mathbf{x}_t^i\|^2, \tag{S3}$$

where $\boldsymbol{\mu}_t^i$ and $\mathbf{x}_t^i$ represent the predicted position and target position of the i-th object at time step t. T_{valid} is the number of valid future time steps for the agent, and N_{det} is the number of detected objects.

A.4. Training Strategy

The end-to-end cooperative perception and prediction model addresses two distinct yet interrelated tasks while integrating information across both temporal and spatial dimensions. Training such an end-to-end model from scratch often results in suboptimal performance, due to the inherent complexity of jointly optimizing these tasks and dimensions. To effectively handle these challenges, we adopt a multi-stage training strategy to progressively refine the model's capabilities.

Multi-Stage Training Strategy. Initially, the end-to-end perception and prediction model is trained in a single-agent setting, focusing on temporal fusion without incorporating multi-agent spatial fusion. It simplifies the optimization process, enabling the model to learn robust temporal features in isolation. The resulting single-agent model then serves as a pre-trained model for subsequent multi-agent spatial fusion training in the V2X environment. This staged training strategy ensures that the model incrementally acquires the ability to handle the complexities of cooperative perception and prediction tasks.

Stage 1: Single-Agent Multi-task Learning. The single-agent model training stage addresses the core challenge of coordinating multi-task learning to capture complex patterns across perception and prediction tasks. Prediction task requires a comprehensive understanding of objects' temporal information and their intricate motion patterns, while detection focuses mainly on identifying objects in the current frame, with historical information providing supplementary context. Training both tasks jointly without proper initialization risks overfitting to simpler current-frame features, thereby neglecting the rich but complex temporal features essential for accurate prediction. Moreover, perception is foundational to prediction, as detecting an object is a prerequisite for predicting its motion. To effectively balance the two tasks, we adopt a task-specific training strategy. *(1) Single-Frame Perception Training:* the training begins by optimizing the model for single-frame perception, establishing a foundation for object detection. *(2) Temporal Prediction Training:* the prediction task is introduced by freezing the parameters of the detection backbone and training an additional temporal network and prediction head, guiding the model to focus more on the prediction task and effectively learn complex temporal dependencies. *(3) Joint Fine-Tuning:* the entire model is unfrozen, enabling end-to-end fine-tuning across both tasks.

Stage 2: Multi-Agent Spatiotemporal Learning. Based on the pre-trained single-agent model, the multi-agent fusion module is introduced and jointly trained with the entire model. At this stage, the primary focus is to balance the two tasks, ensuring that neither perception nor prediction dominates the training process. To achieve this, we employ a dynamic loss-weighting strategy that gradually increases the weight assigned to the prediction loss. This approach ensures balanced optimization, avoiding performance trade-offs between tasks and improving overall effectiveness across both perception and prediction objectives.

Table S1. Additional benchmark results of cooperative perception and prediction models on V2XPnP Sequential (V2XPnP-Seq) Dataset

Dataset	Method	E2E	Map	**AP@0.5** (%) ↑	ADE (m) ↓	FDE (m) ↓	MR (%) ↓	**EPA** (%) ↑
V2XPnP-Seq-VC *(with V+2I at most)*	V2VNet* [13]	✓		48.6	2.10	3.75	42.3	25.3
	F-Cooper* [2]	✓	✓	66.0	1.35	2.56	36.1	38.7
	DiscoNet* [10]	✓	✓	66.8	1.41	2.62	34.4	42.8
	V2XPnP (Ours)	✓	✓	**71.6**	1.35	2.36	31.7	**48.2**
V2XPnP-Seq-IC *(with 2V+I at most)*	V2VNet* [13]	✓		33.6	1.95	3.53	44.2	16.3
	F-Cooper* [2]	✓	✓	60.2	1.21	2.32	36.3	36.3
	DiscoNet* [10]	✓	✓	65.4	1.14	2.18	36.1	40.7
	V2XPnP (Ours)	✓	✓	**71.0**	1.18	2.16	34.0	**46.0**
V2XPnP-Seq-V2V	V2VNet* [13]	✓		43.1	3.10	5.55	46.8	19.4
	F-Cooper* [2]	✓	✓	60.2	1.69	3.22	41.1	34.4
	DiscoNet* [10]	✓	✓	61.2	1.66	3.13	41.2	33.1
	V2XPnP (Ours)	✓	✓	**70.5**	1.78	3.28	39.9	**40.6**
V2XPnP-Seq-I2I	V2VNet* [13]	✓		41.1	1.83	3.34	40.4	23.2
	F-Cooper* [2]	✓	✓	58.6	1.34	2.58	40.0	33.6
	DiscoNet* [10]	✓	✓	63.5	1.15	2.19	37.5	38.4
	V2XPnP (Ours)	✓	✓	**69.2**	1.26	2.31	36.5	**42.8**

Training Details. The model is trained using the Adam optimizer [7] with an initial learning rate of 2×10^{-3} and a weight decay of 1×10^{-4} with early stopping on NVIDIA L40S GPUs. We employ 4 training stages, as detailed before, and each training stage consists of 30 epochs with a batch size of 2. Early stopping is employed to prevent overfitting. We carefully tune the hyperparameters to ensure the stability and efficiency of the training process.

B. Additional Benchmark Results

In this paper, we benchmark different spatiotemporal strategies with 11 fusion models in total:

- **No Fusion**: *No Fusion*, *No Fusion-FaF*
- **Early Fusion**: *Early Fusion*
- **Late Fusion**: *Late Fusion*
- **Intermediate Fusion**: *V2VNet* [13], *F-Cooper* [2], *DiscoNet* [10], *CoBEVFlow* [14], *FFNet* [21], *V2X-ViT* [16], and our proposed *V2XPnP*.

We present additional benchmark results for *V2VNet* [13], *F-Cooper* [2], and *DiscoNet* [10] across all collaboration modes, as shown in Tab. S1. Our proposed *V2XPnP* consistently outperforms these SOAT baselines in terms of EPA and AP across all collaboration modes. Notably, V2VNet* exhibits lower performance, likely due to the absence of a map and the loss of temporal features during explicit feature ROI matching.

C. Cooperative Temporal Perception Task

In addition to the end-to-end perception and prediction task, the sequential nature of our V2XPnP-Sequential dataset facilitates other temporal tasks, including temporal perception and traditional prediction tasks. In this section, we introduce the cooperative temporal perception task and present benchmark results on the V2XPnP-Sequential dataset. Details on the traditional prediction task are provided in Sec. D.

C.1. Problem Formulation

The cooperative temporal perception task is an extension of the single-frame perception task by incorporating historical context. Specifically, given historical T frames raw perception data $\mathbf{P}_i^t, i \in \{1, \cdots, N\}$ from all N agents within the communication range of the ego agent, the objective is to detect the surrounding objects in the current frame. The core challenge lies in effectively leveraging temporal information from T past frames to enhance detection accuracy in the present frame.

C.2. Benchmark Methods

For benchmarking, we adapt our end-to-end model, *V2XPnP*, by removing the prediction head, resulting in a model only for temporal perception. Various V2X fusion strategies are evaluated in this framework, as detailed in Tab. S2. Moreover, we provide another baseline *FaF**, which adopts a combination of 2D and 3D convolutions for temporal fusion. *FaF** further integrates with the F-Cooper intermediate fusion method and early fusion method for V2X fusion comparison. We also provide the results of the *No Temp* model, which excludes temporal fusion and is evaluated using both F-Cooper and early fusion methods. Model parameters and experimental setups for this task are consistent with those used for the end-to-end cooperative perception and prediction task.

Table S2. Benchmark results for cooperative temporal perception. No Temp: single-frame perception, FaF*: temporal perception with alternating 2D and 3D convolutions, V2XPnP: temporal perception with temporal attention modules.

Dataset	No Fusion (AP@0.5 (%) ↑)			Early Fusion (AP@0.5 (%) ↑)			Intermediate Fusion (AP@0.5 (%) ↑)		
	No Temp	FaF*	V2XPnP	No Temp	FaF*	V2XPnP	No Temp	FaF*	V2XPnP
V2XPnP-Seq-VC	43.9	57.1	**60.3**	63.5	67.0	**71.0**	65.1	70.3	**74.0**
V2XPnP-Seq-IC	46.4	61.1	**64.7**	61.0	65.5	**71.4**	61.1	67.1	**73.2**
V2XPnP-Seq-V2V	40.8	53.7	**59.1**	54.9	56.4	**66.6**	58.0	61.4	**69.4**
V2XPnP-Seq-I2I	51.0	61.2	**64.7**	63.4	66.0	**71.6**	58.5	62.9	**72.4**

C.3. Benchmark Results

The results demonstrate that incorporating temporal cues significantly improves perception performance across all multi-agent fusion strategies. Notably, our *V2XPnP* model achieves superior results compared to other baselines, due to the careful design of temporal attention. However, we observe a slight performance drop when the same model is applied to the end-to-end cooperative perception and prediction task, compared to its use solely for temporal perception. The possible reason is the difficulty of optimizing both tasks to achieve optimal performance. Nevertheless, the end-to-end model still outperforms other baselines in both perception and prediction tasks. Future research should focus on optimizing the balance between multiple tasks to further enhance the performance of end-to-end models.

D. Traditional Cooperative Prediction Task

D.1. Problem Formulation

V2XPnP sequential dataset also supports the traditional prediction task. Compared to end-to-end models, which directly infer future states of objects from perception data, the traditional prediction task forecasts their future trajectories from historical trajectories. The cooperative prediction task is formulated as follows: given the map and the historical trajectories of all detected objects obtained from the ego agent and other agents (*e.g.*, CAVs and infrastructure units) within the communication range of the ego agent, the objective is to predict future trajectories of these detected objects.

D.2. Benchmark Methods

To investigate the influence of perception results on prediction tasks, we provide two types of input for the prediction models: 1) Ground-truth historical trajectories of surrounding objects; 2) Perception-based historical trajectories generated by the upstream perception module. The first one is the common setting for the traditional trajectory prediction task, assuming full availability of accurate historical trajectories for prediction. However, it ignores real-world challenges such as occlusions and cumulative errors introduced by separate modules. To address this limitation and enable a more realistic evaluation, we designed the second setting, where CAVs can only derive the historical trajectories from the perception results, and thus the perception uncertainty can propagate to the downstream prediction. Notably, regardless of the input type, the prediction model is trained using the complete future trajectory dataset aggregated from all agents.

In our experiment, the prediction model configuration and experimental setup align closely with the decoupled attention predictor. Following the LSTM baseline setting in the Waymo motion dataset [5], the LSTM model also serves as a strong baseline, which includes an LSTM encoder and LSTM decoder. We report benchmark results under three configurations: *No Fusion*, where no perception information is fused; *Ground Truth*, assuming perfect historical trajectories; *Late Fusion*, where the decoupled pipeline from the traditional prediction task is employed.

D.3. Benchmark Results

The experimental results, summarized in Tab. S3, compare traditional prediction under three input settings: ground truth trajectories, perception without fusion, and perception with late fusion. The results indicate that as perception improves—from no fusion to late fusion—the prediction performance correspondingly increases. When the environment is fully observable, the task simplifies to the traditional prediction setup, achieving the best overall performance for both detection and prediction. A significant drop in performance is observed for perception-based prediction, highlighting the critical dependency of predictive tasks on perception accuracy. Moreover, the Attention predictor shows better robustness compared to the LSTM baseline under noisy perception inputs, thanks to the attention module for complex interaction feature capturing. We anticipate that this temporal prediction task will inspire further exploration of perception-based prediction approaches.

Table S3. Benchmark results for traditional prediction. No Fusion: prediction based on the no-fusion perception results. Late Fusion: prediction based on the late fusion perception results. Ground Truth: prediction based on the ground truth trajectories with no occlusion or perception errors.

Dataset	Method	Attention Predictor				LSTM Predictor			
		AP@0.5(%) ↑	ADE(m) ↓	FDE(m) ↓	MR(%) ↓	AP@0.5(%) ↑	ADE(m) ↓	FDE(m) ↓	MR(%) ↓
V2XPnP-Seq -VC	No Fusion	43.9	1.87	3.24	33.8	43.9	2.91	4.77	35.0
	Late Fusion	58.1	1.59	2.81	34.3	58.1	2.76	4.60	33.7
	Ground Truth	-	0.60	1.26	23.0	-	0.66	1.31	23.0
V2XPnP-Seq -IC	No Fusion	46.4	2.10	3.75	42.3	46.4	2.11	3.67	35.8
	Late Fusion	55.9	1.39	2.44	30.1	55.9	2.61	4.40	32.7
	Ground Truth	-	0.63	1.35	26.2	-	0.61	1.31	25.0
V2XPnP-Seq -V2V	No Fusion	40.8	1.99	3.38	34.0	40.8	2.98	4.82	34.4
	Late Fusion	55.3	1.75	3.07	34.0	55.3	2.87	4.79	35.0
	Ground Truth	-	0.60	1.26	22.9	-	0.66	1.31	22.8
V2XPnP-Seq -I2I	No Fusion	51.0	1.69	3.06	36.2	51.0	2.11	3.67	35.9
	Late Fusion	61.3	1.41	2.50	30.0	61.3	2.44	4.18	32.1
	Ground Truth	-	0.63	1.35	26.2	-	0.61	1.31	25.0

E. V2XPnP Sequential Dataset Details

E.1. Dataset Visualization

Our V2XPnP-Sequential dataset provides two sensor sequences (LiDAR and camera) collected in dense urban environments, capturing diverse interactive behaviors over time. Fig. S1 illustrates two representative interaction scenarios in our dataset, presenting LiDAR and camera data from multiple agents at two key timestamps. The main intersection objects pair has been annotated with red and yellow blocks in different agents' views.

E.2. Data Acquisition

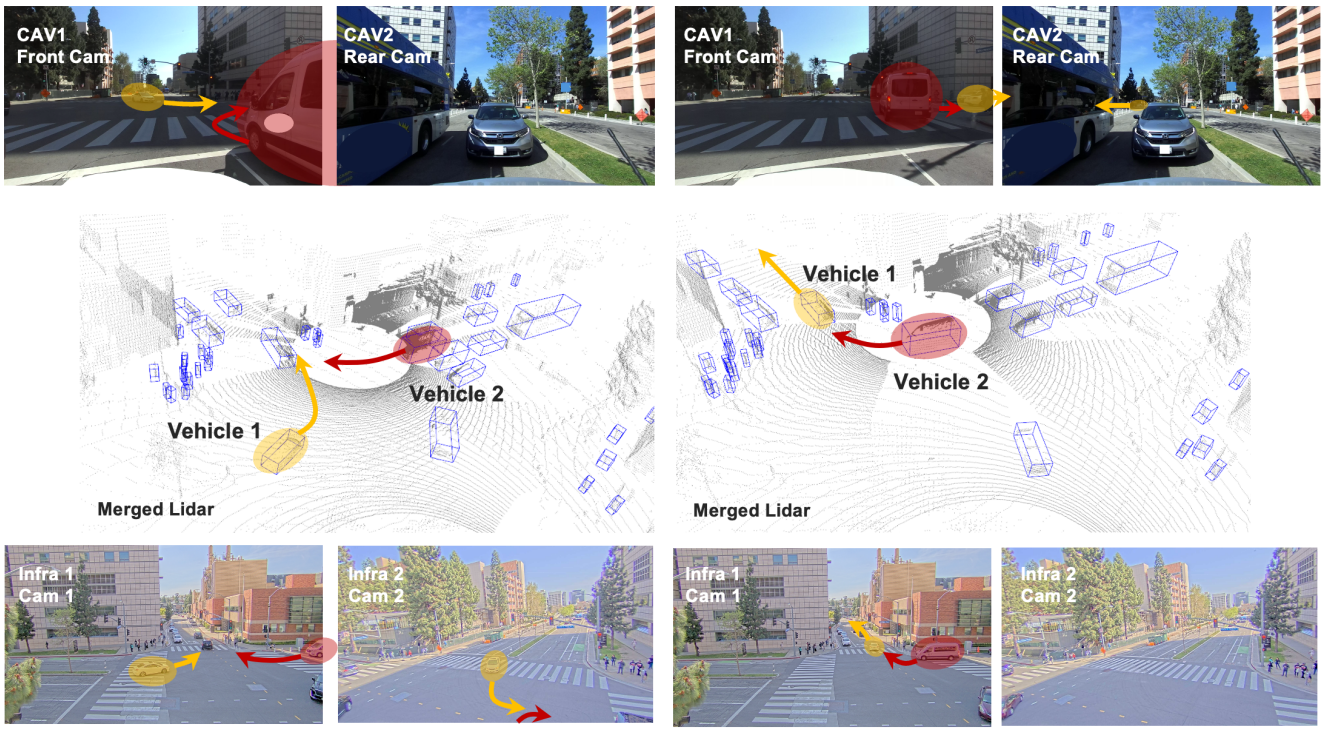
Sensor Specifications. The dataset was collected using four agents - two connected automated vehicles and two smart infrastructure units. Each CAV is equipped with a RoboSense 128-beam LiDAR, four stereo RGB cameras with 1920 × 1080 resolution, and an integrated GPS/IMU system. The four stereo cameras are mounted on the front, rear, left, and right sides of the CAV, providing a complete 360-degree field of view. Similarly, each infrastructure unit is configured with a 128- or 64-beam LiDAR, two Axis cameras with 1920 × 1080 resolution, and a GPS module. The sensor deployment of our data collection system is shown in Fig. 4(a).

Coordinate System. Our V2XPnP-Sequential dataset encompasses three coordinate systems: the LiDAR coordinate system, the camera coordinate system, and the map coordinate system. Each agent (vehicle or infrastructure) maintains its own local LiDAR and camera coordinate systems. The global map coordinate system serves as the reference for all annotations and maps. The transformation from each agent's local LiDAR coordinate to the map coordinate in each frame is achieved with the GPS/IMU data and the offline PCD map. We also conduct the 3D-2D calibration for LiDAR and camera, as shown in Fig. 4(b).

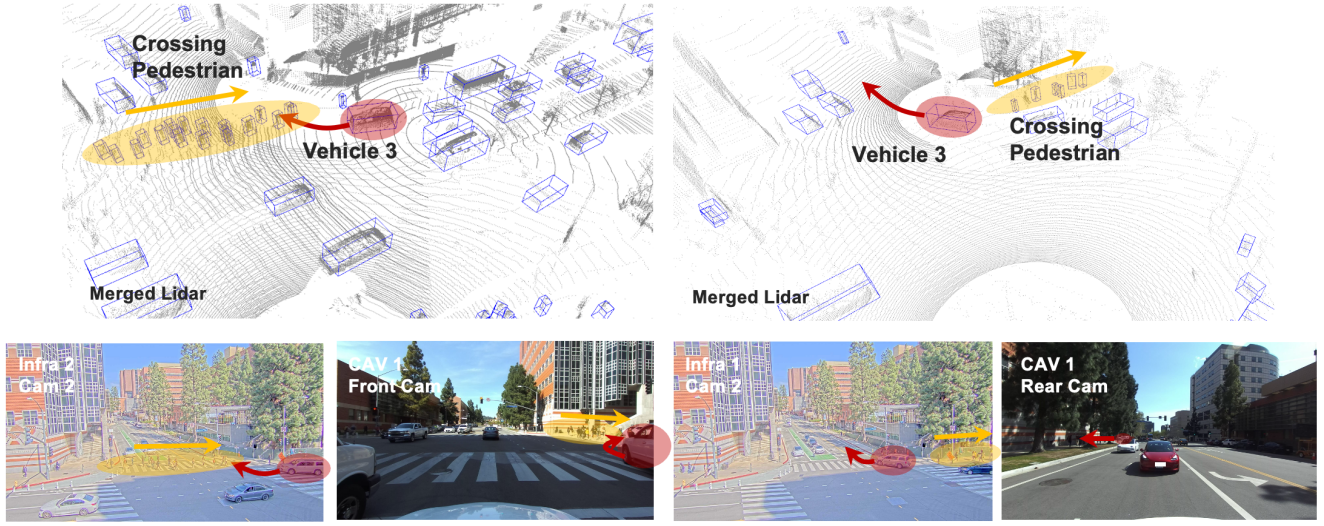
E.3. Data Annotation and Processing

Data Annotation. The 3D bounding boxes in our V2XPnP-Sequential dataset are annotated using an open-source labeling tool, SUSTechPOINTS [9], by expert annotators. The first step is annotating the bounding boxes in the point clouds from the two CAVs and infrastructure units. Then, these bounding boxes, annotated in different agents' coordinate frames, are processed through a V2X sequential pipeline to assign consistent object IDs across agents and temporal frames. To ensure annotation quality, each object is subjected to eight rounds of review and revision. In total, ten object categories are included in our dataset: car, pedestrian, scooter, motorcycle, bicycle, truck, van, concrete truck, bus, and road barrier. Each object annotation includes the center of the bounding box (x, y, z), sizes $(width, length, height)$, and orientation $(roll, yaw, pitch)$ in the global coordinates. Notably, we follow a general object definition in annotation, encompassing stationary objects such as parked vehicles and barriers, which are annotated similarly to movable objects but explicitly labeled as static. This aligns with public datasets like nuScenes [1], where static objects are tracked while maintaining consistent IDs.

Trajectory Generation. In addition to perception data, the dataset provides a ground-truth trajectory dataset derived from the fused perception data of all agents, capturing the trajectories of objects across all frames. This trajectory dataset is primarily utilized in traditional prediction tasks, which assume all history trajectories are observable to the ego agent. However, this assumption ignores the fact that the trajectories obtained from onboard sensors are incomplete due to occlu-



(a) Scene 1: A vehicle slows to wait for a turning flow in its opposite lane.



(b) Scene 2: A vehicle waits for a crossing pedestrian flow and then continue to turn.

Figure S1. Examples of interaction scenarios from the V2XPnP-Sequential dataset. The dataset provides multi-agent perception perspectives and captures diverse interaction behaviors among ten object classes in dense urban traffic environments.

Table S4. Comparison between the V2XPnP-Sequential dataset and other public available driving datasets

Dataset	Year	Type	V2V	V2I	I2I	Trajectory	Map	Agent Number	Tracked Objects/Scene	3D Boxes	RGB Images	LiDAR Frames	Categories
nuScenes [1]	2019	Real				✓	✓	1	75.75	1.4M	1.4M	400k	23
Waymo Open [5]	2019	Real				✓	✓	1	-	12M	1M	200k	4
OPV2V [17]	2022	Sim	✓					2.89	26.5	230k	44k	11k	1
V2X-Sim [11]	2022	Sim	✓	✓		✓		10	-	26.6k	0	10k	1
V2XSet [16]	2022	Sim	✓	✓		✓		2-7	-	230k	44k	11k	1
DAIR-V2X [19]	2022	Real		✓				2	0	464k	39k	39k	10
V2V4Real [18]	2023	Real	✓			✓	✓	2	-	240k	40k	20k	5
V2X-Seq [20]	2023	Real		✓		✓	✓	2	110	464k	71k	-	10
RCooper [6]	2024	Real			✓	✓		4	-	-	50k	30k	10
V2X-Real [15]	2024	Real	✓	✓	✓			4	0	1.2M	171K	33k	10
V2XPnP-Seq	**2024**	**Real**	✓	✓	✓	✓	✓	**4**	**136**	**1.45M**	**208k**	**40K**	**10**

sion and limited perception range, and no specific datasets are designed to support this task. To support research in prediction with real-world sensor constraints, we provide a trajectory retrieval module in the V2XPnP-Sequential dataset to return observable trajectories of surrounding objects based on their actual visibility relationships.

Map Generation. The HD map generation involves two stages: point cloud (PCD) map generation and vector map generation. (1) To generate the PCD map, each LiDAR frame from the CAVs is preprocessed to remove dynamic objects, retaining only static elements essential for mapping. Then, a Normal Distribution Transform (NDT) scan-matching algorithm is employed to compute the relative transformation between consecutive frames, forming the basis of the LiDAR odometry. We also incorporate translation and heading information obtained from the vehicle's GPS/IMU system, integrating them through a Kalman filter to refine the pose estimation, mitigating the drift from the error accumulation in LiDAR data. Finally, the LiDAR sequences are fused to form the PCD map across all collection areas. (2) The aggregated PCD map is imported into RoadRunner [3] to generate vector maps. Road geometry is inferred and annotated based on intensity variations visualized by distinct color mappings within RoadRunner, and all the semantic attribution is annotated based on the collected camera data, such as road type (*e.g.*, driving, sidewalk, and parking) and line type (*e.g.*, solid and broken yellow line combination and solid white line). Finally, the generated maps are exported in the OpenDRIVE (Xodr) format and converted to Waymo map format [5], ensuring compatibility with downstream applications.

E.4. Dataset Analysis

Tab. S4 presents the comparison of the V2XPnP-Sequential dataset with existing driving datasets. Our dataset tracks an average of 136 objects per scene, recording high-density and complex traffic scenarios. Furthermore, the dataset's extensive map and trajectory data further enhance its utility in cooperative perception and prediction research across all collaboration modes. The data distribution of ten object classes is shown in Fig. 4(d). The dataset covers 24 intersections of varying types, including roundabouts, T-junctions, and crossroads, as shown in Fig. S2. Notably, many collection areas have a significant gradient, which can facilitate the detection and prediction research in diverse terrain conditions.

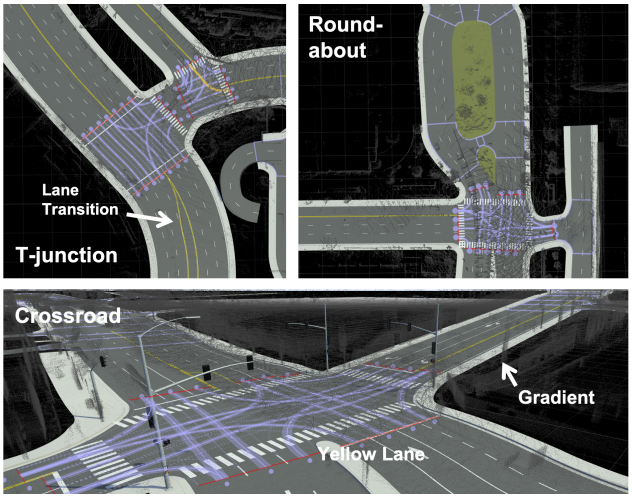


Figure S2. Examples of intersection types in the map, including T-junctions, roundabouts, and crossroads. The gray point clouds in the background represent the PCD map, while lane transitions and gradients are depicted in the map.

E.5. Dataset Privacy Protection

The V2XPnP-Sequential dataset is designed with stringent privacy safeguards to ensure the anonymity of individuals and vehicles. Trajectory data only includes object IDs and

positions, eliminating the possibility of tracking specific entities. All perception data undergoes privacy-preserving processing, with LiDAR annotations retaining only essential attributes such as object ID, agent type, and bounding box pose. Additionally, all image data has been anonymized, with human faces and other potentially sensitive details obscured or removed.

References

[1] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. 5, 7

[2] 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. 1, 3

[3] Valter Crescenzi, Giansalvatore Mecca, Paolo Merialdo, et al. Roadrunner: Towards automatic data extraction from large web sites. In *VLDB*, pages 109–118, 2001. 7

[4] Nachiket Deo and Mohan M. Trivedi. Convolutional Social Pooling for Vehicle Trajectory Prediction. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1549–15498, Salt Lake City, UT, USA, 2018. IEEE. 1

[5] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R Qi, Yin Zhou, et al. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9710–9719, 2021. 4, 7

[6] Ruiyang Hao, Siqi Fan, Yingru Dai, Zhenlin Zhang, Chenxi Li, Yuntian Wang, Haibao Yu, Wenxian Yang, Yuan Jirui, and Zaiqing Nie. Rcooper: A real-world large-scale dataset for roadside cooperative perception. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 7

[7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3

[8] 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

[9] E Li, Shuaijun Wang, Chengyang Li, Dachuan Li, Xiangbin Wu, and Qi Hao. Sustech points: A portable 3d point cloud interactive annotation platform system. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1108–1115. IEEE, 2020. 5

[10] Yiming Li, Shunli Ren, Pengxiang Wu, Siheng Chen, Chen Feng, and Wenjun Zhang. Learning distilled collaboration graph for multi-agent perception. In *Thirty-fifth Conference on Neural Information Processing Systems (NeurIPS 2021)*, 2021. 1, 3

[11] Yiming Li, Dekun Ma, Ziyan An, Zixun Wang, Yiqi Zhong, Siheng Chen, and Chen Feng. V2X-Sim: Multi-Agent Collaborative Perception Dataset and Benchmark for Autonomous Driving. *IEEE Robotics and Automation Letters*, 7(4):10914–10921, 2022. Conference Name: IEEE Robotics and Automation Letters. 7

[12] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 2

[13] Tsun-Hsuan Wang, Sivabalan Manivasagam, Ming Liang, Bin Yang, Wenyuan Zeng, James Tu, and Raquel Urtasun. V2VNet: Vehicle-to-Vehicle Communication for Joint Perception and Prediction. In *arXiv:2008.07519 [cs]*, 2020. 00010 ECC arXiv: 2008.07519. 1, 3

[14] Sizhe Wei, Yuxi Wei, Yue Hu, Yifan Lu, Yiqi Zhong, Siheng Chen, and Ya Zhang. Asynchrony-robust collaborative perception via bird's eye view flow. In *Advances in Neural Information Processing Systems*, 2023. 1, 3

[15] Hao Xiang, Zhaoliang Zheng, Xin Xia, Runsheng Xu, Letian Gao, Zewei Zhou, Xu Han, Xinkai Ji, Mingxi Li, Zonglin Meng, et al. V2x-real: a largs-scale dataset for vehicle-to-everything cooperative perception. *arXiv preprint arXiv:2403.16034*, 2024. 7

[16] Runsheng Xu, Hao Xiang, Zhengzhong Tu, Xin Xia, Ming-Hsuan Yang, and Jiaqi Ma. V2X-ViT: Vehicle-to-Everything Cooperative Perception with Vision Transformer. In *ECCV 2022*, pages 107–124, Cham, 2022. Springer Nature Switzerland. 1, 3, 7

[17] 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, 2022. 7

[18] Runsheng Xu, Xin Xia, Jinlong Li, Hanzhao Li, Shuo Zhang, Zhengzhong Tu, Zonglin Meng, Hao Xiang, Xiaoyu Dong, Rui Song, Hongkai Yu, Bolei Zhou, and Jiaqi Ma. V2V4Real: A Real-World Large-Scale Dataset for Vehicle-to-Vehicle Cooperative Perception. pages 13712–13722, 2023. 7

[19] Haibao Yu, Yizhen Luo, Mao Shu, Yiyi Huo, Zebang Yang, Yifeng Shi, Zhenglong Guo, Hanyu Li, Xing Hu, Jirui Yuan, and Zaiqing Nie. DAIR-V2X: A Large-Scale Dataset for Vehicle-Infrastructure Cooperative 3D Object Detection. pages 21361–21370, 2022. 7

[20] Haibao Yu, Wenxian Yang, Hongzhi Ruan, Zhenwei Yang, Yingjuan Tang, Xu Gao, Xin Hao, Yifeng Shi, Yifeng Pan, Ning Sun, Juan Song, Jirui Yuan, Ping Luo, and Zaiqing Nie. V2X-Seq: A Large-Scale Sequential Dataset for Vehicle-Infrastructure Cooperative Perception and Forecasting. arXiv, 2023. arXiv:2305.05938 [cs]. 7

[21] Haibao Yu, Yingjuan Tang, Enze Xie, Jilei Mao, Ping Luo, and Zaiqing Nie. Flow-based feature fusion for vehicle-infrastructure cooperative 3d object detection. *Advances in Neural Information Processing Systems*, 36, 2024. 1, 3