# CoST: Efficient Collaborative Perception From Unified Spatiotemporal Perspective

Supplementary Material

## 6. Preliminary

### 6.1. Problem Formulation

In a collaborative perception scenario involving $N$ agents, each agent $i$ owns unique observations $\{\mathcal{X}_i\}_{i=1}^N$ and the perception supervision $\{\mathcal{Y}_i\}_{i=1}^N$. The objective is to maximize the collective perception performance of all agents while ensuring that the transmission cost remains within a specified limit $G$. The process can be formulated as follows:

$$\begin{aligned} &\arg\max_{\theta,\mathcal{M}} \sum_{i=1}^{N} g(\psi_\theta(\mathcal{X}_i,\{\mathcal{M}_{i\to j}\}_{j=1}^N),\mathcal{Y}_i), \\ &\text{s.t.} \sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N} |\mathcal{M}_{i\to j}| \leq G, \end{aligned} \tag{7}$$

where $g(\cdot,\cdot)$ represents the perception evaluation metric, $\psi(\cdot)$ is the collaborative perception model with trainable parameter $\theta$, and $\mathcal{M}_{i\to j}$ denotes the message transmitted from the $i$th agent to the $j$th agent.

### 6.2. Collaboration Stages

Collaborative perception in multi-agent systems involves several key stages to ensure comprehensive and accurate environmental understanding. Here, we outline the main stages of our proposed framework and their respective roles.

**V2X Metadata Sharing:** The collaborative process begins with the sharing of meta information. Each agent shares metadata, including 6 degrees of freedom (6DoF) pose, extrinsic parameters, velocity, and agent type, which covers both infrastructure and vehicle type. One vehicle is designated as the ego agent. This shared meta information lays the foundation for subsequent feature extraction and fusion processes.

**Feature Extraction:** Following metadata exchange, each agent extracts relevant features within its own view. We use the efficient PointPillar [11] for LiDAR point cloud feature extraction due to its minimal inference time and optimized memory utilization [32]. PointPillar transforms the sparse point cloud into dense pillar tensors, which are then processed to obtain rich semantic BEV (Bird's Eye View) features $F_i^t \in \mathbb{R}^{H\times W\times C}$ at timestamp $t$ for agent $i$.

**Feature Communication:** After extracting the features, agents exchange these features. Given the data-intensive nature of PointPillar features, reducing transmission bandwidth is crucial. Limited communication bandwidth in practical scenarios makes efficient feature communication a core challenge. To address this, We introduce a spatiotemporal perspective where only dynamic object observations are transmitted, while static object observations are reused, reducing communication load. The ego agent combines selected tokens with historical features from the memory bank to form a reconstructed feature $R_i^t$ in $\mathbb{R}^{H\times W\times C}$, which then updates the memory bank for future use.

**Feature Fusion:** Upon receiving features from other agents, the ego agent performs feature fusion. The goal is to integrate its own information with that received from other agents to derive the most comprehensive perceptual features. We employ a multi-agent fusion module that merges reconstructed features with ego features, producing a collaborative feature $B$ for the current timestamp in the space $\mathbb{R}^{H\times W\times C}$.

**Detection Head** In the final stage, the integrated perceptual features are fed into the detection head to predict the final perception results. This process involves two $1\times 1$ convolution layers: one for box regression, outputting position, dimensions, and yaw angle of prediction boxes, and another for classification, generating a confidence score indicating the likelihood of each box containing an object or being background. The employed loss functions align with those of PointPillar [11], including a smooth $L1$ loss [24] for regression and focal loss [16] for classification.

## 7. Spatiotemporal Transmission

As shown in Figure 8, the STT module in our CoST framework reduces communication bandwidth by transmitting only dynamic object features instead of the full BEV map. Conventional methods reduce feature channels [31, 34], but they neglect the temporal redundancy where static information remains unchanged over time. While Where2comm [8] transmits sparse tokens from key object regions, it does not leverage the fact that static regions can be reconstructed from historical data. In our approach, we focus on dynamic features that change over time, thereby avoiding the re-transmission of redundant static information while ensuring a complete scene representation.

**Robustness to Communication Loss.** To further evaluate the robustness of our framework under lossy communication, we simulate perturbations by randomly dropping a portion of transmitted feature tokens. As shown in Figure 9, CoST exhibits stronger resilience compared to baselines, with significantly smaller degradation in detection accuracy. This robustness stems from our use of temporal memory to mitigate the impact of missing data during transmission.

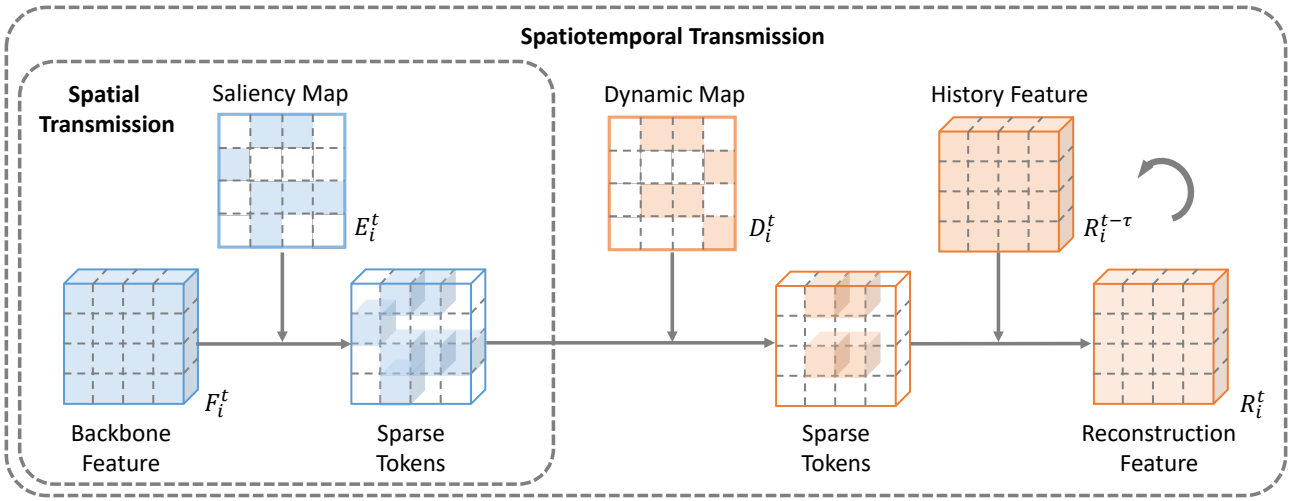


Figure 8. Our spatiotemporal transmission filters tokens based on saliency and motion, similar to selective attention in perception. These tokens are subsequently merged with historical features to reconstruct a comprehensive feature map.

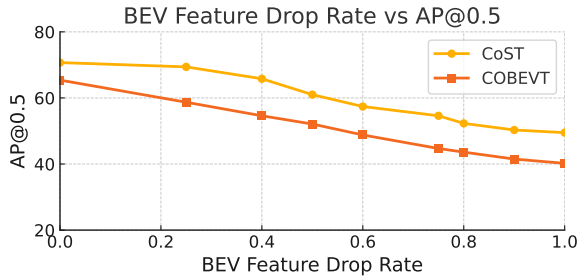


Figure 9. Performance under randomly dropped transmitted features on V2V4Real. CoST shows better robustness with minimal accuracy drop.

## 8. Training Details

All detection models utilize PointPillar [11] to extract the BEV features from the point cloud and the feature channel $C$ is set as 256. The models are trained over 60 epochs with a batch size of 4 per GPU (Tesla V100), a learning rate set at 0.001, and incorporating learning rate decay using a cosine annealing strategy [21]. Regarding the USTF module, during training, we use data samples of three consecutive frames with a time interval $\tau$ set to 2 frames to obtain more context information. During testing, we process the frames sequentially from the first frame to the last. As for the STT compression module, it undergoes an additional two-stage training process with the other pre-trained modules frozen. To ensure consistency in evaluation, standard settings are followed on the V2V4Real [34] dataset, including typical data augmentations for point cloud data. Conversely, no data augmentations are applied to the V2XSet [31] and DAIR-V2X [37] datasets. The model optimization is performed using AdamW [10] with a weight decay of $1 \times 10^{-2}$ to fine-tune the models.

## 9. Datasets

**V2V4Real:** V2VReal is a large-scale real-world V2V dataset collected by two vehicles with multi-modal sensors navigating through diverse scenarios. Covering a driving area of 410 km, V2V4Real includes $20K$ LiDAR frames, $240K$ annotated 3D bounding boxes across 5 classes, and HDMaps that encompass all driving routes.

**V2XSet:** V2XSet integrates V2X collaboration with realistic noise simulation. The dataset comprises a total of $11,447$ frames from CARLA [6] and OpenCDA [29], split into training, validation, and test sets with $6,694$, $1,920$, and $2,833$ frames, respectively. It includes five types of roadways: straight segments, curvy segments, midblocks, entrance ramps, and intersections. Each scene features between two to seven intelligent agents for collaborative perception.

**DAIR-V2X:** DAIR-V2X is a substantial real-world V2I dataset without V2V collaboration. It comprises three parts: DAIR-V2X-C, DAIR-V2X-I, and DAIR-V2XV. Notably, DAIR-V2X-C contains sensor data from vehicles and infrastructure, with $38,845$ frames each from cameras and LiDAR, and around 464,000 3D bounding boxes categorized into 10 distinct classes.

## 10. Qualitative Analysis

Figure 10 illustrates the detection visualization for OPV2V, CoBEVT, V2X-ViT, and CoTT across sparse and dense scenes in V2V4Real. Our model produces precise bounding boxes that closely align with the actual ground truth, contrasting with competing methods that exhibit notable discrepancies. Specifically, in sparse scenes with fewer than three ground truths, observed in the initial and sec-

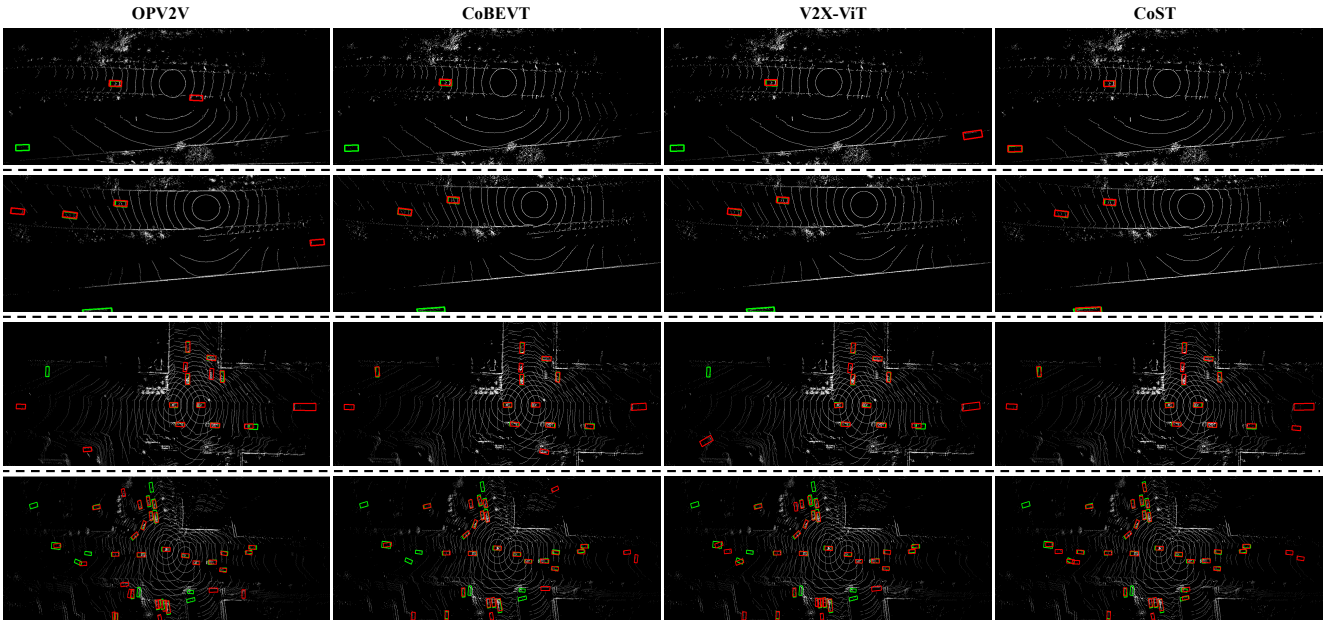


Figure 10. Qualitative comparison between sparse and dense scenes. The green and red 3D bounding boxes represent the ground truth and predictions, respectively. Our approach demonstrates superior accuracy in detection results.

ond rows, alternative methods frequently overlook distant ground truth boxes and may produce false positives. Conversely, our approach delivers accurate predictions in such scenarios. Moreover, our methodology achieves the most thorough detection outcomes in dense environments represented in the last two rows of Figure 10, demonstrating its proficiency in effectively leveraging historical data.

We visualize the ground truth objects/boxes in adjacent frames in Figure 11. Only the six cars positioned in the center of the frame have moved between frame 25 and frame 30, while the state of the other vehicles has remained unchanged. Besides, the temporal context benefits the detection result and reduces the omission of detection results, as shown in the bottom of Figure 11.

## 11. Robustness Analysis

**Time Delay** poses a notable challenge in real-world V2X communications, leading to a lack of synchronization between the ego vehicle's functionalities and information received from collaborative agents. It is essential for collaborative perception techniques to effectively handle such time delays. In this research, we investigate the resilience of our model towards time delays. We enhanced our approach by utilizing a model originally trained in an ideal environment, introducing latencies uniformly distributed between 0 to 500 milliseconds. To assess our model, we conducted tests with fixed latencies. In scenarios featuring a fixed delay (as depicted in Figure 6 in the main text), a consistent latency

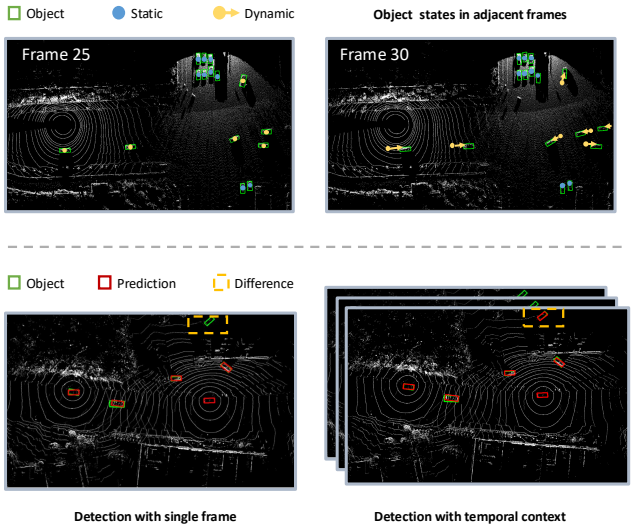


Figure 11. Visualization for object states on adjacent frames and comparing detection with single frame and temporal context.

of up to 500 milliseconds was added to the feature transmission from each agent to the ego vehicle. Our findings reveal that our method demonstrates substantial resilience to latency within V2V4Real. In contrast, both early fusion and late fusion methodologies exhibit a notable decline in performance when confronted with latency issues, indicat-

ing a lack of robustness. Consistently, our method surpasses other techniques in accuracy across various time delay configurations. Even in settings with noisy 100 milliseconds time delays, our method achieves an accuracy of $66.03\%$ AP@0.7, outperforming other cooperative approaches under optimal conditions.

**Pose Error** is a significant issue in collaborative perception, encompassing both heading error and localization error. In our experiments, we selected the V2XSet dataset for validation because the real-world database V2V4Real already contains inherent pose error noise, making the effects of manually added noise less discernible. To mimic real-world inaccuracies in our experiments, we introduced localization noise without modifying the models' settings. This noise follows a Gaussian Distribution with a mean of zero and an adjustable standard deviation, emphasizing the importance of robustness in collaborative perception models against localization inaccuracies. The results from the V2XSet dataset shown in Figure 6 in the main text indicate that heading errors have a more significant impact on performance compared to positional errors. It is evident that introducing localization errors of $0.1$ and $0.2$ has minimal impact on the accuracy of our method. Notably, our method exhibits strong resilience to localization and heading errors, surpassing other methods notably in terms of AP@0.5.