CORE: Cooperative Reconstruction for Multi-Agent Perception

Binglu Wang\textsuperscript{1,2\*}, Lei Zhang\textsuperscript{1\*}, Zhaozhong Wang\textsuperscript{1}, Yongqiang Zhao\textsuperscript{1}, Tianfei Zhou\textsuperscript{2,3†}

\textsuperscript{1} Northwestern Polytechnical University \hspace{0.5em} \textsuperscript{2} Beijing Institute of Technology \hspace{0.5em} \textsuperscript{3} Computer Vision Lab, ETH Zurich

https://github.com/zllxot/CORE

Abstract

This paper presents CORE, a conceptually simple, effective and communication-efficient model for multi-agent cooperative perception. It addresses the task from a novel perspective of cooperative reconstruction, based on two key insights: 1) cooperating agents together provide a more holistic observation of the environment, and 2) the holistic observation can serve as valuable supervision to explicitly guide the model learning how to reconstruct the ideal observation based on collaboration. CORE instantiates the idea with three major components: a compressor for each agent to create more compact feature representation for efficient broadcasting, a lightweight attentive collaboration component for cross-agent message aggregation, and a reconstruction module to reconstruct the observation based on aggregated feature representations. This learning-to-reconstruct idea is task-agnostic, and offers clear and reasonable supervision to inspire more effective collaboration, eventually promoting perception tasks. We validate CORE on two large-scale multi-agent perception dataset, OPV2V and V2X-Sim, in two tasks, i.e., 3D object detection and semantic segmentation. Results demonstrate that CORE achieves state-of-the-art performance, and is more communication-efficient.

1. Introduction

Perception – identifying and interpreting sensory information – is a crucial ability for intelligent agents to sense the surrounding environment. Thanks to continued advances in deep learning, individual perception has demonstrated remarkable achievements in a number of tasks, e.g., detection [26, 45, 19, 24], segmentation [23, 35, 44] and tracking [36, 43]. Though being promising, it tends to suffer from issues (e.g., occlusion) stemmed from limited line-of-sight visibility of individual agents and is challenged by safety concerns. A more compelling paradigm is cooperative perception, i.e., a collection of agents to be

\*The first two authors contribute equally to this work.
\*Corresponding author: Tianfei Zhou.

Figure 1: CORE addresses multi-agent perception from a novel perspective of cooperative reconstruction. Take the Lidar-based autonomous driving scenario as an example, individual vehicles are confined to limited sensing capabilities of onboard sensors, yielding partial observations of the whole scene (i.e., raw BEVs). CORE takes a straightforward but remarkably effective step to address this: in addition to task-aware learning (e.g., object detection, semantic segmentation), CORE explicitly learns to reconstruct the complete scene (i.e., reconstructed BEVs) from incomplete observations of cooperating agents. The reconstruction-aware learning objective serves as a more sensible goal to inspire more effective cooperation of connected agents, ultimately boosting perception performance.

have as a group by exchanging information with each other so as to use their combined sensory experiences to perceive. Along this direction, recent efforts have been made to deliver datasets [40, 39, 15, 5] and cooperative solutions [22, 21, 1, 5, 28, 31, 10, 2, 34, 38, 39].

All these solutions hold a consensus promise that multiple agents together provide a holistic observation to the environment. But there is a practical challenge of performance-bandwidth trade-off to be addressed. On this account, some studies explore adaptive communication architectures to reduce bandwidth requirements by dynamically determining for each agent, e.g., whom to communicate with [22], when to communicate [21], and what message to communicate [10], while others focus more on the design of multi-agent collaborative strategies, e.g., early collaboration [3, 1] to aggregate raw observations of cooperating agents, late collaboration [20, 33] to combine
prediction results only, or the prevailing intermediate collaboration [2, 34, 28, 38, 39] to fuse intermediate feature representations that are easy to compress. Although these approaches find applications in myriad domains, from autonomous driving [15], to automated warehouses [29], to search and rescue [30], a key question remains unanswered: what should the ideal sensory state of each agent look like after information exchanging and aggregation?

Prior approaches rely on task-specific objectives to learn how to communicate or collaborative, which is suboptimal and potentially degrades generalization capabilities of models to a broader array of perception tasks.

This paper advocates an approach from a novel perspective of cooperative reconstruction (see Fig. 1). Our main insight is that, if multiple agents together indeed offer a more complete observation of scene, by absorbing others’ information, an agent will be able to reconstruct missing parts in its partial raw observation. By learning to reconstruct, the model is urged to learn more effective task-agnostic feature representations, and can offer a clearer explanation to the ideal cooperative state (i.e., features) of the agent, that is, from which we are able to reconstruct its complete observation. Moreover, this learning-to-reconstruct idea naturally links to recent advancements in masked data modeling [9, 7, 25], and enables our model to recover complete observation from even more corrupted inputs obtained. e.g., by masking some proportion of raw observations. With this capability, agents will be allowed to exchange spatially sparse features to reduce transmission overhead during inference.

The idea is realized with our CORE framework. Given a collection of agents and their 2D bird’s eye view (BEV) maps, CORE performs cooperative reconstruction with three key modules: a compression module, a collaboration module and a reconstruction module. 1) The compression module computes a compressed feature representation of each BEV for efficient transmission. Unlike most prior works [28, 16, 38] that only consider channel-wise compression, the module imposes more significant compression by masking (sub-sampling) features along the spatial dimension. 2) The collaboration module is a lightweight attention module that encourages knowledge exchanging across cooperating agents to enhance feature representations of each individual agent. 3) The reconstruction module takes a decoder structure to recover complete scene observation from the enhanced feature representation. It is learned in a supervised rather than self-supervised [9, 25, 7, 17] manner, because we can easily obtain the supervision by aggregating raw sensor data of all agents. Note that the communication-costly raw data fusion only works in the training phase, and the entire reconstruction module will be discarded during inference. The reconstruction objective provides an explicit guidance to multi-agent collaboration, leading to ideal cooperative representations that will benefit various perception tasks. For task-specific decoding (e.g., detection or segmentation), CORE applies conventional decoders in parallel with the reconstruction decoder.

To verify CORE, we conduct experiments in the autonomous driving scenarios on the OPV2V [39] dataset. Two popular tasks, i.e., 3D object detection and semantic segmentation, are studied. Results show that CORE is generalizable across tasks, and delivers better trade-off between perception accuracy and communication bandwidth.

2. Related Work

Cooperative Perception enables multiple agents to perceive surrounding environment collaboratively by sharing their observations and knowledge, which offers a great potential for improving individual’s safety, resilience and adaptability [14, 27, 8, 5, 16]. Recent years have witnessed many related systems developed to support a broad range of real-world applications, e.g., vehicle-to-vehicle (V2V)-communication-aided autonomous driving [38, 39, 34, 2, 3, 5], multirobot warehouse automation system [18, 42] and multiple unmanned aerial vehicles (UAVs) for search and rescue [30]. As single-agent perception tasks, the progress is largely driven by high-quality datasets, e.g., OPV2V [39], DAIR-V2X [40], V2X-Sim 1.0 [16] and 2.0 [15].

Based on these datasets, many effective cooperative perception methods are developed. One straightforward solution, widely studied in early efforts [3, 1], is to directly transmit and fuse raw sensor data among cooperating agents. Despite promising performance improvements, the expensive communication bandwidth required for transmitting high-dimensional data limits their deployment in practical situations. Recently, a new family of solutions [2, 34, 22, 21, 16, 39, 10, 28], i.e., intermediate fusion-based methods, which broadcasts the compact intermediate representation, has been extensively studied as they are able to deliver a better trade-off between perception performance and network bandwidth. A mainstream branch [22, 21, 39, 10, 28] of these methods is to employ attention mechanisms to obtain fusion weights, while other methods [34, 16, 46] model the relative relationships between different agents using graph neural networks. Apart from performance, the robustness of system is also critical. Some methods [14, 32, 27] address problems arising from the communication process, for example, latency [14, 38], localization error [32, 41], and communication interruption [27]. Some other approaches [17, 4] investigates self-supervised learning mechanisms to improve the generalization ability of collaboration models.

CORE is closely relevant with DiscoNet [16] in terms that both methods rely on early collaboration to yield holistic-view inputs and leverage them as valuable guidance of network learning. However, CORE addresses the problem based on a learning-to-reconstruction formulation
rather than teacher-student knowledge distillation in [16]. In addition to yielding a simpler and more elegant model design, our formulation has two other advantages: first, by optimizing a reconstruction-aware objective, CORE achieves better generalizability as well as improved perception performance, e.g., it outperforms [16] significantly by about 3% on the task of BEV semantic segmentation (see Table 2); second, the reconstruction idea allows CORE to further masking out spatial features to be transmitted, thereby achieving more efficient communication against [16].

3. Methodology

The framework of CORE is illustrated in Fig. 2. We assume a system of \( N \) agents simultaneously perceiving the environment. Each agent is equipped with an onboard sensor (e.g., RGB or thermal cameras, Lidar) to observe the environment and obtain its local measurement in form of modalities like RGB images or 3D point clouds. As conventions [16, 28, 10], CORE explores perception in BEV space, where agents transform their individual perception information into BEV. Denote \( \mathbf{I}_i \in \mathbb{R}^{H \times W \times C} \) as the BEV representation of the \( i \)-th agent with spatial size \( H \times W \) and channel number \( C \). The agent encodes \( \mathbf{I}_i \) into an intermediate feature representation by \( \mathbf{f}_\text{ENC}(\mathbf{I}_i) \in \mathbb{R}^{H \times W \times C'} \). Here, \( \mathbf{f}_\text{ENC} \) is a feature encoder shared by all agents, and \( H, W, C' \) denote feature height, width, channel, respectively.

CORE compresses the intermediate feature \( \mathbf{I}_i \) of each agent independently to a compact representation \( \mathbf{I}'_i = \mathbf{f}_\text{COM}(\mathbf{I}_i) \) using a shared compressor \( \mathbf{f}_\text{COM} \) (§3.1) and broadcast it along with its pose to other agents. Subsequently, each agent \( i \) aggregates the received features to enhance its representation \( \hat{\mathbf{I}}_i = f_{\text{COL}}(\mathbf{I}'_i, \{\mathbf{I}'_j\}_{j \neq i}) \in \mathbb{R}^{H \times W \times C} \) based on an attentive collaboration component (§3.2). Unlike existing methods that learn \( \hat{\mathbf{I}}_i \) in a task-specific manner, CORE provides additional guidance to learn the ideal state of \( \hat{\mathbf{I}}_i \) from which the BEV representation can be reconstructed, denoted as \( \tilde{\mathbf{I}}_i = f_{\text{REC}}(\mathbf{I}_i) \in \mathbb{R}^{H \times W \times C} \) (§3.3). Here \( f_{\text{REC}} \) is a reconstruction decoder. Next, we elaborate on details of the key modules.

3.1. Feature Compression and Sharing

The goal of feature compression is to reduce the communication bandwidth as much as possible while maintaining perception performance. Previous approaches only compress features (e.g., \( \mathbf{I}_i \) for agent \( i \)) along the channel dimension by, e.g., \( 1 \times 1 \) convolutional autoencoder [34, 16, 38]. However, we argue that simply compressing the channel dimension does not lead to the maximal saving of the bandwidth. To drive more effective reconstruction, CORE additionally performs spatial-wise sampling to features, which can further alleviate the transmission burden. Concretely, for channel-wise compression, we follow [38] to use a series of \( 1 \times 1 \) convolutional layers to progressively compress \( \mathbf{I}_i \), and the compressed feature is with the shape \( (H, W, C') \) and \( C' \ll C \). In our experiments, we follow prior studies [16, 39, 38] to set \( C' = 16 \). For spatial-wise subsampling, a two-step scheme is taken: first, we sum up the channel-compressed feature map along the channel dimension to obtain a 2D activation map \( \mathbf{O}_i \in \mathbb{R}^{H \times W} \) and se-
collect the top $K\%$ of the $HW$ feature points with the highest activation values; second, we further sample $R\%$ of the $(HWK)\%$ points following a uniform distribution, leading to a set of $(HWKR)\%_{occ}$ feature points at final. These features and their corresponding spatial coordinates in $I_i$ are messages that will be transmitted to connected agents nearby. After other agents receiving the feature points, they will reorganize them into an all-zero-initialized feature map $I^*_{j} \in \mathbb{R}^{H \times W \times C}$, and then project them back to $(H, W, C)$ using $1 \times 1$ convolutions.

### 3.2. Attentive Collaboration

Multi-agent collaborative learning focuses on updating each agent’s feature map by aggregating informative messages from its partners. Without loss of generality, we assume the $i$-th agent as the ego agent, and we have its local feature map $I^i$ as well as the features received from connected agents $\{I^j\}_{j=1}^J$. Here $J$ denotes the number of agents connected to $i$. To ensure the sender and the receiver are properly aligned before collaboration, we geometrically warp the sender’s message $I^i$ onto the receiver’s coordinate to derive $I^i_{j \rightarrow i} = \Gamma_j(I^i)$. Here $\Gamma_j$ is a differential transformation and sampling operator [38].

For multi-agent collaboration, we are inspired by [10] and devise a lightweight attention-aware collaboration scheme. Concretely, for each agent $i$, we transform its feature $I^i$ using a series of $1 \times 1$ convolutions followed by a softmax function to generate a confidence map $P_i \in [0, 1]^{H \times W}$. Each element in $P_i$ represents the activation degree at the corresponding spatial location, i.e., higher values are more objective-sensitive. $P_i$ conveys what information agent $i$ can offer to others. Moreover, we compute another confidence map $R_i \in [0, 1]^{H \times W}$ by $R_i = 1 - P_i$. In contrast to $P_i$, the spatial locations with higher values in $R_i$ indicate there is potentially information missing caused by occlusions or limited visibility, hence, it reflects what information agent $i$ needs the most. Given the confidence maps, for ego agent $i$ and its partner agent $j$, we compute an attention map as follows (see Fig. 3):

$$M_{j \rightarrow i} = R_i \odot P_j \in [0, 1]^{H \times W}, \quad (1)$$

where $\odot$ is the Hadamard product. Here the attention map $M_{j \rightarrow i}$ highlights the positions that agent $i$ needs information while agent $j$ can meet $i$’s requirements. The attention map allows us to perform more accurate and adaptive feature aggregation.

Furthermore, we update the feature $I^i_j$ of ego agent $i$ based on the calibrated feature $I^i_{j \rightarrow i}$ from agent $j$ and attention mask $M_{j \rightarrow i}$ as follows:

$$\hat{I}_{j \rightarrow i} = \text{DConv}_l \times 1(A) \odot V \odot M_{j \rightarrow i} + I^i_j \in \mathbb{R}^{H \times W \times C}, \quad (2)$$

where $A$ and $V$ are computed as:

$$A = W_1 [I^i_j, I^i_{j \rightarrow i}] \in \mathbb{R}^{H \times W \times 2C}, \quad (3)$$

$$V = W_2 \hat{I}_{j \rightarrow i} \in \mathbb{R}^{H \times W \times C}. \quad (4)$$

Fig. 4 illustrates the structure of the attentive collaboration scheme. Here $\hat{I}_{j \rightarrow i}$ is the updated message of agent $i$ based on the information from agent $j$, DConv$_l \times 1$ indicates depthwise convolution with kernel size $l \times l$ that generates an output with half the number of input channels. It enhances the receptive field of features while reducing the complexity and computational cost. $[\cdot, \cdot]$ is a channel-wise tensor concatenation operator, combining the message from different agents. $W_1$ and $W_2$ are learnable linear weights that allow cross-channel information interaction. The first term in Eq. 2 computes the attention weight from agent $j$ to $i$, where the attention map $M_{j \rightarrow i}$ makes it more targeted. A residual layer is introduced to avoid information loss in $I^i_j$.

For ego agent $i$, Eq. 2 is individually applied to each of its connected agent. Its final feature is the average of updated features from all agents, i.e., $\hat{I}_i = \text{avg}_j(\{\hat{I}_{j \rightarrow i}\})$.

In this way, our attentive collaboration module can effectively capture spatial and channel-wise dependencies among agents and learns to selectively attend to the most informative messages from neighboring agents.
changing messages across all connected agents, the proposed scheme enables each agent to refine its own representation by taking into account the context of the whole group and eventually improve the overall performance of the group on the target task.

### 3.3. BEV Reconstruction

In §3.1 and §3.2, we reduce the bandwidth consumption by compressing the intermediate representation along the channel and spatial dimensions, and achieve efficient information interaction across agents through attentive collaboration. However, the detailed information contained in intermediate representations is inevitably lost due to compression. In addition, supervising collaborative learning solely through task-specific labels can lead to a high dependence on downstream tasks, which hurts the generalization ability of the model. Our BEV reconstruction scheme alleviates these issues by learning a deep convolutional decoder to reconstruct the multi-view BEV representations from the interacted messages.

To obtain proper supervisory signal, we aggregate raw point clouds of individual agents to yield a holistic-view point cloud and convert into BEV representations that will serve as prediction targets of the reconstruction decoder. A conceptual illustration is presented in Fig. 5. To be more specific, let $S_i$ be the raw 3D point clouds collected from the ego agent $i$. We first project point clouds from all neighbor agents, $\{S_j\}_{j=1}^h$, to the ego’s coordinate system: $\{S_{j\rightarrow i}\}_{j=1}^h = \Gamma_\xi(\{S_j\}_{j=1}^h)$. Next, we aggregate each individual point cloud to produce a multi-view 3D scene: $\tilde{S} = f_{\mathrm{Sta}}(\{S_{j\rightarrow i}\}_{j=1}^h, S_i)$, where $f_{\mathrm{Sta}}(\cdot, \cdot)$ is a stacking operator. Now, all agents are situated within the same coordinate system yet occupy distinct positions. Subsequently, we project back to each local coordinate and crop it based on perceptual range: $\tilde{S}_i = \Gamma_\xi^{-1}(\tilde{S})$. Finally, we convert $\tilde{S}_i$ into its corresponding BEV representations $\tilde{I}_i = f_{\mathrm{REC}}(\tilde{I}_i) \in \mathbb{R}^{h \times w \times c}$. To encourage the reconstructed data to match the original input data, we train our reconstruction decoder using a reconstruction loss. Specifically, given the reconstructed BEV features $\tilde{I}_i$ w.r.t its supervision signal $\hat{I}_i$, the reconstruction loss is computed as the mean squared error (MSE) between them:

$$
L_{\mathrm{REC}} = \frac{1}{h \times w} \sum_{x=1}^h \sum_{y=1}^w (\tilde{I}_i(x, y) - \hat{I}_i(x, y))^2.
$$

By incorporating feature reconstruction into our pipeline, we are able to effectively leverage the benefits of collaboration without sacrificing the independence and flexibility of each individual task.

### 3.4. Detailed Network Architecture

#### Network Architecture

The entire network is comprised of three major components:

- **Feature Extractor $f_{\mathrm{ENC}}$.** The feature extractor $f_{\mathrm{ENC}}$ takes the BEV representation as input and encodes it into intermediate features. Following [39], we implement it through three 2D convolutional layers with kernel size of $3 \times 3$, and a batch normalization layer and a ReLU activation layer are employed after each layer.

- **Reconstruction Decoder $f_{\mathrm{REC}}$.** The reconstruction decoder $f_{\mathrm{REC}}$ is used for generating high-quality BEV features. It consists of three blocks, each of which is composed of a $2 \times 2$ transposed convolutional layer and a $3 \times 3$ normal convolutional layer. Each layer is followed by a batch normalization layer and a ReLU activation layer.

- **Task-specific Decoder $f_{\mathrm{TASK}}$.** We study two tasks: 3D object detection and semantic segmentation. For detection, the decoder comprises three deconvolutional layers followed by two sibling branches, *i.e.*, class prediction and bounding box regression. Following [39], the class prediction branch produces the confidence score for each anchor box, distinguishing between being an object or background, while the regression output represents the predefined anchor boxes’ attributes, including their location, size, and orientation. For segmentation, the decoder consists of three deconvolutional layers for upsampling and a $3 \times 3$ convolutional layer for generating the final semantic segmentation map.

#### Loss Function

CORE is optimized by minimizing the following loss function:

$$
\mathcal{L} = \mathcal{L}_{\mathrm{TASK}} + \lambda \mathcal{L}_{\mathrm{REC}},
$$
where the first term $\mathcal{L}_{\text{TASK}}$ is a task-specific loss. For detection, we follow conventions [16, 39] to use focal loss as the classification loss and smooth $l_1$ loss for bounding box regression. For segmentation, we use cross-entropy loss as [28]. The second one is our reconstruction loss $\mathcal{L}_{\text{REC}}$ (Eq. 6) for supervising the output of reconstruction decoder. The coefficient $\lambda$ balances the two loss terms.

4. Experiment

For comprehensive evaluation, we validate Core on two popular cooperative perception tasks, i.e., 3D object detection and BEV semantic segmentation.

4.1. Experimental Setup

**Datasets.** We use two popular datasets for evaluation. OPV2V [39] is a large-scale public dataset collected together by the co-simulating framework OpenCDA [37] and the CARLA [6] simulator for vehicle-to-vehicle collaborative perception. In total, the dataset contains 11,464 frames of LiDAR point clouds and RGB images. OPV2V can be further split into two subsets: a default CARLA towns and a Culver City digital town. The default CARLA towns subset includes 10,914 frames (6,764/1,980/2,170 for train/val/test splits). This subset provides a diverse range of scenarios with different levels of complexity, which can be used to train and evaluate collaborative perception models. The Culver City subset includes 550 frames and is designed to evaluate the generalization ability of the models. This subset represents a realistic imitation of a real-world city, with a variety of objects and structures that can challenge the perception capabilities of the models.

**V2X-Sim** [15] is a collaborative perception dataset for vehicle-to-everything communication, co-simulated using SUMO [12] and CARLA [6]. It is collected from CARLA towns and comprises a total of 10,000 frames of LiDAR point clouds (8,000/1,000/1,000 for train/val/test splits). This dataset provides a diverse range of scenarios with varying complexities, enabling the development and evaluation of vehicle-to-everything communication-enabled perception models.

**Training.** During the training phase, a set of agents that can establish communication with each other is randomly selected from the scenario, with each agent assigned a communication range of 70 m. We limit the range of point clouds to $[-140.8, 140.8] \times [-40, 40] \times [-3, 1]$ along the $x$, $y$, and $z$-axis for the 3D object detection task, and to $[-51.2, 51.2] \times [-51.2, 51.2] \times [-3, 1]$ for the BEV semantic segmentation task. As the voxel resolution is set to 0.4 m, we can get the BEV map with a resolution of $200 \times 704 \times 128$ and $256 \times 256 \times 128$, respectively. We apply several data augmentation techniques to the training data, including random flipping, scaling in range of [0.95, 1.05], and rotation from $[-45^\circ, 45^\circ]$. We train our Core using Adam [11] optimizer with a learning rate of 0.002 and a batch size of 1. We also employ early stopping based on the validation loss to prevent overfitting. The hyperparameters $\lambda$, $R$, and $K$ are empirically set to 1, 90, and 90, respectively.

**Inference.** At the inference stage, we adopt a fixed group of agents to evaluate the model for fair comparison. We apply a confidence score threshold of 0.25 and non-maximum suppression (NMS) with an IoU threshold of 0.15 to filter

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>Default AP@0.5</th>
<th>AP@0.7</th>
<th>Culver AP@0.5</th>
<th>AP@0.7</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Collaboration</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Early Collaboration</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Late Collaboration</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cooper[3]</td>
<td>VoxelNet[45]</td>
<td>85.2</td>
<td>75.8</td>
<td>81.5</td>
<td>67.7</td>
</tr>
<tr>
<td>F-Cooper[3]</td>
<td></td>
<td>87.6</td>
<td>78.7</td>
<td>84.9</td>
<td>72.3</td>
</tr>
<tr>
<td>AttFuse[39]</td>
<td></td>
<td>90.9</td>
<td>85.2</td>
<td>84.3</td>
<td>74.7</td>
</tr>
<tr>
<td>Core</td>
<td></td>
<td>90.9</td>
<td>88.3</td>
<td>87.8</td>
<td>82.6</td>
</tr>
</tbody>
</table>

Table 1: Quantitative results for the task of 3D object detection on OPV2V [39] test.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>Object Detection AP@0.5</th>
<th>AP@0.7</th>
<th>Segmentation mIoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Collaboration</td>
<td></td>
<td>49.9</td>
<td>44.2</td>
<td>45.9</td>
</tr>
<tr>
<td>Early Collaboration</td>
<td></td>
<td>70.4</td>
<td>67.0</td>
<td>64.1</td>
</tr>
<tr>
<td>When2com[21]</td>
<td></td>
<td>44.0</td>
<td>39.9</td>
<td>48.4</td>
</tr>
<tr>
<td>Who2com[22]</td>
<td></td>
<td>44.0</td>
<td>39.9</td>
<td>48.4</td>
</tr>
<tr>
<td>V2VNet[34]</td>
<td>VoxelNet[45]</td>
<td>68.4</td>
<td>62.8</td>
<td>58.4</td>
</tr>
<tr>
<td>DiscoNet[16]</td>
<td></td>
<td>69.0</td>
<td>63.4</td>
<td>56.7</td>
</tr>
<tr>
<td>Core</td>
<td></td>
<td>70.0</td>
<td>64.8</td>
<td>58.3</td>
</tr>
</tbody>
</table>

Table 3: Quantitative results for the task of object detection and segmentation on V2X-Sim [15] test.
Baselines. For comparison, we build two baseline models. **No Collaboration** refers to a single-agent perception system using only individual sensor data without any information from other agents. **Early Collaboration** directly aggregates raw sensor data from multiple agents at an early stage of the perception pipeline. **Late Collaboration** collects the prediction results of multiple agents and combines them to deliver the final results using NMS. In addition, **CORE** is compared with existing state-of-the-art algorithms, including **Cooper** [3], **F-Cooper** [2], **V2VNet** [34], **DiscoNet** [16], **AttFuse** [39], and **CoBEVT** [28].

Evaluation Metrics. For 3D object detection, we adopt standard evaluation metrics as [16, 39]: average precision (AP) at intersection-over-union (IoU) thresholds of 0.5 and 0.7. For BEV semantic segmentation, we follow [28] to report IoU scores for categories of vehicle, road, and lane.

Reproducibility. We implemented **CORE** using PyTorch and trained it on an NVIDIA RTX 3090Ti GPU with a 24GB RAM. Testing is conducted on the same machine.

4.2. Quantitative Result

**OPV2V.** Table 1 reports the overall detection results of **CORE** against existing approaches in terms of AP@0.5 and AP@0.7 on OPV2V [39]. For comprehensive evaluation, we evaluate methods using two different types of 3D perception backbones, *i.e.*, a voxel-based method VoxelNet [45] and a point-based method PointPillar [13]. Regarding VoxelNet-based methods, the table shows that **CORE** yields state-of-the-art performance across all the metrics on both subsets and surpasses the previous state-of-the-art method, **AttFuse** [39], by a significant margin of, e.g., 3.5% and 7.9% in terms of AP@0.5 and 0.7 on Culver City. Among methods based on PointPillar backbone, **CORE** consistently outperforms **AttFuse** [39] across all the metrics. Compared with a more recent method **CoBEVT** [28], **CORE** outperforms it by solid margins of 1.8%/0.8% in terms of AP@0.5/0.7 on Culver City. Though [28] is better than **CORE** on Default subset, the gap is indeed minor (< 0.5%). In summary, these results suggest that **CORE** is very robust to backbones, and outperforms current state-of-the-art methods in terms of AP@0.5 and AP@0.7, especially on the Culver City subset.

Table 2 presents BEV semantic segmentation results on OPV2V [39]. For fair comparison, we use the same backbone (*i.e.*, VoxelNet) and segmentation head for all methods. The table shows that our proposed method **CORE** outperforms all other methods by a significant margin, achieving an average IoU score of 56.8%, clearly outperforming the second-best method **V2VNet** [34] with an average score of 55.0%. **CORE** consistently achieves the best performance for all three categories, demonstrating its ability out overlapping detections.

Figure 6: Qualitative object detection results on OPV2V [39]. **CORE** is compared with **No collaboration**, **AttFuse** [39] and **CoBEVT** [28] on two regions with sparse observations and occlusions.

Figure 7: Qualitative segmentation results on OPV2V [39]. **CORE** is compared with **DiscoNet** [16], **AttFuse** [39] and **V2VNet** [34].
to handle different segmentation targets. Furthermore, our proposed method shows a significant improvement over the baselines with No Collaboration and Late Collaboration. It improves the No Collaboration baseline by 13.1% and the Late Collaboration baseline by 7.9%. These results demonstrate the effectiveness of CORE in improving the performance of BEV semantic segmentation through collaborative learning. The segmentation results further confirm the perception superiority of CORE, and further demonstrates its strong generalization ability.

**V2X-Sim.** Table 3 presents the quantitative results on V2X-Sim [15]. As observed, for object detection, CORE achieves APs of 70.0% and 64.8% at 0.5 and 0.7 IoU thresholds, which are the highest among all methods. For vehicle segmentation, CORE reaches a 58.3% score, higher than most previous works, except for V2VNet [34] which has 58.4%. In terms of mIoU, V2VNet [34] achieves the best performance at 41.1%, while CORE is competitive at 40.5%. Despite this, CORE still significantly outperforms other notable models such as When2com [21], Who2com [22], and DiscoNet [16]. More importantly, the performance gap between CORE and the Early Collaboration is significantly reduced compared to previous methods.

### 4.3. Qualitative Result

We present qualitative results to visually illustrate the effectiveness of CORE in handling challenging scenarios. As depicted in Fig. 6, our algorithm, CORE, can successfully detect multiple objects in cluttered, sparse point clouds, and occluded environments, and show better results in comparison with strong baselines like AttFuse [39] and CoBEVT [28]. To further validate the generalization capability of CORE, we conducted experiments to evaluate the performance of CORE on BEV semantic segmentation, as shown in Fig. 7. We demonstrate that CORE achieves high-quality segmentation results even in complex scenarios such as crowded and dynamic environments. These results are consistent with our quantitative evaluations, positioning it as a promising solution for various real-world applications.

In addition, we present a visualization of the reconstructed BEV features in Fig. 8 to confirm the high fidelity achieved in capturing the details of the supervisory BEV. As seen, the reconstructed BEV (right) closely resemble the supervisory BEV (left), confirming the effectiveness of the reconstruction process.

### 4.4. Ablation Study

**Key Component Analysis.** Table 4 shows the contribution of individual components in our CORE framework for both detection and segmentation tasks. By adding the attentive collaboration (i.e., ‘Colla.’) module alone, we observe an improvement in AP@0.7 and IoU scores, with a slight impact on AP@0.5 compared to the base case. The addition of the BEV reconstruction (i.e., ‘Recon.’) module on top of the collaboration module leads to further improvement across all metrics, particularly in Culver City, where we see a significant increase of 2.8% in AP@0.5 and 1.6% in AP@0.7. These results highlight the effectiveness of the reconstruction module in enhancing the performance of our CORE framework. Moreover, incorporating the reconstruction module also leads to a significant gain in segmentation performance, with a 1.5% increase in IoU compared to the baseline with collaboration. These findings demonstrate the importance of both collaboration and reconstruction modules in achieving state-of-the-art performance on multi-agent perception tasks.

---

**Table 4: Analysis of key components.** ‘Colla.’ and ‘Recon.’ indicate the attention collaboration module and BEV reconstruction module, respectively.

<table>
<thead>
<tr>
<th>Method</th>
<th>Spatial Compression Ratio</th>
<th>Default Culver AP@0.7 AP@0.7 Data Size (KB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DiscoNet [16]</td>
<td>1.0</td>
<td>87.6</td>
</tr>
<tr>
<td></td>
<td>0.8 (default)</td>
<td>88.6</td>
</tr>
<tr>
<td></td>
<td>0.6</td>
<td>88.3</td>
</tr>
<tr>
<td></td>
<td>0.4</td>
<td>87.5</td>
</tr>
<tr>
<td></td>
<td>0.2</td>
<td>87.1</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>83.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>78.3</td>
</tr>
</tbody>
</table>

---

**Table 5: Analysis of spatial-wise feature compression.** Here compression ratio is defined as the ratio between uncompressed data size and total data size. In our method, the spatial compression ratio can be calculated as $R\% \times K\%$.

---

Figure 8: Visualization of reconstructed BEV features.
Moreover, C is not surprised to see that performance progressively degrades as k increases from 1.0 to 0.1. But it is favorable that C can achieve performance even with a large compression ratio of 0.4. Moreover, ORE with a ratio of 0.6 can achieve comparable performance as DiscoNet [16] with no compression. For a better trade-off between perception performance and communication efficiency, we set the spatial compression ratio to 0.8 by default.

Hyperparameters R and K. Next, we study the impacts of hyperparameters R and K (§3.1), which jointly determine the spatial compression ratio. We vary their values from 10% to 90%, and show the object detection results (AP@0.7 in subsets of Default and Culver City) in Figure 9. As expected, ORE tends to perform worse as R or K becomes smaller. However, we observe that ORE is generally robust to these hyperparameters, as even with a compression ratio as low as 10%, the performance degradation is moderate. These results demonstrate the potential of our ORE to be applied to a wide range of scenarios, where different levels of compression ratios may be required due to varying bandwidth and storage constraints.

Supervisory BEV Generation. Table 6 presents the comparison results of generating supervisory BEV using two methods: CNN-based or grid-based projection. The former adopts existing 3D detectors (e.g., VoxelNet, PointPillar) to generate BEV features. Grid-based projection directly maps points to a two-dimensional grid, where each grid cell represents a region, and the presence or absence of points is used to indicate whether that region contains point cloud data. As seen, both methods can generate promising supervisory BEV, but the CNN-based projection appears to have a slight advantage, especially in more complex environments (i.e., Culver City). This indicates that the features extracted by the CNN-based method encode richer point cloud structural information, resulting in a more semantically meaningful BEV representation, while the grid-based method may suffer some loss during the BEV generation process.

Hyperparameter λ. Table 7 measures the effects of the hyperparameter λ in Eq. 7. As can be seen, the optimal performance is achieved at λ = 1, and we observe a slight degradation in performance when λ is either increased or decreased from this optimal value. However, ORE exhibits robustness to variations in λ, as evidenced by the relatively stable results across different values.

5. Conclusion

This paper has introduced ORE to address cooperative perception in multi-agent scenarios. By solving the task from a cooperative reconstruction view, ORE is able to learn more effective multi-agent collaboration, which is beneficial to specific perception tasks. In addition, cooperative reconstruction naturally links to the idea of masked data modeling, inspiring us to mask spatial features to further reduce transmitted data volume. ORE demonstrates superior performance-bandwidth trade-off on standard benchmarks, i.e., OPV2V and V2X-Sim, in both 3D object detection and BEV semantic segmentation tasks.

Acknowledgements This work was supported by the Postdoctoral Science Foundation of China under Grant 2022M710393, the Fourth Special Grant of China Postdoctoral Science Foundation (in front of the station) 2022TQ0035.
References


[7] Christoph Feichtenhofer, Haoqi Fan, Yanghao Li, and Kaiming He. Masked autoencoders as spatiotemporal learners. In NeurIPS, 2022. 2


[9] Kaiming He, Xinyi Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In CVPR, 2022. 2


[19] Zhijun Li, Qinjian Li, Pengbo Huang, Haisheng Xia, and Guoxin Li. Human-in-the-loop adaptive control of a soft exo-suit with actuator dynamics and ankle impedance adaptation. IEEE Transactions on Cybernetics, 2023. 1


[22] Yen-Cheng Liu, Junjiao Tian, Chih-Yao Ma, Nathan Glaser, Chia-Wen Kuo, and Zsolt Kira. Who2com: Collaborative perception via learnable handshake communication. In ICRA, 2020. 1, 2, 6, 8


[27] Shunli Ren, Zixing Lei, Zi Wang, Siheng Chen, and Wenjun Zhang. Robust collaborative perception against communication interruption. 2

[28] Hao Xiang Wei Shao Bolei Zhou Jiaqi Ma Runsheng Xu, Zhengzhong Tu. Coebot: Cooperative bird’s eye view semantic segmentation with sparse transformers. In CoRL, 2022. 1, 2, 3, 6, 7, 8


[32] Nicholas Vadivelu, Mengye Ren, James Tu, Jingkang Wang, and Raquel Urtasun. Learning to communicate and correct


