remote sensing

MDPI

Article

Co-Fusion: Advancing 3D Object Detection with Collaborative Fusion

Wenxuan Li [1,2], Qin Zou [1,2], Shoubin Chen [1,*], Chi Chen [3] and Qingxiang Meng [4]

[1] Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Shenzhen 518060, China; liwenx@whu.edu.cn (W.L.); qzou@whu.edu.cn (Q.Z.)
[2] School of Computer Science, Wuhan University, Wuhan 430072, China
[3] State Key Laboratory of Surveying, Mapping, and Remote Sensing Information Engineering, Wuhan University, Wuhan 430079, China; chichen@whu.edu.cn
[4] School of Remote Sensing and Information Engineering, Wuhan University, Wuhan 430079, China; mqx@whu.edu.cn
* Correspondence: shoubin.chen@whu.edu.cn

Highlights

What are the main findings?

- Co-Fusion combines standardized sparse-depth guidance, gated image–depth fusion, and three-stage parameter transfer within the Co-Fix3D framework.
- The method achieves 74.1% mAP and 75.3% NDS on the nuScenes test set in the single-frame setting without test-time augmentation.

What are the implications of the main findings?

- Sparse LiDAR measurements projected into camera views can guide feature construction without explicit depth completion.
- LiDAR-trained parameters provide reusable initialization for camera and fusion stages, while sensor-specific depth normalization requires validation for new sensing configurations.

Abstract

LiDAR–camera fusion for 3D object detection combines the metric geometry of LiDAR measurements with the semantic information in images. However, projected LiDAR depth is sparse and provides incomplete geometric guidance to the image branch, while the modality-specific branches follow different learning dynamics during multimodal training. We present Co-Fusion, a LiDAR-dominant collaborative fusion framework that extends the previously published Co-Fix3D detector. Co-Fusion introduces a LiDAR-depth guidance pipeline that standardizes projected LiDAR depth using dataset-level statistics and encodes the sparse measurements into features aligned with the image branch. A gated fusion network applies multiplicative modulation to the joint image–depth representation to produce depth-aware camera features. A three-stage parameter-transfer training schedule selectively transfers parameters from the LiDAR branch to initialize the camera and fusion branches. Experiments on the nuScenes 3D detection benchmark show that Co-Fusion reaches 74.1% mAP and 75.3% NDS in the single-frame multimodal setting without test-time augmentation.

Keywords: 3D object detection; LiDAR-Image fusion; bird's-eye view; depth guidance; deep learning

Check for updates

Academic Editor: Yuli Sun

Received: 29 July 2026
Revised: 15 October 2026
Accepted: 19 October 2026
Published: 28 October 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Multimodal 3D object detection [1] supports scene perception in autonomous driving by combining complementary sensor measurements. Recent methods [2,3] commonly adopt bird's-eye-view (BEV) fusion frameworks [4,5], which transform LiDAR point clouds and image features into a shared spatial representation. Despite continued progress [6,7], effectively integrating LiDAR geometry with image semantics remains challenging.

This study focuses on two limitations of existing BEV-based multimodal fusion methods.

First, the image branch receives limited geometric supervision. Projecting LiDAR points onto the image plane produces sparse depth maps whose valid values vary over a broad metric range and may change abruptly across neighboring pixels, i.e., depth jumps, as illustrated in Figure 1a. These two properties present different learning problems. EA-LSS [8] addresses local spatial discontinuities by constructing fine-grained depth targets and explicit depth-gradient cues, whereas auxiliary depth-estimation networks provide additional geometric supervision [9]. Co-Fusion instead addresses the global numerical scale of the sparse measurements before encoding them while retaining their spatial structure.

Second, the LiDAR and image branches differ in their input characteristics and learning dynamics. Consequently, initialization and the selection of trainable components affect multimodal optimization. BEVFusion [10], for example, adopts staged training, whereas IS-Fusion [11] restricts branch updates in its fusion setting. Figure 1b,c motivates our analysis of stage-wise parameter transfer and branch freezing but does not establish gradient conflict or modality dominance. We therefore assess the proposed schedule using detection-level controls and limit the conclusions to the observed accuracy differences.

To address these limitations, we propose Co-Fusion, a LiDAR-guided multimodal 3D object detector that extends the previously published Co-Fix3D framework [12]. Co-Fix3D operates primarily at the detection stage after construction of the BEV representation. Its parallel Local–Global Enhancement (LGE) branches, progressive query allocation, and spatial masks are designed for heterogeneous and weak-response objects. Its conventional image-to-BEV pathway, however, does not explicitly encode projected LiDAR depth into the image features, leaving the image branch dependent on monocular depth estimation. The LGE parameters learned from LiDAR geometry are also confined to the detection head rather than transferred across the LiDAR, camera, and fusion training stages.

Co-Fusion retains the Co-Fix3D detection head and modifies the upstream representation and training process. The LiDAR-Depth Guidance (LDG) pathway standardizes and encodes projected sparse depth, applies valid-pixel depth supervision, and modulates the joint image–depth features before camera-to-BEV lifting. The three-stage schedule transfers LiDAR-trained decoder and LGE parameters to initialize the camera and fusion stages, reusing the geometric information captured by the inherited modules without introducing a new detection head. The fused BEV tensor B_{F} connects the proposed fusion pathway to the Co-Fix3D detector. The LGE architecture, parallel multi-stage head, mask-based query de-duplication, Transformer decoder, and prediction heads are therefore inherited components rather than contributions of this study.

Co-Fusion uses LiDAR-guided sparse-depth encoding and supervision to provide geometric information to the image branch. LiDAR points are projected onto the image planes, and the resulting sparse depth measurements are standardized using statistics computed only from the training split. A downsampling convolutional network then aggregates local measurements into features at the resolution of the image branch; it does not explicitly reconstruct a dense physical depth map.

A gated fusion network subsequently applies content-adaptive modulation to the joint image–depth representation. Its contribution is evaluated against non-gated fusion operators in the ablation study.

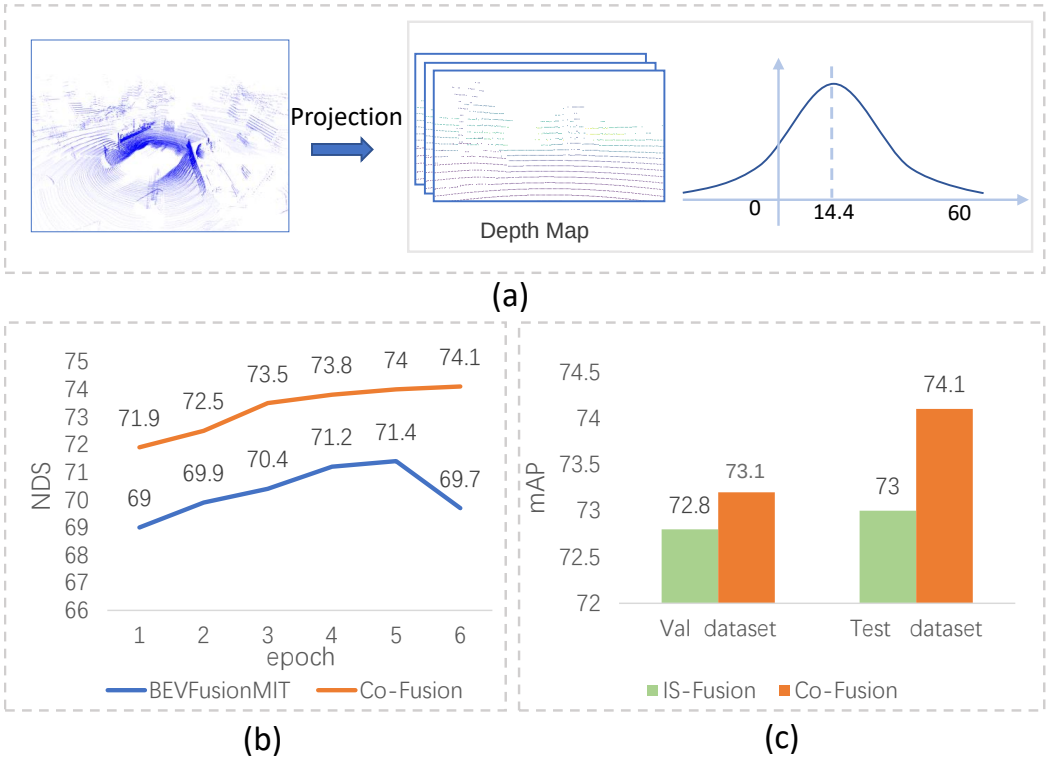


Figure 1. Challenges of BEV-based methods. (**a**) Instability of LiDAR-projected depth maps caused by depth discontinuities. Zero denotes pixels without a projected LiDAR return, rather than a physical depth measurement; valid projected depths are restricted to $[1, 60)$ m, and 14.41 m is the training-set mean computed over valid pixels only. The curve is a schematic illustration of the value range and does not imply that projected depths follow a Gaussian distribution. (**b**) Training performance comparison between BEVFusion-MIT and Co-Fusion on the validation set. (**c**) Performance comparison between IS-Fusion and Co-Fusion on the validation and test sets. Panels (**b**,**c**) use truncated vertical axes for readability; every plotted value is annotated.

The main contributions of this paper are summarized as follows:

- We extend the detection-level Co-Fix3D architecture to fusion-level collaborative modeling. The inherited Co-Fix3D head operates on the fused BEV representation, whereas the new Co-Fusion pathway improves the construction of that representation from LiDAR and camera inputs.
- We develop an LDG pathway that standardizes and encodes sparse projected LiDAR depth, together with gated image–depth modulation and valid-pixel depth-distribution supervision, to improve the geometric conditioning of camera features.
- We design a three-stage parameter-transfer schedule in which LiDAR-trained decoder and LGE parameters initialize the camera and fusion stages, while the modality backbones are frozen during Stage III so that optimization is concentrated on fusion and detection components.

2. Related Work

2.1. LiDAR-Based 3D Object Detection

LiDAR sensors provide direct range measurements that are largely independent of illumination, making them a common input for 3D object detection [13,14]. LiDAR-based detectors can be broadly divided into point-based and voxel-based methods.

Point-based methods, such as PointNet [15], operate directly on raw point clouds without voxel quantization. Their irregular memory access and neighborhood operations

can be computationally demanding for BEV processing. 3DSSD [16] introduces a sampling strategy for efficient point-based detection, while Feature Sparse Detection (FSD) [17] addresses missing center features in sparse representations. Voxel-based methods, including VoxelNet [18] and SECOND [19], quantize point clouds into regular voxel grids. SECOND uses sparse convolutions to avoid computation at empty locations. CenterPoint [20] formulates voxel-based detection as anchor-free center prediction, and TransFusion-L [21] uses Transformer-based query decoding for LiDAR detection.

Point-cloud sparsity and measurement noise remain challenging in low-reflectivity or cluttered environments [22]. This motivates the use of image semantics as a complementary source of information in LiDAR–camera detection.

2.2. Multimodal 3D Object Detection

Multimodal 3D object detection [23] combines measurements with complementary geometric and semantic content. Existing methods are commonly categorized as early, intermediate, or late fusion according to the stage at which cross-modal interaction occurs.

Early fusion methods [24,25] augment input point features with information sampled from corresponding image pixels. AutoAlignV2 [26] performs voxel-level fusion by aggregating image features from multiple sampling points. Such correspondence-based operations depend on sensor calibration and can be affected by projection errors. Late fusion methods combine modality-specific predictions or region proposals after largely independent feature extraction. This limits cross-modal interaction during proposal generation [27]. Intermediate fusion instead exchanges information before final prediction. Its calibration sensitivity depends on the specific alignment and fusion operations and must be assessed experimentally.

LiDAR-guided depth supervision has been used to improve camera-to-BEV lifting. BEVDepth [28], for example, supervises a discretized image-depth distribution using projected LiDAR depths, whereas BEVStereo [29] uses temporal stereo to refine camera depth estimates. EA-LSS [8] targets local depth jumps: it expands sparse depth supervision over fine-grained neighborhoods, computes directional differences in the expanded depth map, and uses the resulting edge response to guide depth learning. Its edge-response normalization is a local spatial operation and is distinct from standardizing the physical values of valid projected depths. SA-BEV [30] introduces semantic-aware BEV pooling for camera-only detection, and UniTR [31] uses a modality-agnostic Transformer with shared parameters for unified multimodal representation. Recent fusion methods, including DAL [9], IS-Fusion [11], MSMDFusion [32], and GraphBEV [33], explore complementary depth guidance, interaction, or alignment mechanisms. BEVFix [34] refines BEV representations using point-cloud-derived masks and wavelet-based feature enhancement to address sparsity and noise. Co-Fusion differs from these approaches by applying training-split Z-score standardization to the global value scale of valid sparse depths before feature encoding, followed by gated joint feature modulation and a staged parameter-transfer schedule. This standardization does not remove local depth discontinuities; it changes the numerical conditioning of the measurements presented to DepthNet. While Co-Fix3D [12] improves detection through multi-stage BEV optimization, its image branch has no LDG module or gated image–depth feature fusion.

3. Method

Figure 2 presents the Co-Fusion architecture. This section describes the overall framework, the LiDAR-depth guidance and depth-supervision components, and the three-stage parameter-transfer training schedule.

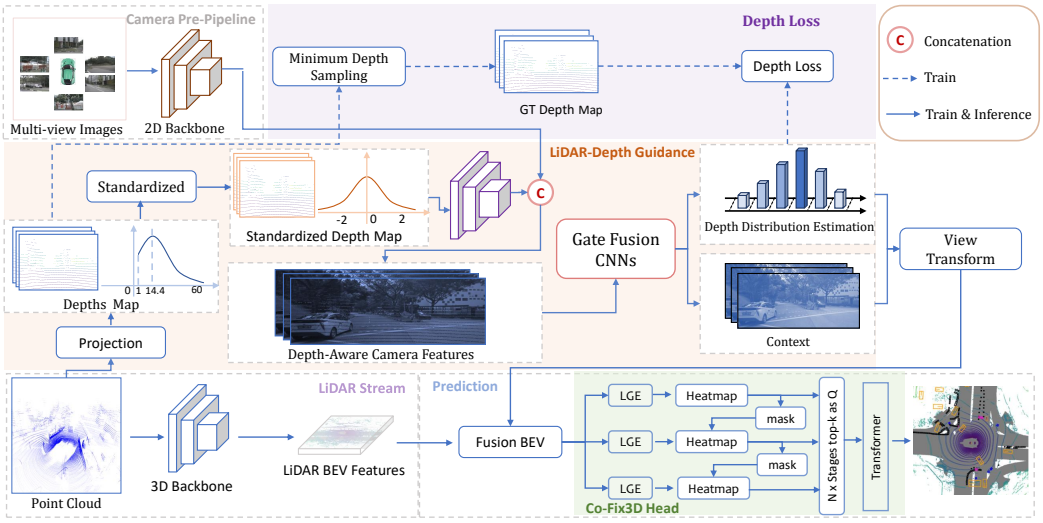


Figure 2. Overview of Co-Fusion. Co-Fusion takes multi-view images and point clouds as input. The LiDAR-Depth Guidance (LDG) module projects and standardizes sparse LiDAR depths, and DepthNet encodes them into 1/8-scale sparse-depth features. These features are concatenated with image features and refined by Gated CNN units. The depth loss supervises the predicted depth logits at valid downsampled LiDAR-depth locations. LSS transforms the resulting camera context features into camera BEV features, which are fused with point-cloud BEV features. The inherited Co-Fix3D [12] head then produces 3D detections.

3.1. Overall Framework

As shown in Figure 2, Co-Fusion receives a LiDAR point cloud P and N synchronized camera images $C = \{C_1, C_2, \ldots, C_N\}$ with known LiDAR-to-camera calibration. Modality-specific encoders produce the point-cloud representation $B_P = f_{\text{point}}(P)$ and perspective-view image representation $F_C = f_{\text{img}}(C)$. Following [35], $f_{\text{point}}(\cdot)$ is implemented with VoxelNet [18], and $f_{\text{img}}(\cdot)$ uses a Swin Transformer. In particular, $B_P \in \mathbb{R}^{W \times H \times C}$ is obtained by compressing the height dimension of the 3D voxel feature as in VoxelNet [18], where W and H are the numbers of BEV grid cells along the x and y axes, and C denotes the channel dimension.

The multi-view camera BEV feature is computed as

$$B_C = f_{VT}(f_{LDG}(F_C)), \tag{1}$$

where $f_{LDG}(\cdot)$ is the LiDAR-Depth Guidance module, $f_{VT}(\cdot)$ is the Lift–Splat–Shoot (LSS) view transformation, and B_C is the resulting camera BEV feature.

We obtain the fused BEV features as

$$B_F = f_{\text{fusion}}(B_P, B_C), \tag{2}$$

where B_F denotes the fused BEV representation, and $f_{\text{fusion}}(\cdot)$ operates on the point-cloud BEV feature B_P and camera BEV feature B_C.

Co-Fix3D [12] processes B_F using a multi-stage head with an LGE module. LGE combines a discrete wavelet transform and attention to refine the BEV representation. Multi-stage filtering identifies weak positive queries, while a spatial mask prevents overlap between newly selected queries and those retained from earlier stages. The aggregated queries are passed to a Transformer for classification and 3D localization.

3.2. LiDAR-Depth Guidance Module

The LiDAR-depth guidance framework supplies the image branch with geometric cues from projected LiDAR measurements. The projected maps are sparse and irregularly

distributed, while their valid entries retain metric units and span a much broader numerical range than the normalized features used by the image branch. LDG therefore standardizes the valid measurements using training-set statistics before encoding the sparse map at the image-feature resolution. DepthNet is the encoder within LDG, not a separate guidance module applied after standardization. Importantly, the LDG module does not perform explicit pixel-domain depth completion or geometric propagation. Its convolutional encoder aggregates sparse measurements over local receptive fields; the depth-supervision target is constructed separately from projected LiDAR depths.

Point Cloud Projection. The raw point cloud is transformed into the coordinate systems of the N calibrated cameras using the known extrinsic and intrinsic parameters. Each transformed point $P = (x, y, z)$ is projected onto the corresponding image plane to form a sparse depth map $P_{\text{depth}} \in \mathbb{R}^{H_{\text{img}} \times W_{\text{img}} \times 1}$. Each valid pixel stores the camera-frame z coordinate, i.e., optical-axis depth rather than Euclidean distance to the camera center. Pixels without a projected LiDAR return are assigned zero as an invalid-value sentinel. The sparse and irregular sampling leaves large regions of P_{depth} without valid measurements, which complicates local feature extraction.

Standardization. To control the input scale of sparse depth, we apply Z-score standardization using dataset-level statistics estimated from valid projected depths in the nuScenes train portion. Invalid pixels are excluded from the statistics. The mean (μ_{depth}) and population standard deviation (σ_{depth}) are computed as

$$\mu_{\text{depth}} = \frac{1}{N} \sum_{i=1}^{N} p_i, \tag{3}$$

$$\sigma_{\text{depth}} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (p_i - \mu_{\text{depth}})^2}, \tag{4}$$

where p_i denotes a valid projected depth and N is the total number of valid pixels used to estimate the statistics.

Let $M \in \{0, 1\}^{H_{\text{img}} \times W_{\text{img}}}$ be the validity indicator, where $M(u, v) = 1$ if a projected LiDAR measurement is available. In the current implementation, invalid pixels are set to zero after normalization:

$$P_{\text{depth-norm}}(u, v) = M(u, v) \frac{P_{\text{depth}}(u, v) - \mu_{\text{depth}}}{\sigma_{\text{depth}}}. \tag{5}$$

The preprocessing routine stores M alongside the normalized map, but the adopted DepthNet receives only $P_{\text{depth-norm}}$ as a single input channel. Thus, the mask is used to zero invalid entries and to select valid depth-supervision locations; it is not an independent input feature. Unlike Batch Normalization, the normalization constants do not depend on the mini-batch or inference data. Z-score standardization centers and rescales the depth values relative to this reference population; it does not make their distribution uniform or smooth spatial depth boundaries.

For a dense, fully observed input, a fixed Z-score transform is an affine reparameterization that can, in principle, be absorbed into the weights and bias of the first convolution. Accordingly, standardization does not increase the representational capacity of DepthNet or introduce additional geometric information. Equation (5), however, applies the mean subtraction only at valid projected-depth locations and resets invalid locations to zero. The resulting masked operation is not equivalent to applying one global affine transform to every pixel because a single convolutional bias cannot simultaneously reproduce the different offsets required at valid and invalid locations. More generally, even when two parameterizations can represent the same function, they need not follow the same optimization trajectory

under finite initialization, nonlinear activations, weight decay, and a fixed learning-rate schedule. Centering and scaling the valid depths limits scale-induced variation in the initial activations and back-propagated gradients of DepthNet and the subsequent fusion layers, thereby providing better-conditioned inputs for finite-step optimization. This mechanism is complementary to the local gradient and edge supervision used by EA-LSS [8]: EA-LSS explicitly models where depth changes abruptly, whereas our standardization controls the global numerical scale without suppressing those boundaries. We therefore interpret standardization as an input-preconditioning choice for the adopted optimization protocol, rather than as a source of new geometry or theoretical model capacity. Because gradient variance is not measured directly, the detection ablations are treated as evidence of optimization sensitivity rather than proof of a smoother loss landscape.

Sparse-depth encoding within LDG. Standardization controls the value scale but does not change the sparsity of the projected measurements. The sparse-depth encoder, denoted as *DepthNet*, progressively aggregates local geometric measurements. It is an integral part of LDG and is not supervised to predict missing pixel depths; hence, it is not a depth-completion network. The *DepthNet* adopts an encoder-style convolutional architecture with four convolutional layers. The first layer uses a 7×7 kernel; the following three layers use 5×5 kernels with stride 2. The output has $1/8$ of the input spatial resolution, matching the image features used for fusion.

The output feature map is expressed as

$$F_S = \text{DepthNet}\big(P_{\text{depth-norm}}\big), \tag{6}$$

where F_S is the sparse-depth feature tensor. Each output location aggregates measurements within its receptive field and is aligned with the $1/8$-scale image feature map. These features provide geometric cues for subsequent image–depth fusion.

Gate Fusion Network. As illustrated in Figure 3, the Gate Fusion Network (GFN) jointly encodes LiDAR-derived sparse-depth and image features to form a depth-aware camera representation. It applies multiplicative feature modulation after joint encoding rather than estimating a separate reliability score for each modality.

The initial fusion operation is formulated as

$$F_0 = \text{Reduce}\left(\text{Cat}(F_S, F_C)\right), \tag{7}$$

where $F_S \in \mathbb{R}^{H' \times W' \times 64}$ and $F_C \in \mathbb{R}^{H' \times W' \times 256}$ are the aligned sparse-depth and image features, respectively. $\text{Cat}(\cdot)$ concatenates along the channel dimension, and $\text{Reduce}(\cdot)$ is a 3×3 convolution followed by Batch Normalization and ReLU that produces $F_0 \in \mathbb{R}^{H' \times W' \times 256}$. Thus, subsequent Gated CNN units operate on a joint representation rather than on one modality in isolation.

The gating operation then modulates and combines these components:

$$\tilde{F}_c = \text{DWConv}_{7\times7}(F_c), \tag{8}$$

$$F_{k+1} = \text{Linear}_2\Big(\text{GELU}(F_g) \odot \text{Cat}(F_i, \tilde{F}_c)\Big) + F_k, \tag{9}$$

where $\odot$ is element-wise multiplication. Both operands of $\odot$ have rC channels because $\text{Cat}(F_i, \tilde{F}_c) \in \mathbb{R}^{H' \times W' \times rC}$. $\text{DWConv}_{7\times7}(\cdot)$ is a depthwise convolution that captures local spatial context, and the residual connection preserves the input dimensionality. No normalization layer is applied inside the Gated CNN unit.

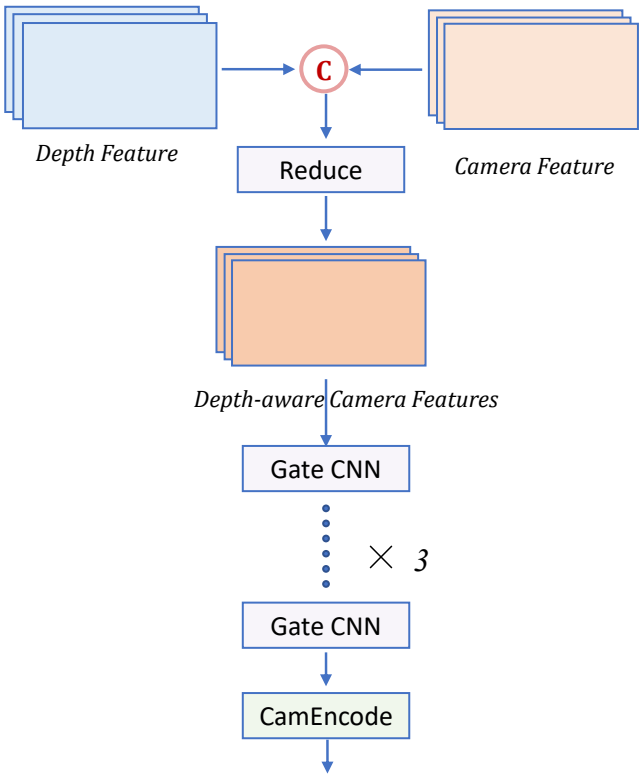


Figure 3. Illustration of the gated fusion network. Sparse-depth features and camera features are first jointly encoded into a depth-aware camera feature. The adopted configuration refines this feature with three Gated CNN units and then applies CamEncode to predict the depth distribution and camera context features.

Three Gated CNN units refine the fused feature, as illustrated in Figure 4. Given $F_k \in \mathbb{R}^{H' \times W' \times C}$, each unit applies a linear projection and partitions the resulting $2rC$ channels into three tensors:

$$F_g, F_i, F_c = \text{Split}\left(\text{Linear}_1(F_k)\right), \tag{10}$$

where $F_g \in \mathbb{R}^{H' \times W' \times rC}$, $F_i \in \mathbb{R}^{H' \times W' \times (r-1)C}$, and $F_c \in \mathbb{R}^{H' \times W' \times C}$. Here, $r > 1$ is the channel-expansion coefficient. In our configuration, $r = 8/3$, $C = 256$, and the hidden width is $\lfloor rC \rfloor = 682$.

The multiplicative interaction in Equation (9) adaptively modulates channels and spatial locations of the joint feature tensor, while the residual path supports optimization. Because the gate is computed after the initial joint encoding in Equation (7), we interpret it as content-adaptive feature modulation rather than as an explicit modality-confidence estimator.

After the three units, the predicted depth distribution and camera context features are obtained as

$$F_{\text{ctx}}, D_{\text{pred}} = \text{CamEncode}(F_3), \tag{11}$$

where $\text{CamEncode}(\cdot)$ predicts the depth distribution and produces the context features. Finally, F_{ctx} and D_{pred} are transformed into BEV features using BEVPoolv2 [36]:

$$B_C = \text{BEVPooling}(F_{\text{ctx}}, D_{\text{pred}}), \tag{12}$$

where B_C denotes the camera BEV feature and $\text{BEVPooling}(\cdot)$ denotes BEVPoolv2.

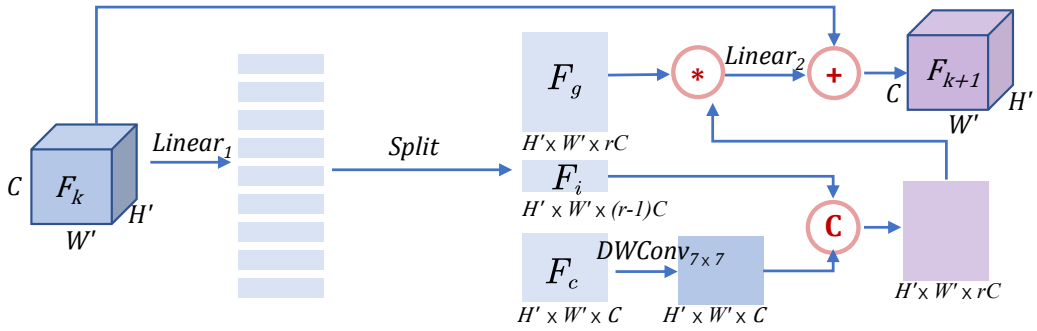


Figure 4. Architecture of one Gated CNN unit. Linear$_1$ expands F_k to $2rC$ channels and splits it into the gating tensor F_g, identity tensor F_i, and spatial-context tensor F_c. GELU activates F_g, while a 7×7 depthwise convolution processes F_c. Concatenating F_i and $\tilde{F}_c$ yields an rC-channel tensor, which is modulated element-wise by the activated gate. Linear$_2$ restores C channels, and the residual shortcut adds F_k to produce F_{k+1}. The adopted configuration uses $C = 256$ and $r = 8/3$.

3.3. Depth Loss Module

The depth-loss module constructs a sparse ground-truth depth target and compares it with the predicted depth distribution.

Minimum Depth Sample. To align the target with the image feature map, P_{depth} is downsampled by a factor of eight using minimum-depth sampling. The resulting ground-truth map is denoted by P_{depthGT}.

Depth focal loss. We construct the target following the sparse LiDAR-supervised depth-classification procedure used in BEVDepth [28]. The projected depth map is first rounded to the input pixel grid. Within each 8×8 window, zero entries are replaced by a large sentinel value and the minimum remaining depth is selected. The selected depth is discretized into H_D bins. Let Ω be the set of pixels with a valid binned target, $y_{i,c} = \mathbf{1}[c = c_i]$ its one-hot label, and $p_{i,c} = \sigma(z_{i,c})$ the sigmoid of the predicted logit $z_{i,c}$. The implemented loss is the sigmoid focal loss:

$$\mathcal{L}_{\text{depth}} = -\frac{\lambda}{|\Omega|} \sum_{i \in \Omega} \sum_{c=1}^{H_D} \alpha_{i,c} (1 - p_{t,i,c})^{\gamma} \log(p_{t,i,c}), \tag{13}$$

where $p_{t,i,c} = y_{i,c} p_{i,c} + (1 - y_{i,c})(1 - p_{i,c})$ and $\alpha_{i,c} = \alpha y_{i,c} + (1 - \alpha)(1 - y_{i,c})$. The implementation uses the MMDetection default values $\alpha = 0.25$ and $\gamma = 2$ and multiplies the resulting loss by $\lambda = 3$. The depth range is $[1, 60)$ m with a bin width of 0.5 m, giving $H_D = 118$. This loss is evaluated only on valid pixels; invalid locations do not contribute to $\mathcal{L}_{\text{depth}}$. Depth supervision is enabled during Stage II camera-branch training.

3.4. Three-Stage Parameter-Transfer Training

LiDAR and camera features are constructed through different geometric processes. LiDAR measurements provide direct spatial coordinates, whereas the camera pathway must estimate a depth distribution to lift perspective-view features into BEV. Under joint optimization, the detection objective constrains the fused prediction but does not require each branch to develop an equally useful representation. This motivates training the camera pathway before optimizing BEV fusion so that it receives detection supervision without relying on the LiDAR BEV features to explain the target objects. Co-Fusion implements this design through the three-stage schedule in Figure 5, combining LiDAR-based initialization, camera-branch training, and fusion training with frozen modality branches.

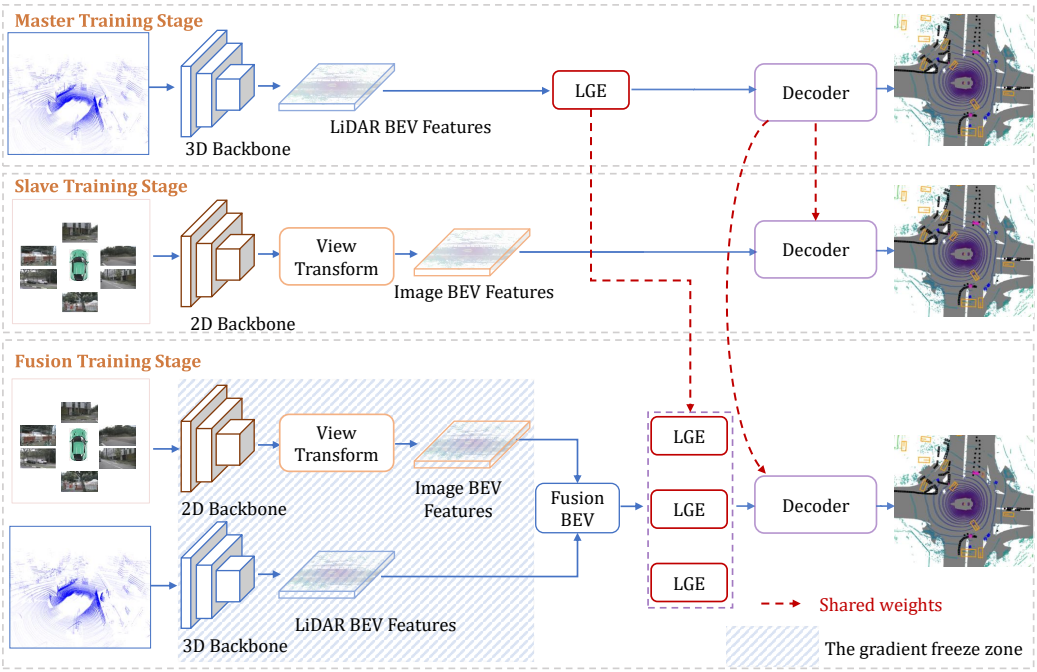


Figure 5. Overview of the three-stage training schedule. Solid blue arrows denote the forward pathways, red dashed arrows denote parameter transfer between stages, and shaded regions indicate frozen modules.

Stage I: LiDAR-branch training. The point-cloud encoder, BEV decoder, LGE module, and detection head are optimized using LiDAR input and the detection objective. The resulting parameters encode a task-specific mapping from BEV features to object predictions. They provide an initialization for structurally compatible modules in the subsequent stages. The rationale for transferring these parameters is that the branches share the BEV coordinate system and detection task, even though their input features differ. This shared task structure makes the LiDAR-trained decoder a relevant starting point for camera-based BEV decoding; it does not imply that the two feature distributions are already aligned.

Stage II: Camera-branch training. The camera decoder is initialized from the Stage-I LiDAR decoder and then adapted together with the image backbone, LDG pathway, LSS view transformation, and camera detection components. The LiDAR detection branch is not updated, and its BEV features are not fused into the camera representation at this stage. Projected LiDAR measurements remain available as sparse-depth input and valid-pixel depth targets. Camera-branch training therefore combines the detection objective with the focal depth loss defined in Section 3.3. These two sources of guidance act at different levels: decoder initialization supplies a starting point learned for BEV detection, while depth supervision constrains the intermediate depth distribution used for view transformation. The transferred decoder remains trainable, allowing it to adapt to the camera-derived features. Parameter transfer denotes copying pretrained weights at stage transitions; the copied parameters are not tied during subsequent optimization.

Stage III: Multimodal fusion training. The pretrained LiDAR and camera branches are assembled with the BEV fusion pathway. The fusion-stage decoder is initialized from the Stage-I LiDAR decoder, and the Stage-I LGE parameters initialize the corresponding LGE modules in the fusion head. Both modality backbones are frozen, and gradients are disabled through the camera LSS pathway. Training updates the BEV fusion modules and detection head using the detection objective, without auxiliary depth optimization. This restricts the trainable parameter space and prevents fusion-loss gradients from changing the pretrained branch parameters. The fusion modules can consequently adapt to the representations

supplied by the two branches without simultaneously updating their encoders. LGE initialization similarly reuses BEV feature-processing parameters learned during LiDAR detection while allowing the fusion head to adapt to multimodal input. Freezing also limits flexibility: an encoder cannot correct its representation in response to the fusion objective, so the value of this constraint depends on the quality of the pretrained branches.

Relation to existing staged training. Pretrained components and sequential optimization are also used in LiDAR–camera detectors such as TransFusion [21] and BEVFusion-MIT [10]. Co-Fusion couples branch-specific training with decoder and LGE parameter transfer, followed by constrained optimization of the fusion components. Parameter transfer changes the starting point of optimization, whereas freezing determines which parameters can subsequently adapt. Neither operation increases the representational capacity of the inference architecture. Together, they define a training prior whose utility is assessed by the schedule and initialization controls in Section 5.3. These controls support a performance comparison under the evaluated protocol; the optimization rationale above does not establish gradient alignment or the elimination of modality dominance.

4. Experiments

4.1. Datasets

We evaluate Co-Fusion on the nuScenes dataset [37], which contains 1000 driving scenes recorded with six cameras and a 32-beam LiDAR sensor. The six cameras provide 360-degree image coverage synchronized with the LiDAR measurements. The official split contains 700 training, 150 validation, and 150 test scenes. Performance is reported using mean average precision (mAP) and the nuScenes Detection Score (NDS).

4.2. Implementation Details

The model is implemented in PyTorch and MMDetection3D [38]. The main camera configuration uses Swin-Tiny and an FPN for multi-scale feature fusion; the module controls use ResNet-50 unless otherwise stated. The archived camera configuration initializes Swin-Tiny from the NuImages-pretrained checkpoint `swint-nuimages-pretrained-e2e.pth`; thus, the absence of test-time augmentation does not imply training without backbone pretraining. The input image resolution is 384×1056 pixels. In the Lift–Splat–Shoot (LSS) configuration, the frustum ranges are $[-54, 54]$ m, $[-54, 54]$ m, and $[-5, 3]$ m along the x, y, and z axes, respectively, with a voxel size of 0.6 m. The depth range is $[1, 60)$ m with a bin width of 0.5 m, yielding $H_D = 118$ depth bins. The implementation uses $\mu_{\text{depth}} = 14.41$ and a variance of 156.89 (equivalently $\sigma_{\text{depth}} \approx 12.53$), estimated from valid projected depths in the train portion. The BEV resolution is 180×180, aligned with the 8× downsampled BEV features generated by VoxelNet [18], with 128 channels. The depth loss uses sigmoid focal loss with $\alpha = 0.25$, $\gamma = 2$, and loss weight $\lambda = 3$.

For statistics estimation, keyframe points and nine aligned historical sweeps are projected into six 256×704 camera views. Points with camera-frame depth greater than 4 m are retained before selecting the minimum depth per pixel. The valid pixel depths are pooled with equal weight, and the resulting statistics are reused across input resolutions.

The training process consists of three stages. Stage I trains the LiDAR branch and detection head for 20 epochs with CBGS [39]; ground-truth sampling is used during the first 15 epochs and disabled during the final 5 epochs. Stage II initializes the camera decoder from the Stage-I LiDAR decoder and trains the camera branch for 20 epochs without CBGS. Stage III initializes LGE modules 1 and 2 from the Stage-I LGE module 0, freezes both modality backbones, and trains only the fusion and detection components for 6 epochs with CBGS. Training uses four NVIDIA RTX 4090 GPUs, a cumulative batch size of 16, AdamW with an initial learning rate of 1.0×10^{-4}, and a one-cycle learning-rate schedule. On this four-

GPU platform, the approximate wall-clock training times for Stages I, II, and III are 34.5, 7.0, and 12.0 h, respectively, giving a total of approximately 53.5 h. Unless otherwise stated, the reported results use a single frame. Runtime and GPU memory are evaluated under a fixed single-GPU inference setting. Measurements are collected over repeated inference runs after warm-up, with the pre-defined evaluation protocol. These measurements characterize the computational cost of the evaluated configuration; comparisons across different hardware or measurement protocols do not establish an efficiency ranking.

Table 1 maps the main nuScenes results to their experimental configurations. The ResNet-50 reference is used for the module controls unless otherwise stated; it does not describe every backbone or resolution variant.

The implementation is being organized for a future release in the existing Co-Fix3D repository at https://github.com/WenxuanLi-whu/Co-Fix3d (accessed on 1 October 2026).

Mapping of the headline result. The 74.1 mAP/75.3 NDS result in Table 2 is produced by the single-frame Swin-Tiny configuration with the light LiDAR encoder and 384 × 1056 image input shown in Table 1. Note that, it is not the full-channel LiDAR configuration that obtains 73.5 mAP/74.9 NDS on the validation split. The test model is trained on train+val and evaluated through the official nuScenes test server without TTA or model ensembling.

Table 1. Experimental configurations for the reported Co-Fusion results on nuScenes.

Configuration	ResNet-50 Reference	Swin-Tiny Reference	Wider LiDAR Encoder	Test Submission
Image backbone	ResNet-50	Swin-Tiny	Swin-Tiny	Swin-Tiny
LiDAR encoder width	Light	Light	Matched width	Light
Training split	train	train	train	train+val
Evaluation split	val	val	val	test
mAP/NDS (%)	71.7/73.5	73.1/74.2	73.5/74.9	74.1/75.3

Shared settings. All four configurations use a single frame and 384 × 1056 images. Light denotes LiDAR encoder channels $((16, 16, 32), (32, 32, 64), (64, 64, 128), (128, 128))$; matched width denotes $((32, 32, 64), (64, 64, 128), (128, 128, 256), (256, 256))$, matching the IS-Fusion comparison. *Initialization and precision*. The ResNet-50 reference uses the default pretrained weights. The Swin-Tiny reference and test configuration use NuImages-pretrained image weights. Image backbone and neck computation uses FP16 automatic mixed precision; their output features are converted to FP32 before subsequent processing, and the LiDAR branch uses FP32. *Evaluation*. The test submission uses neither test-time augmentation nor model ensembling. Runtime measurements are reported separately later.

4.3. Comparison with State-of-the-Art Methods

3D object detection on test set. Table 2 compares Co-Fusion with representative LiDAR-only (L), single-frame LiDAR–camera (LC), and temporal LiDAR–camera (LCT) detectors on the nuScenes test set. Co-Fusion records the highest aggregate mAP and NDS among the listed methods, although it is not best in every class. Relative to BEVFusion-MIT, the absolute differences are 3.9 mAP points and 2.4 NDS points. The mAP differences relative to FocalFormer3D, SparseFusion, and IS-Fusion are 2.5, 2.1, and 1.1 points, respectively. Co-Fusion is also 0.8/0.6 points above BEVFusion4D and 0.5/0.4 points above GAFusion in mAP/NDS while using a single frame. These published comparisons are not uniformly matched for architecture, pretraining, or test-time augmentation; the controlled ablations below are used to attribute improvements to individual design choices.

3D object detection on the validation set. Table 3 shows that the Swin-Tiny variant of Co-Fusion has the highest mAP and NDS among the listed methods on the nuScenes validation set. Its mAP exceeds those of CMT, SparseFusion, and IS-Fusion by 2.8, 2.1, and 0.3 percentage points, respectively. The 0.3-point margin over IS-Fusion should be inter-

preted in the context of run-to-run variation. The controlled experiments below separately examine the contributions of depth guidance, gated fusion, and the training schedule.

Table 2. 3D object detection performance on the nuScenes test set.

Method		Modality	mAP	NDS	Car	Truck	C.V.	Bus	T.L.	B.R.	M.T.	Bike	Ped.	T.C.
TransFusion-L [21]	[CVPR 22]	L	65.5	70.2	86.2	56.7	28.2	66.3	58.8	78.2	68.3	44.2	86.1	82.0
PillarNet-34 [40]	[ECCV 22]	L	66.0	71.4	87.6	57.5	27.9	63.6	63.1	77.2	70.1	42.3	87.3	83.3
FocalFormer3D [41]	[ICCV 23]	L	68.7	72.6	87.2	57.1	34.4	69.6	64.9	77.8	76.2	49.6	88.2	82.3
PointAugment [25]	[CVPR 21]	LC	66.8	71.1	87.5	57.3	28.0	65.2	60.7	72.6	74.3	50.9	87.9	83.6
AutoAlignV2 [26]	[ECCV 22]	LC	68.4	72.4	87.0	59.0	33.1	69.3	59.3	-	72.9	52.1	87.6	-
TransFusion-LC [21]	[CVPR 22]	LC	68.9	71.7	87.1	60.0	33.1	68.3	60.8	78.1	73.6	52.9	88.4	86.7
BEVFusion [35]	[NeurIPS 22]	LC	69.2	71.8	88.1	60.9	34.4	69.3	62.1	78.2	72.2	52.2	89.2	85.5
BEVFusion-MIT [10]	[ICRA 23]	LC	70.2	72.9	88.6	60.1	39.3	69.8	63.8	80.0	74.1	51.0	89.2	86.5
DeepInteraction [42]	[NeurIPS 22]	LC	70.8	73.4	87.9	60.2	37.5	70.8	63.8	80.4	75.4	54.5	91.7	87.2
ObjectFusion [43]	[ICCV 23]	LC	71.0	73.3	89.4	59.0	40.5	71.8	63.1	80.0	78.1	53.2	90.7	87.7
MSMDFusion [32]	[CVPR 23]	LC	71.5	74.0	88.4	61.0	35.2	71.4	64.2	80.7	76.9	58.3	90.6	88.1
FocalFormer3D [41]	[ICCV 23]	LC	71.6	73.9	88.5	61.4	35.9	71.7	66.4	79.3	80.3	57.1	89.7	85.3
SparseFusion [27]	[ICCV 23]	LC	72.0	73.8	88.0	60.2	38.7	72.0	64.9	79.2	78.5	59.8	90.9	87.9
CMT [44]	[ICCV 23]	LC	72.0	74.1	88.0	63.3	37.3	**75.4**	65.4	78.2	79.1	60.6	87.9	84.7
IS-Fusion [11]	[CVPR 24]	LC	73.0	75.2	88.3	62.7	38.4	74.9	67.3	78.1	**82.4**	59.5	89.3	**89.2**
DAL [9]	[ECCV 24]	LC	72.0	74.8	89.1	60.2	34.6	73.3	65.8	80.6	81.7	58.5	89.6	86.6
GraphBEV [33]	[ECCV 24]	LC	71.7	73.6	89.2	60.0	40.8	72.1	64.5	80.1	76.8	53.3	90.9	88.9
MTA [7]	[T-ITS 25]	LC	71.8	73.7	88.6	63.3	38.4	71.9	64.7	79.4	75.5	57.2	90.8	88.5
Co-Fix3D [12]	[RA-L 25]	LC	72.3	74.7	89.6	60.8	37.9	73.3	65.2	80.0	77.6	62.7	88.7	86.7
BEVFusion4D [45]	[ARXIV 23]	LCT	73.3	74.7	89.7	**65.6**	41.1	72.9	66.0	**81.0**	79.5	58.6	90.9	87.7
GAFusion [46]	[CVPR 24]	LCT	73.6	74.9	89.4	65.3	**42.4**	73.7	65.8	79.2	80.8	60.2	**92.3**	87.0
Co-Fusion (Ours)		LC	**74.1**	**75.3**	**90.3**	64.6	40.8	75.3	**67.5**	79.3	81.8	**63.1**	90.6	87.5

Co-Fusion obtains the highest aggregate mAP and NDS among the methods listed. Co-Fusion uses neither test-time augmentation nor model ensembling; competing values follow the evaluation settings reported in their source publications and are not necessarily augmentation-matched. Modality abbreviations are camera (C), LiDAR (L), and temporal input (T). "C.V.", "T.L.", "B.R.", "M.T.", "Ped.", and "T.C." indicate construction vehicle, trailer, barrier, motorcycle, pedestrian, and traffic cone, respectively. The best result in each column is marked in bold.

Table 3. Performance comparison on the nuScenes validation set. The bold font indicates the best results.

Method		Image Encoder	mAP	NDS
FUTR3D [47]	[CVPR 23]	ResNet-101	64.2	68.0
TransFusion-LC [21]	[CVPR 22]	ResNet-50	67.5	71.3
BEVFusion-MIT [10]	[ICRA 23]	Swin-T	68.5	71.4
DeepInteraction [42]	[NeurIPS 22]	Swin-T	69.9	72.6
CMT [44]	[ICCV 23]	VoV-99	70.3	72.9
SparseFusion [27]	[ICCV 23]	Swin-T	71.0	73.1
IS-Fusion [11]	[CVPR 24]	Swin-T	72.8	74.0
Co-Fusion (Ours)		ResNet-50	71.7	73.5
Co-Fusion (Ours)		Swin-Tiny	**73.1**	**74.2**

5. Ablation Studies

In this section, we analyze the components added by Co-Fusion on top of the inherited Co-Fix3D detection head.

5.1. Component and LDG Ablations

Ablation of Co-Fusion modules. In this setup, the image branch operates at a resolution of 384 × 1056 and uses ResNet-50. Table 4 reports the incremental single-run ablation of depth supervision, depth standardization, and Gated CNN fusion. DepthNet is the encoder

within the standardized LDG branch rather than a separate post-processing module. Table 5 reports additional single-run controls that vary standardization, depth supervision, the sparse-depth encoder, the depth loss, or the DepthNet kernels while retaining the same ResNet-50 experimental protocol. Because these are single-run results, small differences are interpreted descriptively rather than as statistically significant effects.

Independent LDG controls. Table 5 shows that removing depth standardization gives 68.9 mAP and 71.1 NDS, whereas removing auxiliary depth supervision gives 71.8 mAP and 73.2 NDS. The latter result has comparable mAP but 0.3 lower NDS than the reference; the depth loss therefore does not yield a uniform gain in both metrics. This comparison differs from Table 4, where supervision is added to the baseline without standardization or gated fusion. The tables estimate conditional effects in different background configurations, so their mAP differences need not have the same sign. Auxiliary supervision provides a training objective for the intermediate depth distribution, but the available detection results do not establish a stable mAP benefit or identify improvements in individual localization, scale, orientation, velocity, or attribute errors. Replacing the learned DepthNet with fixed 8× downsampling and a 1 × 1 projection gives 71.2/73.2 mAP/NDS, 0.5/0.3 points below the reference. Cross-entropy depth supervision gives 71.3/73.3, while replacing the 7-5-5-5 kernels with 3-3-3-3 gives 71.5/73.4. These single-run results provide descriptive support for the selected design. Consistent with the analysis in Section 3.2, the larger decrease observed without standardization is compatible with the proposed numerical-conditioning explanation: raw metric depths expose the encoder to a broader, non-centered input scale, whereas the standardized input controls the activation and gradient scales encountered during finite-step training. The result does not show that Z-score normalization makes the depth distribution uniform, removes spatial depth jumps, or adds geometric information. Nor does it directly establish lower gradient variance, which would require training-curve or gradient-statistics measurements.

Table 4. Ablation of depth supervision, depth standardization, and Gated CNN fusion on the nuScenes validation set. The bold indicates the best results.

Configuration	Depth	Standardized	Gate CNNs	mAP	NDS
Baseline	–	–	–	70.6	73.1
+depth supervision	✓	–	–	71.0	73.2
+depth standardization	✓	✓	–	71.4	73.3
+Gated CNN fusion (Co-Fusion)	✓	✓	✓	**71.7**	**73.5**

"Depth" denotes depth supervision.

Table 5. Independent single-run controls for the LiDAR-Depth Guidance (LDG) branch on the nuScenes validation set. The bold indicates the best results.

Control	Depth Loss	Standardization	Depth Encoder	Fusion	mAP	NDS
Without standardization	Focal	No	DepthNet (7-5-5-5)	3× Gated CNN	68.9	71.1
Without depth supervision	None	Yes	DepthNet (7-5-5-5)	3× Gated CNN	71.8	73.2
Without learned DepthNet	Focal	Yes	Fixed 8× + 1 × 1 projection	3× Gated CNN	71.2	73.2
Alternative depth loss	Cross-entropy	Yes	DepthNet (7-5-5-5)	3× Gated CNN	71.3	73.3
Alternative DepthNet kernels	Focal	Yes	DepthNet (3-3-3-3)	3× Gated CNN	71.5	73.4
Reference Co-Fusion	Focal	Yes	DepthNet (7-5-5-5)	3× Gated CNN	**71.7**	**73.5**

All controls use the same ResNet-50 protocol. "Fixed 8×" denotes non-learned downsampling followed by a 1 × 1 channel projection.

5.2. Backbone and Architectural Choices

Performance with different image encoders. Table 6 shows that Swin-Tiny gives the highest single-run mAP and NDS among the evaluated image encoders and resolutions. The table

does not isolate which architectural property accounts for the difference. With the same Swin-Tiny backbone, increasing the image resolution from 256 × 704 to 384 × 1056 improves mAP and NDS by 0.5 and 0.3 percentage points, respectively. This numerical comparison should be interpreted together with repeated-run variance.

Effect of the point-cloud encoder. Table 7 compares the light and matched-width Co-Fusion encoders and lists IS-Fusion for context. Expanding the light encoder changes Co-Fusion from 73.1/74.2 to 73.5/74.9 mAP/NDS, corresponding to gains of 0.4 and 0.7 points. The matched-width Co-Fusion result is 0.7 mAP points and 0.9 NDS points above the listed IS-Fusion result. The latency values are retained as hardware-specific references; because the implementations and measurement protocols are not fully matched, they are not used to establish an overall efficiency advantage. Causal attribution is limited to the within-method channel-width comparison.

Effect of Gated CNN depth. Table 8 compares the four Gated CNN depths for which archived results are available. Three blocks give the highest reported mAP, while three and four blocks have the same rounded NDS. Because the margins are small, three blocks are used as a practical setting rather than a statistically established optimum. The final implementation uses an expansion ratio of $r = 8/3$; no verified controlled comparison with other ratios is available, so the former ratio-ablation table has been withdrawn.

Table 6. Performance with different image encoders and input resolutions. The bold indicates the best results.

Image Encoder	Resolution	mAP	NDS
ResNet-50	256 × 704	71.2	73.3
ResNet-50	384 × 1056	71.7	73.5
V2-99	256 × 704	71.8	73.6
V2-99	384 × 1056	72.2	73.8
Swin-Tiny	256 × 704	72.6	73.9
Swin-Tiny	384 × 1056	**73.1**	**74.2**

Table 7. Effect of LiDAR-encoder channel width on nuScenes validation performance.

Method	LiDAR Encoder Channels	LiDAR Latency (ms)	Total Latency (ms)	mAP	NDS
IS-Fusion	((32, 32, 64), (64, 64, 128), (128, 128, 256), (256, 256))	109.0	344.0	72.8	74.0
Co-Fusion (light)	((16, 16, 32), (32, 32, 64), (64, 64, 128), (128, 128))	93.0	255.0	73.1	74.2
Co-Fusion (matched width)	((32, 32, 64), (64, 64, 128), (128, 128, 256), (256, 256))	108.0	271.0	**73.5**	**74.9**

Latencies are archived hardware-specific references rather than a controlled cross-method efficiency comparison.

Table 8. Sensitivity to the number of Gated CNN blocks.

Number of Blocks	mAP	NDS
0	71.4	73.3
1	71.5	73.4
3	71.7	73.5
4	71.6	73.5

The expansion ratio is fixed to $r = 8/3$. Values are single-run results; differences of 0.1–0.3 points are treated as descriptive and not statistically distinguished.

5.3. Training Schedule and Parameter Transfer

Effect of the training strategy. Table 9 compares the ResNet-50 training schedules. Separate branch pretraining with continued camera updates gives 68.9 mAP, below the 70.2 mAP obtained by direct fusion with camera updates. Thus, separate pretraining does not consistently improve the final detection result under these settings. The available measurements

do not identify the cause of this reduction, and no gradient-conflict or modality-dominance explanation is inferred. Because the schedules differ in initialization and branch updates, the table evaluates complete strategies rather than isolating freezing from parameter transfer. A control with identical pretrained parent weights and frozen-branch behavior, differing only in decoder/LGE transfer at Stage III, has not been completed.

Table 9. Ablation of the training schedule. The bold indicates the best results.

Training Strategy	Camera Updated in Fusion	mAP	NDS
LiDAR branch only (Stage I)	–	65.9	70.9
Camera branch only (Stage II)	–	46.2	47.8
Direct fusion; camera frozen	No	66.2	71.2
Direct fusion; camera updated	Yes	70.2	72.1
Separate branch pretraining; camera updated	Yes	68.9	71.6
Three-Stage Parameter-Transfer Training; both branches frozen	No	**71.7**	**73.5**

"Camera updated in fusion" specifies whether the camera branch remains trainable during Stage III. Values are single-run results and support a descriptive comparison of the schedules.

Tables 10 and 11 evaluate decoder and LGE initialization, respectively.

Table 10. Effect of LiDAR-decoder initialization during camera-branch and fusion-stage training.

Decoder Initialization	Image Stage		Fusion Stage	
	mAP	NDS	mAP	NDS
Independent	43.6	47.3	71.3	73.2
LiDAR decoder	46.5	49.2	71.7	73.5

Table 11. Effect of LiDAR-based initialization of different numbers of LGE modules.

Initialized LGE Modules	mAP	NDS
None	70.1	72.4
One	71.4	73.4
Three	71.7	73.5

Effect of parameter transfer. Table 10 shows that initializing the camera decoder from the LiDAR decoder improves the reported camera-stage and fusion-stage values. This result supports LiDAR-derived initialization; analyses of feature alignment or gradient conflict require additional quantitative evidence. Table 11 evaluates LGE initialization. Initializing all three LGE modules from the point-cloud branch gives the highest reported values in this experiment.

Additional Stage-III repetitions. A separate Swin-Tiny control omits decoder/LGE transfer in both Stages II and III. Three completed Stage-III runs share the same Stage-II parent checkpoint: seeds 0, 1, and 2 yield mAP/NDS values of 72.96/74.22, 73.03/74.30, and 73.14/74.33, respectively. Their means and sample standard deviations are 73.04 ± 0.09 mAP and 74.28 ± 0.06 NDS (percentage-point scale, $n = 3$). This control also changes the upstream training path and forces the frozen backbones into evaluation mode. It is therefore reported separately from the ResNet-50 schedule table and does not isolate the effect of fusion-stage transfer. These repetitions quantify conditional variation only for this control; they do not estimate variation in the reference model, standardization controls, or the complete three-stage pipeline.

5.4. Fusion Operator Controls

Table 12 compares the detection performance and complexity of image–depth fusion operators.

Table 12. Comparison of image–depth fusion operators on the nuScenes validation set.

Fusion Operator	Parameters (K)	FLOPs (G)	mAP	NDS
Element-wise addition (with 64 → 256 projection)	16.4	1.3	71.1	73.1
Concatenation + 1 × 1 convolution	81.9	6.2	71.2	73.1
Concatenation + 3 × 3 convolution	737.3	56.1	71.4	73.3
Concatenation + 1 × 1 convolution + CBAM attention	90.2	6.3	71.5	73.3
Three-layer conventional convolutional block	1916.9	145.7	71.3	73.2
Proposed 3× Gated CNN blocks (including input projection)	2352.0	178.4	71.7	73.5

All alternatives use a 256-channel image input, a 64-channel sparse-depth input, and a 256-channel output. Parameters and FLOPs are operator-level analytical counts at six 48 × 132 feature maps, with one multiply–add counted as two FLOPs.

Attention-based and higher-capacity convolutional controls. For the attention baseline, the 256-channel image feature and 64-channel sparse-depth feature are concatenated and projected to 256 channels by a 1 × 1 convolution. A CBAM module with channel reduction ratio 16 and a 7×7 spatial-attention kernel is then applied. Under the counting convention in Table 12, this baseline contains 90.2 K parameters and requires 6.3 G FLOPs while obtaining 71.5 mAP and 73.3 NDS. To test whether increasing the capacity of a conventional fusion operator accounts for the gain, we additionally evaluate a three-layer convolutional block. Given the concatenated 320-channel input, it applies three standard 3 × 3 convolutions with channel widths 320 → 256 → 256 → 256, with ReLU after the first two convolutions. The block contains no multiplicative gate, attention operation, or depthwise convolution. Excluding bias terms, it has 1916.9 K parameters and requires 145.7 G FLOPs, compared with 2352.0 K parameters and 178.4 G FLOPs for the proposed three-block Gated CNN. All operator controls use the same ResNet-50 backbone, 384×1056 input resolution, initialization, augmentations, training schedule, and evaluation protocol. The three-layer conventional block obtains 71.3 mAP and 73.2 NDS, which is 0.1/0.1 points below the single-3 × 3 control despite its greater capacity and 0.4/0.3 points below the proposed Gated CNN. These results indicate that repeatedly mixing the concatenated features with standard convolutions does not improve detection under this protocol. One possible explanation is that uniform convolution cannot selectively regulate sparse depth responses relative to image features, whereas the gated operator performs content-dependent modulation. Because no feature-response analysis is provided, we do not interpret the reduction as direct evidence that depth information itself is detrimental. All differences are reported as single-run descriptive results.

5.5. Sensitivity and Cross-Sensor Evaluation

Sensitivity evaluation protocol. Calibration and normalization perturbations are applied only at inference, retaining each method's trained checkpoint. BEVFusion and Co-Fusion are evaluated under the same deterministic calibration perturbations; the released BEVFusion-MIT result is included only as a literature reference. Each calibration condition applies a fixed positive or negative rotation of 0.5° about one camera-coordinate axis, or a translation of 0.10 m along one such axis, to all six cameras and all validation samples. Updated calibration is used consistently for image geometry and sparse-depth projection. Results are averaged across the six signed axis directions, rather than across random perturbation samples. All Co-Fusion conditions use the same Swin-Tiny Stage-III checkpoint with unperturbed 73.1 mAP

and 74.2 NDS. Mean and standard deviation are perturbed independently or jointly by setting $\mu' = (1 + \delta_\mu)\mu$ and $v' = (1 + \delta_\sigma)^2 v$, with $\delta_\mu, \delta_\sigma \in \{-0.2, -0.1, 0, 0.1, 0.2\}$ as listed in Table 13. The implementation stores variance $v = \sigma^2$; a standard-deviation perturbation therefore changes variance quadratically. BEVFusion has no LDG standardization operation. Table 13 summarizes the inference-time perturbation results.

Table 13. Sensitivity to inference-time calibration and depth-statistics perturbations on the nuScenes validation set.

Method	Inference Perturbation	$\Delta\mu/\Delta\sigma$	mAP	NDS	ΔmAP
BEVFusion-MIT (reported) [10]	None	-	68.5	71.4	–
BEVFusion (matched re-run)	None	–	68.3	71.2	–
Co-Fusion (Swin-T)	None	0%/0%	73.1	74.2	–
BEVFusion (matched re-run)	Rotation ±0.5°	–	68.3	71.1	−0.1
Co-Fusion (Swin-T)	Rotation ±0.5°	0%/0%	72.8	74.0	−0.4
BEVFusion (matched re-run)	Translation ±0.10 m	–	68.3	71.1	0.0
Co-Fusion (Swin-T)	Translation ±0.10 m	0%/0%	73.0	74.1	−0.1
Co-Fusion (Swin-T)	None	+10%/0%	70.0	72.1	−3.1
Co-Fusion (Swin-T)	None	−10%/0%	67.9	70.6	−5.2
Co-Fusion (Swin-T)	None	+20%/0%	65.6	69.7	−7.5
Co-Fusion (Swin-T)	None	−20%/0%	55.6	63.9	−17.5
Co-Fusion (Swin-T)	None	0%/ + 10%	71.8	73.4	−1.4
Co-Fusion (Swin-T)	None	0%/ − 10%	71.2	73.0	−1.9
Co-Fusion (Swin-T)	None	0%/ + 20%	69.3	72.0	−3.8
Co-Fusion (Swin-T)	None	0%/ − 20%	67.1	70.6	−6.0
Co-Fusion (Swin-T)	None	+10%/ + 10%	68.5	71.3	−4.7
Co-Fusion (Swin-T)	None	−10%/ − 10%	64.5	68.9	−8.7
Co-Fusion (Swin-T)	None	+20%/ + 20%	62.4	68.1	−10.7
Co-Fusion (Swin-T)	None	−20%/ − 20%	47.9	60.3	−25.2

Rotation and translation results average the $\pm x$, $\pm y$, and $\pm z$ directions; ΔmAP is measured relative to the corresponding unperturbed result. Differences are calculated before rounding the displayed scores.

Sensitivity results. The matched BEVFusion re-run achieves 68.3 mAP and 71.2 NDS without perturbation. Averaged over the six signed coordinate-axis directions, its rotation and translation results are 68.3/71.1 and 68.3/71.1, corresponding to mAP changes of −0.1 and 0.0 points, respectively. The Swin-Tiny Co-Fusion model retains 72.8 mAP and 74.0 NDS under the ±0.5° rotation perturbations (a −0.4-point mAP change), while the analogous ±0.10 m translation perturbations give 73.0 mAP and 74.1 NDS (a −0.1-point mAP change). Starting from the same 73.1 mAP and 74.2 NDS checkpoint, joint +10% and −10% depth-statistics perturbations reduce mAP by 4.7 and 8.7 points, respectively; increasing the magnitude to 20% produces reductions of 10.7 and 25.2 points. The larger degradation for negative perturbations indicates an asymmetric dependence on the fixed normalization constants. Co-Fusion loses more mAP under rotation than the BEVFusion re-run (0.4 versus 0.1 points). An additional dependence on projected depth is a possible explanation but is not isolated by these experiments. Calibration conclusions are limited to the tested ±0.5° and ±0.10 m conditions. Across all four relative perturbation levels, changing only the mean causes larger mAP reductions than changing only the standard deviation. For example, a −20% mean change gives 55.6 mAP, a −20% standard-deviation change gives 67.1, and the joint change gives 47.9. These fixed-checkpoint results demonstrate sensitivity to the normalization constants, rather than general robustness.

Cross-sensor evaluation on KITTI. We use the common 3712/3769 train/validation split, one front camera with a 384 × 1280 input, the point-cloud range $[0, -40, -3, 70.4, 40, 1]$ m, and a voxel size of $[0.05, 0.05, 0.1]$ m. Valid projected depths from the 3712 KITTI training

images have mean $\mu = 16.5155$ m and population standard deviation $\sigma = 12.3949$ m; the implementation stores the corresponding variance, $\sigma^2 = 153.6335$ m^2, and evaluates $(d - \mu)/\sqrt{\sigma^2}$. To isolate the effect of transferring the fixed prior, we evaluate the same Stage-III epoch-5 checkpoint twice. The resolved configurations and evaluation protocol are identical except for the inference-time depth statistics: the KITTI values above are replaced by the nuScenes values $\mu = 14.41$ m and $\sigma^2 = 156.89$ m^2 ($\sigma \approx 12.53$ m). Thus, this control measures sensitivity to a misspecified normalization prior without retraining. The archived Stage-II log does not record the runtime constants, so the Stage-II result is reported only as a camera-stage performance reference and is excluded from all normalization comparisons.

Tables 14 and 15 report the KITTI 3D and BEV detection results, respectively.

Table 14. KITTI validation 3D AP_{R40} (%). The selected Stage-I, Stage-II, and Stage-III entries do not constitute a matched sequential training pipeline: Stage III loads a different archived Stage-II checkpoint from the one displayed. Stage-I/Stage-III differences are descriptive and cannot isolate the benefit of multimodal fusion. The bold font indicates the best results.

Stage	Modality	Car			Pedestrian			Cyclist			Mean AP
		E	M	H	E	M	H	E	M	H	
TransFusion-LC (reported)	LiDAR + camera	86.3	78.9	70.8	59.0	52.1	42.2	76.1	62.3	57.3	65.0
MVXNet (reported by Han et al.)	LiDAR + camera	88.1	78.4	74.0	61.2	55.7	52.1	73.2	54.6	51.4	65.4
Stage I	LiDAR	87.7	**80.0**	**79.2**	55.1	52.3	49.8	85.9	67.4	63.9	69.0
Stage II (reference run)	Camera	58.4	45.2	39.8	16.6	13.5	11.9	28.5	16.1	14.9	27.2
Stage III (nuScenes statistics at inference)	LiDAR + camera	86.4	78.8	76.7	**60.6**	**55.1**	**51.6**	84.0	66.3	62.7	69.1
Stage III (KITTI statistics)	LiDAR + camera	**87.8**	79.7	77.4	59.0	54.0	50.9	**87.1**	**68.1**	**64.5**	**69.8**

The TransFusion-LC and MVXNet rows are external reported references [48,49]. The two Stage-III rows use the same trained checkpoint and evaluation configuration; only the inference-time depth-normalization statistics are changed. The Stage-II row is retained only as a camera-stage performance reference because its archived log does not record the runtime normalization constants; it is not used to support conclusions about standardization. E, M, and H denote the Easy, Moderate, and Hard difficulty levels. Mean AP is the unweighted mean of the nine class–difficulty entries.

Table 14 reports AP^{3D}_{R40} for the KITTI front-camera setting. With KITTI statistics, Stage III obtains a nine-entry mean of 69.8, 0.8 points above the selected Stage I result. It improves all Pedestrian and Cyclist entries but reduces Moderate and Hard Car AP by 0.4 and 1.8 points, respectively. The camera-only Stage-II result is lower than that of the LiDAR branch, consistent with the more limited BEV coverage of one front-facing camera relative to the six-camera nuScenes setting. Table 15 reports the corresponding AP^{BEV}_{R40} results: the KITTI-statistics evaluation reaches 77.2, compared with 74.7 for Stage I. Substituting the nuScenes statistics in the same Stage-III checkpoint gives 69.1 mean 3D AP and 76.6 mean BEV AP, decreases of 0.7 and 0.7 points, respectively. These differences are calculated from the unrounded results. The per-class response is non-uniform: Pedestrian 3D AP increases slightly, whereas the largest reductions occur for Cyclist. The selected Stage-I, Stage-II, and Stage-III entries are representative verified runs and do not form a strictly matched sequential training pipeline; in particular, the selected Stage III loads a different archived Stage-II checkpoint from the Stage-II row shown in the table. This limitation does not affect the paired Stage-III normalization control, for which the checkpoint and all other evaluation settings are fixed. For context, TransFusion-LC and MVXNet report nine-entry means of 65.0/73.2 and 65.4/72.1 for 3D/BEV AP, respectively [48,49]. These values are external literature references; their implementations and optimization protocols are not matched to ours, so no controlled superiority claim is made.

Table 15. KITTI validation AP_{R40}^{BEV} (%). The selected Stage-I, Stage-II, and Stage-III entries do not constitute a matched sequential training pipeline: Stage III loads a different archived Stage-II checkpoint from the one displayed. Stage-I/Stage-III differences are descriptive and cannot isolate the benefit of multimodal fusion. The bold font indicates the best results.

Stage	Modality	Car			Pedestrian			Cyclist			Mean AP
		E	M	H	E	M	H	E	M	H	
TransFusion-LC (reported)	LiDAR + camera	90.2	82.2	75.8	75.0	70.2	62.2	77.0	65.2	60.6	73.2
MVXNet (reported by Han et al.)	LiDAR + camera	92.3	88.0	85.6	70.3	64.4	60.8	75.6	57.4	54.1	72.1
Stage I	LiDAR	91.9	89.0	87.9	61.8	58.7	56.5	**88.9**	70.2	66.8	74.7
Stage III (nuScenes statistics at inference)	LiDAR + camera	92.3	89.4	88.7	**69.4**	65.2	61.5	86.6	69.9	66.1	76.6
Stage III (KITTI statistics)	LiDAR + camera	**93.1**	**89.6**	**88.9**	69.2	**65.4**	**61.9**	87.9	**71.4**	**67.6**	**77.2**

The TransFusion-LC and MVXNet rows are external reported references [48,49]. The two Co-Fusion Stage-III rows use the same trained checkpoint and evaluation configuration; only the inference-time depth-normalization statistics are changed. IoU thresholds are 0.7 for Car and 0.5 for Pedestrian and Cyclist. Mean AP is the unweighted mean of the nine class–difficulty entries.

5.6. Runtime Measurement

Table 16 reports a standardized measurement of Co-Fusion on one RTX 3090. The mean and sample standard deviation were independently recomputed from the 500 recorded latencies.

Table 16. Co-Fusion runtime and peak GPU memory under the recorded measurement protocol.

Setting or Metric	Recorded Value
Model/image input	Swin-Tiny Stage III/six 384 × 1056 views
GPU/batch size	One RTX 3090/1
Precision	Model-native mixed precision
Warm-up/measured iterations	50/500
Latency, mean ± sample SD	357.67 ± 8.72 ms
Latency, P50/P95	357.68/373.86 ms
Peak allocated/reserved memory	1.193/2.031 GiB

Timing includes model preprocessing, CPU-to-GPU transfer, forward inference, decoding, and model post-processing; dataset I/O/pipeline processing, evaluation, and result serialization are excluded. The sequential validation loader supplies the first 550 batches (50 warm-up, 500 measured). After warm-up, the CUDA cache is cleared and peak-memory counters are reset; the memory peaks include the resident model. The environment uses Python 3.10.0, PyTorch 2.0.1+cu117, CUDA 11.7, and cuDNN 8.5.

Measurement scope. This measurement uses the archived checkpoint underlying the 73.1 mAP/74.2 NDS validation result and preserves its internal mixed-precision paths. Components were not independently timed under this protocol, so historical gated-fusion and LSS timings are not combined with the new total. A historical author measurement of CMT [44] gave 266 ms on an RTX 4090D, but its complete timing protocol was not retained. It is therefore omitted from the quantitative comparison table. Different hardware, image resolutions, and timing boundaries prevent an efficiency ranking. LSS computational cost depends on the number of views, depth bins, and feature-map resolution, but the present timing does not isolate these components.

5.7. Qualitative Results

Visualization. Figure 6 presents one qualitative nuScenes scene. It illustrates examples in which Co-Fusion and the BEVFusion baseline make different predictions; quantitative sensitivity is assessed separately in Section 5.5.

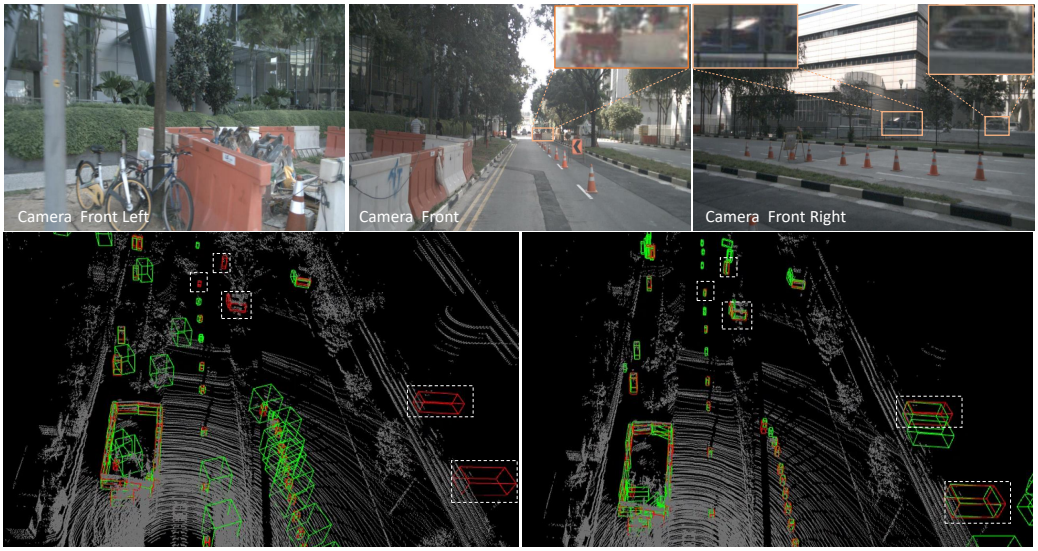


Figure 6. Qualitative results on the nuScenes validation set. Green boxes denote predictions, and red boxes denote ground truth (GT). White dashed boxes in the point cloud and orange boxes in the image highlight differences between *Co-Fusion* and the *BEVFusion* baseline.

6. Discussion

The current design depends on projected LiDAR measurements and therefore cannot provide the same guidance when LiDAR is unavailable or severely degraded. Its normalization constants are also sensor- and dataset-dependent and should be estimated from the training split of each target platform. On nuScenes, jointly decreasing the inference-time mean and standard deviation by 20% reduces the Swin-Tiny configuration from 73.1 to 47.9 mAP while jointly increasing them by 20% gives 62.4 mAP. In the matched KITTI control, replacing the KITTI statistics with the numerically closer nuScenes statistics reduces the mean 3D and BEV AP by 0.7 points each. These results show that the effect depends on the magnitude and direction of the statistic mismatch; they support validating fixed statistics for the target sensor rather than assuming invariance across datasets. The broader KITTI experiment reports non-uniform class-wise differences between representative stages; their unmatched training histories prevent attributing the aggregate difference solely to fusion or establishing general cross-sensor transfer.

DepthNet takes a single-channel standardized sparse depth map, with invalid pixels set to zero; the validity mask is not supplied as an additional input channel. A preliminary depth-plus-mask experiment did not improve detection performance, so the single-channel input was retained. The representation of sparse-measurement confidence remains open for further study. The three-stage schedule also increases total training cost relative to a single end-to-end run. In addition, LSS entails a resolution–latency trade-off because the cost of image-to-BEV lifting grows with image resolution, the number of depth bins, and the number of camera views. The present evaluation is limited to single-frame detection. Temporal fusion, broader calibration errors, and sensor-degradation conditions require further investigation. The LDG formulation may be applicable to other co-calibrated active–passive sensing platforms, including mobile-mapping and UAV LiDAR–camera systems, but its sensor-specific normalization would need to be re-estimated and validated.

Most component and schedule ablations remain single-run experiments. The three Stage-III runs apply only to a separate Swin-Tiny no-transfer control and do not establish the variability of the reference or standardization ablations. Differences of 0.1–0.5 points should therefore be interpreted descriptively; their stability across training seeds remains

unresolved. Freezing and parameter transfer are not fully separated by the current schedule controls. Sensitivity to depth-statistics mismatch is a practical limitation: a joint −20% shift causes a 25.2-point mAP loss, and the mean-only perturbations cause larger losses than corresponding standard-deviation-only perturbations. These results motivate checking the target sensor's depth statistics before deployment.

7. Conclusions

This paper presents Co-Fusion, an extension of Co-Fix3D for LiDAR–camera 3D object detection. The new components are a LiDAR-depth guidance pipeline, gated joint image–depth feature modulation, and a three-stage parameter-transfer schedule. Rather than explicitly completing a dense depth map, LDG encodes normalized sparse projected LiDAR measurements into features aligned with the image branch, while a valid-pixel depth loss supervises the predicted depth distribution. On nuScenes, the reported experiments show that these additions improve the detection performance of the Co-Fix3D-based system. The strong sensitivity to mismatched depth statistics, unresolved small-margin variability, LiDAR dependence, training cost, and single-frame evaluation limit the current evidence for deployment on other platforms.

Author Contributions: Conceptualization, W.L., Q.Z. and S.C.; methodology, W.L. and Q.Z.; software, W.L.; validation, W.L.; formal analysis, W.L., C.C. and Q.M.; investigation, W.L.; resources, Q.Z. and S.C.; data curation, W.L.; writing—original draft preparation, W.L.; writing—review and editing, Q.Z., S.C., C.C. and Q.M.; visualization, W.L.; supervision, Q.Z. and S.C.; project administration, Q.Z. and S.C.; funding acquisition, Q.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Open Research Fund from Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ) under Grant No. GML-KF-24-35.

Data Availability Statement: The datasets used in this study are available from the official nuScenes website at https://www.nuscenes.org/nuscenes (accessed on 1 October 2026) and KITTI website at https://www.cvlibs.net/datasets/kitti/ (accessed on 1 October 2026), subject to their respective access requirements and terms of use. The Co-Fusion implementation has not yet been publicly released. The planned release location is the existing Co-Fix3D repository at https://github.com/WenxuanLi-whu/Co-Fix3d (accessed on 1 October 2026). The intended Co-Fusion package comprises the LDG and gated-fusion modules, three-stage configurations and training/evaluation commands, parameter-transfer and freezing rules, depth-statistics processing instructions, checkpoint-to-configuration mappings, pretrained checkpoints, and the sensitivity and runtime measurement scripts. The Co-Fusion-specific source code will be released within the next two months, by 12 December 2026. Release preparation is ongoing; the availability date and download links for pretrained checkpoints have not yet been finalized.

Acknowledgments: The authors would like to thank all members of the laboratory for their technical assistance and valuable discussions during this study. During the preparation of this manuscript, the authors used ChatGPT-4o (OpenAI) solely for English language editing to improve grammar, clarity, and readability. The authors reviewed and revised all suggested edits and take full responsibility for the content of the manuscript.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Mao, J.; Xue, Y.; Niu, M.; Bai, H.; Feng, J.; Liang, X.; Xu, H.; Xu, C. Voxel transformer for 3D object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 3164–3173.
2. Jia, F.; Jia, C.; Liu, A.; Xu, S.; Xia, Q.; Liu, L.; Yang, L.; Gong, Y.; Song, Z. DGFusion: Dual-guided Fusion for Robust Multi-Modal 3D Object Detection. *IEEE Trans. Circuits Syst. Video Technol.* **2025**, *36*, 4956–4970. [CrossRef]

3. Song, Z.; Liu, L.; Jia, F.; Luo, Y.; Jia, C.; Zhang, G.; Yang, L.; Wang, L. Robustness-aware 3D object detection in autonomous driving: A review and outlook. *IEEE Trans. Intell. Transp. Syst.* **2024**, *25*, 15407–15436. [CrossRef]
4. Ding, R.; Kuang, Z.; Ji, Y.; Yang, M.; Zheng, X.; Hua, G. Multi-Modal Decouple and Recouple Network for Robust 3D Object Detection. *IEEE Trans. Circuits Syst. Video Technol.* **2026**, *36*, 5815–5827. [CrossRef]
5. Liu, H.; Du, J.; Zhang, Y.; Zhang, H.; Zeng, J. PVConvNet: Pixel-voxel sparse convolution for multimodal 3D object detection. *Pattern Recognit.* **2024**, *149*, 110284. [CrossRef]
6. Li, H.; Zhao, Y.; Zhong, J.; Wang, B.; Sun, C.; Sun, F. Delving into the secrets of BEV 3D object detection in autonomous driving: A comprehensive survey. *IEEE Trans. Intell. Transp. Syst.* **2025**, *27*, 119–144. [CrossRef]
7. Lin, C.; He, Z.; Qiu, Y.; Huang, Y. A method of time alignment in bev features for multimodal fusion object detection of intelligent vehicles. *IEEE Trans. Intell. Transp. Syst.* **2025**, *26*, 12432–12445. [CrossRef]
8. Hu, H.; Wang, F.; Su, J.; Wang, Y.; Hu, L.; Fang, W.; Xu, J.; Zhang, Z. Ea-lss: Edge-aware lift-splat-shot framework for 3D bev object detection. *arXiv* **2023**, arXiv:2303.17895.
9. Huang, J.; Ye, Y.; Liang, Z.; Shan, Y.; Du, D. Detecting as labeling: Rethinking lidar-camera fusion in 3D object detection. *arXiv* **2023**, arXiv:2311.07152.
10. Liu, Z.; Tang, H.; Amini, A.; Yang, X.; Mao, H.; Rus, D.L.; Han, S. Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. In *Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA)*; IEEE: Piscataway, NJ, USA, 2023; pp. 2774–2781.
11. Yin, J.; Shen, J.; Chen, R.; Li, W.; Yang, R.; Frossard, P.; Wang, W. Is-fusion: Instance-scene collaborative fusion for multimodal 3D object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 16–22 June 2024; pp. 14905–14915.
12. Li, W.; Zou, Q.; Chen, C.; Du, B.; Chen, L.; Zhou, J.; Yu, H. Co-Fix3D: Enhancing 3D Object Detection With Collaborative Refinement. *IEEE Robot. Autom. Lett.* **2025**, *10*, 4970–4977. [CrossRef]
13. Cui, Y.; Li, Z.; Fang, Z. Dynamic Clustering Transformer for LiDAR-Based 3D Object Detection. *Pattern Recognit.* **2025**, *172*, 112444. [CrossRef]
14. He, Q.; Wang, Z.; Zeng, H.; Zeng, Y.; Liu, Y.; Liu, S.; Zeng, B. Stereo RGB and deeper LiDAR-based network for 3D object detection in autonomous driving. *IEEE Trans. Intell. Transp. Syst.* **2022**, *24*, 152–162. [CrossRef]
15. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. Pointnet: Deep learning on point sets for 3D classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660.
16. Yang, Z.; Sun, Y.; Liu, S.; Jia, J. 3DSSD: Point-based 3D single stage object detector. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11040–11048.
17. Fan, L.; Wang, F.; Wang, N.; Zhang, Z.X. Fully sparse 3D object detection. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 351–363. [CrossRef]
18. Zhou, Y.; Tuzel, O. Voxelnet: End-to-end learning for point cloud based 3D object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4490–4499.
19. Yan, Y.; Mao, Y.; Li, B. Second: Sparsely embedded convolutional detection. *Sensors* **2018**, *18*, 3337. [CrossRef] [PubMed]
20. Yin, T.; Zhou, X.; Krahenbuhl, P. Center-based 3D object detection and tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 11784–11793.
21. Bai, X.; Hu, Z.; Zhu, X.; Huang, Q.; Chen, Y.; Fu, H.; Tai, C.L. Transfusion: Robust lidar-camera fusion for 3D object detection with transformers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 1090–1099.
22. Song, Z.; Wei, H.; Jia, C.; Xia, Y.; Li, X.; Zhang, C. VP-Net: Voxels as points for 3-D object detection. *IEEE Trans. Geosci. Remote Sens.* **2023**, *61*, 1–12. [CrossRef]
23. Wang, X.; Li, K.; Chehri, A. Multi-sensor fusion technology for 3D object detection in autonomous driving: A review. *IEEE Trans. Intell. Transp. Syst.* **2023**, *25*, 1148–1165. [CrossRef]
24. Vora, S.; Lang, A.H.; Helou, B.; Beijbom, O. Pointpainting: Sequential fusion for 3D object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 4604–4612.
25. Wang, C.; Ma, C.; Zhu, M.; Yang, X. Pointaugmenting: Cross-modal augmentation for 3D object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 11794–11803.
26. Chen, Z.; Li, Z.; Zhang, S.; Fang, L.; Jiang, Q.; Zhao, F. Deformable feature aggregation for dynamic multi-modal 3D object detection. In *Proceedings of the European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 628–644.
27. Xie, Y.; Xu, C.; Rakotosaona, M.J.; Rim, P.; Tombari, F.; Keutzer, K.; Tomizuka, M.; Zhan, W. Sparsefusion: Fusing multi-modal sparse representations for multi-sensor 3D object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 1–6 October 2023; pp. 17591–17602.
28. Li, Y.; Ge, Z.; Yu, G.; Yang, J.; Wang, Z.; Shi, Y.; Sun, J.; Li, Z. Bevdepth: Acquisition of reliable depth for multi-view 3D object detection. *Proc. Aaai Conf. Artif. Intell.* **2023**, *37*, 1477–1485. [CrossRef]

29. Li, Y.; Bao, H.; Ge, Z.; Yang, J.; Sun, J.; Li, Z. BEVStereo: Enhancing Depth Estimation in Multi-View 3D Object Detection with Temporal Stereo. *Proc. Aaai Conf. Artif. Intell.* **2023**, *37*, 1486–1494. [CrossRef]
30. Zhang, J.; Zhang, Y.; Liu, Q.; Wang, Y. SA-BEV: Generating Semantic-Aware Bird's-Eye-View Feature for Multi-view 3D Object Detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 1–6 October 2023; pp. 3348–3357.
31. Wang, H.; Tang, H.; Shi, S.; Li, A.; Li, Z.; Schiele, B.; Wang, L. UniTR: A Unified and Efficient Multi-Modal Transformer for Bird's-Eye-View Representation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 1–6 October 2023; pp. 6792–6802.
32. Jiao, Y.; Jie, Z.; Chen, S.; Chen, J.; Ma, L.; Jiang, Y.G. Msmdfusion: Fusing lidar and camera at multiple scales with multi-depth seeds for 3D object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 21643–21652.
33. Song, Z.; Yang, L.; Xu, S.; Liu, L.; Xu, D.; Jia, C.; Jia, F.; Wang, L. GraphBEV: Towards Robust BEV Feature Alignment for Multi-Modal 3D Object Detection. In *Proceedings of the European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2024; pp. 347–366. [CrossRef]
34. Li, W.; Zhou, J.; Chen, C.; Yu, H.; Du, B.; Zou, Q. BEVFix: Deep feature enhancement for robust 3D object detection. *Neural Netw.* **2025**, *190*, 107675. [CrossRef] [PubMed]
35. Liang, T.; Xie, H.; Yu, K.; Xia, Z.; Lin, Z.; Wang, Y.; Tang, T.; Wang, B.; Tang, Z. Bevfusion: A simple and robust lidar-camera fusion framework. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 10421–10434. [CrossRef]
36. Huang, J.; Huang, G. Bevpoolv2: A cutting-edge implementation of bevdet toward deployment. *arXiv* **2022**, arXiv:2211.17111.
37. Caesar, H.; Bankiti, V.; Lang, A.H.; Vora, S.; Liong, V.E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; Beijbom, O. nuscenes: A multimodal dataset for autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11621–11631.
38. Contributors, M. MMDetection3D: OpenMMLab Next-Generation Platform for General 3D Object Detection. 2022. Available online: https://github.com/open-mmlab/mmdetection3d (accessed on 1 November 2021).
39. Zhu, B.; Jiang, Z.; Zhou, X.; Li, Z.; Yu, G. Class-balanced grouping and sampling for point cloud 3D object detection. *arXiv* **2019**, arXiv:1908.09492.
40. Shi, G.; Li, R.; Ma, C. Pillarnet: Real-time and high-performance pillar-based 3D object detection. In *Proceedings of the European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 35–52.
41. Chen, Y.; Yu, Z.; Chen, Y.; Lan, S.; Anandkumar, A.; Jia, J.; Alvarez, J.M. Focalformer3D: Focusing on hard instance for 3D object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 1–6 October 2023; pp. 8394–8405.
42. Yang, Z.; Chen, J.; Miao, Z.; Li, W.; Zhu, X.; Zhang, L. Deepinteraction: 3D object detection via modality interaction. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 1992–2005. [CrossRef]
43. Cai, Q.; Pan, Y.; Yao, T.; Ngo, C.W.; Mei, T. Objectfusion: Multi-modal 3D object detection with object-centric fusion. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 1–6 October 2023; pp. 18067–18076.
44. Yan, J.; Liu, Y.; Sun, J.; Jia, F.; Li, S.; Wang, T.; Zhang, X. Cross modal transformer: Towards fast and robust 3D object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 1–6 October 2023; pp. 18268–18278.
45. Cai, H.; Zhang, Z.; Zhou, Z.; Li, Z.; Ding, W.; Zhao, J. Bevfusion4d: Learning lidar-camera fusion under bird's-eye-view via cross-modality guidance and temporal aggregation. *arXiv* **2023**, arXiv:2303.17099.
46. Li, X.; Fan, B.; Tian, J.; Fan, H. GAFusion: Adaptive Fusing LiDAR and Camera with Multiple Guidance for 3D Object Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 16–22 June 2024; pp. 21209–21218.
47. Chen, X.; Zhang, T.; Wang, Y.; Wang, Y.; Zhao, H. Futr3D: A unified sensor fusion framework for 3D detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 172–181.
48. Liu, H.; Duan, T. Real-Time Multimodal 3D Object Detection with Transformers. *World Electr. Veh. J.* **2024**, *15*, 307. [CrossRef]
49. Han, Y.; Yu, M.; Liu, J. Adaptive Multiscale Attention Feature Aggregation for Multi-Modal 3D Occluded Object Detection. *IET Comput. Vis.* **2025**, *19*, e70035. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.