IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p-ISSN: 2278-8727, Volume 28, Issue 3, Ser. III (May. – June. 2026), PP 78-86
www.iosrjournals.org

HAT-Point Pillars: Hierarchical Attention Transformer for Vehicle-Side and Infrastructure-Side LiDAR 3D Detection

Le Wang
(Transportation Institute of Inner Mongolia University, Hohhot 010070, China)

Abstract:
Background*: LiDAR-based 3D object detection is important for intelligent transportation, autonomous driving and roadside perception. PointPillars provides an efficient pillar-based pipeline, but its compact representation may be affected by background noise, sparse small-object points, weak pillar-context modeling and limited proposal refinement, especially when vehicle-side and infrastructure-side scenes are evaluated together.*
Materials and Methods*: This paper proposes HAT-PointPillars, a hierarchical attention transformer framework for vehicle-side and infrastructure-side LiDAR 3D detection. The method includes Depth-Adaptive Density Preprocessing (DADP), Intra-Inter Dual Context Attention (IDCA), an Attention-Residual Feature Backbone with Pyramid Multi-scale Fusion (ARFB-PMF), and Channel-weighted Transformer Refinement (CTRF). Experiments are conducted on KITTI and DAIR-V2X-I using an OpenPCDet-based implementation.*
Results*: Under the reported validation protocol, KITTI results are reported as 3D AP_R40 at the Moderate difficulty level with IoU thresholds of 0.7 for Car and 0.5 for Pedestrian/Cyclist. HAT-PointPillars obtains 79.87% Car AP, 54.92% Pedestrian AP and 72.56% Cyclist AP while maintaining 25.60 FPS. On the customized KITTI-style DAIR-V2X-I infrastructure-side validation split, it obtains 71.08% Moderate mAP and 25.20 FPS.*
Conclusion: *The proposed method improves vehicle, pedestrian and cyclist detection under the reported protocol and provides a practical balance between detection accuracy and real-time inference. The study focuses on single-side LiDAR detection on vehicle-side and infrastructure-side datasets rather than explicit V2X feature-level fusion.*
Key Word*: LiDAR 3D object detection; PointPillars; vehicle-side perception; infrastructure-side perception; proposal refinement; BEV pseudo-image.*

Date of Submission: 08-06-2026 Date of Acceptance: 19-06-2026

I. Introduction

LiDAR-based 3D object detection is a core perception task in autonomous driving and intelligent transportation systems. Vehicle-mounted LiDAR provides direct geometric measurements for ego-vehicle perception, while infrastructure-side LiDAR can extend the observation range and reduce blind areas caused by occlusion. In both cases, accurate and real-time detection of vehicles, pedestrians and cyclists is essential for traffic understanding and decision support.

Existing LiDAR 3D detectors include point-based, voxel-based, pillar-based and point-voxel methods. Point-based methods such as PointRCNN preserve raw point geometry but usually require higher computation [1-3]. Voxel-based detectors such as VoxelNet and SECOND convert irregular point clouds into regular voxel representations and improve sparse 3D feature extraction [4,5]. Point-voxel methods such as PV-RCNN combine voxel features and point-wise abstraction to improve proposal refinement [6]. PointPillars remains attractive because it converts point clouds into vertical pillars and uses efficient 2D convolution on the generated bird's-eye-view (BEV) pseudo-image [7].

Recent studies have improved LiDAR detection through stronger backbones, residual structures, attention mechanisms, transformer-based aggregation and proposal refinement. Voxel R-CNN, PillarNet, Point-GNN and CenterPoint show the benefit of improved feature aggregation and detection heads [8-11]. Attention modules such as SE and SimAM provide lightweight feature reweighting for convolutional networks [12,13]], while residual learning stabilizes deep feature extraction [14]. Transformer-style attention has been widely adopted for long-range dependency modeling [15]. Focal loss is widely used in dense detection to address class imbalance [16]. Multimodal methods such as MVX-Net, PointPainting, EPNet and CLOCs demonstrate the potential of image-LiDAR fusion, although they introduce additional calibration and computation requirements

[17-21]. Datasets such as KITTI, DAIR-V2X and nuScenes provide important benchmarks for vehicle-side, infrastructure-side and multimodal perception [22-24].

Despite these advances, PointPillars still has limitations in vehicle-side and infrastructure-side LiDAR scenes. First, fixed preprocessing may remove valid sparse points of distant pedestrians and cyclists or retain irrelevant background points. Second, the original pillar feature encoder mainly aggregates points within each pillar and does not explicitly model neighboring-pillar context. Third, BEV features need multi-scale enhancement for objects with different sizes. Fourth, single-stage predictions may have limited refinement ability for small, distant or partially occluded targets.

To address these issues, this paper proposes HAT-PointPillars, a hierarchical attention transformer framework for vehicle-side and infrastructure-side LiDAR 3D detection. The contributions are problem-driven: (1) DADP is designed for depth-adaptive noise suppression and sparse small-object point preservation without using ground-truth labels; (2) IDCA models both intra-pillar point relations and inter-pillar context in a lightweight single-head attention form; (3) ARFB-PMF strengthens multi-scale BEV representations through residual attention learning and pyramid fusion; and (4) CTRF refines high-confidence proposals using transformer-based proposal encoding and channel-weighted decoding. The paper reports focused KITTI Moderate validation and a customized DAIR-V2X-I infrastructure-side validation protocol.

Table no 1: Method-level difference between HAT-PointPillars and related baselines.

Method	Preprocess	Pillar context	BEV backbone	Proposal refinement	Role in comparison
PointPillars	Fixed filtering	No explicit attention	2D CNN	No	Efficient baseline
DADP-IDCA PP	Depth-adaptive filtering	Intra-inter attention	2D CNN	No	Improves local context
Transformer-refined PointPillars	Fixed filtering	PointPillars encoder	2D CNN	Transformer refinement	Accuracy gain with high cost
HAT-PointPillars	DADP	IDCA	ARFB-PMF	CTRF	Balanced accuracy and speed

Table no 1 summarizes the design differences. The proposed method is not presented as a pure replacement of existing detectors, but as a lightweight PointPillars-based framework that jointly addresses preprocessing, pillar context, BEV feature representation and proposal refinement.

II. Material And Methods

2.1 Overall Framework

The proposed HAT-PointPillars follows a pipeline of point-cloud preprocessing, pillar feature encoding, BEV feature extraction, proposal generation, proposal refinement and final 3D detection. The raw point cloud is first processed by DADP. The retained points are then pillarized and encoded by IDCA. The encoded pillar features are scattered to a BEV pseudo-image and processed by ARFB-PMF. The first-stage detection head generates proposals, and high-confidence proposals are further refined by CTRF.

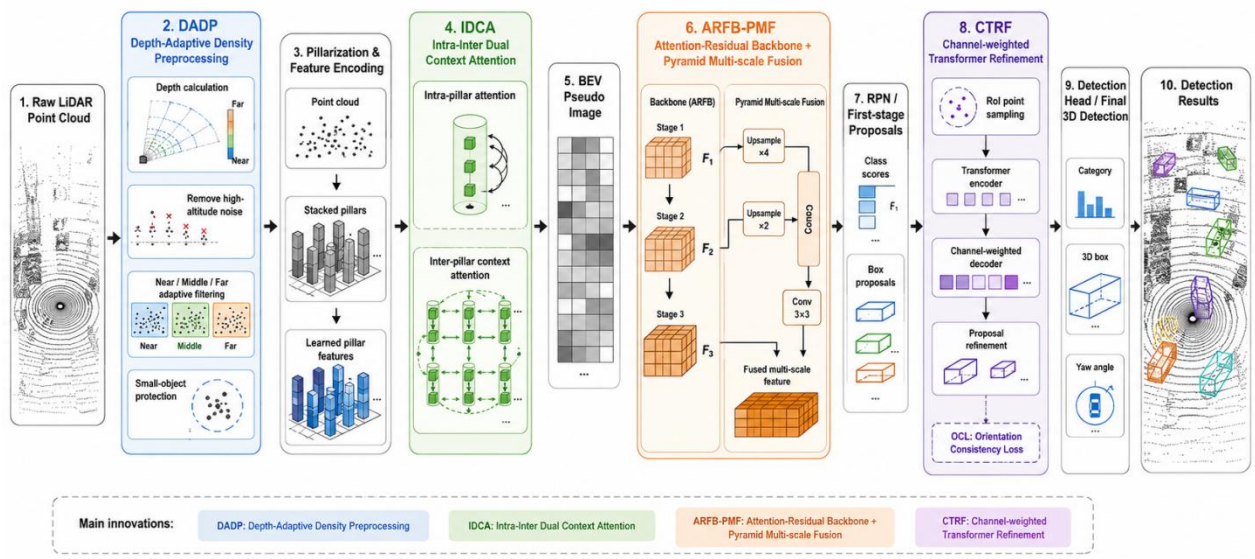

Figure 1. Overall architecture of the proposed HAT-PointPillars**.**

Here, P denotes the raw LiDAR point cloud, and X_out denotes the final detection output, including category, 3D location, object size and yaw angle. The framework does not perform explicit V2X feature-level fusion; instead, it is evaluated on both vehicle-side and infrastructure-side LiDAR data to examine applicability across the two sensing viewpoints.

2.2 Depth-Adaptive Density Preprocessing

Point clouds in road scenes have non-uniform density. Near objects usually contain dense returns, while distant pedestrians and cyclists may contain only a few valid points. A fixed density threshold may therefore remove useful small-object points or retain unnecessary background points. DADP addresses this issue by applying density-based, depth-aware filtering after range filtering and high-altitude background removal [25] .

$$D_i = \sqrt{x_i^2 + y_i^2 + z_i^2} \qquad (1)$$

A point with z_i> 2.0 m is treated as a high-altitude background point and removed before density filtering. The remaining points are divided into three depth regions, and different density parameters are applied:

$$(\varepsilon_i, M_i) = \begin{cases} (0.7,3), D_i < 20\,\text{m} \\ (0.5,2), 20\,\text{m} \le D_i \le 50\,\text{m} \\ (0.3,1), D_i > 50\,\text{m} \end{cases} \qquad (2)$$

The far-region setting uses a smaller neighborhood radius and a relaxed minimum-point threshold to avoid deleting extremely sparse long-range pedestrian and cyclist points. This does not mean that all isolated points are preserved, because the operation is performed after valid-range filtering, high-altitude point removal and local density consistency checking. The small-object protection rule is based only on geometric sparsity and local density; no ground-truth box, class label or annotation-derived signal is used during either training or inference.

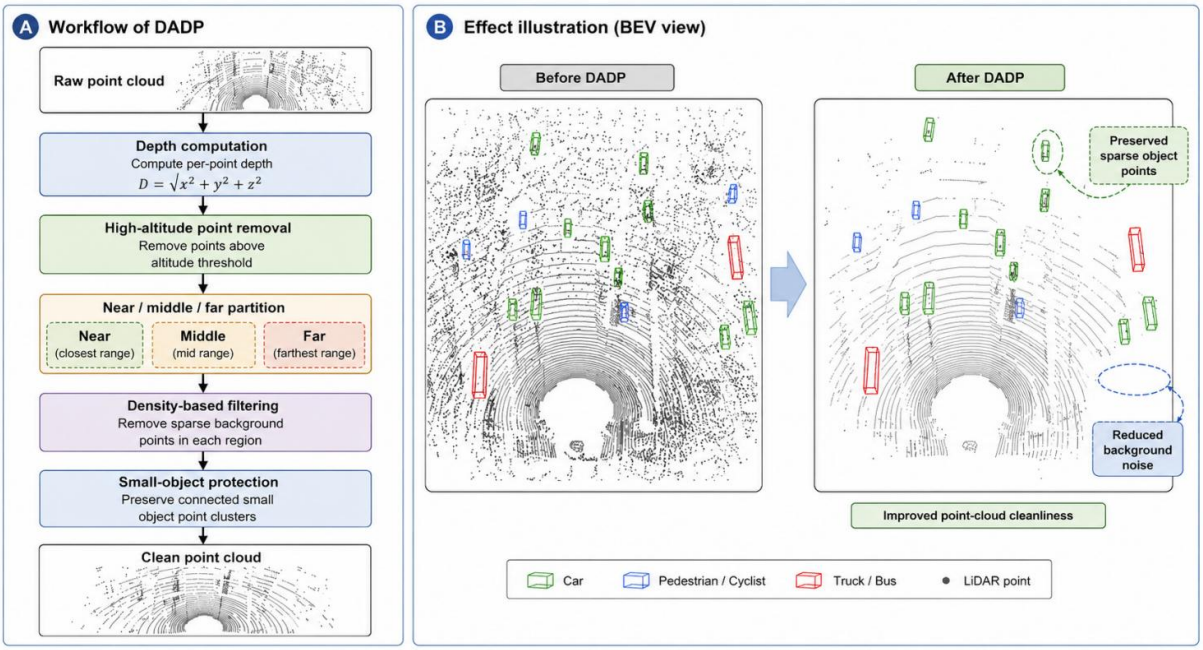

Figure 2. Workflow and schematic effect illustration of the DADP module.

The BEV panels in Figure 2 provide a schematic illustration of the expected filtering behavior of DADP. No independent quantitative filtering rate is claimed from this figure.

Algorithm 1. Depth-Adaptive Density Preprocessing.

Input: raw point cloud P; valid point-cloud range; height threshold z_h = 2.0 m.
1. Remove points outside the valid detection range and remove high-altitude points with $z_i > z_h$.
2. Compute $D_i = \sqrt{x_i^2 + y_i^2 + z_i^2}$ for each remaining point.
3. Assign each point to near, middle or far depth region according to 20 m and 50 m thresholds.
4. Apply region-specific density filtering with (ε_i, M_i) = (0.7,3), (0.5,2) or (0.3,1).
5. Preserve sparse small-object candidate points using geometry-only local density consistency checking.
Output: cleaned point cloud P_{clean} for pillarization.

2.3 Intra-Inter Dual Context Attention

The original PointPillars feature encoder uses PointNet-style aggregation to encode points within each pillar [2,7]. IDCA improves the pillar representation by modeling both point-wise relations within a pillar and contextual dependencies among neighboring pillars. Each non-empty pillar contains at most 32 sampled points. Each point is represented by a 10-dimensional extended feature consisting of raw coordinates, reflectance, cluster offsets and pillar-center offsets. A lightweight single-head attention design is used to control additional computation.

For intra-pillar modeling, Q, K and V are generated from the input pillar feature by learnable linear projections, and the attention output is calculated as:

$$A_{intra} = \text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V \quad (3)$$

For inter-pillar context modeling, the current pillar attends to non-empty neighboring pillars within a 3 x 3 BEV neighborhood. Empty neighboring positions are masked and do not participate in the attention calculation.

$$A_{inter} = \text{Softmax}\left(\frac{Q_c K_n^T}{\sqrt{d}}\right)V_n \quad (4)$$

$$F_{IDCA} = \phi(\text{Concat}(A_{intra}, A_{inter}, F)) \quad (5)$$

Here, Q_c is the query feature of the current pillar, K_n and V_n are features from the 3 x 3 neighboring non-empty pillars, and phi denotes a lightweight projection with normalization and activation. The output is sent to BEV scatter as the enhanced pillar feature.

2.4 Attention-Residual Backbone and Pyramid Multi-scale Fusion

After IDCA, pillar features are scattered into a BEV pseudo-image. ARFB-PMF is designed to strengthen BEV feature extraction while keeping the detector lightweight. ARFB consists of three residual downsampling stages. Each residual attention block includes two 3 x 3 convolution layers, batch normalization, ReLU activation, a residual shortcut and a lightweight attention activation. Residual learning helps stabilize the backbone [14], and attention activation enhances discriminative feature responses [12,13].

$$Y = \sigma(\text{BN}(\text{Conv}_2(\text{BN}(\text{Conv}_1(X)))) + R(X)) \quad (6)$$

$$X_{att} = M(X) \odot X \quad (7)$$

PMF aggregates multi-scale BEV features from different stages. Lower-resolution features are upsampled to a common BEV resolution and concatenated before channel fusion. This design helps represent both large vehicles and small objects.

$$F_{PMF} = \psi(\text{Concat}(F_1, \text{Up}(F_2), \text{Up}(F_3))) \quad (8)$$

2.5 Channel-weighted Transformer Refinement

Single-stage predictions may be inaccurate for small, distant or occluded targets. CTRF is used to refine high-confidence proposals while controlling inference cost. After first-stage NMS, proposals with confidence scores higher than 0.3 are selected for refinement according to the OpenPCDet detection pipeline; if the number of retained proposals exceeds the configured proposal cap, the standard OpenPCDet cap is used. For each proposal region, 128 points are sampled from its cylindrical RoI and embedded into a 128-dimensional proposal-to-point feature representation.

The refinement network contains one transformer encoder layer and one channel-weighted decoder layer. Four attention heads are used in the transformer block. The encoder models proposal-to-point relations, and the decoder reweights encoded proposal features along the channel dimension before box residual prediction.

$$Z' = \text{Transformer}(Z), \quad b_{ref} = b + \Delta b, \quad \Delta b = g(Z') \quad (9)$$

$$w_c = \text{Sigmoid}(\text{MLP}(\text{GAP}(Z'))), \quad Z_c = w_c \odot Z' \quad (10)$$

The refinement head predicts residuals for center location, size and yaw angle. The same confidence threshold, RoI sampling strategy, transformer configuration and NMS settings are used for all compared variants that include CTRF, so the comparison isolates the contribution of the proposed modules rather than different post-processing rules.

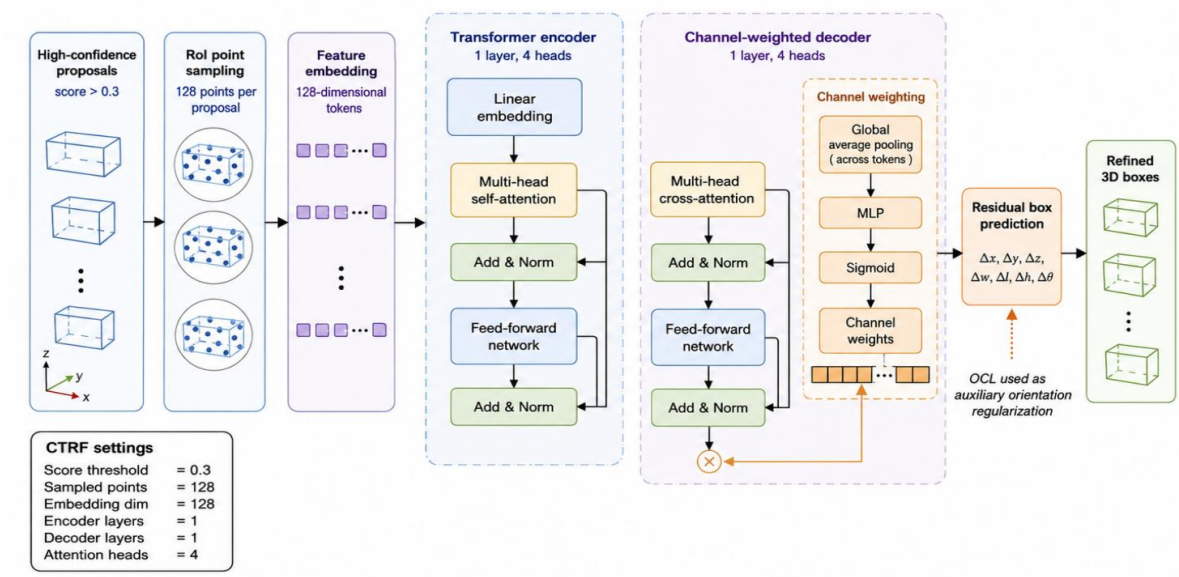

Figure 3. Structure of the Channel-weighted Transformer Refinement (CTRF) module.

2.6 Loss Function and Orientation Regularization

The detector is trained with classification, box regression and confidence losses following the OpenPCDet-based PointPillars framework. An Orientation Consistency Loss (OCL) is used as an auxiliary yaw regularization term in the refinement stage:

$$L_{OCL} = \frac{1-\cos(\theta_p-\theta_g)}{2} \qquad (11)$$

$$L = L_{cls} + L_{reg} + L_{conf} + \lambda L_{OCL} \qquad (12)$$

In Eq. (12), L_{cls} is the classification loss, L_{reg} is the bounding-box regression loss, L_{conf} is the proposal confidence loss, and lambda is the weight of OCL. In this study, lambda is set to 0.5. Since no separate OCL ablation is reported, the independent contribution of OCL is not isolated; it is discussed as part of the complete CTRF refinement design.

2.7 Datasets and Evaluation Protocol

KITTI is used as the vehicle-side benchmark [22]. The 7,481 labeled training samples are divided into 3,712 training samples and 3,769 validation samples following the commonly used train/validation split. Three categories are evaluated: Car, Pedestrian and Cyclist. The detection range is set to [0, 69.12] m x [-39.68, 39.68] m x [-3, 1] m, and the voxel size is [0.16, 0.16, 4]. The IoU threshold is 0.7 for Car and 0.5 for Pedestrian/Cyclist. KITTI results in this paper are reported as 3D AP_R40 at the Moderate difficulty level. Because Easy and Hard results were not recorded in the current experimental log, the KITTI comparison should be interpreted as focused Moderate validation rather than a full official benchmark submission.

DAIR-V2X-I is used as the infrastructure-side benchmark [23]. A total of 7,058 publicly available training and validation frames are used with a fixed 7:3 train/validation split, and the same split is used for all compared methods. Only infrastructure-side LiDAR point clouds are used. To maintain a KITTI-style evaluation pipeline, tricycle, motorcycle and cart-like targets are merged into the Cyclist category, and annotations are converted into KITTI-style format. This customized protocol is used for controlled comparison in this study and should not be directly compared with the official DAIR-V2X leaderboard.

2.8 Implementation Details

All experiments are implemented using an OpenPCDet-based framework [26]. The environment is Linux 5.15.0-139-generic, NVIDIA GeForce RTX 4090 D GPU, x86_64 CPU, 125.6 GB memory, Python 3.8.20 and PyTorch 2.4.1+cu121. The Adam optimizer is used with beta1 = 0.9 and beta2 = 0.999. The batch size is 4, the initial learning rate is 0.00075, the weight decay is 0.01, and the model is trained for 80 epochs. Each pillar contains at most 32 points, and the maximum number of non-empty pillars is 16,000. Standard augmentations include random flipping, rotation and scaling. FPS is measured with batch size 1 on the same RTX 4090 D workstation and includes voxelization, pillar feature encoding, backbone inference, detection head, NMS and CTRF refinement.

III. Results

3.1 KITTI Moderate Validation Results

Table no 2 reports KITTI validation results under the Moderate difficulty setting. All AP values in this table are 3D AP_R40. AOS is reported as an orientation similarity metric and should not be directly compared with 3D AP values. The Transformer-refined PointPillars row denotes a PointPillars variant equipped with transformer-based proposal refinement but without the full DADP-IDCA-ARFB-PMF front-end design.

Table no 2: KITTI validation comparison under Moderate difficulty (3D AP_R40, %).

Method	Car 3D AP	Ped. 3D AP	Cyc. 3D AP	Car AOS	Cyc. AOS	FPS
PointPillars	71.84	49.55	55.47	88.67	58.70	48.08
DADP-IDCA PP	75.32	50.21	56.87	90.73	65.23	28.50
Transformer-refined PointPillars	79.35	54.28	71.93	89.12	61.45	28.74
HAT-PointPillars	79.87	54.92	72.56	90.89	65.56	25.60

HAT-PointPillars obtains the highest Moderate 3D AP for all three categories in this table. Compared with PointPillars, Car AP increases from 71.84% to 79.87%, Pedestrian AP increases from 49.55% to 54.92%, and Cyclist AP increases from 55.47% to 72.56%. The result suggests that hierarchical attention and proposal refinement are helpful for sparse and small objects under the reported Moderate validation setting.

3.2 DAIR-V2X-I Validation Results

Table no 3 reports the infrastructure-side DAIR-V2X-I results under the customized KITTI-style protocol. Moderate mAP is calculated as the average of Car, Pedestrian and Cyclist AP.

Table no 3: DAIR-V2X-I validation comparison under the customized KITTI-style protocol (%).

Method	Car AP	Ped. AP	Cyc. AP	mAP	FPS
PointPillars	78.51	57.99	60.59	65.70	30.67
DADP-IDCA PP	79.65	58.76	61.93	66.78	27.30
Transformer-refined PointPillars	80.41	68.49	62.70	70.53	14.60
HAT-PointPillars	80.89	69.15	63.21	71.08	25.20

HAT-PointPillars obtains 71.08% mAP on DAIR-V2X-I, improving the PointPillars baseline by 5.38 percentage points. Compared with the Transformer-refined PointPillars baseline, it achieves slightly higher mAP while increasing FPS from 14.60 to 25.20. This indicates a better speed-accuracy balance under the reported infrastructure-side validation protocol.

3.3 Ablation Study

Table no 4 reports module-level ablation on DAIR-V2X-I. OCL is included in the complete CTRF setting, and no separate OCL-only ablation is reported in this version.

Table no 4: Ablation study on DAIR-V2X-I under Moderate difficulty (%).

DADP	IDCA	ARFB-PMF	CTRF	Car	Ped.	Cyc.	mAP	FPS
x	x	x	x	78.51	57.99	60.59	65.70	30.67
✓	x	x	x	79.23	58.65	61.21	66.36	29.80
x	✓	x	x	79.65	59.32	61.87	66.95	28.50
x	x	✓	x	78.69	59.46	62.12	66.76	29.20
x	x	x	✓	80.35	67.13	61.35	69.61	14.60
✓	✓	x	x	79.65	58.76	61.93	66.78	27.30
✓	✓	✓	x	80.12	60.78	62.85	67.92	27.30
✓	✓	x	✓	80.67	68.56	62.43	70.55	26.10
✓	x	✓	✓	80.59	68.72	62.28	70.53	26.50
✓	✓	✓	✓	80.89	69.15	63.21	71.08	25.20

DADP, IDCA and ARFB-PMF provide moderate gains with limited speed loss, while CTRF provides a larger AP improvement but increases computation. The complete HAT-PointPillars obtains the best mAP in this ablation table while maintaining above 25 FPS. These results support the complementarity of the proposed components under the reported setting.

3.4 Qualitative Visualization

To complement the numerical comparison, representative qualitative examples are shown in Figure 4 and Figure 5. Because ground-truth boxes are not overlaid in these visualization panels, the figures are used as qualitative baseline-versus-HAT illustrations rather than formal visual accuracy evaluation.

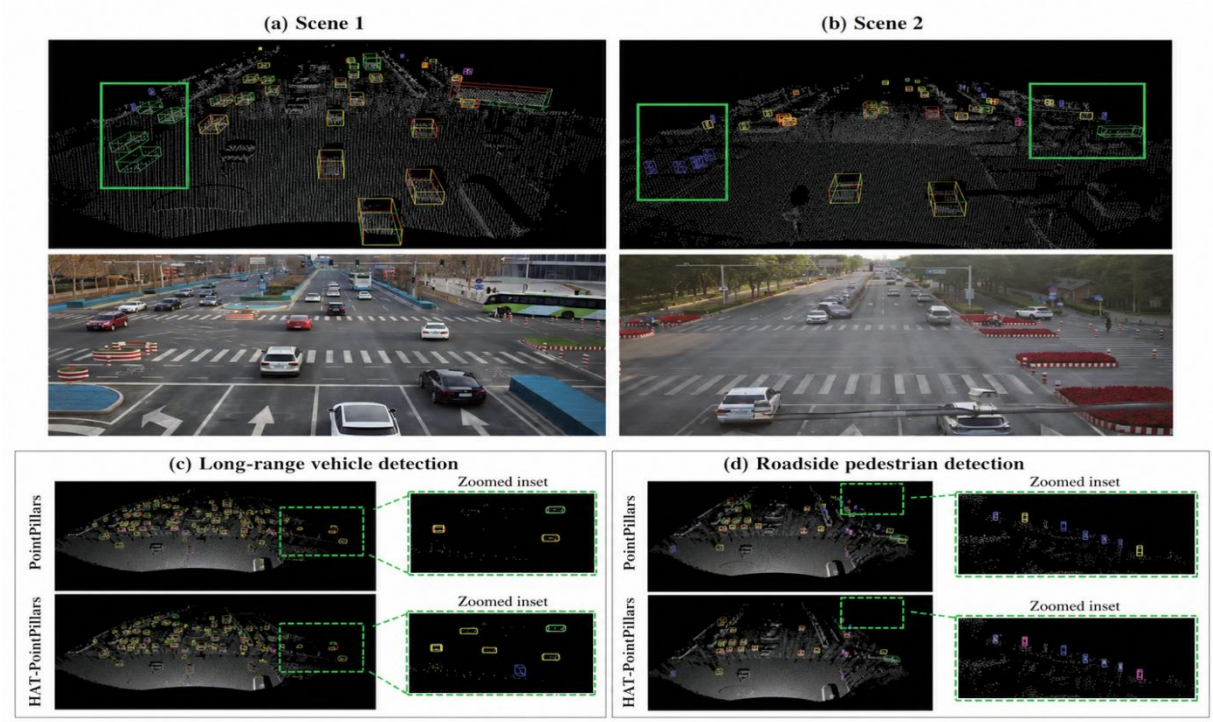

Figure 4. Qualitative detection examples on DAIR-V2X-I roadside scenes. Colored boxes indicate predicted 3D bounding boxes; dashed and enlarged boxes indicate highlighted regions. No ground-truth boxes are overlaid.

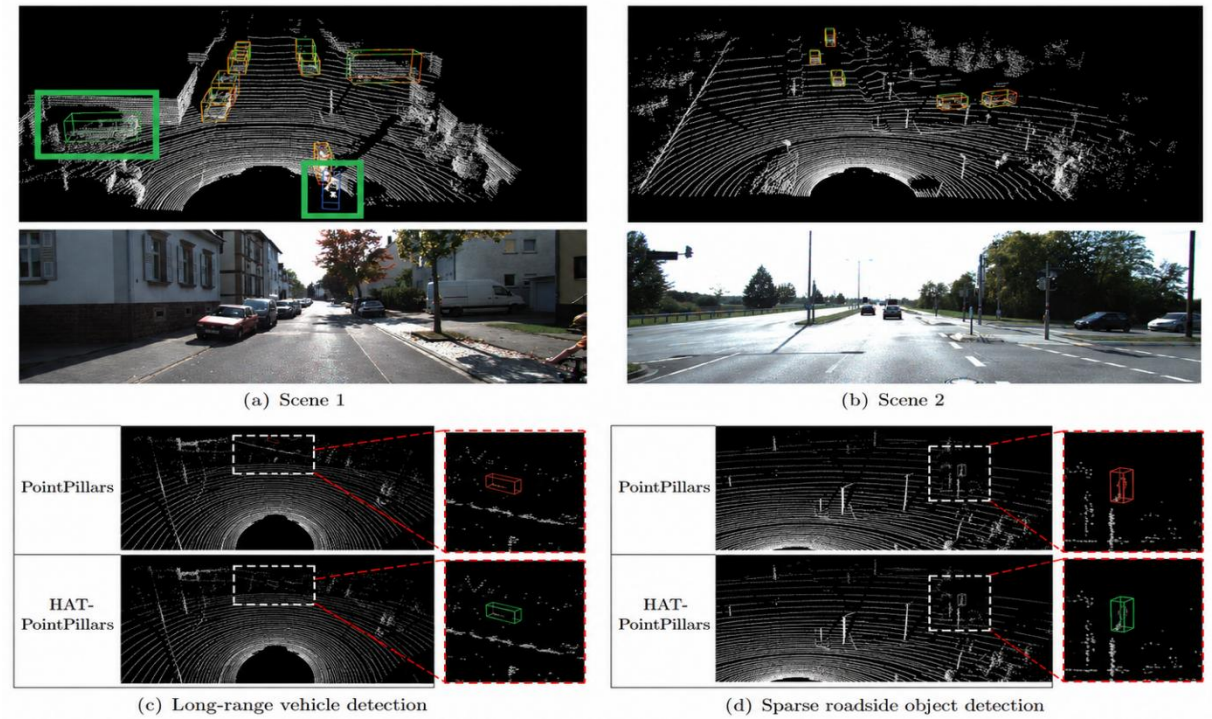

Figure 5. Qualitative detection examples on KITTI vehicle-side scenes. Colored boxes indicate predicted 3D bounding boxes; dashed and enlarged boxes provide visual context for sparse or distant-object regions. No ground-truth boxes are overlaid. Camera images are shown only for scene context and are not used as model input.

Figure 4 provides qualitative examples of detected boxes in DAIR-V2X-I roadside scenes. The examples include roadside scenes, long-range vehicle regions and sparse pedestrian regions. The highlighted areas are used only to guide visual inspection of difficult regions; no ground-truth-based visual accuracy claim is made from this figure. Figure 5 provides qualitative examples from KITTI vehicle-side scenes. The examples provide visual context for the reported Moderate AP results, especially in sparse or distant-object cases. Since

ground-truth boxes are not overlaid, the figure should be interpreted as qualitative illustration rather than formal visual accuracy evaluation.

3.5 Accuracy and Inference-Speed Analysis

Figure 6 summarizes the accuracy and inference-speed comparison on DAIR-V2X-I. PointPillars is the fastest baseline but has lower mAP. Transformer-refined PointPillars obtains higher accuracy but lower speed. HAT-PointPillars achieves the best mAP while maintaining real-time inference speed under the reported validation protocol.

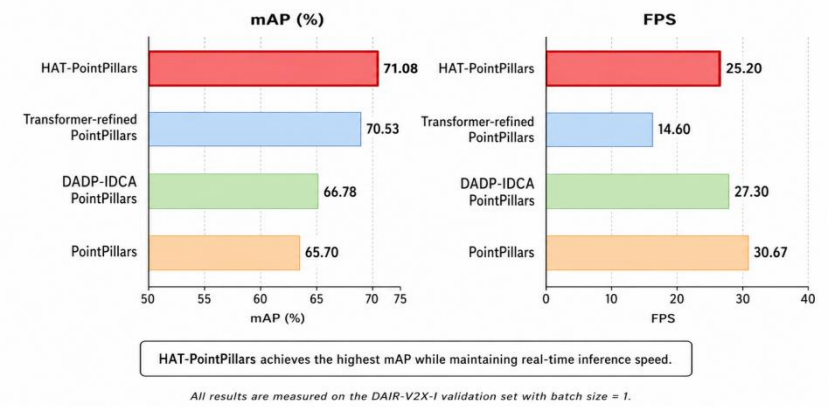

Figure 6. Accuracy and inference-speed comparison on the DAIR-V2X-I validation set.

IV. Discussion

The results indicate that HAT-PointPillars improves detection accuracy under the reported validation protocol. DADP reduces background interference while preserving sparse small-object points. IDCA improves the pillar encoder by jointly modeling intra-pillar point relations and inter-pillar context. ARFB-PMF strengthens multi-scale BEV features, and CTRF refines high-confidence proposals with channel-weighted transformer decoding.

The speed behavior of the transformer-refined variant deserves clarification. The CTRF-only variant is slower because transformer refinement is applied after the original PointPillars branch, which may generate more low-quality or redundant proposals for refinement. In the complete HAT pipeline, DADP, IDCA and ARFB-PMF improve point-cloud quality and first-stage feature representation before refinement, which can reduce the effective refinement burden under the same confidence threshold. This may partly explain why the complete model can be faster than the simple transformer-refined variant while still achieving higher mAP. Because proposal counts and per-module latency logs were not recorded as original experimental outputs, this explanation is given as a mechanism-level interpretation rather than a separate latency-breakdown experiment.

V. Conclusion

The method is designed for both vehicle-side and infrastructure-side LiDAR scenes, but it does not perform explicit vehicle-infrastructure feature fusion. Therefore, the title and claims are limited to vehicle-side and infrastructure-side LiDAR 3D detection rather than full V2X cooperative detection. This distinction is important because cooperative detection usually requires cross-sensor calibration, temporal synchronization and feature- or object-level fusion between vehicle and infrastructure sensors.

There are still limitations. First, KITTI results are reported only at the Moderate difficulty level because Easy and Hard results were not recorded in the current experimental log. Second, DAIR-V2X-I uses a customized fixed 7:3 split and category mapping, so it should not be compared directly with official leaderboard results. Third, the numerical comparison focuses on PointPillars-based variants under the same OpenPCDet-style setting; stronger external detectors are discussed in the related work but are not rerun in this study. Fourth, no separate OCL ablation, repeated-run standard deviation or parameter-sensitivity experiment is reported, so the independent effects of OCL and some hyperparameters remain to be verified. Fifth, the current method uses only LiDAR point clouds and does not exploit camera semantic information. Future work should include full difficulty-level reporting, repeated runs with statistical variance, parameter sensitivity analysis, stronger external baselines and multimodal or cooperative fusion experiments.

References

[1] Shi S, Wang X, Li H. PointRCNN: 3D object proposal generation and detection from point cloud. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2019 Jun 15-20; Long Beach, CA, USA. p. 770-779.

[2] Qi CR, Su H, Mo K, Guibas LJ. PointNet: deep learning on point sets for 3D classification and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2017. p. 652-660.

[3] Qi CR, Yi L, Su H, Guibas LJ. PointNet++: deep hierarchical feature learning on point sets in a metric space. In: Advances in Neural Information Processing Systems; 2017. p. 5099-5108.

[4] 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; 2018 Jun 18-23; Salt Lake City, UT, USA. p. 4490-4499.

[5] Yan Y, Mao Y, Li B. SECOND: sparsely embedded convolutional detection. Sensors (Basel). 2018;18(10):3337.

[6] Shi S, Guo C, Jiang L, Wang Z, Shi J, Wang X, Li H. PV-RCNN: point-voxel feature set abstraction for 3D object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2020 Jun 13-19; Seattle, WA, USA. p. 10526-10535.

[7] Lang AH, Vora S, Caesar H, Zhou L, Yang J, Beijbom O. PointPillars: fast encoders for object detection from point clouds. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2019 Jun 15-20; Long Beach, CA, USA. p. 12697-12705.

[8] Deng J, Shi S, Li P, Zhou W, Zhang Y, Li H. Voxel R-CNN: towards high performance voxel-based 3D object detection. In: Proceedings of the AAAI Conference on Artificial Intelligence; 2021. p. 1201-1209.

[9] 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; 2022. p. 35-52.

[10] Shi W, Rajkumar R. Point-GNN: graph neural network for 3D object detection in a point cloud. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2020. p. 1711-1719.

[11] 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; 2021. p. 11784-11793.

[12] Yang L, Zhang RY, Li L, Xie X. SimAM: a simple, parameter-free attention module for convolutional neural networks. In: Proceedings of the 38th International Conference on Machine Learning; 2021. p. 11863-11874.

[13] Hu J, Shen L, Sun G. Squeeze-and-excitation networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2018. p. 7132-7141.

[14] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2016. p. 770-778.

[15] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. In: Advances in Neural Information Processing Systems; 2017. p. 5998-6008.

[16] Lin TY, Goyal P, Girshick R, He K, Dollar P. Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision; 2017. p. 2980-2988.

[17] Chen X, Ma H, Wan J, Li B, Xia T. Multi-view 3D object detection network for autonomous driving. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2017. p. 1907-1915.

[18] Sindagi VA, Zhou Y, Tuzel O. MVX-Net: multimodal VoxelNet for 3D object detection. In: Proceedings of the IEEE International Conference on Robotics and Automation; 2019. p. 7276-7282.

[19] Vora S, Lang AH, Helou B, Beijbom O. PointPainting: sequential fusion for 3D object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2020. p. 4604-4612.

[20] Huang T, Liu Z, Chen X, Bai X. EPNet: enhancing point features with image semantics for 3D object detection. In: Proceedings of the European Conference on Computer Vision; 2020. p. 35-52.

[21] Pang S, Morris D, Radha H. CLOCs: camera-LiDAR object candidates fusion for 3D object detection. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems; 2020. p. 10386-10393.

[22] Geiger A, Lenz P, Stiller C, Urtasun R. Vision meets robotics: the KITTI dataset. Int J Robot Res. 2013;32(11):1231-1237.

[23] Yu H, Luo Y, Shu M, Huo Y, Yang Z, Shi Y, et al. DAIR-V2X: a large-scale dataset for vehicle-infrastructure cooperative 3D object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2022. p. 21361-21370.

[24] Caesar H, Bankiti V, Lang AH, Vora S, Liong VE, Xu Q, et al. nuScenes: a multimodal dataset for autonomous driving. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2020. p. 11621-11631.

[25] Ester M, Kriegel HP, Sander J, Xu X. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining; 1996. p. 226-231.

[26] OpenPCDet Development Team. OpenPCDet: an open-source toolbox for LiDAR-based 3D object detection. Available from: https://github.com/open-mmlab/OpenPCDet. Accessed 2026 Jun 15.