FusionNeXt: Sequence-First 3D Multi-Modal Fusion in the Era of LLMs

Yu Hong[1,2] **Songbur Wong**[1] **Yanhao Liu**[1,2] **Wenlong Liao**[4]
Tao He[4] **Xiaosong Jia**[3,✉] **Junchi Yan**[1,2,✉]
[1]Shanghai Jiao Tong University [2]Shanghai Innovation Institute
[3]Fudan University [4]COWAROBOT [✉]Corresponding authors

Abstract

Modern LLMs/VLMs have largely converged to a unified architecture: representing heterogeneous inputs as a single token sequence and processing it with a deep stack of generic operators such as attention and feed-forward layers. This sequence-first interface simplifies system designs and thus enables extensive algorithmic and hardware optimization, making it a compelling blueprint for modern model development. In contrast, many 3D multi-modal fusion models still rely on complex and specialized designs such as dense feature representations, sparse 3D convolutions, deformable attention, and view transformations. In this paper, we ask: **Can 3D multi-modal fusion embrace the same design path as modern LLMs/VLMs and benefit from their mature and rapidly evolving ecosystem?** We answer this affirmatively and propose **FusionNeXt**, a modern multi-modal fusion paradigm that (i) unifies camera and LiDAR features into a shared token representation, (ii) serializes tokens into packed, locality-preserving 1D sequences, and (iii) performs feature fusion using a deep stack of vanilla LLM/VLM-style blocks—FlashAttention, pre-norm residuals, SwiGLU FFNs, etc. FusionNeXt achieves 75.9 test NDS on nuScenes and 56.62 mIoU on Occ3D-nuScenes with fast inference, with its Tiny variant reaching 73.5 val NDS at 18.9 FPS on a single RTX 4090. Under matched attention, its packed sequence interface further yields a 1.16–1.35× backbone speedup over prior attention-based fusion methods. SSM substitution further demonstrates the flexibility of the sequence-first interface. Together, these results suggest a scalable direction for 3D perception. We will open-source our code.

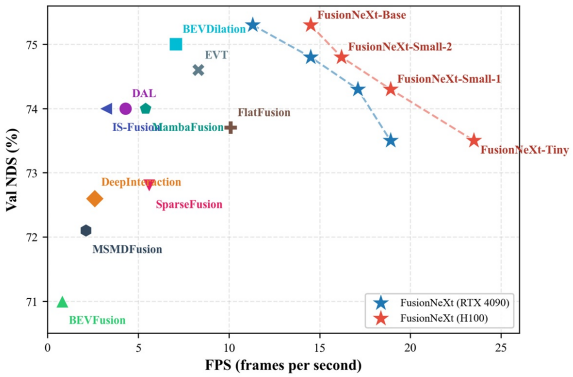


Figure 1: **Val NDS vs. FPS on nuScenes.** FusionNeXt achieves leading accuracy among multi-modal fusion methods without test-time augmentation or ensembling, while delivering high inference throughput. Blue and red FusionNeXt points use RTX 4090 and H100, respectively.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

1 Introduction

Modern large language models (LLMs) and vision–language models (VLMs) have recently converged to a remarkably simple, unified, and scalable architectural paradigm [62, 4, 15, 51, 1, 34, 22, 8]. At the backbone level, they map heterogeneous inputs into a *common token interface* and then process the resulting 1D token sequence with a deep stack of generic operators like attention and feed-forward layers. This "tokens-in, generic blocks, tokens-out" recipe has proven powerful not only for accuracy, but also for efficiency: the reuse of standardized operators simplifies system design, improves hardware utilization, and enables scaling with model size, data, and compute [14, 54, 3, 40, 11, 53].

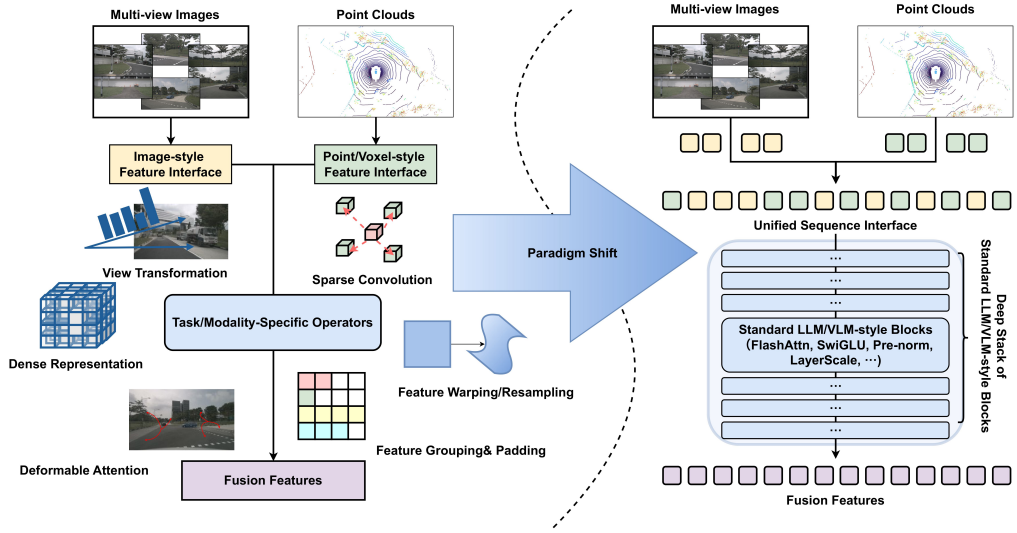


Figure 2: **From traditional fusion to sequence-first fusion.** Traditional 3D multi-modal fusion systems typically couple heterogeneous representations (dense image grids, sparse point/voxel sets, and/or dense BEV grids) via specialized alignment and fusion operators inside the backbone. In contrast, our sequence-first design first unifies heterogeneous modalities into a shared sequence interface $\mathbf{X} \in \mathbb{R}^{N \times C}$, and then devotes the majority of computation to a deep stack of reusable sequence blocks (attention/FFN), improving operator reuse and systems portability.

In contrast, 3D multi-modal fusion for autonomous driving has not followed this convergence yet. Despite substantial progress and strong empirical performance, existing camera-LiDAR fusion systems are still largely built around specialized representations and handcrafted operator choices, such as sparse 3D convolutions [69], deformable attention variants [81], view transformations [49], and explicit warping/resampling [37, 74]. This specialization makes it difficult for fusion models to reuse the highly optimized operators and modules developed by the LLM/VLM community.

This contrast raises a fundamental question: **Can 3D multi-modal fusion embrace the same design path as modern LLMs/VLMs and benefit from their mature and rapidly evolving ecosystem?** We answer this question affirmatively. As in Figure 2, rather than designing ever more intricate fusion-specific operators, we take a different route: we build a principled bridge between traditional multi-modal fusion and the modern LLM/VLM paradigm, and reformulate multi-modal fusion as standardized sequence modeling.

In this paper, we propose **FusionNeXt, a sequence-first multi-modal fusion paradigm** for autonomous driving. FusionNeXt modernizes multi-modal fusion by adopting the same blueprint used in LLMs/VLMs: (i) unify camera and LiDAR features into a shared token representation, (ii) serialize sparse multi-modal tokens into packed, locality-preserving 1D sequences, and (iii) perform multi-modal fusion using a deep stack of vanilla LLM/VLM-style blocks (*e.g.*, FlashAttention, pre-norm residuals, and SwiGLU FFNs). As in Figure 1, this paradigm shift yields both strong accuracy and high efficiency: FusionNeXt reaches 75.9 NDS on the nuScenes test set, and its Tiny variant attains 73.5 val NDS at 18.9 FPS on a single RTX 4090.

Our contributions are summarized as follows:

- We present FusionNeXt, a modern 3D multi-modal fusion paradigm that reshapes 3D multi-modal fusion with the sequence-first principle of LLMs/VLMs.
- We revisit the paradigm gap between traditional camera-LiDAR fusion and modern LLM/VLM stacks, and propose a principled roadmap to reformulate 3D multi-modal fusion into standardized sequence modeling to directly benefit from modern LLM/VLM infrastructure. Attention/SSM substitution demonstrates that the interface is compatible with different sequence operators.
- FusionNeXt achieves 75.9 test NDS on nuScenes and 56.62 mIoU on Occ3D-nuScenes with fast inference, suggesting a scalable direction for 3D multi-modal fusion.

2 Related Work

2.1 Sequence-First Stacks in Modern LLMs and VLMs

Modern large language models (LLMs) are characterized by a *unified, sequence-first* computation pattern: inputs are mapped into tokens, and a backbone processes a 1D token sequence using a deep stack of largely identical blocks [62, 4, 15, 14, 54]. In practice, their scalability and engineering success comes from repeatedly composing a small set of general-purpose, hardware-friendly primitives—pre-normalized residual blocks, efficient attention kernels (*e.g.*, FlashAttention [14]), and feed-forward layers (*e.g.*, SwiGLU [54]). VLMs extend the same recipe by converting visual inputs into tokens and coupling them with language tokens via lightweight bridging modules or joint sequence processing [51, 1, 34, 30, 13, 5]. This "everything is a sequence" perspective has also been explored in vision settings with minimal task-specific inductive bias [17]. This sequence-first abstraction has also been explored beyond language, *e.g.*, unifying multiple vision tasks under a shared token-sequence interface [8] and scaling masked-token pretraining for vision [22]. These developments suggest a general design principle: a unified tokenization interface plus a small set of standard sequence operators offers a route to efficient, scalable models that are easy to reuse across tasks. FusionNeXt follows this sequence-first principle for 3D multi-modal fusion by serializing multi-modal tokens into sequences and performing fusion with native sequence modules, avoiding fusion-specific handcrafted operators.

2.2 3D Multi-Modal Fusion in Autonomous Driving

Autonomous driving perception benefits from combining cameras and LiDAR, where cameras provide dense semantics while LiDAR provides accurate geometry [18, 60, 6]. Existing multi-modal 3D detectors can be roughly categorized by *when* and *where* fusion is performed. *Shallow fusion* integrates modalities with limited coupling, including *early fusion* and *late fusion*. Early fusion typically establishes explicit geometric correspondence (*e.g.*, projecting LiDAR to image or attaching image cues to points), as in MV3D [9], Frustum PointNets [50], PointPainting [63], PointAugmenting [64], and MVP [74]. Late fusion merges modality-specific outputs at the decision stage, *e.g.*, via candidate matching and score refinement as in CLOCs [48].

In contrast, *deep fusion* learns cross-modal interaction inside the network. BEV-based representations unify different modalities in a metric ground-plane space, making fusion easier after modality-specific encoders; BEVFusion [42] and BEVFormer [38, 70] highlight the power of BEV as a unified representation with attention-based aggregation across multi-view cameras [76, 57, 59]. Lift-splat style view transformation and BEV lifting are also widely adopted to connect image features to BEV for downstream fusion [49, 52, 26, 20]. Transformer-style architectures enable flexible association via attention [7, 81, 78, 32, 33, 28]. Object-query decoders refine a set of object hypotheses by attending to modality features, as in TransFusion [2] and FUTR3D [10]; token-level interaction further improves robustness and efficiency, *e.g.*, CMT [68], SparseFusion [67], MSMDFusion [29], FusionFormer [25], AutoAlignV2 [12], and GAFusion [35]. Meanwhile, recent methods [35, 12, 57, 59] explore stronger modality interaction and unified representations for fusion, such as DeepFusion [37], DeepInteraction [71], UVTR [36], and IS-Fusion [73]. Despite strong progress, existing fusion systems often rely on specialized designs (*e.g.*, sparse 3D convolutions, deformable attention, view transforms), which can complicate scaling and deployment [69, 81, 49, 52, 26, 42, 38, 20]. More recently, several methods have begun to explore serialization or sorting of sparse features for more efficient processing [43, 82, 65, 41, 66]. However, in these approaches serialization still serves mainly as a preprocessing step for subsequent custom operators—sorted tokens are immediately consumed by spatial grouping, padded windowing, or bespoke attention variants that

Table 1: High-level paradigm comparison between modern LLM/VLM stacks and traditional 3D multi-modal fusion pipelines.

Aspect	LLMs/VLMs (sequence-first)	3D multi-modal fusion (traditional)
Interface	Unified token matrix $\mathbf{X} \in \mathbb{R}^{N \times C}$	Multi-view images $\mathcal{I}$ + 3D points/voxels/BEV features $\mathcal{V}$
Alignment	Primarily at the input interface	Repeated inside the backbone
Operators	Reusable blocks (attn/FFN)	Specialized fusion/alignment ops
Blueprint	Repeated homogeneous blocks	Heterogeneous, stage-specific ops

are still tailored to the 3D fusion setting. By contrast, FusionNeXt treats serialization as the actual backbone interface: the ordered token sequence is fed directly into unmodified LLM/VLM-style blocks that perform both contextual encoding and cross-modal interaction.

2.3 Discussion: What Sets FusionNeXt Apart?

Despite their diversity, existing 3D fusion methods largely rely on fusion-specific operators and representations that are designed and optimized in isolation from the broader LLM/VLM ecosystem. FusionNeXt departs from this convention in two fundamental ways: **(i) Paradigm shift to sequence-first fusion.** FusionNeXt adopts the sequence-first principle of modern LLMs/VLMs as the core design philosophy: heterogeneous camera and LiDAR features are unified into a single token sequence *before* entering the backbone, and all subsequent feature encoding and cross-modal interaction are performed by a deep stack of generic, reusable sequence blocks. **(ii) Native LLM/VLM operators, no fusion-specific attention kernel.** The fusion backbone operator stack uses off-the-shelf operators from the LLM/VLM recipe—FlashAttention with sliding windows, pre-norm residuals, SwiGLU FFNs—without fusion-specific attention kernels. **The main contribution of FusionNeXt is the bridge** between 3D multi-modal fusion and the sequence-first paradigm. By reformulating camera-LiDAR fusion as geometry-aware token sequences, FusionNeXt enables standard LLM/VLM operators to be used directly, without fusion-specific redesign. This is the key boundary with prior serialization-based 3D fusion methods: they still rely on 3D-fusion-specific operator stacks after sorting, whereas FusionNeXt keeps the operator stack generic and moves the task adaptation into the token interface itself.

3 Method

3.1 Revisiting the Paradigm of Modern LLMs/VLMs and 3D Multi-Modal Fusion

Modern LLMs/VLMs follow a *sequence-first* blueprint: map heterogeneous inputs into a unified token matrix and process it with reusable sequence blocks. Formally,

$$\mathbf{X}_0 = \mathcal{T}(\text{inputs}), \tag{1}$$

$$\mathbf{X}_{\ell+1} = \mathbf{X}_\ell + \mathcal{F}_\ell(\mathbf{X}_\ell), \qquad \ell = 0, \ldots, L-1, \tag{2}$$

where $\mathcal{T}(\cdot)$ tokenizes inputs and $\mathcal{F}_\ell(\cdot)$ is a reusable operator such as attention or FFN. In contrast, 3D multi-modal fusion reconciles dense 2D images and sparse 3D points under different coordinate systems, which has led to specialized representations and operators such as BEV warping, deformable attention, and grouped spatial attention [42, 10, 65]. Table 1 summarizes this dominant paradigm gap. This motivates our goal: reformulate 3D multi-modal fusion as a sequence-first pipeline with a unified token interface, so that multi-modal fusion can be realized by LLM/VLM-style sequence blocks.

3.2 From Traditional Fusion to Sequence-First Fusion

Based on the above paradigm gap, we propose a roadmap to reformulate 3D multi-modal fusion into a sequence-first pipeline. The core lies in how to transform heterogeneous inputs—*e.g.*, multi-view images $\mathcal{I} \in \mathbb{R}^{K \times H \times W \times 3}$ and LiDAR point sets $\mathcal{P} \in \mathbb{R}^{N_{\text{pts}} \times 3}$ into a unified token matrix $\mathbf{X} \in \mathbb{R}^{N \times C}$. This involves two key steps: (i) *multi-modal token generation*, which tokenizes each modality and projects them into a shared feature space; and (ii) *geometry-aware token serialization*, which converts image and LiDAR tokens into a 1D variable-length sequence while preserving spatial locality. After this, we are able to apply *unified feature encoding and fusion* using standard sequence operators

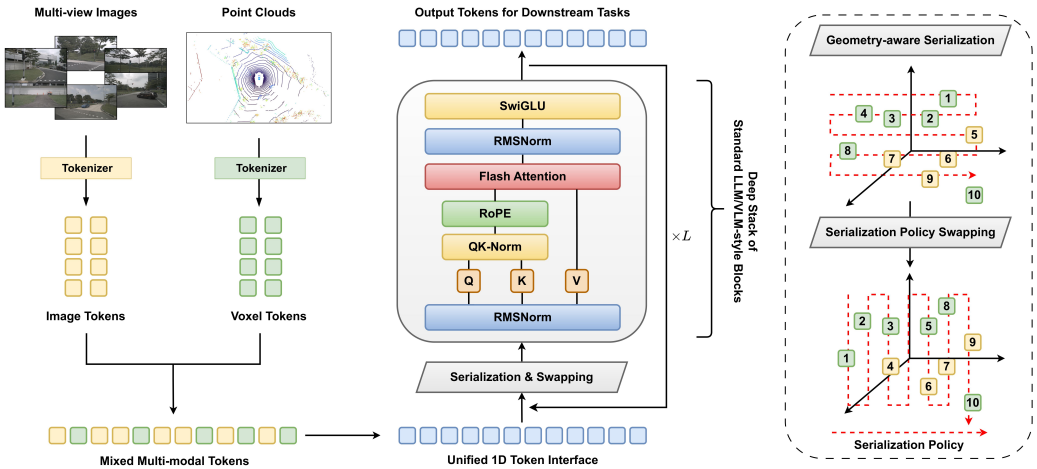


Figure 3: **FusionNeXt framework overview.** FusionNeXt builds a sequence-first multi-modal fusion pipeline: (1) *Multi-modal token generation* extracts tokens from LiDAR points and multi-view images and projects them into a shared feature space; (2) *Geometry-aware token serialization* computes coordinate-derived permutation indices and transforms tokens into locality-preserving 1D sequences with alternated ordering policies; (3) *Vanilla LLM/VLM blocks* (attention/FFN-style) process the serialized sequences for contextual encoding and cross-modal interaction.

(*e.g.*, attention/FFN blocks) on the serialized token sequences. Figure 3 provides an overview of this end-to-end framework, including token generation, geometry-aware serialization, and sequence-style fusion blocks.

3.3 Multi-Modal Token Generation

Given synchronized multi-view images $\mathcal{I} = \{I^{(k)}\}_{k=1}^{K}$ ($I^{(k)} \in \mathbb{R}^{H\times W\times 3}$) and a LiDAR sweep $\mathcal{P} = \{\mathbf{p}_j\}_{j=1}^{N_{\text{pts}}}$ ($\mathbf{p}_j \in \mathbb{R}^3$), lightweight modality-specific encoders produce sparse tokens with their grid coordinates:

$$(\mathbf{V}, \mathbf{C}^v) = f_{\text{lidar}}(\mathcal{P}), \qquad (\mathbf{T}, \mathbf{C}^t) = f_{\text{img}}(\mathcal{I}), \tag{3}$$

where $\mathbf{V} \in \mathbb{R}^{N_v\times C_v}$ with voxel coordinates $\mathbf{C}^v \in \mathbb{Z}^{N_v\times 3}$ ($[x, y, z]$), and $\mathbf{T} \in \mathbb{R}^{N_t\times C_t}$ with image-plane coordinates $\mathbf{C}^t \in \mathbb{Z}^{N_t\times 3}$ ($[k, u, v]$ for view index and patch). We then project both modalities to a shared channel dimension C and concatenate:

$$\mathbf{X}_0 = [\mathbf{V}\mathbf{W}_v; \mathbf{T}\mathbf{W}_t] \in \mathbb{R}^{(N_v+N_t)\times C}, \qquad \mathbf{C}_0 = [\mathbf{C}^v; \mathbf{C}^t], \tag{4}$$

where $\mathbf{W}_v \in \mathbb{R}^{C_v\times C}$ and $\mathbf{W}_t \in \mathbb{R}^{C_t\times C}$ are linear projections. This yields a single sparse multi-modal token set with unified channels and their discrete coordinates, which will be reordered into local 1D sequences by our serialization module (Section 3.4).

3.4 Geometry-Aware Token Serialization

The key to make multi-modal fusion compatible with LLM/VLM-style backbones is to convert camera and LiDAR tokens into a *1D variable-length sequence* using only coordinate-derived indices. Instead of densifying grids or explicitly warping features, FusionNeXt computes permutation indices that order multi-modal tokens according to their 2D/3D coordinates into 1D sequence.

Why ordering matters. For natural language and other sequential inputs, tokens come with an inherent order and strong locality: nearby tokens in the 1D sequence are typically semantically coherent. This property makes *local* sequence operators (*e.g.*, sliding-window attention with official FlashAttention support) highly effective and efficient, avoiding the need for full global attention. In contrast, 3D multi-modal fusion does not provide such a canonical "language-like" ordering—raw LiDAR/image tokens are fundamentally spatial rather than sequential, and arbitrary 1D layouts would break locality. Therefore, FusionNeXt explicitly *injects a spatial order* via geometry-aware serialization, so that adjacent tokens in the serialized sequence are also nearby in 2D/3D space, enabling efficient local attention while preserving meaningful neighborhood interactions.

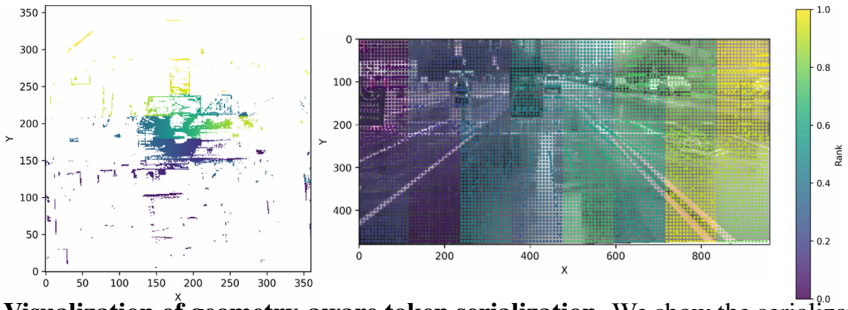


Figure 4: **Visualization of geometry-aware token serialization.** We show the serialization order in voxel space (left) and image space (right). Color gradient indicates the position in the serialized sequence.

Coordinate sorting with a spatial scan prior. Given continuous token coordinates, we construct compact sorting keys (*e.g.*, by quantizing coordinates into coarse bins and applying a serpentine or hilbert scan within each bin) and sort tokens by these keys. This introduces a locality prior—tokens that are close in 2D/3D space become adjacent in the resulting 1D sequence—while keeping the representation sparse and avoiding any feature resampling. Specifically, we use the serpentine sorting as that in [43]. We vary the scan pattern across layers to broaden geometric neighborhood coverage and mitigate directional bias.

Alternating image-plane (2D) and voxel-space (3D) ordering. In 3D multi-modal fusion, LiDAR points and image patches naturally live in heterogeneous coordinate systems (3D metric space vs. 2D image plane). Accordingly, we alternate between two coordinate-based ordering policies to define local neighborhoods for serialization: (i) *Image-plane ordering (2D):* we associate each token with an image-plane coordinate. Image patch tokens use their native 2D grid positions, while voxel tokens obtain 2D coordinates by projecting the voxel center $\mathbf{x} \in \mathbb{R}^3$ to pixel coordinates:

$$(u, v) = \pi\big(\mathbf{x}; \mathbf{K}_k, \mathbf{T}_k\big). \tag{5}$$

If a point is visible in multiple camera views, we select one view index k to define its 2D ordering coordinate. These coordinates are used only to compute sorting keys and attention window indices; the exact depth construction for voxel-space ordering and the multi-view tie-breaking rule are given in Appendix A. (ii) *Voxel-space ordering (3D):* we associate each token with a 3D coordinate. Voxel tokens use their native voxel-space coordinates, while image patch tokens are assigned 3D coordinates by back-projecting pixel/patch coordinates using a depth map $D^{(k)}$, obtained by projecting the LiDAR sweep into view k:

$$\mathbf{x} = \pi^{-1}\big(u, v, D^{(k)}(u, v); \mathbf{K}_k, \mathbf{T}_k\big), \tag{6}$$

where π^{-1} denotes back-projection using camera intrinsics $\mathbf{K}_k$ and extrinsics $\mathbf{T}_k$, and $\mathbf{x} \in \mathbb{R}^3$ can be discretized to a voxel index for sorting/windowing. These two ordering policies differ only in the coordinate space used for sorting/windowing; both are realized by index computation and gather/scatter, without explicit feature resampling or warping. We provide a qualitative visualization of this scan-based sorting in Figure 4.

3.5 Unified Feature Encoding and Fusion via Vanilla Sequence Operations

After serialization, FusionNeXt processes the resulting token sequences using a standard and optimized LLM/VLM-style operator stack. Specifically, we stack identical pre-normalized residual blocks:

$$\mathbf{X} \leftarrow \mathbf{X} + \text{FlashAttn}(\text{RMSNorm}(\mathbf{X})), \tag{7}$$

$$\mathbf{X} \leftarrow \mathbf{X} + \text{SwiGLU}(\text{RMSNorm}(\mathbf{X})). \tag{8}$$

FlashAttention [14] is applied on the serialized 1D sequence, and the same block structure is reused regardless of which serialization strategy (2D or 3D) produced the sequences. In practice, we use FlashAttention's built-in *sliding-window attention*, with window size set to N tokens (a hyperparameter), to restrict attention to local neighborhoods on the serialized sequence. Additionally, we adopt QK-Norm [23], RMSNorm [75], and rotary positional embeddings (RoPE) [58], following common practice in modern LLM/VLM stacks. The fusion backbone operator stack relies on off-the-shelf optimized implementations (*e.g.*, FlashAttention [14]), without fusion-specific attention kernels.

3.6 Task Backbone & Heads

The fusion backbone outputs refined token sequences, which are restored to sparse voxel features (and optionally image patch features) and passed to standard task-specific backbones and heads. In our setting, we attach (i) a BEV backbone with a TransFusion [2] head for 3D detection and (ii) an occupancy pipeline following UniVision [24]; these task modules are independent of the fusion backbone and can be replaced without modifying it. Architectural details are provided in Appendix A.

4 Experiments

4.1 Benchmarks, Tasks, and Metrics

nuScenes dataset. We evaluate FusionNeXt on the nuScenes benchmark [6], a large-scale autonomous driving dataset with synchronized multi-sensor suites. Each scene provides multi-view cameras and a LiDAR sensor, together with 3D box annotations for surrounding traffic participants.

3D object detection. For the nuScenes 3D detection task, the model predicts 3D bounding boxes and class labels. We report the standard nuScenes detection metrics, including mean Average Precision (mAP) and the nuScenes Detection Score (NDS). NDS combines mAP with a set of true-positive quality terms that measure translation, scale, orientation, velocity, and attribute errors (mATE/mASE/mAOE/mAVE/mAAE).

Occupancy prediction. We evaluate 3D semantic occupancy prediction on the Occ3D benchmark [61]. Occ3D formalizes occupancy prediction as jointly estimating the occupancy state of each voxel (occupied/free/unobserved) and semantic labels for occupied voxels. We report per-class IoU and mean IoU (mIoU) over semantic classes.

Detailed FusionNeXt configurations, backbone choices, heads, training settings, and the FPS benchmarking protocol are provided in Appendix A.

4.2 Main Results

3D object detection. We summarize representative fusion 3D detectors on nuScenes, and report the inference speed as stated in their original papers.

As shown in Table 2, FusionNeXt achieves leading accuracy under this single-model, no-TTA, no-ensembling protocol, with strong efficiency. FusionNeXt-Base achieves 75.3 NDS and 73.7 mAP on the validation set, and 75.9 NDS and 74.0 mAP on the test set, comparing favorably with prior and concurrent methods such as BEVDilation (75.0 val NDS). More notably, FusionNeXt-Tiny achieves 73.5 val NDS at 18.9 / 23.5 FPS (RTX 4090 / H100), demonstrating a favorable accuracy–efficiency trade-off. We additionally reproduce FlatFusion, DAL, and MambaFusion on our RTX 4090 / H100 environment (parenthesized FPS in Table 2). Under these conditions, FusionNeXt-Small-1 surpasses all three in val NDS while running 1.4–$2.6\times$ faster. As these system-level measurements also reflect implementation choices, Section 4.4 further isolates the efficiency of the sequence interface under matched attention.

3D occupancy prediction. As shown in Table 3, FusionNeXt achieves 56.62 mIoU, outperforming all compared multi-modal fusion baselines such as DAOcc (54.33 mIoU) and EFFOcc-Base (54.08 mIoU). FusionNeXt demonstrates particularly strong performance on challenging classes such as vegetation (73.11%), manmade structures (74.14%), and terrain (65.17%). For dynamic objects, FusionNeXt achieves competitive results on pedestrian (61.59%) and motorcycle (54.70%), benefiting from the effective fusion through our unified sequence modeling approach.

4.3 Ablation Studies

Ablation on progressive design choices. We ablate reformulation-specific design choices, focusing on how 3D fusion is mapped into a sequence-first pipeline, by comparing progressive baselines in Table 4 and alternative serialization strategies in Table 5. Since the core sequence operators are standard LLM/VLM defaults, these ablations intentionally target the bridge itself: token unification, geometry-aware ordering, ordering-swap policy, fusion depth, and sliding-window size.

Table 2: Multi-modal fusion methods on nuScenes detection. We report NDS/mAP on validation (val) and test (test). FPS is reported from original papers for prior methods; FusionNeXt FPS is reported with RTX 4090 / H100 GPU. *For a fairer comparison, we additionally reproduce FlatFusion, DAL, and MambaFusion under the same RTX 4090 / H100 environment, shown in parentheses.

Method	Modality	Val NDS	Val mAP	Test NDS	Test mAP	FPS ↑
TransFusion [2]	L+C	71.3	67.5	71.6	68.9	-
UVTR [36]	L+C	70.2	65.4	71.1	67.1	-
BEVFusion [42]	L+C	71.0	67.9	71.8	69.2	0.8
DeepInteraction [71]	L+C	72.6	69.9	73.4	70.8	2.6
FUTR3D [10]	L+C	68.3	64.5	-	-	-
MSMDFusion [29]	L+C	72.1	69.3	74.0	71.5	2.1
SparseFusion [67]	L+C	72.8	70.4	73.8	72.0	5.6
CMT [68]	L+C	72.9	70.3	74.1	72.0	6.4
UniTR [65]	L+C	73.3	70.5	74.5	70.9	-
FlatFusion [82]	L+C	73.7	70.6	74.7	71.1	10.1 (10.0 / 13.2)*
EVT [31]	L+C	74.6	72.1	75.3	72.6	8.3
DAL [27]	L+C	74.0	71.5	74.8	72.0	4.3 (9.7 / 10.8)*
BEVDilation [76]	L+C	75.0	73.0	75.4	73.1	7.08
IS-Fusion [73]	L+C	74.0	72.8	75.2	73.0	3.2
MambaFusion [66]	L+C	74.0	71.6	75.0	72.0	5.4 (6.6 / 7.2)*
FusionNeXt-Tiny	L+C	73.5	70.2	-	-	**18.9 / 23.5**
FusionNeXt-Small-1	L+C	74.3	72.4	75.2	73.1	**17.1 / 18.9**
FusionNeXt-Small-2	L+C	74.8	73.3	-	-	**14.5 / 16.2**
FusionNeXt-Base	L+C	**75.3**	**73.7**	**75.9**	**74.0**	**11.3 / 14.5**

Table 3: Multi-modal fusion 3D occupancy on Occ3D-nuScenes. We report overall mIoU and per-class IoU.

Method	Modality	mIoU	others	barrier	bicycle	bus	car	cons. vehi.	motorcycle	pedestrian	traffic_cone	trailer	truck	driv. surf.	other flat	sidewalk	terrain	manmade	vegetation
OccFusion [45]	C+R	38.26	10.11	43.48	26.03	48.25	53.29	24.98	33.35	31.75	27.31	29.88	40.15	77.59	35.47	43.64	48.01	42.73	33.81
TEOcc [39]	C+R	42.90	10.82	50.33	24.28	48.99	57.32	29.38	24.41	30.14	28.46	36.46	43.01	83.96	43.09	56.00	59.34	54.18	49.16
REOcc [56]	C+R	45.33	13.32	51.53	31.82	52.46	58.71	29.55	32.78	32.99	32.08	39.92	44.88	84.54	46.44	57.69	60.32	54.04	47.50
OccFusion [45]	C+L	46.79	11.65	47.81	32.07	57.27	57.51	31.80	40.11	47.35	33.74	45.81	50.35	78.79	37.17	44.36	53.36	63.18	63.20
OccFusion [45]	C+L+R	46.67	12.40	50.30	31.50	57.60	58.80	34.00	41.00	47.20	29.70	42.00	48.00	78.40	35.70	47.30	52.70	63.50	63.30
RadOcc-LC [77]	C+L	49.38	10.93	58.23	25.01	57.89	62.85	34.04	33.45	50.07	32.05	48.87	52.11	82.90	42.73	55.27	58.34	68.64	66.01
SDG-KL [16]	C+L	50.16	12.26	57.12	23.69	58.77	62.74	34.55	36.19	50.10	32.05	49.89	51.24	84.10	46.05	57.20	61.45	69.56	65.78
SDG-Fusion [16]	C+L	51.66	13.21	57.77	24.30	60.33	64.28	36.21	39.44	52.36	35.80	50.91	53.65	84.56	47.45	58.00	61.61	70.67	67.65
EFFOcc-TC [55]	C+L	50.46	-	57.22	40.82	57.60	61.99	34.93	50.18	55.92	42.90	40.05	50.09	77.84	38.60	47.78	54.90	67.36	65.31
EFFOcc-Base [55]	C+L	54.08	-	**60.98**	36.21	62.24	66.42	**38.68**	43.88	52.12	42.40	50.29	**56.08**	84.92	48.00	58.60	61.99	71.29	69.48
DAOcc [72]	C+L	54.33	13.00	60.70	39.80	**64.00**	**66.50**	36.30	49.00	60.10	44.30	50.70	55.90	82.90	44.60	56.80	60.60	70.10	68.30
FusionNeXt-Base	C+L	**56.62**	**14.11**	58.14	**43.56**	63.32	65.33	38.15	**54.70**	**61.59**	**49.29**	**51.37**	54.79	**85.04**	**49.75**	**60.90**	**65.17**	**74.14**	**73.11**

Starting from a baseline (67.6 NDS / 63.9 mAP) where tokens are directly fed to detection backbone and heads, applying FusionNeXt blocks to voxel tokens only in early layers steadily improves performance (up to 70.0 NDS / 66.9 mAP with 4 layers), and enabling cross-modal interaction in later layers (still with the same block architecture, now operating on the unified token set) further boosts accuracy (73.5 NDS / 70.2 mAP with 8 layers). These results indicate that our sequence-first blocks are effective for both unimodal feature enhancement and efficient cross-modal interaction, providing a unified building block for 3D multi-modal fusion.

The 256-channel configuration reaches 74.8 NDS / 73.3 mAP. Disabling geometry-aware serialization ("Del geometry-aware" in Table 4) and using a random 1D order drops performance to 71.2 NDS / 68.1 mAP, highlighting the importance of locality-preserving token ordering for effective fusion. We additionally vary the cross-modal fusion block depth (6/8/10) and observe an 8-layer setting performs best under our Small-2 configuration (Table 4).

Robustness to serialization strategy choices. Among the geometry-aware strategies, we explore two design dimensions: (i) *ordering pattern*—sequential ($\mathrm{I}_4 \rightarrow \mathrm{L}_4$: 4 layers of image-plane (2D) ordering followed by 4 layers of voxel-space (3D) ordering) vs. alternating ($(\mathrm{I} \rightarrow \mathrm{L})_4$: alternating between image-plane (2D) and voxel-space (3D) ordering for 4 iterations); and (ii) *fusion mode*—serial ($\rightarrow$: image-plane (2D) and voxel-space (3D) orderings are applied sequentially) vs. parallel ($\|$: the two orderings are applied in parallel and then merged). Except for the no-swapping baseline,

Table 4: **Progressive ablation.** We progressively add LiDAR-only and cross-modal stages (using the same FusionNeXt block architecture), optionally enabling geometry-aware serialization, and further vary the cross-modal depth.

Settings	#L	#LC	Geo-aware	Base feat.	NDS ↑	mAP ↑
Baseline	0	0	✓	128	67.6	63.9
Add LiDAR-only blocks	2	0	✓	128	69.4	66.0
	4	0	✓	128	70.0	66.9
Add cross-modal blocks	4	4	✓	128	72.9	70.8
	4	8	✓	128	73.5	70.2
Del geometry-aware	4	8	×	128	71.2	68.1
256-channel configuration	4	8	✓	256	**74.8**	**73.3**
Vary cross-modal layers	4	6	✓	256	74.7	73.0
	4	8	✓	256	**74.8**	**73.3**
	4	10	✓	256	74.4	72.5

Table 5: **Serialization strategy.** Ablation of geometry-aware strategies (I: image-plane (2D) ordering, L: voxel-space (3D) ordering; subscript: number of layers) applied to the 8 fusion blocks.

Strategy	Swapping	NDS ↑
Seq.+Serial ($\mathrm{I}_4 \rightarrow \mathrm{L}_4$)	×	74.0
Seq.+Serial ($\mathrm{I}_4 \rightarrow \mathrm{L}_4$)	✓	**74.8**
Alt.+Serial ($(\mathrm{I} \rightarrow \mathrm{L})_4$)	✓	74.7
Seq.+Parallel ($\mathrm{I}_4 \parallel \mathrm{L}_4$)	✓	74.7
Alt.+Parallel ($(\mathrm{I} \parallel \mathrm{L})_4$)	✓	74.6

Table 6: **Sliding-window size N.** Ablation of the FlashAttention sliding-window size N for local attention computation.

N	NDS ↑	latency (ms)↓
40	74.5	1.16
80	**74.8**	1.16
160	74.2	1.64
1024	73.2	4.58

these configurations achieve similar performance (74.6–74.8 NDS). These results indicate that our framework is robust and insensitive to the specific ordering-swap schedule. No delicate hand-designed swapping strategy is required.

Sliding-window size in FlashAttention. In FlashAttention, we use the in-built sliding-window attention mechanism with a window size of N tokens for efficient local attention computation. We ablate the sliding-window size N with 40, 80, 160, and 1024 tokens on the FusionNeXt-**Small-2** model and find that a moderate window size performs best: $N = 80$ achieves 74.8 NDS at 1.16 ms, while a much larger window ($N = 1024$) increases latency to 4.58 ms and degrades accuracy to 73.2 NDS (Table 6).

4.4 Efficiency Attribution and Operator Compatibility

The packed interface is intended to reduce intermediate processing and support reusable sequence operators. We evaluate these properties through fusion-encoder latency comparisons, including a matched-attention control, and an operator replacement.

Fusion encoder latency. We compare the FusionNeXt fusion encoder with FlatFusion [82] on H100 (PyTorch 2.1, CUDA 12.1, batch size 4) across token and layer counts (Figure 5). Timing covers serialization and the two stacked sequence blocks (attention and FFN layers) on synthetic tokens, excluding tokenizers and task heads. In the optimized setting, both methods use fused attention kernels (FlashAttention / xFormers); FusionNeXt calls the variable-length kernel directly on the packed sequence, whereas FlatFusion groups and pads tokens around it. Under these optimized implementations, FusionNeXt is 1.24×–1.73× faster; the gain is largest at small workloads (1.73× at one layer, 1.54× at 10k tokens) and remains 1.24×–1.28× at the largest settings. Because these measurements also reflect kernel choices, we further replace attention in both models with the same manual PyTorch attention, fixing width 128, 8 heads, and window size $K = 80$, and report latency.

Under matched manual attention, FusionNeXt remains **1.20×–1.26× faster across token counts** and **1.16×–1.35× across layer counts** (Figure 5). The measurements support a constant-factor benefit from the simplified processing path. As shared attention/FFN work grows with depth, the relative benefit of avoiding grouping, padding, and layout recovery decreases.

Operator compatibility. We replace attention with Mamba/SSM [19] using compiled `mamba_ssm 2.2.2` ($d_{\mathrm{state}} = 16$, $d_{\mathrm{conv}} = 4$, expansion 2). Both operators use local BF16 autocast and return FP32

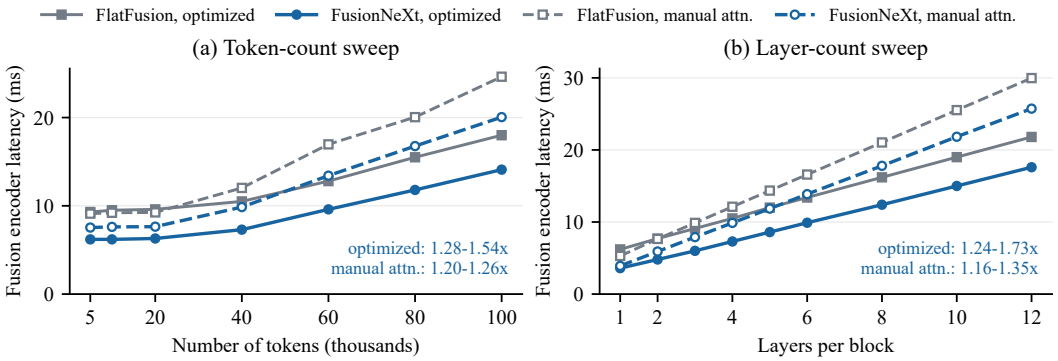


Figure 5: **Fusion encoder latency of FusionNeXt and FlatFusion.** Solid lines use each system's optimized attention implementation; dashed lines use the same unfused manual attention in both encoders. Runs use four layers per block in (a) and 40k tokens in (b).

outputs to the FP32 detector. The SSM variant retains 89.0% throughput with -0.33 NDS/-0.46 mAP changes (Table 7(a)). This supports reuse of the packed interface across both operator families, including fusion without FlashAttention.

System comparison with MambaFusion. We profile released MambaFusion [66] and FusionNeXt–SSM on the same H100 and software environment, matching the Mamba implementation, mixer settings, and BF16 precision above. Each model retains its own encoders, width, and layer schedule. Table 7(b) separates image/LiDAR feature processing, fusion processing, and explicit cross-modal feature transfer/warping; task backbones and heads are excluded. Fusion processing includes SSM mixing and other operations such as projection, grouping/padding, and gather/scatter. Although FusionNeXt–SSM has higher SSM latency (11.42 vs. 7.04 ms), its profiled total is 64.3% lower (50.44 vs. 141.27 ms). Together, these results show that our framework reduces modality-specific processing and auxiliary fusion/transfer overhead, making the overall processing pipeline more efficient.

Table 7: **Operator compatibility and system comparison with MambaFusion.** (a) nuScenes val, epoch 5; H100, batch 1. (b) Profiled component latencies with the same Mamba implementation on H100. Indented rows partition fusion processing; — indicates no separately profiled stage. Displayed sums may differ from totals due to rounding.

(a) Operator replacement

Operator	NDS	mAP	FPS
Attention	74.82	73.03	15.23
SSM	74.49	72.57	13.55

(b) System latency breakdown (ms)

Profiled component	MambaFusion	FusionNeXt (SSM)
Modality-specific feature processing	84.14	21.76
Fusion processing (total)	38.40	28.68
SSM mixing (included)	7.04	11.42
Other fusion operations	31.36	17.25
Cross-modal feature transfer/warping	18.73	—
Total profiled latency	141.27	**50.44**

5 Conclusion

We presented FusionNeXt, a multi-modal fusion framework that brings the sequence-first design of LLMs/VLMs to 3D perception. It processes unified camera and LiDAR tokens with vanilla sequence operators (FlashAttention, SwiGLU FFN), achieving 75.9 test NDS on nuScenes and 56.62 mIoU on Occ3D with fast inference. Matched attention comparisons substantiate the sequence-first interface's efficiency, while SSM substitution demonstrates flexibility across sequence implementations. More broadly, FusionNeXt suggests that multi-modal 3D perception need not remain a separate design space with its own specialized operators and tooling. By sharing the sequence interface of LLMs/VLMs, it can join a broader, reusable ecosystem and evolve together with it, benefiting from advances in operators, systems, and model design as they emerge.

Acknowledgments and Disclosure of Funding

This work was supported by the National Science and Technology Major Project (Grant No. 2025ZD1606203), the Shanghai Municipal Universities Industry-Academia-Research Joint Innovation Program, and the National Natural Science Foundation of China (Grant No. 62272433). The authors declare no competing interests.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning, 2022. URL `https://arxiv.org/abs/2204.14198`.

[2] Xuyang Bai, Zeyu Hu, Xinge Zhu, Qingqiu Huang, Yilun Chen, Hongbo Fu, and Chiew-Lan Tai. Transfusion: Robust lidar-camera fusion for 3d object detection with transformers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[3] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer, 2020. URL `https://arxiv.org/abs/2004.05150`.

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL `https://arxiv.org/abs/2005.14165`.

[5] Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. UniVLA: Learning to Act Anywhere with Task-centric Latent Actions, May 2025.

[6] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11618–11628. IEEE, June 2020. doi: 10.1109/CVPR42600.2020.01163.

[7] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision (ECCV)*, 2020.

[8] Ting Chen, Saurabh Saxena, Lala Li, Tsung-Yi Lin, David J. Fleet, and Geoffrey Hinton. A Unified Sequence Interface for Vision Tasks, October 2022.

[9] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. MV3D: Multi-View 3D Object Detection Network for Autonomous Driving. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6526–6534. IEEE, July 2017. doi: 10.1109/CVPR.2017.691.

[10] Xuanyao Chen, Tianyuan Zhang, Yue Wang, Yilun Wang, and Hang Zhao. Futr3d: A unified sensor fusion framework for 3d detection. *arXiv preprint arXiv:2203.10642*, 2022.

[11] Xuanyao Chen, Zhijian Liu, Haotian Tang, Li Yi, Hang Zhao, and Song Han. SparseViT: Revisiting Activation Sparsity for Efficient High-Resolution Vision Transformer, March 2023.

[12] Zehui Chen, Zhenyu Li, Shiquan Zhang, Liangji Fang, Qinhong Jiang, and Feng Zhao. Autoalignv2: Deformable feature aggregation for dynamic multi-modal 3d object detection. In *European Conference on Computer Vision (ECCV)*, 2022. URL `https://arxiv.org/abs/2207.10316`. arXiv:2207.10316.

[13] Haohan Chi, Huan-ang Gao, Ziming Liu, Jianing Liu, Chenyu Liu, Jinwei Li, Kaisen Yang, Yangcheng Yu, Zeda Wang, Wenyi Li, Leichen Wang, Xingtao Hu, Hao Sun, Hang Zhao, and Hao Zhao. Impromptu VLA: Open Weights and Open Data for Driving Vision-Language-Action Models, May 2025.

[14] Tri Dao, Daniel Y Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[15] DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. DeepSeek-V3 Technical Report, February 2025.

[16] ZaiPeng Duan, ChenXu Dang, Xuzhong Hu, Pei An, Junfeng Ding, Jie Zhan, YunBiao Xu, and Jie Ma. Sdgocc: Semantic and depth-guided bird's-eye view transformation for 3d multimodal occupancy prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6751–6760, 2025.

[17] Yuxin Fang, Bencheng Liao, Xinggang Wang, Jiemin Fang, Jiyang Qi, Rui Wu, Jianwei Niu, and Wenyu Liu. You Only Look at One Sequence: Rethinking Transformer in Vision through Object Detection, October 2021.

[18] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI Vision Benchmark Suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361. IEEE, June 2012. doi: 10.1109/CVPR.2012.6248074.

[19] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023. URL `https://arxiv.org/abs/2312.00752`.

[20] James Gunn, Zygmunt Lenyk, Anuj Sharma, Andrea Donati, Alexandru Buburuzan, John Redford, and Romain Mueller. Lift-Attend-Splat: Bird's-eye-view camera-lidar fusion using transformers, May 2024.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[22] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked Autoencoders Are Scalable Vision Learners, December 2021.

[23] Alex Henry, Prudhvi Raj Dachapally, Shubham Pawar, and Yuxuan Chen. Query-key normalization for transformers. *arXiv preprint arXiv:2010.04245*, 2020.

[24] Yu Hong, Qian Liu, Huayuan Cheng, Danjiao Ma, Hang Dai, Yu Wang, Guangzhi Cao, and Yong Ding. Univision: A unified framework for vision-centric 3d perception. *arXiv preprint arXiv:2401.06994*, 2024.

[25] Chunyong Hu, Hang Zheng, Kun Li, Jianyun Xu, Weibo Mao, Maochun Luo, Lingxuan Wang, Mingxia Chen, Qihao Peng, Kaixuan Liu, Yiru Zhao, Peihan Hao, Minzhe Liu, and Kaicheng Yu. FusionFormer: A Multi-sensory Fusion in Bird's-Eye-View and Temporal Consistent Transformer for 3D Object Detection, October 2023.

[26] Junjie Huang, Guan Huang, Zheng Zhu, Yun Ye, and Dalong Du. Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. *arXiv preprint arXiv:2112.11790*, 2021.

[27] Junjie Huang, Yun Ye, Zhujin Liang, Yi Shan, and Dalong Du. Detecting as labeling: Rethinking lidar-camera fusion in 3d object detection. In *European Conference on Computer Vision (ECCV)*, 2024. arXiv:2311.07152.

[28] Ding Jia, Yuhui Yuan, Haodi He, Xiaopei Wu, Haojun Yu, Weihong Lin, Lei Sun, Chao Zhang, and Han Hu. DETRs with Hybrid Matching, May 2023.

[29] Yang Jiao, Zequn Jie, Shaoxiang Chen, Jingjing Chen, Lin Ma, and Yu-Gang Jiang. 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 (CVPR)*, pages 21643–21652, 2023.

[30] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An Open-Source Vision-Language-Action Model, September 2024.

[31] Yongjin Lee, Hyeon-Mun Jeong, Yurim Jeon, and Sanghyun Kim. Evt: Efficient view transformation for multi-modal 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. URL `https://arxiv.org/abs/2411.10715`. arXiv:2411.10715.

[32] Feng Li, Hao Zhang, Shilong Liu, Jian Guo, Lionel M. Ni, and Lei Zhang. DN-DETR: Accelerate DETR Training by Introducing Query DeNoising, December 2022.

[33] Feng Li, Hao Zhang, Huaizhe Xu, Shilong Liu, Lei Zhang, Lionel M. Ni, and Heung-Yeung Shum. Mask DINO: Towards A Unified Transformer-based Framework for Object Detection and Segmentation. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3041–3050, Vancouver, BC, Canada, June 2023. IEEE. ISBN 979-8-3503-0129-8. doi: 10.1109/CVPR52729.2023.00297.

[34] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models, June 2023.

[35] Xiaotian Li, Baojie Fan, Jiandong Tian, and Huijie Fan. 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 (CVPR)*, 2024.

[36] Yanwei Li, Yilun Chen, Xiaojuan Qi, Zeming Li, Jian Sun, and Jiaya Jia. Unifying voxel-based representation with transformer for 3d object detection. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[37] Yingwei Li, Adams Wei Yu, Tianjian Meng, Ben Caine, Jiquan Ngiam, Daiyi Peng, Junyang Shen, Yifeng Lu, Denny Zhou, Quoc V. Le, Alan Yuille, and Mingxing Tan. Deepfusion: Lidar-camera deep fusion for multi-modal 3d object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[38] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *European Conference on Computer Vision (ECCV)*, 2022.

[39] Zhiwei Lin, Hongbo Jin, Yongtao Wang, Yufei Wei, and Nan Dong. Teocc: Radar-camera multi-modal occupancy prediction via temporal enhancement. *arXiv preprint arXiv:2410.11228*, 2024. URL `https://arxiv.org/abs/2410.11228`.

[40] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows, 2021. URL `https://arxiv.org/abs/2103.14030`.

[41] Zhe Liu, Jinghua Hou, Xinyu Wang, Xiaoqing Ye, Jingdong Wang, Hengshuang Zhao, and Xiang Bai. LION: Linear Group RNN for 3D Object Detection in Point Clouds, July 2024.

[42] Zhijian Liu, Haotian Tang, Alexander Amini, Xinyu Yang, Huizi Mao, Daniela Rus, and Song Han. Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. *arXiv preprint arXiv:2205.13542*, 2022.

[43] Zhijian Liu, Xinyu Yang, Haotian Tang, Shang Yang, and Song Han. FlatFormer: Flattened Window Attention for Efficient Point Cloud Transformer. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1200–1211, Vancouver, BC, Canada, June 2023. IEEE. ISBN 979-8-3503-0129-8. doi: 10.1109/CVPR52729.2023.00122.

[44] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

[45] Zhenxing Ming, Julie Stephany Berrio, Mao Shan, and Stewart Worrall. Occfusion: Multi-sensor fusion framework for 3d semantic occupancy prediction. *arXiv preprint arXiv:2403.01644*, 2024. URL `https://arxiv.org/abs/2403.01644`.

[46] MMDetection3D Contributors. MMDetection3D: OpenMMLab next-generation platform for general 3D object detection. `https://github.com/open-mmlab/mmdetection3d`, 2020.

[47] OpenPCDet Development Team. OpenPCDet: An open-source toolbox for 3D object detection from point clouds. `https://github.com/open-mmlab/OpenPCDet`, 2020.

[48] Su Pang, Daniel Morris, and Hayder Radha. CLOCs: Camera-LiDAR Object Candidates Fusion for 3D Object Detection. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10386–10393. IEEE, October 2020. doi: 10.1109/IROS45743.2020.9341791.

[49] Jonah Philion and Sanja Fidler. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *European Conference on Computer Vision (ECCV)*, 2020. arXiv:2008.05711.

[50] Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum PointNets for 3D Object Detection from RGB-D Data. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 918–927. IEEE, June 2018. doi: 10.1109/CVPR.2018.00102.

[51] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. URL `https://arxiv.org/abs/2103.00020`.

[52] Cody Reading, Ali Harakeh, Julia Chae, and Steven L. Waslander. Categorical depth distribution network for monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. arXiv:2103.01100.

[53] Sudhakar Sah, Ravish Kumar, Honnesh Rohmetra, and Ehsan Saboori. Token Pruning using a Lightweight Background Aware Vision Transformer, October 2024.

[54] Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

[55] Yining Shi, Kun Jiang, Jinyu Miao, Ke Wang, Kangan Qian, Yunlong Wang, Jiusi Li, Tuopu Wen, Mengmeng Yang, Yiliang Xu, and Diange Yang. Effocc: Learning efficient occupancy networks from minimal labels for autonomous driving. *arXiv preprint arXiv:2406.07042*, 2024. URL `https://arxiv.org/abs/2406.07042`.

[56] Chaehee Song, Sanmin Kim, Hyeonjun Jeong, Juyeb Shin, Joonhee Lim, and Dongsuk Kum. Reocc: Camera-radar fusion with radar feature enrichment for 3d occupancy prediction. *arXiv preprint arXiv:2511.06666*, 2025. URL `https://arxiv.org/abs/2511.06666`.

[57] Ziying Song, Feiyang Jia, Hongyu Pan, Yadan Luo, Caiyan Jia, Guoxin Zhang, Lin Liu, Yang Ji, Lei Yang, and Li Wang. ContrastAlign: Toward Robust BEV Feature Alignment via Contrastive Learning for Multi-Modal 3D Object Detection, June 2024.

[58] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

[59] Jiachen Sun, Haizhong Zheng, Qingzhao Zhang, Atul Prakash, Z. Morley Mao, and Chaowei Xiao. CALICO: Self-Supervised Camera-LiDAR Contrastive Pre-training for BEV Perception, November 2023.

[60] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurélien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo Open Dataset. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2443–2451. IEEE, June 2020. doi: 10.1109/CVPR42600.2020.00252.

[61] Xiaoyu Tian, Tao Jiang, Longfei Yun, Yucheng Mao, Huitong Yang, Yue Wang, Yilun Wang, and Hang Zhao. Occ3d: A large-scale 3d occupancy prediction benchmark for autonomous driving. *arXiv preprint arXiv:2304.14365*, 2023.

[62] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

[63] Sourabh Vora, Alex H. Lang, Bassam Helou, and Oscar Beijbom. Pointpainting: Sequential Fusion for 3D Object Detection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4603–4611. IEEE, June 2020. doi: 10.1109/CVPR42600.2020.00466.

[64] Chunwei Wang, Chao Ma, Ming Zhu, and Xiaokang Yang. Pointaugmenting: Cross-modal augmentation for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[65] Haiyang Wang, Hao Tang, Shaoshuai Shi, Aoxue Li, Zhenguo Li, Bernt Schiele, and Liwei Wang. 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 (ICCV)*, 2023.

[66] Hanshi Wang, Jin Gao, Weiming Hu, and Zhipeng Zhang. Height-fidelity dense global fusion for multi-modal 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. arXiv:2507.04369.

[67] Yichen Xie, Chenfeng Xu, Marie-Julie Rakotosaona, Patrick Rim, Federico Tombari, Kurt Keutzer, Masayoshi Tomizuka, and Wei Zhan. Sparsefusion: Fusing multi-modal sparse representations for multi-sensor 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. arXiv:2304.14340.

[68] Junjie Yan, Yingfei Liu, Jianjian Sun, Fan Jia, Shuailin Li, Tiancai Wang, and Xiangyu Zhang. Cross modal transformer: Towards fast and robust 3d object detection. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[69] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10): 3337, 2018.

[70] Chenyu Yang, Yuntao Chen, Hao Tian, Chenxin Tao, Xizhou Zhu, Zhaoxiang Zhang, Gao Huang, Hongyang Li, Yu Qiao, Lewei Lu, Jie Zhou, and Jifeng Dai. BEVFormer v2: Adapting Modern Image Backbones to Bird's-Eye-View Recognition via Perspective Supervision, November 2022.

[71] Zeyu Yang, Jiaqi Chen, Zhenwei Miao, Wei Li, Xiatian Zhu, and Li Zhang. Deepinteraction: 3d object detection via modality interaction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[72] Zhen Yang, Yanpeng Dong, Jiayu Wang, Heng Wang, Lichao Ma, Zijian Cui, Qi Liu, Haoran Pei, Kexin Zhang, and Chao Zhang. Daocc: 3d object detection assisted multi-sensor fusion for 3d occupancy prediction. *IEEE Transactions on Circuits and Systems for Video Technology*, 2025. doi: 10.1109/TCSVT.2025.3610634. URL `https://arxiv.org/abs/2409.19972`. arXiv:2409.19972.

[73] Junbo Yin, Jianbing Shen, Runnan Chen, Wei Li, Ruigang Yang, Pascal Frossard, and Wenguan Wang. Is-fusion: Instance-scene collaborative fusion for multimodal 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[74] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Multimodal virtual point 3d detection. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. arXiv:2111.06881.

[75] Biao Zhang and Rico Sennrich. Root mean square layer normalization. *arXiv preprint arXiv:1910.07467*, 2019.

[76] Guowen Zhang, Chenhang He, Liyi Chen, and Lei Zhang. Bevdilation: Lidar-centric multi-modal fusion for 3d object detection, 2025. arXiv:2512.02972.

[77] Haiming Zhang, Xu Yan, Dongfeng Bai, Jiantao Gao, Pan Wang, Bingbing Liu, Shuguang Cui, and Zhen Li. Radocc: Learning cross-modality occupancy knowledge through rendering assisted distillation. *arXiv preprint arXiv:2312.11829*, 2023. URL `https://arxiv.org/abs/2312.11829`.

[78] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M. Ni, and Heung-Yeung Shum. DINO: DETR with Improved DeNoising Anchor Boxes for End-to-End Object Detection, July 2022.

[79] Yin Zhou, Pei Sun, Yu Zhang, Dragomir Anguelov, Jiyang Gao, Tom Ouyang, James Guo, Jiquan Ngiam, and Vijay Vasudevan. End-to-end multi-view fusion for 3d object detection in lidar point clouds. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 923–932, 2020. URL `https://proceedings.mlr.press/v100/zhou20a.html`.

[80] Benjin Zhu, Zhengkai Jiang, Xiangxin Zhou, Zeming Li, and Gang Yu. Class-balanced grouping and sampling for point cloud 3d object detection. *arXiv preprint arXiv:1908.09492*, 2019.

[81] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *International Conference on Learning Representations (ICLR)*, 2021.

[82] Yutao Zhu, Xiaosong Jia, Xinyu Yang, and Junchi Yan. Flatfusion: Delving into details of sparse transformer-based camera-lidar fusion for autonomous driving. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2025. doi: 10.48550/arXiv.2408.06832. arXiv:2408.06832.

A Implementation Details

Our implementation builds on OpenPCDet [47] and MMDetection3D [46], with task-specific backbone and head adaptation. The LiDAR tokenizer uses DynamicVFE [79], while the image tokenizer uses ResNet [21]; on a single H100 GPU, these two lightweight modules each consume about 5 ms. The reported models use 4 LiDAR-only blocks followed by 8 multimodal blocks. The eight fusion layers include four image-plane and four voxel-space orderings. The sequential and alternating schedules evaluated in Table 5 specify their arrangement.

All layers share the same block architecture; only the token composition and serialization order vary across layers, implemented via lightweight matrix multiplications and coordinate-based sorting. For detection task, we use the Transfusion head [2], and implement four configurations: **Tiny**, **Small-1**, **Small-2**, and **Base** as Table 8.

Table 8: FusionNeXt configurations.

Config.	Image	LiDAR	Feat. Ch.
Tiny	320×640, R18	0.3m, DynamicVFE	128
Small-1	320×640, R18	0.3m, DynamicVFE	256
Small-2	320×640, R50	0.3m, DynamicVFE	256
Base	480×960, R50	0.3m, DynamicVFE	256

For occupancy task, we use the UniVision head [24] and use the **Base** configuration as described above. Detection uses AdamW [44], 5 epochs, a learning rate of 3×10^{-4}, and CBGS [80]. Occupancy uses AdamW for 20 epochs at 3×10^{-4}. Both task heads are reused from their reference implementations and keep their standard objectives: the detection head optimizes focal classification, heatmap-based object localization, and box regression, while the occupancy head optimizes voxel-wise cross-entropy with Dice regularization. Progressive ablations vary feature width and stage depth as specified in Table 4; the serialization and window studies use Small-2 configuration.

Experiments are conducted on 8 NVIDIA H100 GPUs. For FPS reporting, we benchmark both RTX 4090 and H100 to demonstrate inference efficiency on comparable hardware and on more advanced hardware.

Compute budget. All training is performed on 8×NVIDIA H100 (80 GB) GPUs. A full training run of FusionNeXt-Small-2 on nuScenes takes approximately **5.9** hours (batch size **4** per GPU), with peak per-GPU memory usage of about **53.6** GB. The main ablation studies in Tables 4 to 6 comprise approximately **16** distinct training runs, corresponding to roughly **30** H100 GPU-days of compute under the Small-2 configuration.

FPS benchmarking protocol. The following protocol applies to our measurements in Table 2. Original-paper FPS values retain their original evaluation conditions. We use batch size 1 and vanilla PyTorch. The separate operator replacement experiment has its own precision and timing scope (Section 4.4).

Timing uses CUDA events with `torch.cuda.synchronize()` after every forward pass, following the NVIDIA-recommended methodology. We disable `cudnn.benchmark` and enable `cudnn.deterministic` to eliminate kernel auto-tuning jitter, run 100 warmup iterations, and then time 500 samples using trained checkpoints (never random weights). Data are pre-moved to the GPU so that dataloader I/O is excluded from timing. The reported FPS is the filtered mean after removing 2-σ outliers. The measured latency includes the *full* end-to-end forward pass—tokenizers, fusion backbone, detection head, NMS, and box decoding—rather than backbone-only. For each reproduced baseline (FlatFusion, DAL, MambaFusion) we reuse the authors' released code and checkpoints without modification, only wrapping the call site with our timing harness. All measurements are repeated on both NVIDIA RTX 4090 and NVIDIA H100 under identical software stacks (PyTorch 2.1, CUDA 12.1).

A.1 Design and Ablation Clarifications

Depth map for 3D ordering. The depth map $D^{(k)}$ used in the back-projection of image patch tokens to voxel space (Section 3.4) is obtained by projecting the LiDAR point cloud onto each camera

view to form a sparse depth map, which is then passed through a single lightweight convolutional layer that simultaneously smooths the LiDAR-covered regions and predicts depth for the remaining pixels. In this way, every image patch receives a valid depth and thus a well-defined 3D coordinate, without introducing a separate depth-prediction network.

View selection for multi-view-visible voxels. When a voxel token projects into more than one camera view, its 2D ordering coordinate requires a single view index k. During training, we sample k uniformly at random among the valid views, which acts as a mild view-level augmentation and encourages the model to be robust to the specific camera assignment. At inference, we deterministically pick the smallest valid camera index in canonical camera order, so that results are fully reproducible and independent of randomness.

B Additional Results

We provide additional experimental results and visualizations.

B.1 Inference Efficiency Scaling

We benchmark the fusion backbone latency under varying token counts and layer depths. As shown in Figure 6, the token scaling (left) grows sub-quadratically with the token count (4 layers per block). The layer scaling (right) is linear with a slope of 1.28 ms per layer (40k tokens). These measurements describe inference cost over the tested workloads.

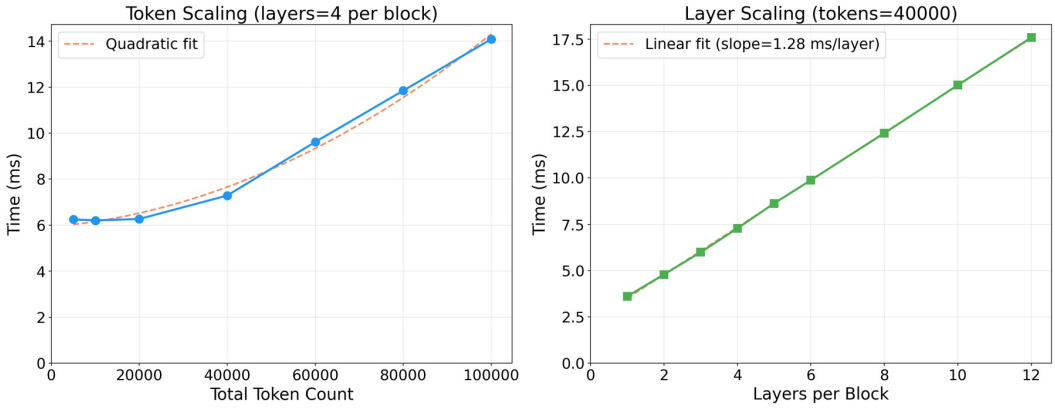


Figure 6: **Inference latency scaling of the FusionNeXt fusion backbone.** Left: latency vs. token count (4 layers per block). The actual latency (blue) scales sub-quadratically, outperforming the quadratic fit (dashed). Right: latency vs. layer count (40k tokens). The latency scales linearly with a slope of 1.28 ms/layer.

B.2 Inference Latency Breakdown

We profile the major forward-pass components of FusionNeXt-Small on an NVIDIA H100 GPU. As shown in Figure 7, the FusionNeXt Blocks account for 54.6% (31.9 ms) of the profiled latency, followed by the Detection Backbone & Heads at 27.1% (15.8 ms). The Image Tokenizer (ResNet) and Voxel Tokenizer (DynamicVFE) together take 18.3% (10.7 ms).

B.3 Visualization of Sorting Strategy Swap

We visualize the four sorting patterns used by FusionNeXt on two nuScenes samples in Figure 8. For each sample, the **top row (a–d)** shows the sorting order under voxel-space (3D) ordering in BEV space, and the **bottom row (e–h)** shows the sorting order under image-plane (2D) ordering on the front-camera image with LiDAR points overlaid. The color gradient (purple→yellow) indicates the position in the serialized 1D sequence. Within each ordering space, patterns alternate between X-primary and Y-primary serpentine scans, and each pair differs by a half-bin shift.

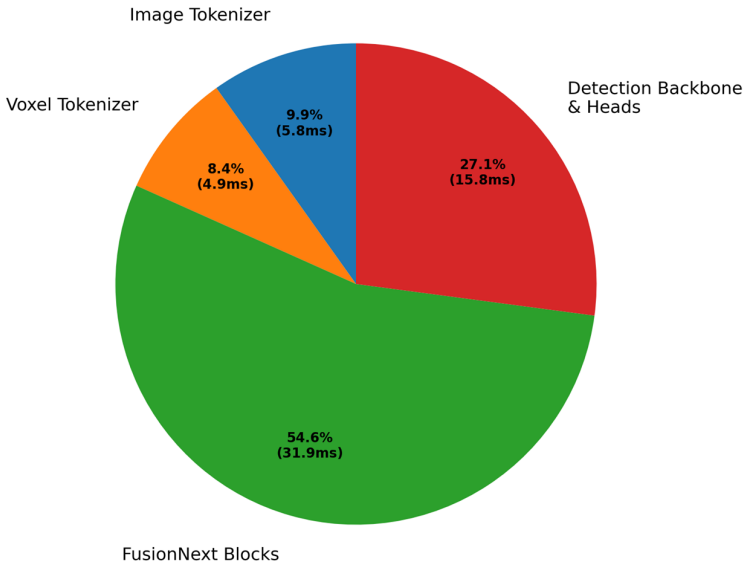


Figure 7: **Inference latency breakdown of FusionNeXt-Small.** Only the primary forward-pass components are profiled.

B.4 Visualization of Query-Key Interactions

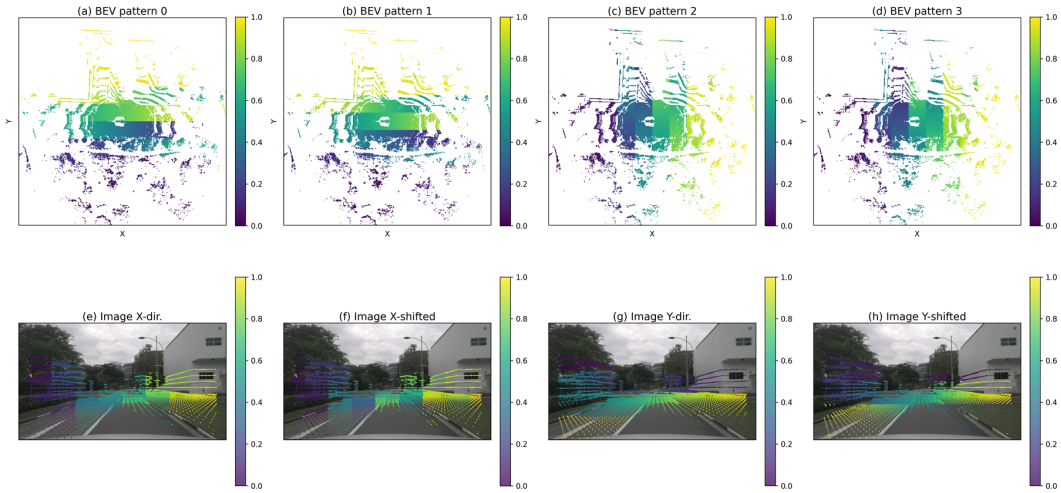
We visualize the query-key interactions in BEV space under the four sorting strategies. We select 5 query tokens (marked as $\star$) and show the key tokens (colored dots) that each query can attend to within the sliding window (Figure 9). Different colors represent different queries (Q1–Q5). X-direction sorting (a) produces horizontally-elongated receptive fields, while Y-direction sorting (c) yields vertically-elongated ones. The shifted variants (b, d) offset the bin boundaries, grouping the same query with a different set of spatial neighbors.

B.5 Layer-Wise Attention Map Visualization

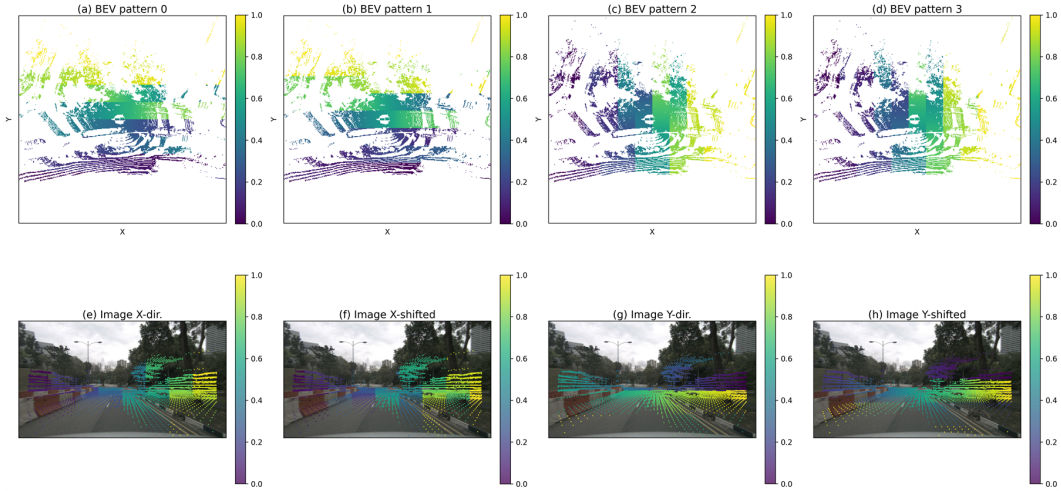
To validate that our 1D sequence modeling achieves spatial modeling comparable to traditional 2D/3D operators, we visualize the attention map of a selected query token across fusion layers. Specifically, we select a query token in 3D space (marked as $\star$) and visualize its attention distribution in both BEV (3D) and image (2D) space after layers 2, 4, 6, and 8 of the cross-modal fusion stage. Figure 10 shows results on two different samples.

Several key observations emerge:

- **Progressive spatial expansion.** In early layers, the attention is concentrated in a small local neighborhood around the query token in both BEV and image space. As the network deepens, the attended region gradually expands, eventually covering a broad spatial extent that approaches global coverage. This progressive diffusion is analogous to the expanding receptive field achieved by stacking 2D/3D convolutions or multi-scale attention.
- **Cross-modal fusion emerges naturally.** In the image-space views, the attention on camera patches starts sparse and localized near the geometric projection of the query token, and progressively spreads to semantically relevant regions. This demonstrates that cross-modal interaction is not imposed by specialized alignment operators but rather *emerges naturally* from the interplay of geometry-aware serialization and standard sliding-window attention.
- **Alternating serialization enables omnidirectional coverage.** The BEV attention maps reveal that alternating between image-plane (2D) and voxel-space (3D) ordering across layers produces complementary spatial patterns. This effectively eliminates the directional bias of any single serialization strategy, yielding an isotropic effective receptive field similar to that of 2D/3D convolutions.

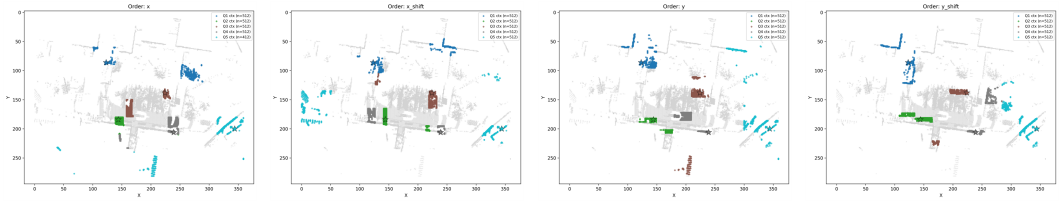


(a) Sample 1



(b) Sample 2

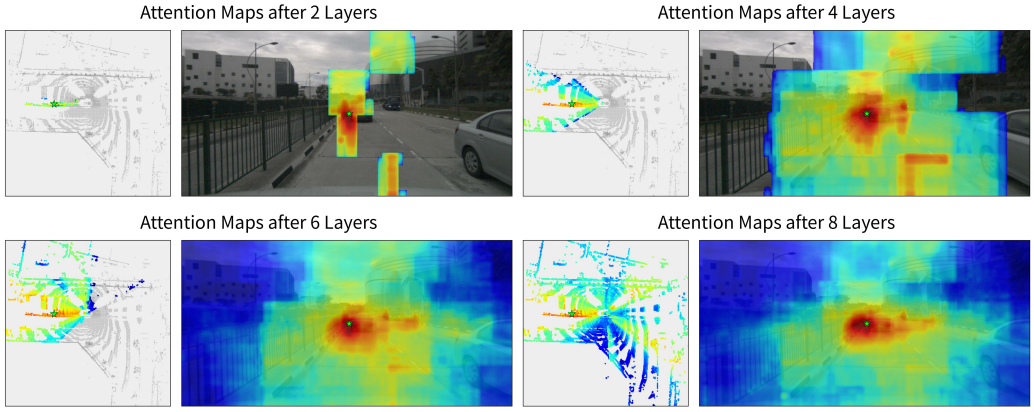
Figure 8: **Visualization of sorting strategy swap on two samples.** For each sample, **top row (a–d):** four sorting patterns under voxel-space (3D) ordering in BEV space; **bottom row (e–h):** four sorting patterns under image-plane (2D) ordering on the front-camera image with LiDAR points overlaid. The color gradient (purple→yellow) indicates the 1D sequence position.



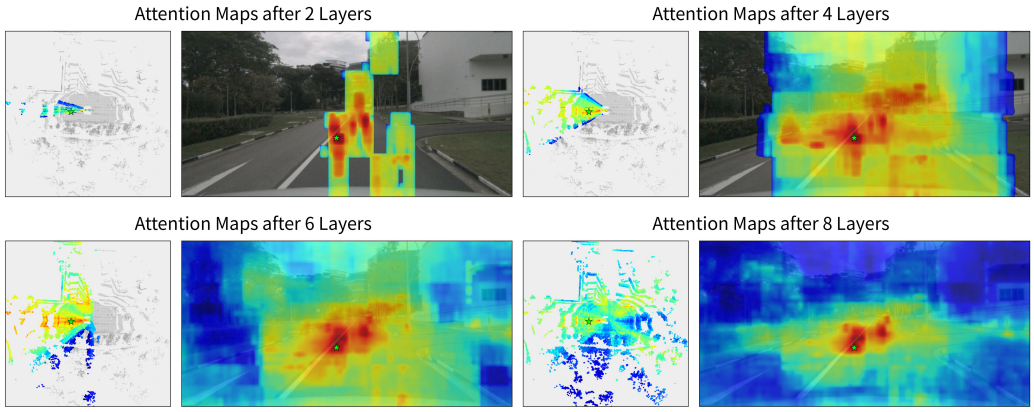
(a) X-direction (b) X-dir. shifted (c) Y-direction (d) Y-dir. shifted

Figure 9: **Query-key interaction visualization under different sorting strategies.** We select 5 query tokens ($\star$) and visualize the key tokens (colored dots) each query attends to within the sliding window. X-direction sorting (a) produces horizontally-elongated receptive fields, Y-direction sorting (c) produces vertically-elongated ones, and the shifted variants (b, d) change the spatial groupings.

These observations confirm that our sequence-first approach—combining 1D sliding-window attention with geometry-aware serialization and strategy swapping—achieves the spatial modeling capability of traditional 2D/3D operators while maintaining a purely sequential computation paradigm.



(a) Sample 1



(b) Sample 2

Figure 10: **Layer-wise attention map visualization.** We select a query token ($\star$) and visualize its attention distribution in BEV space (odd columns) and image space (even columns) after fusion layers 2, 4, 6, and 8 (top to bottom). The attention progressively expands from a local neighborhood to broad spatial coverage in both 3D and 2D space, demonstrating that 1D sliding-window attention with geometry-aware serialization recovers meaningful spatial neighborhoods equivalent to traditional 2D/3D operators. The consistent pattern across two different scenes confirms the generality of the observation.

C Additional Controlled Experiments

C.1 Geometry Robustness

We use fixed Small-2 EMA weights on the full nuScenes validation set and a shared perturbation seed of 0. Depth replacement and noise affect all depth consumers. Noise is applied after projection and before rasterization as $\widetilde{d} = \mathrm{clip}(d \exp(\epsilon), d_{\min}, d_{\max})$, with $\epsilon \sim \mathcal{N}(0, \sigma^2)$. For projected-point sparsification, the LiDAR encoder retains the complete point cloud, and only points constructing the projected depth/lifting coordinates are subsampled. Depth filling can supply missing pixels; otherwise the valid image-token count decreases. DVGT provides dense pseudo-GT depth. The BEVDepth-R50 replacement supplies monocular depth at inference.

Table 9: **Fixed-checkpoint geometry tests.** Calibration perturbations affect the geometric paths; opposite-camera reassignment corrupts correspondence. Deltas are relative to the reference.

Intervention	NDS	mAP	ΔNDS	ΔmAP
LiDAR-projected depth (reference)	74.759	73.284	—	—
DVGT dense pseudo-GT	74.712	73.210	-0.047	-0.074
BEVDepth-R50 monocular depth	74.705	73.265	-0.054	-0.019
Depth noise, $\sigma = 0.1$	74.848	73.421	$+0.089$	$+0.136$
Depth noise, $\sigma = 0.2$	74.810	73.355	$+0.051$	$+0.070$
Retain 50% of projected points	74.650	73.056	-0.109	-0.228
Calibration, $1^\circ + 5$ cm	74.008	72.059	-0.751	-1.225
Calibration, $5^\circ + 50$ cm	71.099	66.700	-3.659	-6.584
Opposite-camera assignment	69.326	62.998	-5.433	-10.287

The tested source and noise changes alter either metric by less than 0.14 points (Table 9). Retaining half of the projected points lowers NDS/mAP by 0.109/0.228. Large calibration and correspondence corruptions produce substantially larger losses, indicating that useful geometry remains important. These tests address the stated depth and projection interventions with a complete LiDAR encoder input.

D Limitations

Scope of evaluation. The primary evaluation covers nuScenes detection and Occ3D-nuScenes occupancy. Broader full-system evaluation on Waymo Open, Argoverse 2, and non-driving scenes remains future work. Operator differences are descriptive results from the reported runs.

Known camera–LiDAR calibration. Geometry-aware serialization assumes a known camera–LiDAR projection to assign 2D and 3D coordinates to each token. Under this assumption, fixed-checkpoint tests (Appendix C.1) show robustness to the depth source, depth noise, and projected-point sparsity, whereas large calibration errors cause substantial degradation. Operating without known calibration, as in recent pose-free models, remains untested.

Scaling beyond the studied range. Controlled width and depth studies show improvements within the evaluated nuScenes regime, while fusion depth saturates beyond eight layers. These results characterize a finite range. General scaling laws, emergence, and comparative scaling advantages require further evidence.

Efficiency and deployment scope. Optimized-system speed reflects both architecture and implementation. The manual-attention controls isolate a measurable benefit under their specific workload and software settings. Automotive hardware and complete sensor-to-output deployment remain to be evaluated.

E Broader Impacts

FusionNeXt targets 3D perception for autonomous driving, where improved accuracy and efficiency may contribute to safer and more accessible autonomous mobility. On the positive side, more efficient fusion backbones can reduce the compute and energy footprint of on-vehicle perception stacks, and the sequence-first design enables the broader 3D perception community to directly benefit from algorithmic and systems-level advances developed for LLMs/VLMs.

On the negative side, like most driving-perception methods, FusionNeXt may inherit distribution biases from the nuScenes and Occ3D datasets (*e.g.*, geography, weather, sensor configuration, traffic-agent demographics), and should not be deployed without additional out-of-distribution testing, safety validation, and domain-specific auditing. The underlying sequence-first architecture itself is domain-neutral and does not introduce misuse pathways beyond those common to 3D perception models.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction center on a packed sequence interface, competitive detection/occupancy performance, matched manual-attention efficiency, and operator compatibility. Section 4.4 and Appendix C specify the corresponding experimental scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Appendix D discusses dataset coverage, the known-calibration assumption, finite-range model results, statistical scope, and the efficiency measurement scope.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper contains no theoretical results, so no theorem assumptions or proofs are required.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [No]

Justification: Section 4 and Appendix A describe the architecture, losses, feature dimensions, training settings, and task-head paths; Appendix C reports the fixed-checkpoint geometry tests. Complete resolved configurations and cache definitions are not yet included in the current version.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The current version does not yet include public code, checkpoints, or reproduction scripts; they will be open-sourced later. Section 4 and Appendix A describe the method and training settings.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4 and Appendices A and C describe the datasets, architecture variants, detection/occupancy training settings, and the timing scope of the matched-attention and operator-replacement controls.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The paper reports descriptive differences between configurations and does not provide formal confidence intervals or significance tests for the main claims.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: Appendix A reports hardware, per-run time and memory for Small-2, and approximately 30 H100 GPU-days for the original ablations. Additional inference protocols specify their GPUs and measurement counts. A complete aggregate compute accounting for the added training experiments is not yet included in the current version.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses standard public benchmarks and routine model development, with no human-subjects component or other obvious ethics conflict. Nothing in the manuscript appears to violate the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

Justification: A Broader Impacts discussion is provided in Appendix E. It outlines positive impacts—contributions to safer, more compute-efficient on-vehicle perception and easier reuse of LLM/VLM systems advances by the 3D perception community—and negative considerations—dataset-induced distribution biases from nuScenes/Occ3D and the need for additional out-of-distribution testing, safety validation, and domain-specific auditing before deployment.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release a high-risk dual-use model or a scraped dataset, so dedicated release safeguards are not applicable.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We use public research datasets and codebases only for academic evaluation. nuScenes is cited and used under its official terms of use / CC BY-NC-SA 4.0 license. Occ3D-nuScenes is cited as a nuScenes-derived occupancy benchmark; its generated annotations/toolkit are released under the MIT License, while use of the underlying nuScenes data follows the nuScenes terms. MMDetection3D is cited and used under the Apache-2.0 License. We do not redistribute these third-party datasets or assets in this submission.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: The submission does not release a separate new asset package with documentation; the new model is described in the paper and appendix, but not shipped as a documented asset release here.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or human-subjects experiments.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No human-subjects study is involved, so IRB approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method does not use an LLM as an original component; LLM/VLM blocks are used as architectural inspiration rather than as a deployed language model in the method.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.