Revisiting Token Compression for Accelerating ViT-based Sparse Multi-View 3D Object Detectors

Supplementary Material

In this document, we first provide more implementation details (Sec. 1). Then, we present additional experiments (Sec. 2) and more visualization results (Sec. 3) to further validate the effectiveness of our method. Finally, we discuss the limitations of our approach and provide insights into potential future research directions (Sec. 4).

1. More Implementation Details

During training, on the nuScenes set, we adopt the default settings of StreamPETR [8], including the optimizer, learning rate, and data augmentation strategies. The model is trained for a total of 36 epochs on 4 $\times$ RTX 3090 GPUs. Specifically, the first 24 epochs are used to train the detector with the flexible patch embedding module [2], aiming to ensure the detector achieves normal performance under a single patch size. The subsequent 12 epochs focus on the training of our proposed three modules. On the Argoverse 2 set, the model is trained for a total of 12 epochs on 4 $\times$ RTX 3090 GPUs. The optimizer, learning rate, and data augmentation strategies follow the configurations in Far3D [4].

Since historical queries are required by the SPSS and IPS modules, SPSS adopts a predefined initial patch size (P_s) at the first frame as a safe fallback to ensure stable performance, and is activated in subsequent frames once temporal information becomes available. Meanwhile, IPS relies solely on current-frame image features to guide informative region selection in the absence of historical cues.

2. Additional Experiments

Comparison to 2D token compression methods. We further compare SEPatch3D with representative 2D token compression approaches, including SparseDETR [7], Cropr [1], and SViT [6], which perform token pruning at 2D tasks (e.g., classification and object detection). For fair comparison, we re-implement these methods on StreamPETR and apply token pruning at layers [6, 12, 18] with a fixed pruning ratio of 0.5, resulting in similar inference time. As shown in Tab. 1, although token pruning effectively reduces computation, it consistently leads to a noticeable performance degradation. In contrast, our method achieves up to +2 *pp* NDS and +1 *pp* mAP under similar latency. This highlights the advantage of patch-level enlargement over token-level pruning and quantitatively validates the core motivation of our design.

Table 1. Comparison to token pruning methods.

Methods	NDS (%) ↑	mAP (%) ↑	Inference Time (ms) ↓
StreamPETR	61.2	52.1	317.0
+ SparseDETR [7]	59.5	51.2	251.3
+ Cropr [1]	60.1	51.4	251.6
+ SViT [6]	60.3	51.1	250.0
Ours-fast	61.2	52.1	250.2

Quantitative validation of hard negatives in token pruning. To provide a quantitative analysis of how token pruning affects false positive predictions in sparse multi-view 3D detection, we compare StreamPETR equipped with Cropr [1] and our SEPatch3D framework in terms of mFP under a fixed recall level. As shown in Tab. 2, applying token pruning with Cropr (pruning ratio 0.5) leads to a substantial increase in mFP, from 35.4% to 53.2%, indicating that aggressive removal of background tokens significantly degrades the detector's ability to suppress hard negatives. In contrast, SEPatch3D maintains a much lower mFP (40.5%) while achieving comparable inference efficiency, demonstrating improved robustness to background-induced false alarms. We further visualize the effect in Fig. 1, where many background patches pruned by Cropr correspond to regions that provide useful negative evidence. Although these patches do not directly contribute to foreground object representations, they play an important role in constraining the decision boundary and suppressing hard negatives in sparse query-based 3D detectors. These results quantitatively support our design choice of retaining all background patches and avoiding token-level pruning.

Table 2. Comparison w.r.t. mFP @ recall=0.2.

Methods	Ratio	mFP (%) ↓
StreamPETR	0	35.4
+ Cropr	0.5	53.2
Ours-fast	0	40.5

Figure 1. Background patches contain useful hard negative information for 3D detection.

Impact of patch size diversity on detection performance in response to: "why do we only use two patch sizes instead of more?". As shown in Fig. 2, using two patch sizes (16, 18) achieves comparable accuracy to the single-size baseline. However, increasing the number of selectable patch sizes to three (16, 18, 20) or four (16, 18, 20, 22) leads to a gradual decrease in performance measured at patch size 16, with the degradation becoming more pronounced as patch size diversity increases. We analyze that introducing too many selectable patch sizes increases the complexity of

token distributions, which forces the detection head to generalize across multiple granularities but weakens optimization for any specific one. In this case, the detection head struggles to learn specific features well for individual patch configurations, leading to suboptimal performance. Therefore, adopting fewer of the two patch sizes not only allows effective dynamic adjustment of patches but also preserves strong specialization, making it an optimal design choice for our SEPatch3D.

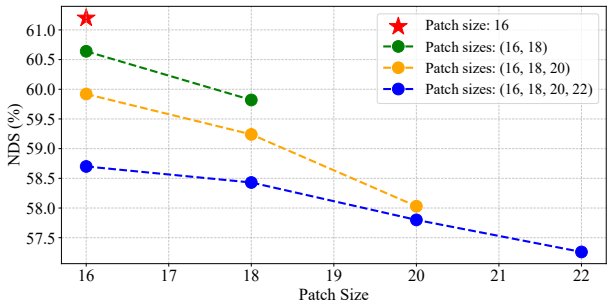

Figure 2. Impact of patch size diversity on detection performance.

Impact of spatiotemporal cues in the SPSS module. In Tab. 3, we analyze the contribution of spatial and temporal cues in the SPSS module. When either spatial or temporal cues are used individually, the inference time is notably reduced compared to the baseline, demonstrating that adaptive patch size selection effectively improves efficiency. However, both configurations suffer from performance degradation in NDS and mAP, indicating that relying on a single cue is insufficient for accurate object motion estimation. When both cues are jointly incorporated, the SPSS module achieves a good trade-off between accuracy and efficiency, further validating the effectiveness of the SPSS module.

Table 3. Ablations of spatiotemporal cues in the SPSS module.

Spat. Cues	Temp. Cues	NDS (%) ↑	mAP (%) ↑	Infe. Time (ms) ↓
		61.2	52.1	317.0
✓		59.7	51.2	195.5
	✓	59.5	51.3	191.3
✓	✓	60.3	51.6	194.3

Analysis of patch size distribution. We analyze the distribution of dynamically selected patch sizes to better understand the behavior of SPSS in practical scenarios. Fig. 3 illustrates the temporal evolution of selected patch sizes together with the average object depth in a representative nuScenes scene (ID: `9f1f69646d644e35be4fe0122a8b91ef`), where a clear positive correlation can be observed. Specifically, as the average object depth increases, SPSS tends to select larger patch sizes, indicating that distant-dominant scenes are processed with coarser patches to reduce redundant background computation. Conversely, scenes with closer objects are more likely to be assigned smaller patch sizes to preserve fine-grained semantic details. To further quantify this behavior, Tab. 4 reports the frequency of selected small (P_s) and large (P_l) patch sizes on the nuScenes validation set under 320×800 resolution. Both SEPatch3D-*fast* and SEPatch3D-*faster* variants exhibit a diverse usage of P_s and P_l, rather than collapsing to a single patch size. This confirms that SPSS performs effective dynamic selection instead of relying on a fixed configuration. Overall, these results demonstrate that the proposed spatiotemporal-aware patch size selection adapts to scene-level depth variations in a stable manner, validating its practical effectiveness and robustness across diverse driving scenarios.

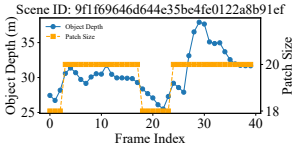

Figure 3. Temporal distribution.

Table 4. Number (frequency) of P_s and P_l under 320×800 resolution.

Methods	P_s	P_l
Ours-fast	2654 (44.1%)	3365 (55.9%)
Ours-faster	2565 (42.6%)	3454 (57.4%)

Impact of the adaptive patch selection strategy in the IPS module. As shown in Tab. 5, fixed-ratio Top-K selection performs worse than our adaptive strategy in both accuracy and latency. We attribute this to its inability to accommodate diverse scene conditions. When the keep ratio is too small, many patches that require refinement fail to be selected, leading to missing detail enhancement. Conversely, when the keep ratio is large, redundant patches are included, introducing unnecessary computation and slowing inference. These results demonstrate the necessity and effectiveness of our adaptive patch selection strategy.

Table 5. Ablations of adaptive patch selection strategy in the IPS module.

Strategies	Keep Ratio	NDS (%) ↑	mAP (%) ↑	Infe. Time (ms) ↓
- (Baseline)	-	61.2	52.1	317.0
Top-K	30%	59.6	51.0	198.4
	50%	59.8	51.1	202.7
	70%	59.9	51.3	207.6
Adaptive (Ours)	-	60.3	51.6	194.3

Impact of the IPS module placement. As shown in Tab. 6, placing the IPS module after the fine patches yields the best trade-off between accuracy and efficiency. Applying IPS only after coarse patches slightly improves accuracy but introduces additional computation, resulting in 14.9 ms increase in inference time. This is because selection performed on coarse patches must later be mapped back to the fine patches for enhancement, which inevitably brings in irrelevant fine patches that were not truly informative, thereby adding unnecessary computation. Applying IPS after both

fine and coarse patches further increases redundancy, offering negligible accuracy gains while significantly slowing down inference. These results indicate that informative patch selection is most effective when performed on fine patch features, where informative details are better preserved and redundant patches can be avoided. Hence, we adopt placing IPS after the fine patches as the default design.

Table 6. Ablations of the IPS module placement.

Position	NDS (%) ↑	mAP (%) ↑	Infe. Time (ms) ↓
After fine patches only	60.3	51.6	194.3
After coarse patches only	60.2	51.7	209.2
After both fine and coarse patches	60.4	51.7	213.8

3. More Visualization Results

In Fig. 4, we show the partial patch grids overlaid on the upper and left regions of the images. In the first row, the patch size increases from 20 to 22 when the ego-car turns into a new road segment. This adjustment occurs because new and farther objects (marked by the red box) appear in the new road segment. In contrast, the second row depicts the patch size decreasing from 22 to 20 as the objects gradually approach and occupy a larger region of the image. The visualizations indicate that SPSS dynamically adjusts patch sizes according to the spatiotemporal changes of objects, confirming its effectiveness.

In Fig. 5, we provide additional visualizations of the selected tokens and feature responses, further extending the analyses presented in the main manuscript to confirm that our IPS module consistently focuses on informative regions and our CGFE module further enhances the feature representations within these selected patch regions.

In Fig. 6, we compare the detection results between the baseline and our SEPatch3D-*faster*. The results indicate that our faster variant achieves comparable detection quality without notable missed detections, demonstrating the effectiveness of our approach in maintaining accuracy while improving efficiency.

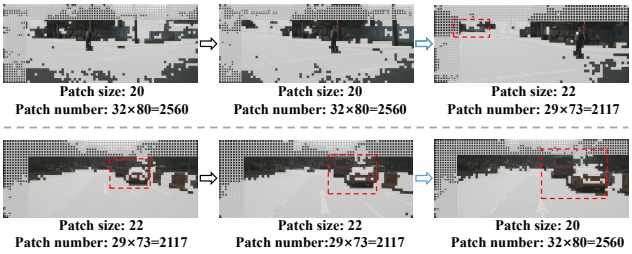

Figure 4. Visualization of the patch size selection process in the SPSS module. The blue arrows mark the frames where the patch size is adjusted. The red boxes indicate the regions exhibiting noticeable changes.

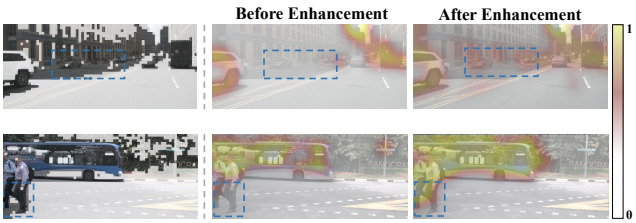

Figure 5. More visualizations of informative regions and feature responses.

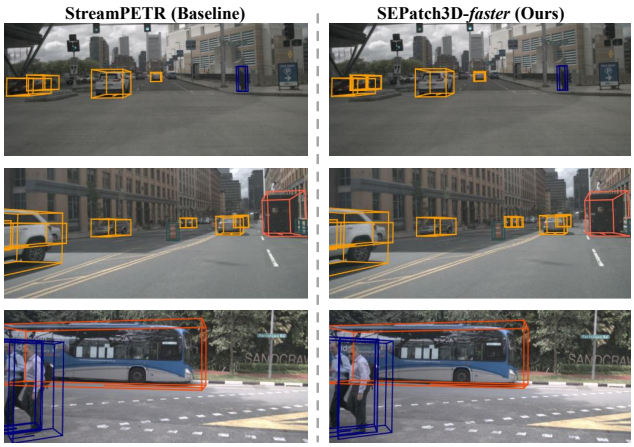

Figure 6. Visualization of detection results between StreamPETR (baseline) and our SEPatch3D-*faster*. Bounding box colors indicate different object categories.

4. Limitation and Future Work

While SEPatch3D effectively improves efficiency by adaptively adjusting patch sizes, the current patch size selection is still based on the pre-defined heuristic priors. In complex scenes with highly dynamic layouts, such heuristics may not always yield the optimal granularity. In future work, we plan to explore learnable patch size selection strategies that allow the network to automatically determine patch sizes in a data-driven manner. Moreover, we plan to further accelerate ViT-based multi-view 3D detectors by incorporating quantization or even binarization techniques [3], enabling real-time perception while maintaining competitive accuracy. Beyond 3D detection, we also plan to extend our SEPatch3D to other query-based spatiotemporal scene understanding tasks, such as 3D occupancy prediction [5] and world model [9].

References

[1] Benjamin Bergner, Christoph Lippert, and Aravindh Mahendran. Token cropr: Faster vits for quite a few tasks. In *CVPR*, 2025. 1

[2] Lucas Beyer, Pavel Izmailov, Alexander Kolesnikov, Mathilde Caron, Simon Kornblith, Xiaohua Zhai, Matthias Minderer, Michael Tschannen, Ibrahim Alabdulmohsin, and Filip Pavetic. Flexivit: One model for all patch sizes. In *CVPR*, 2023. 1

[3] Tian Gao, Yu Zhang, Zhiyuan Zhang, Huajun Liu, Kaijie Yin, Chengzhong Xu, and Hui Kong. Bhvit: Binarized hybrid vision transformer. In *CVPR*, 2025. 3

[4] Xiaohui Jiang, Shuailin Li, Yingfei Liu, Shihao Wang, Fan Jia, Tiancai Wang, Lijin Han, and Xiangyu Zhang. Far3d: Expanding the horizon for surround-view 3d object detection. In *AAAI*, 2024. 1

[5] Zhihao Li, Shanshan Zhang, and Jian Yang. Ashsr: Enhancing query-based occupancy prediction via anti-occlusion sampling and hard sample reweighting. *Neurocomputing*, 2026. 3

[6] Yifei Liu, Mathias Gehrig, Nico Messikommer, Marco Cannici, and Davide Scaramuzza. Revisiting token pruning for object detection and instance segmentation. In *WACV*, 2024. 1

[7] Byungseok Roh, JaeWoong Shin, Wuhyun Shin, and Saehoon Kim. Sparse detr: Efficient end-to-end object detection with learnable sparsity. *arXiv preprint arXiv:2111.14330*, 2021. 1

[8] Shihao Wang, Yingfei Liu, Tiancai Wang, Ying Li, and Xiangyu Zhang. Exploring object-centric temporal modeling for efficient multi-view 3d object detection. In *ICCV*, 2023. 1

[9] Yupeng Zheng, Pengxuan Yang, Zebin Xing, Qichao Zhang, Yuhang Zheng, Yinfeng Gao, Pengfei Li, Teng Zhang, Zhongpu Xia, Peng Jia, et al. World4drive: End-to-end autonomous driving via intention-aware physical latent world model. In *ICCV*, 2025. 3