SToRe3D: Sparse Token Relevance in ViTs for Efficient Multi-View 3D Object Detection

Supplementary Material

7. Implementation Details

We follow standard multi-view 3D detection settings on nuScenes using 6 cameras, synchronized frames, and known camera intrinsics and extrinsics. Backbones are ViT-based, and we evaluate both medium and large variants. There is no encoder after the backbone, and the decoder follows a DETR-style design with multi-scale features. We measure end-to-end latency at batch size 1 with the standard PyTorch profiler on a single RTX3090 GPU.

The ViT-L and ViT-B backbones follow EVA-02 [10], which is used for pretraining. The ViT-L backbone has 1024 embedding channels and 24 transformer layers with a mix of global and windowed attention, 16 attention heads, and a 0.3 drop path rate. The ViT-B backbone has 768 embedding channels and 12 transformer layers with a mix of global and windowed attention, 12 attention heads, and a 0.1 drop path rate. Both backbones use a patch size of 16. The nominal window size is 16 for ViT-L and 14 for ViT-B.

The detection head uses 6 decoder layers with 256-dimensional query embeddings. In Table 5, we list training hyperparameters and backbone configuration for all experiments, following EVA-02 [10] and StreamPETR [52] where possible. Hyperparameter and design choices specific to SToRe3D are described in the main text.

Base Config	**Value**	
Learning rate	4e-4	
Learning rate schedule	cosine decay	
Optimizer	AdamW	
Weight decay	0.01	
Batch size	16	
Training epochs	24	
Max detections	300	
ViT Config	**ViT-L**	**ViT-B**
Checkpoint	EVA-02 [10] Objects365	
Patch size	16	16
Embedding dimension	1024	768
Window attention layers	16	6
Global attention layers	8	6
Attention heads	16	12
Window size	16	14
Drop path	0.3	0.1

Table 5. Model configuration and training hyperparameters used in all experiments.

Pruning in the backbone occurs in the layer preceding a subset of global-attention blocks. For ViT-L with global attention at layers $[3, 6, 9, 12, 15, 18, 21, 24]$, pruning is applied before the later global layers. For ViT-B with global attention at layers $[2, 4, 6, 8, 10, 12]$, pruning is similarly applied before a subset of these layers.

The pruning schedule for the number of tokens kept in each layer follows a quadratic schedule,

$$\mathrm{LKR}_l = \mathrm{TKR} + (1 - \mathrm{TKR})(1 - l/L)^2, \tag{9}$$

where the layer keep ratio (LKR) is the fraction of tokens processed at layer l, the total keep ratio (TKR) is the final target fraction of tokens processed across the backbone, and L is the total number of layers. Table 6 lists the resulting per-layer keep ratios and the mean keep ratio (MKR), which approximates the average fraction of tokens processed through the backbone.

ViT-L Models	**TKR**	**MKR**	**LKR$_6$**	**LKR$_8$**	**LKR$_{10}$**	**LKR$_{12}$**
ToC3D-fast	0.5	0.70	0.70	0.50	0.50	0.50
SToRe3D-1/2	0.5	0.76	0.63	0.53	0.50	1.00
ToC3D-faster	0.3	0.58	0.50	0.40	0.30	0.30
SToRe3D-1/3	0.3	0.67	0.48	0.32	0.30	1.00
SToRe3D-1/10	0.1	0.58	0.33	0.16	0.10	1.00
ViT-B Models	**TKR**	**MKR**	**LKR$_9$**	**LKR$_{15}$**	**LKR$_{21}$**	**LKR$_{24}$**
ToC3D-fast	0.5	0.70	0.70	0.50	0.50	0.50
SToRe3D-1/2	0.5	0.74	0.63	0.53	0.50	1.00
ToC3D-faster	0.3	0.58	0.50	0.40	0.30	0.30
SToRe3D-1/3	0.3	0.64	0.48	0.34	0.30	1.00
SToRe3D-1/10	0.1	0.53	0.33	0.16	0.10	1.00

Table 6. Token pruning schedules for SToRe3D model variants and ToC3D [61]. TKR is the total keep ratio, MKR is the mean keep ratio across layers, and LKR_ℓ is the layer keep ratio at selected global-attention layers.

8. Profiling Analysis

We now analyze latency sensitivity to image tokens and object queries in isolation. Starting from the dense baseline with 900 object queries and 127,500 image tokens across all camera views, we systematically subsample each axis. We reduce the number of object queries by 50% and 90%, and measure the resulting latency of the ViT-L backbone and the detection decoder. We then repeat the experiment by reducing the number of image tokens by 50% and 90% while keeping the number of queries fixed. Figure 7 shows the measured latencies.

From these experiments, we estimate simple sensitivities $\Delta t/\Delta i$ and $\Delta t/\Delta o$, where t is the latency, i is the number of image tokens, and o is the number of object queries. For the decoder, the sensitivity to image tokens is modest, with $\Delta t/\Delta i \approx 7\%$, while the sensitivity to object queries is much higher, with $\Delta t/\Delta o \approx 50\%$. For the ViT-L backbone, the sensitivity to image tokens is dominant, with $\Delta t/\Delta i \approx 95\%$, and the sensitivity to object queries is effectively zero, since queries are not used in the backbone.

These results support our design choice. Pruning image tokens primarily reduces backbone latency, while pruning object queries primarily reduces decoder latency. Joint sparsity over both axes is therefore necessary to reach the strongest latency improvements.

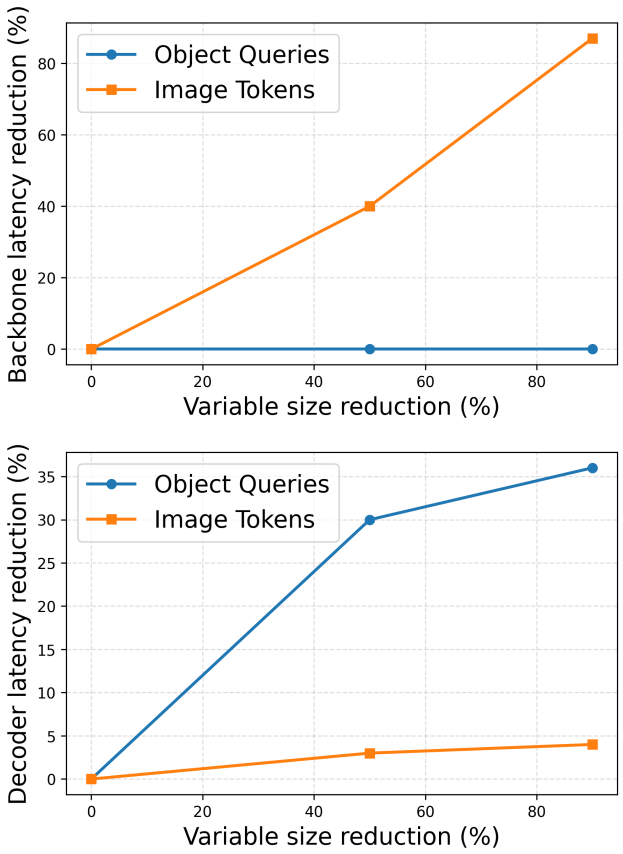


Figure 7. Latency sensitivity of the ViT backbone and the detection decoder when varying the number of image tokens and object queries independently. The backbone is almost entirely driven by the number of image tokens, while the decoder is much more sensitive to the number of object queries.

9. Additional Metrics

For the planning-relevance metrics introduced in the main paper, an agent is labeled relevant if the closest distance d_C between its swept corridor and the ego vehicle's swept corridor is below a buffer threshold d_{RM}. To choose d_{RM}, we first compute the empirical distribution of the closest ego–agent distances d_C across the entire nuScenes dataset. The cumulative distribution is shown in Figure 8. We select d_{RM} as the 10th percentile of this distribution, which corresponds to approximately 10% of agents being labeled relevant. This yields $d_{RM} = 1.2$ m over a 5-second horizon, which in turn leads to roughly 3 relevant agents per frame on average and a maximum of 31 relevant agents.

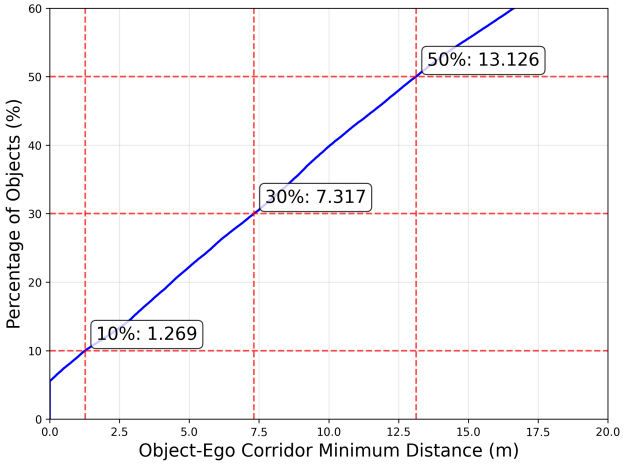


Figure 8. Cumulative distribution of closest ego–agent distances on nuScenes, used to select the relevance buffer for our nuScenes-R benchmark.

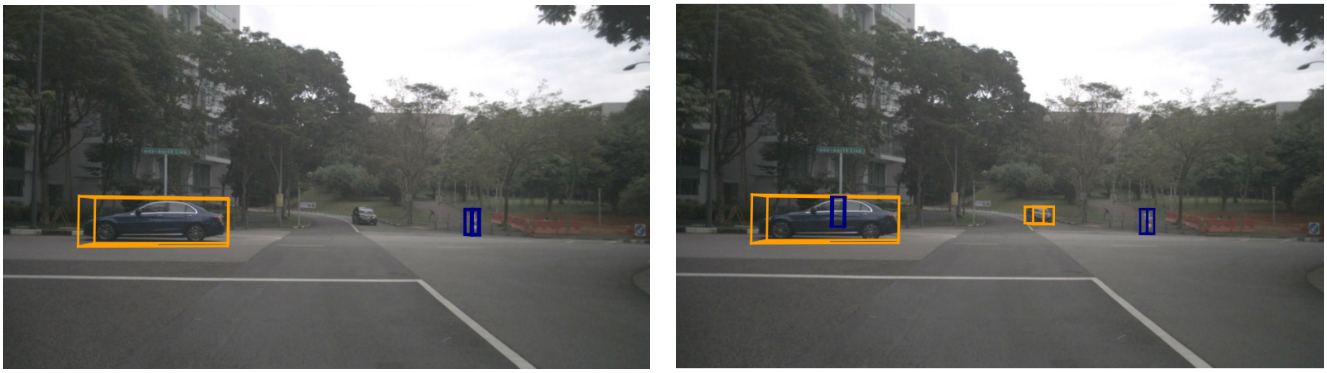
10. Additional Qualitative Results

Figure 9 provides additional qualitative comparisons between SToRe3D and the baseline StreamPETR model. We highlight three representative cases where highly relevant objects are missed by the baseline ResNet-50 model but detected by our similar-latency variant SToRe3D-1/10-ViT-B.

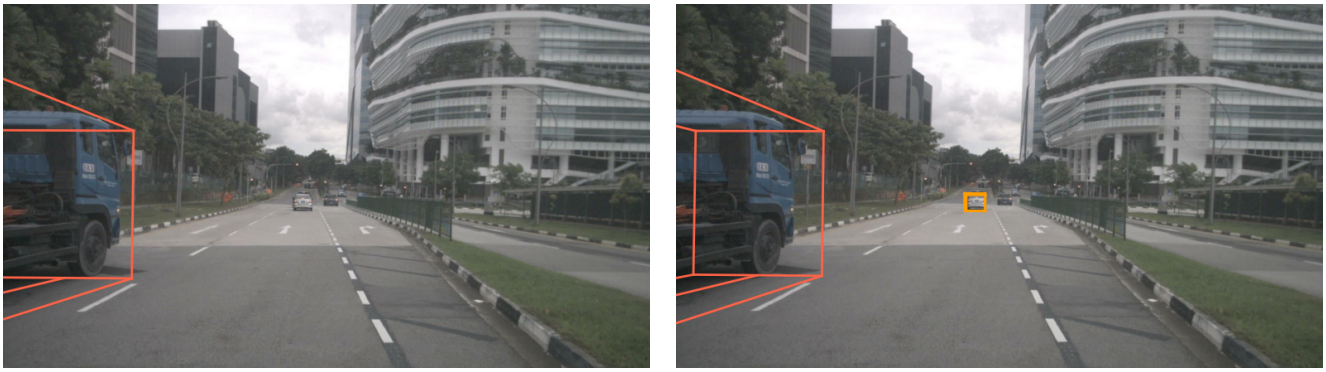
In all three scenes, the highlighted objects interact with the ego vehicle and have predicted future trajectories that pass close to the ego path. Examples include a car in the oncoming lane at an intersection (top row), a car stopped ahead near an upcoming intersection (middle row), and a nearby pedestrian about to cross the road (bottom row). These cases illustrate that the higher-capacity ViT backbone with relevance-aware pruning detects planning-critical objects that a lower-capacity dense backbone misses, while operating at comparable latency.

11. Limitations and Future Work

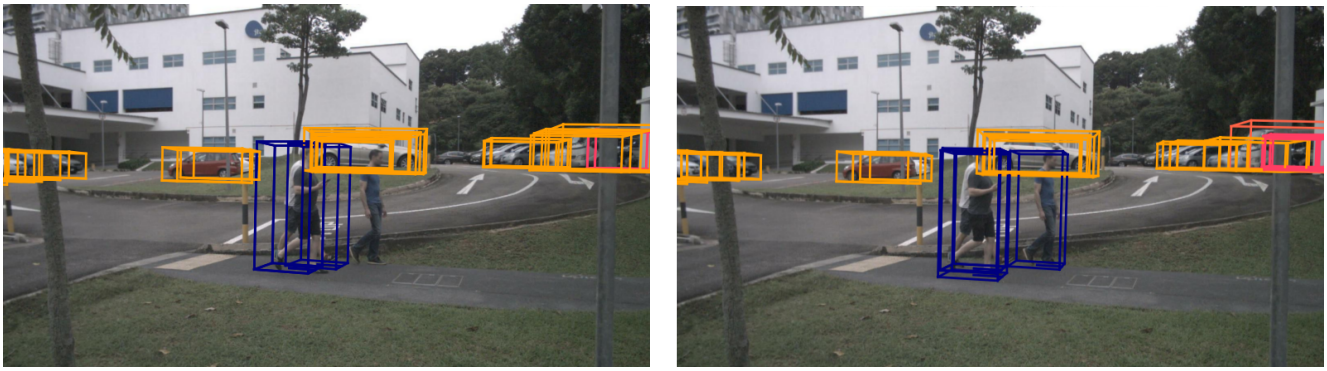
While SToRe3D achieves strong accuracy–latency trade-offs, it has several limitations. First, our evaluation is restricted to the nuScenes dataset and camera-only multi-view 3D detection. The relevance heads and pruning schedules are



(a) Front view of an oncoming car that is missed by the baseline detector but correctly detected by SToRe3D.



(b) Front view of a lead vehicle ahead that is missed by the baseline but detected by SToRe3D.



(c) Left side view of a nearby pedestrian that is missed by the baseline but detected by SToRe3D.

Figure 9. Additional qualitative examples of false negatives for the baseline StreamPETR-R50 (left in each pair), where SToRe3D-1/10-ViT-B at a similar latency (right in each pair) correctly detects the planning-critical objects.

tuned for this setting. Performance and sparsity behavior on other datasets, sensing setups (for example, LiDAR or radar fusion), and driving domains are not evaluated.

Second, the approach still relies on relatively heavy ViT backbones and multi-scale DETR-style decoding. Although relevance-aware sparsity yields substantial speedups, absolute latency and memory usage may remain challenging for some embedded or low-power deployments, especially at higher input resolutions.

Finally, the nuScenes-R protocol for planning-critical evaluation uses privileged future information to define interaction corridors. This is reasonable for offline benchmarking, but the corridor parameters and labeling procedure may not perfectly match the objectives of a downstream planner in a closed-loop system. In addition, we focus only on open-loop detection and do not assess downstream effects on full autonomous driving stacks in closed-loop simulation or real-world testing. Studying how relevance-aware sparsity interacts with planning and control policies, along with joint training of SToRe3D for end-to-end driving are important direction for future work.