VASC: VALUE-AWARE SPARSE ATTENTION WITH CROSS-LAYER MEMORY FOR EFFICIENT 3D RECONSTRUCTION

Anonymous authors
Paper under double-blind review



Figure 1: **Reconstruction quality with sparse attention.** Left: RGB references and aligned stride-10 reconstructions; red boxes mark shared regions. For each scene, FasterVGGT and VASC use the same sparsity. RGB and rendered views are not pixel-aligned. Right: dataset-level metrics normalized to dense VGGT. Higher values indicate lower errors or higher normal consistency (NC).

ABSTRACT

Feed-forward 3D vision models such as VGGT have achieved remarkable progress, unifying camera estimation and dense scene reconstruction in a single pass. However, their quadratic global attention makes long image sequences expensive, while existing sparse methods may favor highly attended yet value-redundant regions. To address these limitations, we introduce **VASC**, a training-free sparse attention method combining **value-aware block selection** and **execution-aware cross-layer memory**. Our value-aware block selection integrates pooled query–key relevance with neighboring value contrast, reducing redundancy while preserving query-relevant and distinctive content. Cross-layer memory tracks unserved demand across layers and updates this state according to actual execution, enabling previously underserved blocks to compete under a fixed computation budget. Experiments on 7Scenes and NeuralRGB-D with VGGT and π^3 demonstrate improved pose estimation and reconstruction quality compared with FasterVGGT, together with up to $2.29\times$ faster inference than dense VGGT. Code and evaluation scripts will be made available.

1 INTRODUCTION

Recovering three-dimensional geometry from images is increasingly becoming a problem of learned, feed-forward inference. Building on transformer representations (Vaswani et al., 2017; Dosovitskiy et al., 2021), DUSt3R casts pairwise geometry as pointmap regression, while MASt3R connects this representation to dense matching (Wang et al., 2024; Leroy et al., 2024). VGGT jointly predicts cameras, depth, point maps, and tracks, and π^3 develops permutation-equivariant geometry

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

prediction (Wang et al., 2025a; 2026b). Scaling these models requires both accurate predictions and efficient processing of long inputs.

Global cross-view attention is a central obstacle to this scalability: its interactions grow quadratically with the number of visual tokens. Efficient exact-attention kernels reduce memory traffic and improve hardware utilization (Dao et al., 2022; Dao, 2024), but do not decide which interactions can be omitted. Recurrent and streaming models address long inputs through spatial memory, persistent state, or causal processing (Wang & Agapito, 2025; Wang et al., 2025b; Wu et al., 2025b; Zhuo et al., 2026); TTT3R and InfiniteVGGT further study long-term memory (Chen et al., 2026b; Yuan et al., 2026). We consider a complementary setting: accelerating an existing backbone's global attention without retraining or replacing its prediction architecture.

Training-free acceleration offers several ways to reduce this cost. Token merging consolidates similar representations (Bolya et al., 2022; Shen et al., 2025); sparse global attention retains selected block interactions (Wang et al., 2026a); layer scheduling changes where cross-view computation is spent (Sun et al., 2026) (see also geometry quantization and video-generation caching; Feng et al., 2026; Wu et al., 2025a). For sparse execution, a key question remains: which interactions should receive a fixed computation budget? Selection must protect geometry prediction without an overhead that erases the savings.

Pooled query–key scores provide a natural relevance estimate, but sparse selection faces two linked limitations in content and execution history. Attention weights describe query-dependent relevance, whereas value vectors supply the aggregated content, a distinction also explored in value-aware approximation (Gupta & Berant, 2021). Strong attention can coincide with locally similar values, so relevance alone does not indicate how distinct the selected content is. Under a tight budget, relevant but distinctive interactions may consequently be omitted, as illustrated by the attention ranking in Figure 3. This limitation also extends across network depth: independent selection at each layer does not account for whether previously assigned demand was actually served. Historical scores and reusable sparse indices provide useful context (Zhang et al., 2023; Bai et al., 2026; Han et al., 2026), but do not by themselves encode the gap between requested and executed computation. The challenge is therefore to account for both value content and unmet computational demand while preserving the efficiency of fixed-budget sparse attention. Sparse attention must account for both what deserves computation and whether previously assigned demand was actually served.

We introduce **VASC**, **Value-Aware Sparse Attention with Cross-Layer Memory**, to address these two aspects jointly. Value-aware selection combines pooled query–key relevance with the RMS contrast of adjacent pooled value blocks, distinguishing content at native sparse-block granularity. Cross-layer memory maintains one scalar balance per block pair, accumulating unserved demand and subtracting actual execution. This computational deficit participates in subsequent ranking alongside current demand. Both components preserve per-query capacity, the native sparse kernel, special-token handling, and geometry heads.

Experiments on 7Scenes and NeuralRGB-D evaluate VGGT and π^3 across sparsity levels and input strides. VASC improves all six pose and reconstruction metrics over the corresponding sparse baselines at S75 in the main comparisons, while retaining up to $2.29\times$ speedup over dense VGGT. Figures 1 and 2 summarize quality and efficiency. Component ablations, denser inputs, ScanNet transfer, and scheduler adaptations further examine effectiveness and generality.

Our contributions are fourfold:

- **Observation.** We illustrate a mismatch between attention priority and local value variation in a geometry transformer, motivating their complementary use.
- **Value-aware selection.** We combine pooled query–key relevance with adjacent-value contrast for training-free selection of distinctive, query-relevant blocks.
- **Cross-layer memory.** An execution-updated pair-level state carries unserved demand across layers to compete with current demand under an unchanged budget.
- **Effectiveness and efficiency.** Across two geometry backbones, VASC improves sparse reconstruction and pose estimation, with controlled ablations and up to $2.29\times$ faster inference than dense VGGT.

2 Related Work

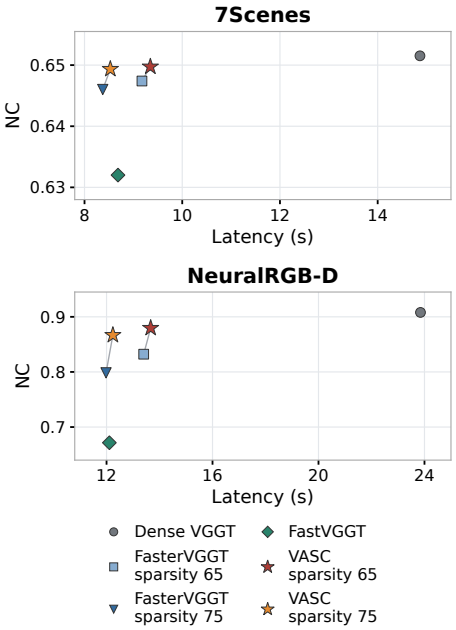

Figure 2: **Quality–latency comparison.** VGGT at stride 10. Higher NC and lower latency are better.

Feed-forward and streaming visual geometry. DUSt3R predicts pairwise pointmaps and aligns them across views; MASt3R augments this geometry-based formulation with dense matching (Wang et al., 2024; Leroy et al., 2024). VGGT and π^3 advance joint multi-view geometry prediction through global attention (Wang et al., 2025a; 2026b). Spann3R, CUT3R, and Point3R instead maintain spatial or recurrent state for incremental reconstruction (Wang & Agapito, 2025; Wang et al., 2025b; Wu et al., 2025b). StreamVGGT adapts geometry prediction to streaming inputs, while TTT3R and InfiniteVGGT address the maintenance of long-term state (Zhuo et al., 2026; Chen et al., 2026b; Yuan et al., 2026). VASC targets interaction selection inside the global attention of pretrained, multi-view geometry backbones.

Efficient attention and token reduction. FlashAttention improves exact-attention execution through IO-aware computation and parallelism (Dao et al., 2022; Dao, 2024). Structured sparsity reduces the interaction set through local, factorized, or mixed connectivity (Child et al., 2019; Beltagy et al., 2020; Zaheer et al., 2020). DynamicViT, EViT, and TokenLearner learn to prune, reorganize, or aggregate visual tokens (Rao et al., 2021; Liang et al., 2022; Ryoo et al., 2021); ToMe merges similar representations (Bolya et al., 2022). For language models, MInference identifies sparse attention patterns for prefill and Quest makes sparse access query-dependent (Jiang et al., 2024; Tang et al., 2024). These works motivate efficient selection but operate at different training, token, or inference interfaces. VASC preserves child-block identity and changes the support passed to the existing sparse attention kernel.

Training-free acceleration of geometry models. FastVGGT adapts token merging to visual geometry, FasterVGGT uses pooled-QK proposals and block-sparse execution, and AVGGT changes the allocation of global-attention computation (Shen et al., 2025; Wang et al., 2026a; Sun et al., 2026). QuantVGGT addresses model compression through post-training quantization (Feng et al., 2026). GHOST exploits predicted geometry for online KV-token eviction in streaming reconstruction (Chen et al., 2026a). VASC focuses on selecting value-sensitive block interactions and accounting for their execution across depth.

Value-aware and history-aware selection. Value-aware Approximate Attention incorporates values when approximating attention outputs (Gupta & Berant, 2021). VASC uses adjacent-value contrast to refine pooled relevance at low cost. H_2O and SnapKV compress language-model KV state using attention-derived importance, and StreamingLLM preserves attention sinks with a recent-context window (Zhang et al., 2023; Li et al., 2024; Xiao et al., 2024). IndexCache reuses sparse indices across layers, while EvoSparse exploits token-importance dynamics (Bai et al., 2026; Han et al., 2026). Our memory instead records the gap between assigned demand and actual execution. Its nonnegative recurrence follows classical queue accounting (Lindley, 1952; Neely, 2010), providing an execution-aware summary of allocation history.

3 Analyzing Attention and Value in VGGT

Reconstruction under attention-only selection. Which connections should be preserved when only a small fraction of attention can be retained? We investigate this question through reconstruction experiments on scenes from NeuralRGB-D and 7Scenes. Starting from pretrained VGGT, we compare dense attention with an attention-only setting that retains the highest-ranked 25% of patch-key connections for each patch query and head in the global attention layers. The retained weights are renormalized, while frame attention, special-token handling, and model parameters remain unchanged. As shown in Figure 3(a), retaining the strongest attention responses alone can still degrade reconstruction quality. This motivates examining both the relevance and the value content of selected connections.

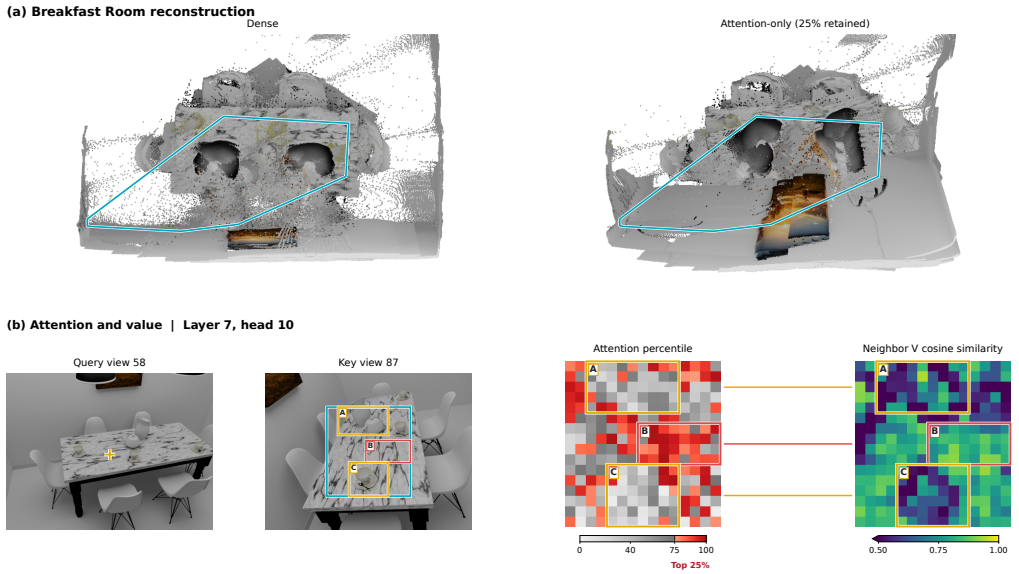

Figure 3: **Attention priority and local value variation.** (a) Dense and attention-only reconstruction with 25% of connections retained. (b) Attention and neighboring-value similarity reveal different spatial patterns; lower similarity indicates greater directional variation.

Attention relevance and value variation. We compare attention priorities with local value variation, visualized using each value vector's mean cosine similarity to its four spatial neighbors. Lower similarity indicates greater directional variation. Figure 3(b) illustrates that the two maps need not agree: highly attended connections can carry locally similar values, while lower-ranked connections can contain more distinct content. Attention-only selection can therefore favor redundant content under a tight budget. Visualization details are given in Appendix B.

Competition under a fixed budget. Retaining a connection consumes capacity that could otherwise serve another candidate. A high attention rank therefore does not, by itself, indicate how much distinct content is preserved by that choice. Conversely, falling below the selection threshold does not imply irrelevance to the query. The resulting trade-off concerns not only how much attention is retained, but also which value content is represented within the available capacity.

Implications for sparse selection. In an attention output $o_i = \sum_j a_{ij} v_j$, the weights describe query–key relevance, while the values supply the aggregated content. Ranking solely by a_{ij} does not account for similarity among values. This motivates **joint QKV-informed ranking**: query–key relevance identifies relevant interactions, and value variation helps distinguish their content. Such a ranking expresses which interactions deserve computation; the fixed budget still determines which can actually be executed. This distinction between demand and execution underlies VASC's selection and feedback mechanism.

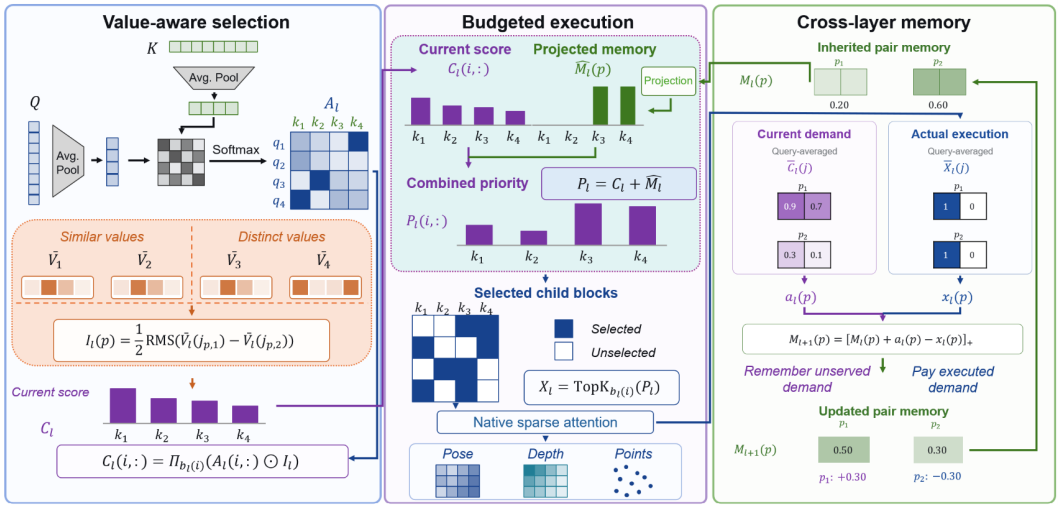

Figure 4: **VASC overview.** Value contrast and cross-layer memory jointly guide block selection within a fixed budget. Actual execution updates memory for the next layer.

4 METHOD

VASC combines query–key relevance and value contrast into current demand, allocates a fixed execution budget, and carries their accumulated mismatch across layers (Figure 4). Its low-cost, training-free realization preserves the native sparse attention kernel and geometry heads.

Blocks and capacity. At layer l, i indexes a query block and j a key/value child block, the unit selected for sparse execution. Each query has capacity $b_l(i)$. Adjacent key/value children form pairs $\mathcal{C}(p)$ with fixed membership across layers; $p(j)$ denotes the pair containing child j. Pairs share a value-contrast score and a memory state, but their children remain separate execution candidates. Batch and head indices are omitted.

4.1 VALUE-AWARE BLOCK SELECTION

Section 3 shows that attention-only selection can degrade reconstruction. Attention priority also differs from local value variation: highly attended regions can contain similar values, while lower-ranked regions carry more distinct content. This motivates combining query relevance with value contrast.

We express this joint QKV-informed scoring as

$$C_l = \mathcal{F}(Q_l, K_l, V_l; b_l). \tag{1}$$

Here $C_l(i,j)$ is **normalized computation demand** for interaction (i,j). QK measures relevance to the query; value contrast distinguishes the content from nearby alternatives.

A low-cost realization. VASC instantiates $\mathcal{F}$ with average-pooled block features. Pooled queries and keys provide relevance:

$$A_l(i,j) = \mathrm{Softmax}_j\left(\frac{\bar{Q}_l(i)^\top \bar{K}_l(j)}{\sqrt{d}}\right). \tag{2}$$

For each adjacent pair, RMS contrast between pooled values supplies a shared content signal. We use it to modulate the relevance of both children:

$$\begin{aligned} I_l(p) &= \tfrac{1}{2}\,\mathrm{RMS}\big(\bar{V}_l(j_{p,1}) - \bar{V}_l(j_{p,2})\big)\,, \\ C_l(i,:) &= \Pi_{b_l(i)}\big(A_l(i,:) \odot I_l(p(:))\big)\,. \end{aligned} \tag{3}$$

Budget rescaling and clipping by Π_b give $0 \le C_l(i,j) \le 1$ and $\sum_j C_l(i,j) \le b_l(i)$, making demand comparable to unit execution. Contrast is shared across queries; relevance remains query-specific. RMS captures these value differences with a low-cost blockwise reduction.

4.2 BUDGETED EXECUTION

From demand to execution. Value-aware scores describe which interactions deserve computation; the fixed budget determines which are executed. To account for both current demand and past service, we combine C_l with a compact memory M_l of accumulated demand–execution imbalance:

$$X_l = \mathcal{G}(C_l, M_l; b_l), \qquad \sum_j X_l(i,j) = b_l(i). \tag{4}$$

Each $X_l(i,j) \in \{0,1\}$ supplies one unit of execution when selected. Memory redistributes execution within this fixed capacity.

A budget-preserving realization. VASC adds a bounded copy $\widetilde{M}_l$ of pair memory to current demand, then selects children within each query's capacity:

$$\begin{aligned} P_l(i,j) &= C_l(i,j) + \widetilde{M}_l(p(j)), \\ X_l(i,:) &= \mathrm{TopKMask}_{b_l(i)}(P_l(i,:)) \,. \end{aligned} \tag{5}$$

Each query selects its own children from priority P_l and passes the binary mask X_l directly to the native sparse attention kernel. We preserve the host's special-token handling: camera and register queries use dense attention, and special-only key blocks remain always kept. These always-kept blocks do not compete for the patch-child budget above.

4.3 CROSS-LAYER MEMORY

Refine allocation through past service. Current demand evaluates present content; memory adjusts allocation using accumulated demand–execution imbalance at the block-pair level. This gives previously underserved blocks another opportunity to compete for the fixed budget. In particular, near the selection cutoff, past service can distinguish candidates with similar current demands.

A compact update. We apply the same averaging over query blocks and each pair's children to C_l and X_l, obtaining pair demand $a_l(p)$ and service $x_l(p)$, both in $[0,1]$. Starting from zero, we retain their accumulated nonnegative imbalance:

$$M_{l+1}(p) = [M_l(p) + a_l(p) - x_l(p)]_+, \qquad M_1(p) = 0. \tag{6}$$

Demand adds to the balance; service subtracts from it and can reduce earlier deficits. Clipping prevents negative memory, but receiving some computation need not clear the balance. The resulting pair-level state summarizes demand and service across queries and paired children.

One scalar per pair, batch entry, and head is retained for updating; its bounded copy informs selection. Each layer uses its own Q, K, and V to compute attention on the selected support.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Datasets and metrics. We evaluate VGGT and π^3 on all 18 test sequences of 7Scenes (Shotton et al., 2013) and all nine NeuralRGB-D (NRGBD) sequences (Azinović et al., 2022) at strides $s \in \{5, 10\}$, retaining every s-th frame without a frame cap. Generalization uses 50 ScanNet (Dai et al., 2017) scenes with 100 frames each: the first frame and 99 uniformly sampled remaining frames. On 7Scenes and NRGBD, we report absolute trajectory error (ATE), translational and rotational relative pose error (RPE-t/RPE-r), reconstruction accuracy (Acc), completeness (Comp), and normal consistency (NC). Translation and reconstruction distances are in meters; RPE-r is in degrees. Lower is better except for NC. Results are sequence macro averages, displayed to three decimal places. Boldface uses unrounded values.

Baselines and implementation. S55–S85 denote target sparsity of 55–85% of patch-child interactions; realized sparsity accounts for block rounding and always-kept support. Within each backbone, baseline/VASC comparisons share checkpoints, inputs, evaluators, geometry heads, and sparse kernels. We compare with FasterVGGT (Wang et al., 2026a) and our π^3 adaptation, FasterPi3; tables label both FasterVGGT. FastVGGT (Shen et al., 2025) uses its official reconstruction pipeline at merge ratio 0.9. Timing covers full VGGT forward passes, including geometry heads, at input size 518 and stride 10 on A6000 and RTX 4090. We average per-sequence medians of three passes after one warm-up, excluding loading and evaluation; comparisons use the same physical GPU.

5.2 Main Results

Table 1: **Comparison at strides 10 and 5.** Sequence means on 7Scenes (18 sequences) and NRGBD (9). Bold marks the better unrounded value within each FasterVGGT–VASC pair at the same backbone, stride, and sparsity. FasterVGGT (Wang et al., 2026a) is adapted to π^3 in its rows. Dense is the unpruned reference; OOM: out of memory.

Method	Sparsity (%)	Stride	7Scenes						NeuralRGB-D					
			ATE↓	RPE-t↓	RPE-r↓	Acc↓	Comp↓	NC↑	ATE↓	RPE-t↓	RPE-r↓	Acc↓	Comp↓	NC↑
VGGT														
Dense	0	10	0.069	0.031	0.859	0.017	0.028	0.652	0.030	0.016	0.179	0.014	0.015	0.908
FasterVGGT	65	10	0.081	0.035	0.960	**0.017**	0.029	0.647	0.059	0.026	0.427	0.034	0.024	0.832
VASC	65	10	**0.073**	**0.033**	**0.911**	0.018	**0.028**	**0.650**	**0.044**	**0.019**	**0.239**	**0.018**	**0.017**	**0.880**
FasterVGGT	75	10	0.084	0.038	1.080	0.018	0.028	0.646	0.073	0.030	0.503	0.041	0.027	0.798
VASC	75	10	**0.074**	**0.034**	**0.963**	**0.018**	**0.028**	**0.649**	**0.047**	**0.020**	**0.262**	**0.019**	**0.018**	**0.867**
Dense	0	5	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM
FasterVGGT	65	5	0.081	0.024	0.682	**0.016**	0.027	0.610	0.066	0.024	0.383	**0.030**	**0.021**	0.817
VASC	65	5	**0.073**	**0.023**	**0.623**	0.016	**0.027**	**0.612**	**0.062**	**0.018**	**0.263**	0.033	0.022	**0.849**
FasterVGGT	75	5	0.084	0.027	0.757	0.017	0.027	0.608	0.076	0.027	0.446	0.031	0.022	0.787
VASC	75	5	**0.075**	**0.024**	**0.670**	**0.017**	**0.027**	**0.611**	**0.054**	**0.019**	**0.266**	**0.028**	**0.019**	**0.840**
π^3														
Dense	0	10	0.057	0.024	0.769	0.020	0.025	0.810	0.025	0.012	0.133	0.019	0.018	0.959
FasterVGGT	65	10	0.064	0.026	0.835	0.029	0.026	0.719	0.063	0.019	0.252	0.050	0.031	0.794
VASC	65	10	**0.060**	**0.025**	**0.789**	**0.022**	**0.025**	**0.786**	**0.046**	**0.015**	**0.173**	**0.029**	**0.024**	**0.900**
FasterVGGT	75	10	0.066	0.027	0.855	0.031	0.027	0.709	0.074	0.020	0.277	0.056	0.033	0.779
VASC	75	10	**0.061**	**0.026**	**0.807**	**0.022**	**0.025**	**0.779**	**0.058**	**0.017**	**0.196**	**0.034**	**0.026**	**0.881**
Dense	0	5	0.057	0.016	0.507	0.020	0.025	0.811	0.025	0.010	0.108	0.019	0.018	0.959
FasterVGGT	65	5	0.064	0.017	0.535	0.029	0.026	0.719	0.060	0.015	0.196	0.050	0.030	0.793
VASC	65	5	**0.060**	**0.016**	**0.512**	**0.022**	**0.024**	**0.788**	**0.047**	**0.012**	**0.136**	**0.029**	**0.023**	**0.901**
FasterVGGT	75	5	0.066	0.018	0.544	0.030	0.026	0.709	0.069	0.016	0.213	0.055	0.032	0.779
VASC	75	5	**0.061**	**0.017**	**0.520**	**0.022**	**0.025**	**0.780**	**0.056**	**0.013**	**0.152**	**0.033**	**0.025**	**0.881**

Reconstruction and camera estimation. Table 1 compares S65/S75 at strides 10 and 5 on both backbones. VASC improves all three pose metrics over FasterVGGT and FasterPi3 in every setting, and all six metrics at S75. On π^3, all six improve at both sparsities and strides.

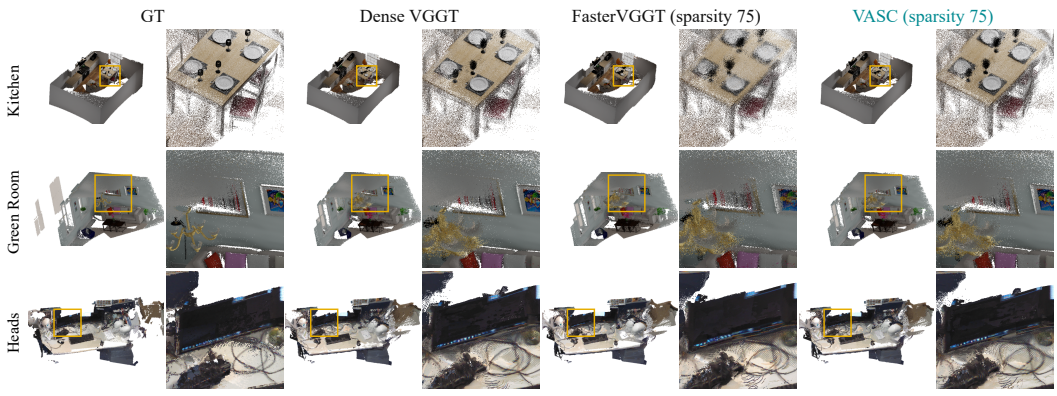

Figure 5: **Qualitative reconstruction at sparsity 75.** Kitchen and Green Room from NRGBD, and Heads from 7Scenes, using full stride-10 sequences. Each method shows an overview and a detail crop; yellow boxes mark shared regions. Cameras and crop bounds are shared after rigid alignment to GT.

Figure 5 compares local structure around tableware, wall decorations, and monitor edges, complementing the broader reconstruction comparisons in Figure 1.

Denser inputs and generalization. Gains persist with denser inputs: at stride 5 and S75 on NRGBD, VASC improves VGGT ATE and Comp, and π^3 Acc and NC (Table 1). On ScanNet-50, VASC at S75 reduces ATE, translational RPE, and rotational RPE by 13.98%, 14.40%, and 18.10%, respectively, relative to FasterVGGT, while Chamfer distance remains nearly unchanged.

Efficiency. On NRGBD at S85, VASC achieves $2.22\times$ and $2.29\times$ full-model speedups over dense VGGT on A6000 and RTX 4090, respectively (Table 2), with 1.66–2.62% latency overhead over FasterVGGT.

Table 2: **Full-model efficiency on NeuralRGB-D.** Mean seconds per complete stride-10 sequence. Speedup is relative to dense VGGT; overhead is relative to FasterVGGT (Wang et al., 2026a).

GPU	Sparsity (%)	Dense (s)	FasterVGGT (s)	VASC (s)	Speedup	Overhead
A6000	75	23.849	11.981	12.239	$1.950\times$	2.150%
	85	23.849	10.506	10.739	$2.220\times$	2.220%
RTX 4090	75	15.631	7.256	7.446	$2.100\times$	2.620%
	85	15.631	6.650	6.822	$2.290\times$	2.600%

5.3 Ablation Study

Table 3 compares three variants at 75% sparsity and frame stride 10: Value-aware only (no memory), Faster+memory (value contrast set to one), and Full VASC. Full VASC improves all six dataset-level metrics over FasterVGGT on both backbones and datasets.

Table 3: **S75 component ablation.** Frame stride 10. FasterVGGT results are from Table 1. Bold marks the best unrounded value within each backbone–dataset group.

Backbone	Dataset	Setting	ATE ↓	RPE-t ↓	RPE-r ↓	Acc ↓	Comp ↓	NC ↑
VGGT	7Scenes	FasterVGGT	0.084	0.038	1.080	0.018	0.028	0.646
		Value-aware only	0.078	0.034	**0.957**	0.018	0.028	0.649
		Faster+memory	0.078	0.035	0.975	**0.018**	0.028	0.648
		Full VASC	**0.074**	**0.034**	0.963	0.018	**0.028**	**0.649**
VGGT	NRGBD	FasterVGGT	0.073	0.030	0.503	0.041	0.027	0.798
		Value-aware only	**0.045**	0.020	0.278	0.020	0.018	**0.871**
		Faster+memory	0.049	0.022	0.325	0.023	0.020	0.848
		Full VASC	0.047	**0.020**	**0.262**	**0.019**	**0.018**	0.867
Pi3	7Scenes	FasterVGGT	0.066	0.027	0.855	0.031	0.027	0.709
		Value-aware only	**0.061**	0.026	0.815	0.025	0.025	0.743
		Faster+memory	0.064	0.026	0.827	0.025	0.026	0.753
		Full VASC	0.061	**0.026**	**0.807**	**0.022**	**0.025**	**0.779**
Pi3	NRGBD	FasterVGGT	0.074	0.020	0.277	0.056	0.033	0.779
		Value-aware only	0.062	0.018	0.228	0.042	0.029	0.831
		Faster+memory	0.064	0.019	0.226	0.040	0.028	0.849
		Full VASC	**0.058**	**0.017**	**0.196**	**0.034**	**0.026**	**0.881**

Value-aware selection. The benefit of value-aware selection grows with sparsity on NRGBD. Adding value contrast to Faster+memory reduces Acc by 5.20%, 10.09%, 15.32%, and 49.07% at S55, S65, S75, and S85 on VGGT; the corresponding gains on π^3 are 3.96%, 13.85%, 13.89%, and 19.97%. Value contrast delivers larger reconstruction gains on NRGBD as the execution budget shrinks.

Cross-layer memory. At S75, adding memory to Value-aware only reduces π^3/NRGBD Acc by 19.79% and Comp by 11.08%, while NC increases from 0.831 to 0.881. Full VASC improves all six metrics over both single-component variants on π^3/NRGBD. Relative to Value-aware only, memory improves five of six metrics on π^3/7Scenes, with slightly higher ATE, and five and four metrics on VGGT/7Scenes and VGGT/NRGBD, respectively, with trade-offs in the remaining metrics.

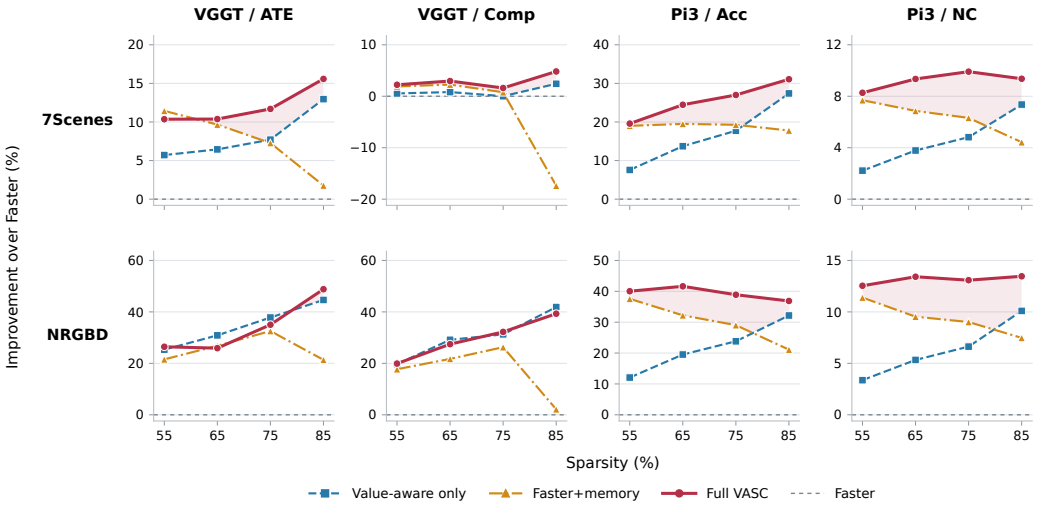

Figure 6: **Component trends across sparsity.** Relative improvements over FasterVGGT at stride 10; higher is better. Shading marks gains over the stronger single component. Each panel uses its own y-axis scale.

Trends across sparsity levels. Figure 6 extends the component comparison to S55–S85 on selected metrics. Improvements denote relative error reductions for ATE, Acc, and Comp, and relative increases for NC. Full VASC improves VGGT Comp over either component at all four sparsities on 7Scenes and at S55/S75 on NRGBD; on π^3, it improves Acc and NC across all four sparsities on both datasets. The two components improve reconstruction together across backbones and sparsity levels.

Memory adaptation to other schedulers. We add memory to FastVGGT's token protection without changing its cosine matcher, merge ratio (0.9), or token count, and to our AVGGT-R reproduction at 85% retained K/V service (Table 4). AVGGT-R uses the same selector with and without memory. Acc and Comp improve on both datasets; FastVGGT Comp improves by 1.20% while Acc worsens by 1.27%. FastVGGT adaptation is evaluated within the 7Scenes collection used for adapter selection.

Table 4: **Memory adaptation across hosts.** All 18 7Scenes and nine NRGBD sequences at stride 10; settings in Section 5.3. FastVGGT follows Shen et al. (2025). Bold marks the better value in each pair.

Host	Dataset	Configuration	Acc ↓	Comp ↓	NC1 ↑	NC2 ↑
FastVGGT	7Scenes	Official	**0.018**	0.027	0.616	0.648
		Memory adapter	0.018	**0.027**	**0.617**	**0.650**
AVGGT-R	7Scenes	w/o memory	0.018	0.028	**0.647**	0.650
		w/ memory	**0.017**	**0.027**	0.646	**0.651**
AVGGT-R	NRGBD	w/o memory	0.020	0.018	0.844	0.841
		w/ memory	**0.019**	**0.018**	**0.849**	**0.842**

6 Conclusion

We presented VASC, a training-free method for efficient 3D reconstruction. VASC combines pooled query–key relevance with adjacent-value contrast, and uses cross-layer memory to carry unserved demand into subsequent selection. Both components operate within a fixed budget while preserving the native sparse kernel and geometry heads. Experiments on 7Scenes and NeuralRGB-D with VGGT and π^3 demonstrate improved pose estimation and reconstruction over sparse baselines, with up to $2.29\times$ faster inference than dense VGGT.

Reproducibility statement

All reported cells use frozen weights and repositories. We record frame stride, input length, realized sparsity, layer budget, execution switches, source revision, and clean completion; failed or partial runs are excluded.

AI use statement

Generative AI assisted with methodology feedback, analysis code, figure composition, and language editing. The authors executed and audited all experiments and remain responsible for the final content.

References

Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. Neural rgb-d surface reconstruction. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6280–6291. IEEE, 2022.

Yushi Bai, Qian Dong, Ting Jiang, Xin Lv, Zhengxiao Du, Aohan Zeng, Jie Tang, and Juanzi Li. Indexcache: Accelerating sparse attention via cross-layer index reuse. *arXiv preprint arXiv:2603.12201*, 2026.

Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*, 2022.

Leyang Chen, Junyi Wu, Zhiteng Li, and Yulun Zhang. Ghost: Geometry-hierarchical online streaming token eviction for efficient 3d reconstruction. *arXiv preprint arXiv:2605.15852*, 2026a.

Xingyu Chen, Yue Chen, Yuliang Xiu, Andreas Geiger, and Anpei Chen. Ttt3r: 3d reconstruction as test-time training. In *International Conference on Learning Representations*, volume 2026, pp. 50694–50718, 2026b.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 2432–2443. IEEE, 2017.

Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations*, volume 2024, pp. 35549–35562, 2024.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35:16344–16359, 2022.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Weilun Feng, Haotong Qin, Mingqiang Wu, Chuanguang Yang, Yuqi Li, Xiangqi Li, Zhulin An, Libo Huang, Yulun Zhang, Michele Magno, et al. Quantized visual geometry grounded transformer. In *International Conference on Learning Representations*, volume 2026, pp. 59817–59838, 2026.

Ankit Gupta and Jonathan Berant. Value-aware approximate attention. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 9567–9574, 2021.

Ruizi Han, Miao Zhang, Ziyue Qiao, and Liqiang Nie. Evolving sparsity: Leveraging token importance dynamics for efficient llm decoding with sparse attention. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11554–11566, 2026.

Huiqiang Jiang, Yucheng Li, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Zhenhua Han, Amir H Abdi, Dongsheng Li, Chin-Yew Lin, et al. Minference 1.0: Accelerating pre-filling for long-context llms via dynamic sparse attention. *Advances in Neural Information Processing Systems*, 37:52481–52515, 2024.

Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *European conference on computer vision*, pp. 71–91. Springer, 2024.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. Snapkv: Llm knows what you are looking for before generation. *Advances in Neural Information Processing Systems*, 37:22947–22970, 2024.

Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Not all patches are what you need: Expediting vision transformers via token reorganizations. *arXiv preprint arXiv:2202.07800*, 2022.

D. V. Lindley. The theory of queues with a single server. *Mathematical Proceedings of the Cambridge Philosophical Society*, 48(2):277–289, 1952. doi: 10.1017/S0305004100027638.

Michael Neely. *Stochastic network optimization with application to communication and queueing systems*. Morgan & Claypool Publishers, 2010.

Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *Advances in neural information processing systems*, 34:13937–13949, 2021.

Michael S Ryoo, AJ Piergiovanni, Anurag Arnab, Mostafa Dehghani, and Anelia Angelova. Tokenlearner: What can 8 learned tokens do for images and videos? *arXiv preprint arXiv:2106.11297*, 2021.

You Shen, Zhipeng Zhang, Yansong Qu, Xiawu Zheng, Jiayi Ji, Shengchuan Zhang, and Liujuan Cao. Fastvggt: Training-free acceleration of visual geometry transformer. *arXiv preprint arXiv:2509.02560*, 2025.

Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2930–2937, 2013.

Xianbing Sun, Zhikai Zhu, Zhengyu Lou, Bo Yang, Jinyang Tang, Liqing Zhang, He Wang, and Jianfu Zhang. Avggt: Rethinking global attention for accelerating vggt. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 251–260, 2026.

Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. Quest: Query-aware sparsity for efficient long-context llm inference. *arXiv preprint arXiv:2406.10774*, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Chung-Shien Brian Wang, Christian Schmidt, Jens Piekenbrinck, and Bastian Leibe. Block-sparse global attention for efficient multi-view geometry transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14546–14555, 2026a.

Hengyi Wang and Lourdes Agapito. 3d reconstruction with spatial memory. In *2025 International Conference on 3D Vision (3DV)*, pp. 78–89. IEEE, 2025.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5294–5306. IEEE, 2025a.

Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10510–10522. IEEE, 2025b.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20697–20709. IEEE, 2024.

Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Permutation-equivariant visual geometry learning. In *International Conference on Learning Representations*, volume 2026, pp. 10481–10497, 2026b.

Junyi Wu, Zhiteng Li, Zheng Hui, Yulun Zhang, Linghe Kong, and Xiaokang Yang. Quantcache: Adaptive importance-guided quantization with hierarchical latent and layer caching for video generation. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15035–15044. IEEE, 2025a.

Yuqi Wu, Wenzhao Zheng, Jie Zhou, and Jiwen Lu. Point3R: Streaming 3D reconstruction with explicit spatial pointer memory. In *Advances in Neural Information Processing Systems*, volume 38, pp. 69675–69699, 2025b. doi: 10.52202/085713-2344.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, volume 2024, pp. 21875–21895, 2024.

Shuai Yuan, Yantai Yang, Xiaotian Yang, Xupeng Zhang, Zhonghao Zhao, Lingming Zhang, and Zhipeng Zhang. Infinitevggt: Visual geometry grounded transformer for endless streams. *arXiv preprint arXiv:2601.02281*, 2026.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297, 2020.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in neural information processing systems*, 36:34661–34710, 2023.

Dong Zhuo, Wenzhao Zheng, Jiahe Guo, Yuqi Wu, Jie Zhou, and Jiwen Lu. Streaming visual geometry transformer. In *International Conference on Learning Representations*, volume 2026, pp. 88055–88072, 2026.

Organization of the appendix. Appendix A gives implementation details; Appendix B describes observation and rendering protocols. Benchmark results, ablations, and backbone adaptations appear in Appendices C, D, and E, respectively.

A Implementation Details

A.1 Block Scores and Budgeted Selection

At global layer l, i indexes a pooled query block and j a physical key/value child block. Adjacent children in native patch-block order form stable pairs $\mathcal{C}(p) = \{j_{p,1}, j_{p,2}\}$; $p(j)$ identifies the pair containing child j. A final unpaired child is a singleton. Pairing is fixed across layers for a given layout. Batch and head indices are omitted below.

Equations 2–3 use average-pooled queries, keys, and values with feature dimension d. RMS is the square root of the mean squared feature entries, and Π_b is the proportional clipping operator defined below. The neighboring-value cosine similarity in Section 3 is a directional-variation diagnostic; it is not the RMS block score used for selection.

For N_q query blocks, demand and execution use identical averaging:

$$\begin{aligned} a_l(p) &= \frac{1}{N_q|\mathcal{C}(p)|} \sum_i \sum_{j \in \mathcal{C}(p)} C_l(i,j), \\ x_l(p) &= \frac{1}{N_q|\mathcal{C}(p)|} \sum_i \sum_{j \in \mathcal{C}(p)} X_l(i,j). \end{aligned} \tag{7}$$

The realization of $\mathcal{G}$ broadcasts bounded pair memory across queries and children, then selects within each query's fixed capacity:

$$P_l(i,j) = C_l(i,j) + \widetilde{M}_l(p(j)). \tag{8}$$

$$X_l(i,j) = \mathbf{1}\Big[j \in \mathrm{TopKIndices}_{b_l(i)}(P_l(i,:))\Big], \qquad \sum_j X_l(i,j) = b_l(i). \tag{9}$$

The memory update uses the raw balance M_l, while selection uses only the bounded copy $\widetilde{M}_l$.

A.2 Proportional Clipping

For a nonnegative vector r and capacity $b \geq 0$, the implementation uses

$$[\Pi_b(r)]_j = \min\left(1, \frac{b r_j}{\sum_k r_k}\right), \qquad \Pi_b(0) = 0. \tag{10}$$

This bounds each score by one and the total mass by b; saturation can leave unused score mass. The binary Top-K mask in Equation 9 separately enforces exact execution capacity. Memory uses $c_l = \sum_p a_l(p)$ per batch and head, giving $0 \leq \widetilde{M}_l(p) \leq 1$ and $\sum_p \widetilde{M}_l(p) \leq c_l$.

A.3 Value Contrast and Pairing

For pooled values u, v and scalar weights α, β,

$$\alpha u + \beta v = (\alpha + \beta)\frac{u+v}{2} + (\alpha - \beta)\frac{u-v}{2}. \tag{11}$$

The two terms separate common and differential content. RMS contrast measures the latter, complementing query-dependent relevance; common content can remain informative even at zero contrast. This interpretation motivates selection rather than bounding attention-output error. Pairs follow Section 4. An odd final child is paired with itself for zero contrast and counted as a singleton in demand and service.

A.4 Memory Normalization and Initialization

For the settings studied here, the memory copy used for ranking is

$$\widetilde{M}_l = \Pi_{c_l}(M_l), \qquad c_l = \sum_p a_l(p), \qquad M_1 = \widetilde{M}_1 = 0. \tag{12}$$

Normalization is performed separately for each batch entry and head. Only the ranking copy is normalized; the raw balance follows Equation 6 and is carried to the next layer.

A.5 Residual Accounting and State Lifetime

Define excess service by $\rho_l(p) = [x_l(p) - M_l(p) - a_l(p)]_+$. The nonnegative update in Equation 6 can then be written exactly as

$$M_{l+1} = M_l + a_l - x_l + \rho_l. \tag{13}$$

With $M_1 = 0$, summing gives $M_{L+1} = \sum_{l=1}^{L}(a_l - x_l + \rho_l)$ pair-wise. Since $a_l, x_l \in [0, 1]$, induction gives $0 \le M_l(p) \le l - 1$ for a fixed layout. The raw state retains accumulated deficits; Π_{c_l} bounds the copy used for ranking. Executing both children supplies one unit per query before averaging, leaving a residual if accumulated demand exceeds service.

Execution conventions. State is initialized per inference and reset on a layout mismatch. A pending layer is settled upon entry to the next; the final layer is settled after execution. The selected patch-child mask supplies x_l. Camera and register queries retain native dense attention; special-only sink key blocks retain the host's always-kept treatment. Handling of mixed boundary blocks follows the native mask configuration and is shared with the baseline. Selected child support is passed directly to native sparse execution, with pose, depth, and point heads unchanged.

B Observation and Visualization Protocols

We detail the intervention in Figure 3 and the rendering protocol in Figure 5.

B.1 Attention-Only Reconstruction and Local Value Analysis

Reconstruction intervention. We compare dense and attention-only VGGT inference on `office/seq-06` from 7Scenes (100 frames) and `breakfast_room` from NeuralRGB-D (117 frames), using full stride-10 sequences. Each input image has resolution 518×392, with a 28×37 patch grid and five special tokens. For each patch query and head, the attention-only condition retains the highest-ranked 25% of patch keys across all input views in each of the 24 global layers. Attention is renormalized over retained patch and protected special keys. Special queries, frame attention, weights, and prediction heads are unchanged. Selection uses each layer's current hidden states, and the resulting activations propagate through subsequent layers.

Controls include native dense inference and unmasked inference with matched chunking. Figure 3 uses the latter, with shared inputs, GT-defined camera, rendering settings, and tabletop outline. This token-level diagnostic is separate from the FasterVGGT/VASC benchmarks and is evaluated for reconstruction, not speed.

Attention and value maps. Figure 3(b) uses Breakfast Room, global layer 7 and head 10 (one-based), query patch $(14, 18)$ in frame 58, and key view 87 (zero-based coordinates). Pre-mask percentiles rank all 121,212 patch keys, including the query view but excluding special keys; 30,303 connections are retained per query and head. Higher percentiles indicate stronger attention; red and gray show retained and omitted connections. Value similarity averages cosine similarity to valid four-neighbors in the same image and head. Display values are clipped to $[0.5, 1]$; statistics use unclipped values. Cosine describes directional variation, whereas the method uses paired RMS contrast.

Spatial correspondence and selection. The 240-token crop spans rows $[6, 22)$ and columns $[11, 26)$. Crop-relative boxes are A: $[0, 5) \times [2, 11)$, B: $[6, 10) \times [7, 15)$, and C: $[10, 16) \times [4, 11)$. Coordinates are shared across maps and mapped to the image at 14×14 pixels per token. Boxes mark spatial regions; Table 5 reports their actual masks.

Table 5: **Regional attention, value similarity, and actual selection.** Statistics for the spatial boxes in Figure 3(b). Retained counts refer to query–key connections in the illustrated head, not tokens omitted from all network computation.

Region	Keys	Retained	Mean attention percentile	Mean V cosine
A	45	9	54.500	0.610
B	32	26	83.400	0.810
C	42	8	43.900	0.660

Selection and interpretation. The layer and head were chosen by inspecting saved maps; the three regions were manually annotated. The maps illustrate an attention–value mismatch, while reconstruction reflects the full-network intervention. They are qualitative evidence, not a prevalence estimate or an isolated head-level causal test. Matched-budget benchmarks evaluate method performance.

B.2 Qualitative Reconstruction Rendering

Figure 5 uses full stride-10 reconstructions of Kitchen (152 frames), Green Room (145 frames), and Heads/seq-01 (100 frames); Figure 1 uses Breakfast Room, Office/seq-06, and Stairs/seq-04 (50 frames). Stairs uses S85; the other sparse reconstructions and the teaser's dataset-level radar plots use S75. Predictions are rigidly aligned to GT by point-to-point ICP (threshold 0.1, at most 30 iterations, deterministic uniform registration samples). All saved points are rendered with shared cameras, point footprints, and crop bounds within each scene.

C Additional Benchmark Results

We report the full benchmark comparisons first, followed by denser input sampling, ScanNet-50, the FastVGGT comparison, and forward timing. Component controls are grouped separately in Appendix D.

C.1 Complete Main Comparison

Table 6: **Complete stride-10 comparison on VGGT and π^3.** Dense is the unpruned reference for each backbone. FasterVGGT (Wang et al., 2026a) denotes the original baseline on VGGT and our adaptation on π^3. Sparse baseline/VASC pairs have matched realized capacity; bold marks the better raw value. Gain is VASC's arithmetic mean over six direction-normalized relative changes, with metric wins in parentheses. Every row uses complete native sequences (18/18 for 7Scenes, 9/9 for NRGBD).

Dataset	Method	S	ATE ↓	RPE-t ↓	RPE-r ↓	Acc ↓	Comp ↓	NC ↑	Gain
7Scenes	Dense VGGT	0	0.069	0.031	0.859	0.017	0.028	0.652	–
	FasterVGGT	S55	0.079	0.033	0.922	**0.017**	0.028	0.650	–
	VASC	S55	**0.071**	**0.032**	**0.895**	0.018	**0.028**	**0.650**	+2.500% (5/6)
	FasterVGGT	S65	0.081	0.035	0.960	**0.017**	0.029	0.647	–
	VASC	S65	**0.073**	**0.033**	**0.911**	0.018	**0.028**	**0.650**	+3.600% (5/6)
	FasterVGGT	S75	0.084	0.038	1.080	0.018	0.028	0.646	–
	VASC	S75	**0.074**	**0.034**	**0.963**	**0.018**	**0.028**	**0.649**	+6.200% (6/6)
	FasterVGGT	S85	0.091	0.043	1.273	0.021	0.029	0.643	–
	VASC	S85	**0.077**	**0.037**	**1.028**	**0.019**	**0.028**	**0.646**	+10.500% (6/6)
NRGBD	Dense VGGT	0	0.030	0.016	0.179	0.014	0.015	0.908	–
	FasterVGGT	S55	0.049	0.022	0.324	0.025	0.020	0.854	–
	VASC	S55	**0.036**	**0.018**	**0.203**	**0.016**	**0.016**	**0.888**	+24.300% (6/6)
	FasterVGGT	S65	0.059	0.026	0.427	0.034	0.024	0.832	–
	VASC	S65	**0.044**	**0.019**	**0.239**	**0.018**	**0.017**	**0.880**	+29.200% (6/6)
	FasterVGGT	S75	0.073	0.030	0.503	0.041	0.027	0.798	–
	VASC	S75	**0.047**	**0.020**	**0.262**	**0.019**	**0.018**	**0.867**	+34.900% (6/6)
	FasterVGGT	S85	0.118	0.040	0.650	0.057	0.036	0.757	–
	VASC	S85	**0.061**	**0.024**	**0.298**	**0.026**	**0.022**	**0.843**	+41.300% (6/6)
7Scenes	Dense Pi3	0	0.057	0.024	0.769	0.020	0.025	0.810	–
	FasterVGGT	S55	0.062	0.025	0.815	0.026	0.025	0.739	–
	VASC	S55	**0.058**	**0.024**	**0.774**	**0.021**	**0.024**	**0.800**	+7.700% (6/6)
	FasterVGGT	S65	0.064	0.026	0.835	0.029	0.026	0.719	–
	VASC	S65	**0.060**	**0.025**	**0.789**	**0.022**	**0.025**	**0.786**	+9.400% (6/6)
	FasterVGGT	S75	0.066	0.027	0.855	0.031	0.027	0.709	–
	VASC	S75	**0.061**	**0.026**	**0.807**	**0.022**	**0.025**	**0.779**	+10.400% (6/6)
	FasterVGGT	S85	0.071	0.029	0.902	0.034	0.028	0.692	–
	VASC	S85	**0.064**	**0.027**	**0.835**	**0.024**	**0.025**	**0.757**	+12.300% (6/6)
NRGBD	Dense Pi3	0	0.025	0.012	0.133	0.019	0.018	0.959	–
	FasterVGGT	S55	0.046	0.016	0.208	0.041	0.028	0.824	–
	VASC	S55	**0.037**	**0.014**	**0.154**	**0.025**	**0.021**	**0.927**	+22.800% (6/6)
	FasterVGGT	S65	0.063	0.019	0.252	0.050	0.031	0.794	–
	VASC	S65	**0.046**	**0.015**	**0.173**	**0.029**	**0.024**	**0.900**	+25.600% (6/6)
	FasterVGGT	S75	0.074	0.020	0.277	0.056	0.033	0.779	–
	VASC	S75	**0.058**	**0.017**	**0.196**	**0.034**	**0.026**	**0.881**	+23.000% (6/6)
	FasterVGGT	S85	0.107	0.023	0.344	0.068	0.040	0.751	–
	VASC	S85	**0.081**	**0.021**	**0.239**	**0.043**	**0.031**	**0.852**	+23.100% (6/6)

C.2 DENSER INPUT SAMPLING

The stride-5 evaluation retains every fifth frame, with no input cap.

Table 7: **Complete stride-5 comparison across sparsity.** FasterVGGT (Wang et al., 2026a) and VASC are paired at identical realized block capacity. Bold marks the better raw value within each pair. Gain uses the same six-metric signed mean as Table 6, with wins in parentheses. All sequences are native and uncapped (18/18 for 7Scenes, 9/9 for NRGBD).

Dataset	Method	S	ATE ↓	RPE-t ↓	RPE-r ↓	Acc ↓	Comp ↓	NC ↑	Gain
7Scenes	FasterVGGT	S55	0.080	0.024	0.650	**0.016**	**0.027**	0.611	–
	VASC	S55	**0.072**	**0.022**	**0.609**	0.016	0.027	**0.612**	+3.000% (4/6)
	FasterVGGT	S65	0.081	0.024	0.682	**0.016**	0.027	0.610	–
	VASC	S65	**0.073**	**0.023**	**0.623**	0.016	**0.027**	**0.612**	+4.100% (5/6)
	FasterVGGT	S75	0.084	0.027	0.757	0.017	0.027	0.608	–
	VASC	S75	**0.075**	**0.024**	**0.670**	**0.017**	**0.027**	**0.611**	+6.200% (6/6)
	FasterVGGT	S85	0.110	0.032	1.142	0.028	0.035	0.605	–
	VASC	S85	**0.077**	**0.027**	**0.737**	**0.018**	**0.027**	**0.608**	+23.400% (6/6)
NRGBD	FasterVGGT	S55	0.056	0.022	0.339	0.028	0.019	0.835	–
	VASC	S55	**0.045**	**0.017**	**0.258**	**0.026**	**0.019**	**0.861**	+11.800% (6/6)
	FasterVGGT	S65	0.066	0.024	0.383	**0.030**	**0.021**	0.817	–
	VASC	S65	**0.062**	**0.018**	**0.263**	0.033	0.022	**0.849**	+8.500% (4/6)
	FasterVGGT	S75	0.076	0.027	0.446	0.031	0.022	0.787	–
	VASC	S75	**0.054**	**0.019**	**0.266**	**0.028**	**0.019**	**0.840**	+21.700% (6/6)
	FasterVGGT	S85	0.134	0.036	0.605	0.057	0.036	0.736	–
	VASC	S85	**0.067**	**0.020**	**0.268**	**0.031**	**0.023**	**0.820**	+40.600% (6/6)

C.3 SCANNET-50

We evaluate the fixed 50-scene collection with 100 images per scene, retaining the first frame and uniformly sampling the remaining frames following the FastVGGT evaluation protocol. All three methods use identical frame IDs within each scene, the same evaluator, depth-confidence threshold 1.0, and Chamfer distance cutoff 0.5. Unlike the stride-based benchmarks, this protocol fixes the input count. Both sparse methods use their S75 settings from Table 1.

Table 8: **ScanNet-50 quality.** Macro means across all 50 scenes; 100 frames per scene. All metrics are lower-is-better.

Method	ATE	RPE-t	RPE-r	ARE	Chamfer
Dense VGGT	0.092	0.055	1.206	3.950	0.441
FasterVGGT, S75	0.119	0.077	1.590	5.012	0.447
VASC, S75	0.102	0.066	1.303	4.253	0.447
VASC wins vs. FasterVGGT	46/50	48/50	50/50	42/50	23/50

VASC improves all four pose means over FasterVGGT by 13.98%, 14.40%, 18.10%, and 15.15%, with scene-level wins reported in Table 8. Mean Chamfer is nearly unchanged (0.08% improvement), with improvements on 23 of 50 scenes.

Evaluation protocol. The two sparse methods are paired on the same host for every scene: 41 scenes on RTX 4090 and nine on RTX A6000. For scene0056_00, Dense runs on RTX 4090 and the sparse pair on RTX A6000; input IDs and evaluator settings match. These runs use one forward measurement without warm-up and report quality only; latency follows the protocol in Appendix C.5.

C.4 Reconstruction Comparison with FastVGGT

FastVGGT (Shen et al., 2025) uses the authors' implementation with merge ratio 0.9. Its A6000 timings in Figure 2 were measured separately from the FasterVGGT/VASC timing runs. Table 9 compares reconstruction medians at both strides. On NRGBD, VASC improves Comp and NC medians over FastVGGT; trade-offs remain in Acc and in 7Scenes NC at stride 5.

Table 9: **Reconstruction medians versus FastVGGT.** Sequence-averaged point-level medians on all 18/9 sequences. FastVGGT (Shen et al., 2025) uses merge ratio 0.9; VASC uses VGGT. Bold: best unrounded value per stride and dataset.

Method	Stride	7Scenes			NeuralRGB-D		
		Acc Med↓	Comp Med↓	NC Med↑	Acc Med↓	Comp Med↓	NC Med↑
FastVGGT	10	0.008	0.010	0.716	**0.011**	0.010	0.793
VASC (S65)	10	**0.007**	0.010	**0.733**	0.011	**0.007**	**0.974**
VASC (S75)	10	0.007	**0.009**	0.732	0.012	0.008	0.969
FastVGGT	5	0.008	0.010	**0.709**	**0.012**	0.010	0.790
VASC (S65)	5	**0.006**	0.008	0.674	0.018	0.008	**0.945**
VASC (S75)	5	0.006	**0.008**	0.672	0.018	**0.008**	0.943

C.5 Forward Timing

The timed operation is `model(images)`, including its geometry heads. Loading, pose alignment, point-cloud evaluation, and ICP lie outside this region. The evaluator's pose-only flag affects post-forward evaluation only; all task heads remain in the timed forward. Input size is 518 with native aspect-ratio handling and stride 10: 1,700 frames over 18 7Scenes sequences and 1,101 over nine NRGBD sequences. One warm-up precedes three synchronized measured forwards per sequence. We average sequence medians, and compute speedup as the ratio of dataset mean Dense latency to dataset mean VASC latency. The baseline and VASC use the same physical GPU, native kernel, and fixed-count selection. A6000 has 48 GB and RTX 4090 has 24 GB of device memory.

Table 10: **Full-model forward latency.** Seconds per complete stride-10 sequence, averaged over the dataset. Speedup is Dense/VASC; overhead is VASC/FasterVGGT-1 (Wang et al., 2026a). Each sequence uses one warm-up and the median of three measured forwards. Only complete, uncontaminated matched comparisons are included.

GPU	Dataset	S	Dense (s)	FasterVGGT (s)	VASC (s)	Speedup	Overhead (%)
A6000	7Scenes	55	14.865	9.951	10.124	1.470×	1.740
A6000	7Scenes	65	14.865	9.175	9.347	1.590×	1.870
A6000	7Scenes	75	14.865	8.373	8.528	1.740×	1.850
A6000	NRGBD	55	23.849	14.830	15.076	1.580×	1.660
A6000	NRGBD	65	23.849	13.402	13.666	1.750×	1.970
A6000	NRGBD	75	23.849	11.981	12.239	1.950×	2.150
A6000	NRGBD	85	23.849	10.506	10.739	2.220×	2.220
RTX 4090	NRGBD	55	15.631	8.468	8.675	1.800×	2.450
RTX 4090	NRGBD	65	15.631	7.852	8.052	1.940×	2.560
RTX 4090	NRGBD	75	15.631	7.256	7.446	2.100×	2.620
RTX 4090	NRGBD	85	15.631	6.650	6.822	2.290×	2.600

Timing coverage. The table includes complete matched runs with no detected interference. A6000 7Scenes VASC-S85 and RTX 4090 7Scenes Dense have no qualifying completed records. The RTX 4090 7Scenes FasterVGGT-S75 interval is excluded because of interference; none of these missing entries is imputed.

D COMPONENT ABLATIONS

The component study examines value-aware scoring and cross-layer memory. The three component variants share the same implementation and all settings except the component being varied. Faster-VGGT reference rows in Table 3 are taken from the main benchmark runs in Table 1; they are not an additional both-components-off run of the ablation implementation.

Tables 11–14 report all six metrics at stride 10, averaged over all 18 7Scenes or nine NeuralRGB-D sequences. Value-aware only removes memory; Faster+memory sets paired-value contrast to one. S denotes sparsity (%). Bold marks the best unrounded value in each three-row group, including when displayed values coincide after rounding.

D.1 VGGT

Table 11: **VGGT components on 7Scenes.** Frame stride 10.

S	Variant	ATE ↓	RPE-t ↓	RPE-r ↓	Acc ↓	Comp ↓	NC ↑
55	Value-aware only	0.075	0.032	0.886	0.017	0.028	0.650
	Faster+memory	**0.070**	**0.031**	**0.873**	**0.017**	0.028	0.650
	Full VASC	0.071	0.032	0.895	0.018	**0.028**	**0.650**
65	Value-aware only	0.076	**0.033**	**0.910**	**0.017**	0.028	0.649
	Faster+memory	0.074	0.033	0.927	0.018	0.028	0.649
	Full VASC	**0.073**	0.033	0.911	0.018	**0.028**	**0.650**
75	Value-aware only	0.078	0.034	**0.957**	0.018	0.028	0.649
	Faster+memory	0.078	0.035	0.975	**0.018**	0.028	0.648
	Full VASC	**0.074**	**0.034**	0.963	0.018	**0.028**	**0.649**
85	Value-aware only	0.079	**0.037**	1.030	**0.019**	0.029	**0.646**
	Faster+memory	0.090	0.042	1.232	0.023	0.035	0.644
	Full VASC	**0.077**	0.037	**1.028**	0.019	**0.028**	0.646

Table 12: **VGGT components on NeuralRGB-D.** Frame stride 10.

S	Variant	ATE ↓	RPE-t ↓	RPE-r ↓	Acc ↓	Comp ↓	NC ↑
55	Value-aware only	0.036	0.018	0.228	0.017	0.016	**0.888**
	Faster+memory	0.038	0.018	0.212	0.016	0.016	0.884
	Full VASC	**0.036**	**0.018**	**0.203**	**0.016**	**0.016**	0.888
65	Value-aware only	**0.041**	0.019	0.254	0.018	**0.017**	**0.880**
	Faster+memory	0.043	0.020	0.253	0.020	0.019	0.871
	Full VASC	0.044	**0.019**	**0.239**	**0.018**	0.017	0.880
75	Value-aware only	**0.045**	0.020	0.278	0.020	0.018	**0.871**
	Faster+memory	0.049	0.022	0.325	0.023	0.020	0.848
	Full VASC	0.047	**0.020**	**0.262**	**0.019**	**0.018**	0.867
85	Value-aware only	0.066	**0.023**	0.325	**0.024**	**0.021**	**0.845**
	Faster+memory	0.093	0.031	0.456	0.051	0.036	0.814
	Full VASC	**0.061**	0.024	**0.298**	0.026	0.022	0.843

D.2 π^3

Table 13: π^3 **components on 7Scenes.** Frame stride 10.

S	Variant	ATE ↓	RPE-t ↓	RPE-r ↓	Acc ↓	Comp ↓	NC ↑
55	Value-aware only	0.060	0.025	0.801	0.024	0.025	0.755
	Faster+memory	0.059	0.025	**0.774**	0.021	**0.024**	0.795
	Full VASC	**0.058**	**0.024**	0.774	**0.021**	0.024	**0.800**
65	Value-aware only	0.061	0.025	0.811	0.025	0.025	0.746
	Faster+memory	0.063	0.026	0.803	0.023	0.025	0.768
	Full VASC	**0.060**	**0.025**	**0.789**	**0.022**	**0.025**	**0.786**
75	Value-aware only	**0.061**	0.026	0.815	0.025	0.025	0.743
	Faster+memory	0.064	0.026	0.827	0.025	0.026	0.753
	Full VASC	0.061	**0.026**	**0.807**	**0.022**	**0.025**	**0.779**
85	Value-aware only	**0.062**	**0.027**	**0.833**	0.025	0.025	0.743
	Faster+memory	0.068	0.029	0.904	0.028	0.027	0.723
	Full VASC	0.064	0.027	0.835	**0.024**	**0.025**	**0.757**

Table 14: π^3 **components on NeuralRGB-D.** Frame stride 10.

S	Variant	ATE ↓	RPE-t ↓	RPE-r ↓	Acc ↓	Comp ↓	NC ↑
55	Value-aware only	0.042	0.016	0.194	0.036	0.026	0.852
	Faster+memory	0.042	0.015	0.160	0.026	0.022	0.918
	Full VASC	**0.037**	**0.014**	**0.154**	**0.025**	**0.021**	**0.927**
65	Value-aware only	0.054	0.017	0.217	0.040	0.028	0.836
	Faster+memory	0.056	0.017	0.196	0.034	0.025	0.869
	Full VASC	**0.046**	**0.015**	**0.173**	**0.029**	**0.024**	**0.900**
75	Value-aware only	0.062	0.018	0.228	0.042	0.029	0.831
	Faster+memory	0.064	0.019	0.226	0.040	0.028	0.849
	Full VASC	**0.058**	**0.017**	**0.196**	**0.034**	**0.026**	**0.881**
85	Value-aware only	0.081	**0.020**	0.250	0.046	0.032	0.827
	Faster+memory	0.089	0.022	0.291	0.053	0.036	0.807
	Full VASC	**0.081**	0.021	**0.239**	**0.043**	**0.031**	**0.852**

E MEMORY ADAPTATION TO OTHER ATTENTION-REDUCTION METHODS

Table 4 summarizes memory adaptation to FastVGGT and AVGGT-R, retaining each host's attention-reduction mechanism.

E.1 FASTVGGT

The adapter derives arrivals from the native source-to-destination merge residual and counts independent source retention as service. It modifies source-protection priority while preserving destination-protection slots, the cosine matcher, merge ratio 0.9, and realized attention-token count. We use current scale 5, memory weight 0.25, and arrival load 1.0, selected on two four-sequence subsets of the evaluated 18-sequence 7Scenes collection. Results characterize in-collection adaptation, rather than held-out generalization.

Table 15: **FastVGGT adaptation on 7Scenes.** Frame stride 10. Means are macro averages of per-sequence statistics; "median" rows first take the within-sequence median, then average across sequences.

Variant	Acc ↓	Comp ↓	NC1 ↑	NC2 ↑
Official, mean	0.018	0.027	0.616	0.648
Adapted, mean	0.018	0.027	0.617	0.650
Official, median	0.008	0.010	0.682	0.750
Adapted, median	0.008	0.010	0.684	0.754

E.2 AVGGT-R

AVGGT-R is our independent reproduction. Memory starts at the first retained global layer (zero-based layer 9) and propagates through the remaining 15 global layers. Both arms use the same QK-derived current priority, factor-4 retained K/V grid, first-frame retention, diagonal correction, and omitted-token mean correction. The 85% service-retention ratio applies to this host's retained service support; it is distinct from the S85 patch sparsity in the main experiments. No paired-value factor is added.

Reconstruction and pose. On 7Scenes, Acc and Comp improve by 3.11% and 1.72%; on NeuralRGB-D they improve by 6.44% and 3.80%. At 90% retained service on 7Scenes, all eight reconstruction aggregates improve, while the three pose means regress by 0.09–0.43%.