SparseVILA-R1: Decoupling Visual Sparsity for Efficient VLM Reasoning

Samir Khaki[4] **Junxian Guo**[2] **Jiaming Tang**[2] **Shang Yang**[2] **Yukang Chen**[1]

Konstantinos N. Plataniotis[4] **Yao Lu**[1] **Song Han**[1,2] **Zhijian Liu**[1,3]

[1]NVIDIA [2]MIT [3]UC San Diego [4]University of Toronto

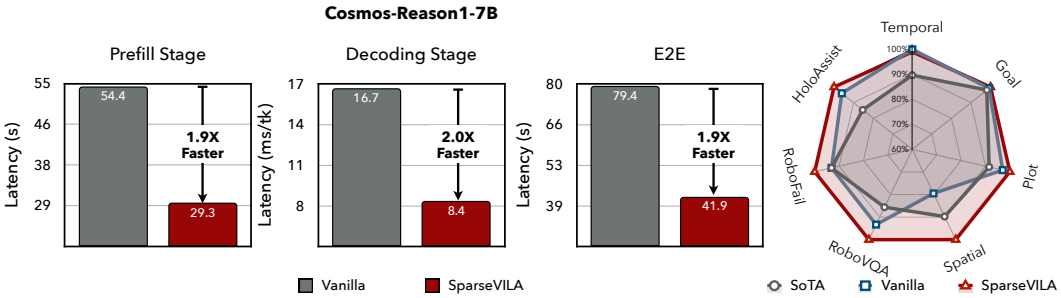

Figure 1: Overview of SparseVILA-R1. By decoupling prefill and decoding sparsity, SparseVILA-R1 achieves faster decoding throughput, hence tailored to the compute workload of large reasoning models. We achieve comparable performance on physical reasoning and video reasoning benchmarks while accelerating end-to-end latency by up to 1.9×.

Abstract

Enabling Vision Language models (VLMs) to *reason* requires operating over long chains of multimodal evidence grounded in video and physical interaction. The computation profile of such reasoning VLMs differs starkly from standard VQA-style inference (visual question answering). Reasoning VLMs typically generate large numbers of decoding tokens, hence shifting the latency distribution to the decoding stage and bottlenecking inference cost with token throughput. We present SparseVILA-R1, an inference-time, token sparsity approach tailored to visual reasoning. Through *decoupling* prefill and decoding sparsities, SparseVILA-R1 is able to strategically target token reduction, achieving up to 1.9× speedup whith lossless performance. By aligning sparsity with the compute profile of visual reasoning models, SparseVILA-R1 preserves cross-modal grounding while improving end-to-end efficiency, operating at the speed-accuracy Pareto frontier for long-context visual reasoning.

1 Introduction

In recent months, Vision Language models (VLMs) have been progressing from VQA-style systems – which map images or short video clips to concise answers – toward reasoning VLMs that maintain explicit chain-of-thought (CoT) over long multimodal contexts, enabling video understanding, robotic decision-making, and physically grounded reasoning [1, 2, 3]. As the field moves beyond retrieval and short-answer prediction, the runtime profile shifts in ways that stress current systems. First, prefill latency scales with the spatial and temporal resolution of modern inputs (high-resolution images, long videos). Second, reasoning workloads emit substantially more decoding tokens than VQA [4],

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Efficient Reasoning.

making decoding token throughput the dominant bottleneck. Realizing grounded, long-context, CoT-heavy interactions, therefore, requires efficiency by design.

We introduce SparseVILA-R1, a sparsity framework built on SparseVILA [5] and tailored to multimodal reasoning. Its core design is stage-decoupled sparsity tailored to multimodal reasoning: sparsity policies that treat prefill and decoding as distinct operating regimes with different compute profiles and pruning sensitivities. During prefill, SparseVILA-R1 performs modality-aware compression of spatiotemporal media – reducing redundant tokens before the LLM/token processor – to achieve significantly faster prefilling. During decoding, we retrieve and maintain a highly sparse contextually-aware subset of KV tokens that supports long chain-of-thought traces without eroding visual grounding. Taken together, these choices accelerate the processing of long, high-resolution visual media and sustain higher tokens-per-second during CoT generation, enabling faster, grounded video analysis and embodied decision-making – without retraining.

2 Preliminaries

Token pruning has proven effective in accelerating inference across a variety of tasks, including image classification [6, 7, 8] and semantic segmentation [9, 10]. With the rise of generative AI, these techniques have been further extended to diffusion models [11, 12], large language models [13], and vision-language models [14, 15, 16]. In this work, we examine how these techniques translate to *reasoning-oriented* workloads. Compared to standard VQA, reasoning VLMs generally process larger context and emit longer chains of thought, thereby shifting the cost profile towards decode-time throughput. In particular, we show that these characteristics necessitate a *stage-decoupled* approach to sparsity tailored to the compute profile of reasoning models, rather than relying solely on prefill-phase context compression as is common in VQA settings.

3 SparseVILA-R1

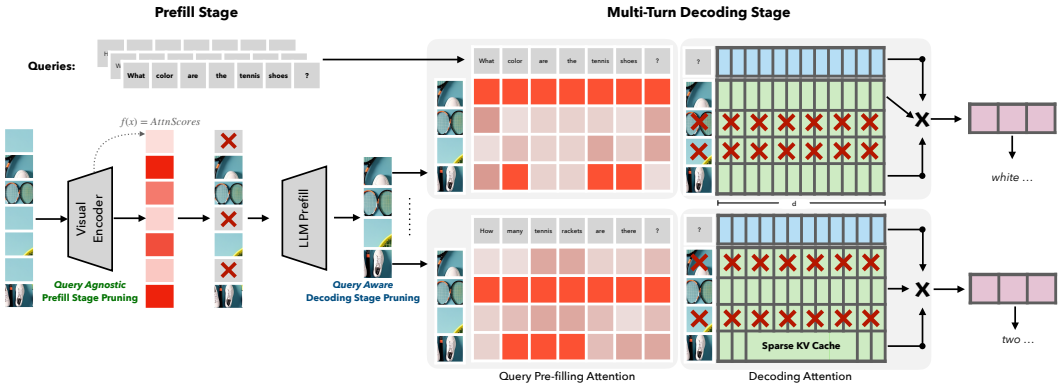

Figure 2: Overview of SparseVILA's decoupled sparsity framework. In the prefill stage, query-agnostic pruning removes redundant visual tokens based on salience scores from the visual encoder, yielding a compact representation shared across conversation turns. During decoding, query-aware retrieval selects only the most relevant visual tokens from the KV cache for attention computation, accelerating generation while maintaining multi-turn fidelity.

Prior work in visual token pruning predominantly targets compression in the prefill stage, with computational savings propagating into decoding. Subsequently, many prefill-pruning approaches remove tokens either prior to the LLM [17, 15] or within its early layers [18]. As a result, the removed tokens are excluded from most of the question's causal attention graph; the model cannot effectively attend to or recover information from them at decoding time, making aggressive prefill pruning comparatively lossy for reasoning (see Sec. 3.2). Accordingly, we introduce **SparseVILA-R1**, a reasoning-centric extension of SparseVILA [5], focusing on visual sparsity that *decouples* compression across the prefill and decoding stages. By shifting more aggressive compression into the decoding stage, where the question is known and relevant tokens can be retained, we better align with the decoding-dominated latency profile of reasoning workloads. This design reduces visual redundancy early without sacrificing coverage and concentrates decoding on contextually relevant tokens, thereby improving token throughput and end-to-end latency.

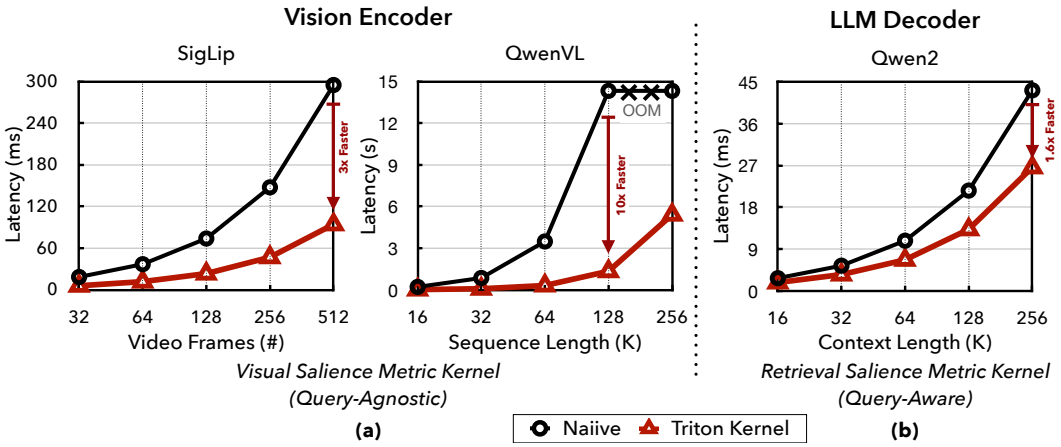

Figure 3: **Salience Metric Kernels.** Latency comparison between the naïve and custom Triton implementations across two settings: **(a)** query-agnostic salience computation for the SigLIP and QwenVL vision encoders, and **(b)** query-aware retrieval salience for the Llama2 and Qwen2 decoder backbones. Our custom kernels consistently accelerate both query-agnostic and retrieval salience computations, achieving up to 10× and 1.6× speedups, respectively.

3.1 Prefill Phase: Query-Agnostic Pruning

During the prefill stage, the vision–language model (VLM) encodes the system prompt, visual tokens, and optionally the first user query to construct the multimodal context. To ensure stable performance across multiple dialogue turns, pruning at this stage must remain strictly *query-agnostic*—guided only by visual redundancy or salience rather than any text-conditioned correlation. Since the visual context is computed once and reused throughout the conversation, pruning must retain sufficient coverage for future queries while minimizing redundant information.

Token Salience Estimation. We estimate token importance directly from the visual encoder's self-attention maps, providing a query-independent measure of visual salience. Following prior work [19, 20, 21], we aggregate attention signals to quantify each token's contribution to the overall representation, pruning those with the lowest aggregate salience. For models without summary tokens (e.g., SigLIP, QwenVL), importance is estimated by averaging intra-visual attention across all tokens, effectively capturing the same global aggregation behavior.

Efficient Implementation. For long-context inputs such as video sequences, attention-based salience estimation can be memory- and latency-intensive. To address this, we implement a custom Triton [22] kernel that streams softmax normalization and salience accumulation without explicitly forming the full attention matrix. This enables efficient salience computation even for hundreds of thousands of tokens. Empirically, the kernel yields up to a **3**× acceleration for SigLIP-style encoders and up to **10**× for QwenVL-style encoders (Figure 3a), forming the computational foundation for SparseVILA-R1's scalable prefill pruning.

3.2 Decode Phase: Query-Aware Retrieval

During the decoding phase, the VLM becomes memory-bound as it repeatedly computes next-token predictions using the pre-filled KV cache. To accelerate this process, SparseVILA-R1 selectively activates only the most query-relevant visual tokens during decoding attention, while preserving the rest of the visual information in the KV cache for potential use in later turns. This design enables query-conditioned sparsity without permanently discarding context, maintaining the flexibility required for multi-turn reasoning.

Query-Aware Token Selection. Before decoding begins, SparseVILA-R1 estimates the relevance of each visual token to the current query using attention-based salience. Specifically, it measures the aggregate attention strength between the query embeddings and visual entries in the KV cache,

providing a query-aware signal that highlights which tokens the model is most likely to reference during generation. Tokens with the highest relevance scores are retained for decoding, while less relevant tokens remain cached but inactive. This dynamic retrieval process effectively narrows the attention scope to the most informative subset of visual tokens, improving efficiency without compromising context consistency. We extend the Triton kernel from the prefill stage to stream the relevance computation directly between the query and cached visual tokens. This operation executes concurrently with the FlashAttention2 [23] path during prefill, yielding up to a **1.6**× speedup over a naïve implementation (Figure 3b). Once salience scores are obtained, the selected visual KV entries are compactly packed into a contiguous memory region, avoiding irregular sparse access patterns during autoregressive decoding.

3.3 Decoupled Prefill–Decode Visual Sparsity

SparseVILA-R1 introduces a decoupled sparsity framework that explicitly separates *where* and *how* visual compression is applied across the inference pipeline. This design is motivated by the distinct computational characteristics of the two stages: the *prefill stage* executes once per visual input to build the multimodal context, while the *decoding stage* performs iterative next-token prediction and typically dominates end-to-end latency. Applying uniform sparsity across both is therefore suboptimal—aggressive prefill pruning can permanently discard visual information required for later turns, whereas decoding remains the primary runtime bottleneck.

To address this imbalance, SparseVILA-R1 decouples sparsity between stages: lightweight, query-agnostic pruning is applied during prefill to remove globally redundant tokens while retaining sufficient visual coverage, and aggressive, query-aware retrieval is applied during decoding to focus computation on the most relevant visual cues. This adaptive allocation introduces sparsity where it yields the greatest efficiency gain, without compromising contextual grounding for future queries.

We compare the decoupled design with a *prefill-only* sparsity baseline on RoboVQA [24] (Table 1). When tuned for equivalent end-to-end speedup, reallocating sparsity toward decoding consistently improves task performance. The prefill stage retains enough visual tokens to maintain context integrity, while decoding sparsity effectively targets the dominant latency source in multimodal generation.

Sparsity		Speedup			RoboVQA
Prefill	Decode	Prefill	Decode	E2E	
0%	0%	1.0×	1.0×	1.0×	86.4
90%	0%	**14.6**×	1.1×	**1.4**×	80.0
70%	**85%**	4.9×	**1.2**×	**1.4**×	**89.1**

Table 1: Decoupled Prefill–Decode Sparsity

4 Experiments

Baselines. We evaluate SparseVILA-R1 against two categories of baselines: (i) vision-only pruning methods, which reduce redundancy purely in the visual domain without considering the text query, and (ii) text-aware methods, which adapt pruning decisions based on the language context. Representative baselines include VisionZip[15], PruMerge[17], and FastV [18]. Many baselines incorporate token salience metrics based on the attention weights of a particular layer. Although effective, these approaches are prone to high latency and memory demands that exceed the available hardware, particularly in the Vision Encoder, where attention is computed non-causally. In such cases, to maintain a fair comparison, we apply a chunked computation of their attention weights, trading off latency to remain within the memory bounds of the hardware (49GiB on 1xNVidia A6000). Fortunately, SparseVILA-R1 employs custom kernels to fuse our metric computation, enabling low-latency computation as shown in Figure 3 and Table 2. Our results demonstrate that SparseVILA-R1 achieves stronger adaptation to reasoning-oriented workloads and superior end-task performance compared to both categories. Additional details are included in Section A of the Appendix.

Inference Setting. We build an optimized inference pipeline based on TinyChat. Specifically, we apply W8A8 quantization to the visual encoder following SmoothQuant [25], and W4A16 quantization to the LLM following AWQ [26]. This quantized version achieves a 2.4× end-to-end speedup over the vanilla one, with negligible accuracy degradation, as verified in preliminary experiments. All subsequent results in this work are reported on top of this quantized version. Unless otherwise stated, inference is performed on a single NVIDIA A6000 GPU using greedy decoding with a batch size of 1.

Latency Evaluation. We measure the end-to-end inference runtime, including the visual encoder (E), language model prefilling (P), and decoding throughput (D). Total latency (E2E) is defined as the sum of prefill time and per-token decoding time, with decoding lengths fixed to ensure consistency across tasks. Reasoning models are evaluated in a single-turn setting, where total latency is computed as the sum of prefill and decoding times over 1,500 tokens, consistent with the typical 1-2K token output length of reasoning tasks.

Sparsity Ratio. Our sparsity configuration adopts a straightforward approach for both prefill and decoding, ensuring efficient implementation and cross-model compatibility. Specifically, we set a constant prefill sparsity before the LLM and a uniform decoding sparsity across all layers. More granular strategies, such as layer-wise or head-aware sparsity, may yield further optimization but introduce additional complexity and tuning overhead. We prioritize simplicity and generalization, leaving these refinements for future work.

	Sparsity		Speedup				HoloAssist	RoboFail	RoboVQA	Average
	P	D	E	P	D	E2E				
Cosmos-Reason1-7B (4fps)	0	0	1.0×	1.0×	1.0×	1.0×	65.0	60.0	86.4	70.5
+ PruMerge	.90	0	0.2×	2.2×	1.1×	0.7×	41.0	39.0	52.3	44.2
+ VisionZip	.90	0	0.2×	14.6×	1.1×	0.8×	**66.0**	54.0	80.3	66.7
+ FastV	.71	0	1.0×	2.2×	1.1×	1.3×	46.0	37.0	80.9	52.5
+ **SparseVILA-R1**	.70	.85	0.7×	4.9×	1.2×	1.4×	64.0	**63.0**	**89.1**	**72.0**
Cosmos-Reason1-7B (24fps)	0	0	1.0×	1.0×	1.0×	1.0×	72.0	54.0	88.2	71.4
+ PruMerge	.97	0	0.04×	13.8×	1.6×	0.3×	46.0	43.0	70.0	53.0
+ VisionZip	.97	0	0.04×	73.4×	1.6×	0.3×	64.0	54.0	80.9	66.6
+ **SparseVILA-R1**	.75	.95	0.4×	7.6×	2.0×	1.9×	**75.0**	**58.0**	**94.5**	**75.9**

Table 2: **Physical Reasoning Benchmark Results**. SparseVILA-R1 delivers up to **7.6**× faster language model prefill, **2.0**× faster decoding, and **1.9**× end-to-end speedup, while outperforming prior methods and the baseline model at 24 frames-per-second.

	Sparsity		Speedup				Temporal	Goal	Plot	Spatial	Overall
	P	D	E	P	D	E2E					
LongVILA-R1-7B (2048f)	0	0	1.0×	1.0×	1.0×	1.0×	68.7	87.8	73.1	48.1	74.1
+ PruMerge	.97	0	0.9×	89.1×	1.5×	1.6×	60.5	82.3	64.7	54.3	67.9
+ VisionZip	.97	0	0.9×	89.6×	1.5×	1.6×	61.6	86.8	68.9	49.4	70.4
+ **SparseVILA-R1**	.85	.95	1.0×	12.3×	1.6×	1.6×	**68.0**	**88.5**	**75.3**	**60.5**	**75.8**

Table 3: **Video Reasoning Benchmark Results**. SparseVILA-R1 maintains competitive performance on long-video reasoning tasks while delivering up to **1.6**× end-to-end speedup.

4.1 Results on Reasoning Benchmarks

Physical Reasoning. As shown in Table 2, our SparseVILA-R1 performs comparably to the baseline model on all benchmarks, further establishing the retained reasoning capability of our method. SparseVILA-R1 significantly outperforms state-of-the-art approaches on Cosmos-Reason [27], with scores on HA and RFail outperforming even the unquantized model. In these cases, SparseVILA-R1 operates at the Pareto-frontier of efficiency and performance, achieving a lossless 1.9× E2E speedup with a 4.5% boost in performance from the Vanilla baseline.

LongVideo Reasoning. We evaluate SparseVILA-R1 on a subset of the long video reasoning benchmark, which contains carefully curated question-answer pairs that require long-context reasoning [4]. Table 3 presents the performance comparison of SparseVILA-R1 with SoTA approaches [17, 15] and baselines. FastV [18] does not support chunk pre-filling, hence is not included in this comparison. By strategically shifting higher sparsity into the decoding stage, SparseVILA-R1 once again achieves an optimal efficiency-to-accuracy tradeoff, outperforming the vanilla baseline by +1.7% on average while accelerating the model by upwards of 1.6×.

Discussion. In Table 2 and Table 3, we have illustrated the robustness of SparseVILA-R1 in achieving strong compression/speedup while maintaining performance. Additionally, we have shown that, although existing SoTA methods reduce the effective compute (token sparsity), this does not always translate to overal latency reduction due to overheads in their metric computation. This concretely emphasizes the need for efficient and scalable approaches to context compression, as shown with SparseVILA-R1. In certain cases, we have shown that our approach improves upon baseline accuracies. We attribute this phenomenon to improving the model's retrieval ability by effectively reducing the processed context length. Our decoding sparsity approach effectively compresses the KV cache; the resulting cache constitutes a smaller, more information-dense context over which the model can reason. Similar findings were echoed in StreamingLLM [28], with the use of local-cache position IDs. We further explore SparseVILA-R1's retrieval capacity with Visual Needle in a Haystack in Section B of the Appendix.

5 Conclusion

We introduce SparseVILA-R1, a decoupled visual sparsity approach to accelerate large reasoning VLMs. By leveraging query-agnostic context stage pruning with query-aware generation stage retrieval, SparseVILA-R1 achieves the Pareto frontier of VLM reasoning efficiency. Considering the entire VLM inference stack – visual embedding, prefill, and decoding – SparseVILA-R1 achieves up to **12.3**$\times$ faster language model prefilling, **2.0**$\times$ faster decoding, and **1.9**$\times$ end-to-end speedup, while preserving or improving accuracy on reasoning benchmarks. Unlike prior pruning methods that trade speed for capability, SparseVILA-R1 maintains fidelity across modalities and architectures through decoupled sparsity allocation and efficient kernel design. This establishes a scalable, training-free foundation for accelerating the next generation of multimodal systems, enabling VLMs to efficiently reason about longer contexts, accommodating thousands of images and frames.

References

[1] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2024. 1

[2] Ruohong Zhang, Bowen Zhang, Yanghao Li, Haotian Zhang, Zhiqing Sun, Zhe Gan, Yinfei Yang, Ruoming Pang, and Yiming Yang. Improve vision language model chain-of-thought reasoning, 2024. 1

[3] Guowei Xu, Peng Jin, Ziang Wu, Hao Li, Yibing Song, Lichao Sun, and Li Yuan. Llava-cot: Let vision language models reason step-by-step, 2025. 1

[4] Yukang Chen, Wei Huang, Baifeng Shi, Qinghao Hu, Hanrong Ye, Ligeng Zhu, Zhijian Liu, Pavlo Molchanov, Jan Kautz, Xiaojuan Qi, Sifei Liu, Hongxu Yin, Yao Lu, and Song Han. Scaling rl to long videos, 2025. 1, 5, 9

[5] Samir Khaki, Junxian Guo, Jiaming Tang, Shang Yang, Yukang Chen, Konstantinos N. Plataniotis, Yao Lu, Song Han, and Zhijian Liu. Sparsevila: Decoupling visual sparsity for efficient vlm inference, 2025. 2

[6] 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. 2

[7] Zhenglun Kong, Peiyan Dong, Xiaolong Ma, Xin Meng, Mengshu Sun, Wei Niu, Xuan Shen, Geng Yuan, Bin Ren, Minghai Qin, et al. Spvit: Enabling faster vision transformers via soft token pruning. *arXiv preprint arXiv:2112.13890*, 2021. 2

[8] Samir Khaki and Konstantinos N Plataniotis. The need for speed: Pruning transformers with one recipe. In *The Twelfth International Conference on Learning Representations*, 2024. 2

[9] Xuanyao Chen, Zhijian Liu, Haotian Tang, Li Yi, Hang Zhao, and Song Han. Sparsevit: Revisiting activation sparsity for efficient high-resolution vision transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2061–2070, June 2023. 2

[10] Quan Tang, Bowen Zhang, Jiajun Liu, Fagui Liu, and Yifan Liu. Dynamic token pruning in plain vision transformers for semantic segmentation, 2023. 2

[11] Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating diffusion transformers with token-wise feature caching. In *The Thirteenth International Conference on Learning Representations*, 2025. 2

[12] Daniel Bolya and Judy Hoffman. Token merging for fast stable diffusion. *CVPR Workshop on Efficient Deep Learning for Computer Vision*, 2023. 2

[13] Sehoon Kim, Sheng Shen, David Thorsley, Amir Gholami, Woosuk Kwon, Joseph Hassoun, and Kurt Keutzer. Learned token pruning for transformers, 2022. 2

[14] Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, et al. Sparsevlm: Visual token sparsification for efficient vision-language model inference. *arXiv preprint arXiv:2410.04417*, 2024. 2

[15] Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. Visionzip: Longer is better but not necessary in vision language models. *arXiv preprint arXiv:2412.04467*, 2024. 2, 4, 5

[16] Kai Huang, Hao Zou, Ye Xi, BoChen Wang, Zhen Xie, and Liang Yu. Ivtp: Instruction-guided visual token pruning for large vision-language models. In *European Conference on Computer Vision*, pages 214–230. Springer, 2025. 2

[17] Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. Llava-prumerge: Adaptive token reduction for efficient large multimodal models. *arXiv preprint arXiv:2403.15388*, 2024. 2, 4, 5

[18] Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models, 2024. 2, 4, 5

[19] Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. VisionZip: Longer is Better but Not Necessary in Vision Language Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 3

[20] Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. LLaVA-PruMerge: Adaptive Token Reduction for Efficient Large Multimodal Models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. 3

[21] Kazi Hasan Ibn Arif, JinYi Yoon, Dimitrios S Nikolopoulos, Hans Vandierendonck, Deepu John, and Bo Ji. HiRED: Attention-Guided Token Dropping for Efficient Inference of High-Resolution Vision-Language Models. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2025. 3

[22] Philippe Tillet, Hsiang-Tsung Kung, and David Cox. Triton: An Intermediate Language and Compiler for Tiled Neural Network Computations. In *Proceedings of the ACM SIGPLAN International Workshop on Machine Learning and Programming Languages (MAPL)*, 2019. 3

[23] Tri Dao. FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024. 4

[24] Pierre Sermanet, Tianli Ding, Jeffrey Zhao, Fei Xia, Debidatta Dwibedi, Keerthana Gopalakrishnan, Christine Chan, Gabriel Dulac-Arnold, Sharath Maddineni, Nikhil J Joshi, Pete Florence, Wei Han, Robert Baruch, Yao Lu, Suvir Mirchandani, Peng Xu, Pannag Sanketi, Karol Hausman, Izhak Shafran, Brian Ichter, and Yuan Cao. RoboVQA: Multimodal Long-Horizon Reasoning for Robotics. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2024. 4

[25] Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, 2023. 4

[26] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-Aware Weight Quantization for LLM Compression and Acceleration. In *Proceedings of the Conference on Machine Learning and Systems (MLSys)*, 2024. 4

[27] NVIDIA, :, Alisson Azzolini, Junjie Bai, Hannah Brandon, Jiaxin Cao, Prithvijit Chattopadhyay, Huayu Chen, Jinju Chu, Yin Cui, Jenna Diamond, Yifan Ding, Liang Feng, Francesco Ferroni, Rama Govindaraju, Jinwei Gu, Siddharth Gururani, Imad El Hanafi, Zekun Hao, Jacob Huffman, Jingyi Jin, Brendan Johnson, Rizwan Khan, George Kurian, Elena Lantz, Nayeon Lee, Zhaoshuo Li, Xuan Li, Maosheng Liao, Tsung-Yi Lin, Yen-Chen Lin, Ming-Yu Liu, Xiangyu Lu, Alice Luo, Andrew Mathau, Yun Ni, Lindsey Pavao, Wei

Ping, David W. Romero, Misha Smelyanskiy, Shuran Song, Lyne Tchapmi, Andrew Z. Wang, Boxin Wang, Haoxiang Wang, Fangyin Wei, Jiashu Xu, Yao Xu, Dinghao Yang, Xiaodong Yang, Zhuolin Yang, Jingxu Zhang, Xiaohui Zeng, and Zhe Zhang. Cosmos-reason1: From physical common sense to embodied reasoning, 2025. 5

[28] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks, 2024. 6

[29] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11975–11986, 2023. 9

[30] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025. 9

[31] Fuzhao Xue, Yukang Chen, Dacheng Li, Qinghao Hu, Ligeng Zhu, Xiuyu Li, Yunhao Fang, Haotian Tang, Shang Yang, Zhijian Liu, Yihui He, Hongxu Yin, Pavlo Molchanov, Jan Kautz, Linxi Fan, Yuke Zhu, Yao Lu, and Song Han. Longvila: Scaling long-context visual language models for long videos. 2024. 9

[32] Peiyuan Zhang, Kaichen Zhang, Bo Li, Guangtao Zeng, Jingkang Yang, Yuanhan Zhang, Ziyue Wang, Haoran Tan, Chunyuan Li, and Ziwei Liu. Long context transfer from language to vision. *arXiv preprint arXiv:2406.16852*, 2024. 9

A Details on Experimental Setup

In this section we include additional details on the models used and the inference setting.

Models We study compression in the context of two reasoning-focused VLMs: LongVILA and Cosmos-Reason. LongVILA [4] employs a SigLIP [29] vision encoder followed by spatial token pooling before the LLM. For vision-only compression approaches (e.g., VisionZip, PruMerge, and our prefill-stage compression), we enforce a floored square token count to maintain compatibility with LongVILA's media processing pipeline. In contrast, Cosmos-Reason – built on the Qwen2.5VL [30] architecture – processes spatial and temporal tokens jointly along a unified sequence dimension. Consequently, compression in this setting operates in a spatio-temporal manner, enabling baselines and our method to reduce redundancy across both visual and temporal modalities. This setup ensures a fair comparison across architectures with distinct vision-language processing pipelines, while highlighting the robustness of SparseVILA-R1 across both vision-only and text-aware strategies.

Additional Details on Inference Setting. For inference, we employ chunked prefilling on the token processor (LLM), with a chunk size of 32768 tokens. While the vision tower typically processes the entire context embedding in a single forward pass, the memory footprint of LongVILA [4] exceeds the GPU capacity on 1xNVidia A6000 when evaluating with 2048 frames. In such cases, we account for the additional overhead introduced by *embedding-prefill disaggregation*, where both the embedding and prefilling stages are executed in chunks.

B Visual Needle in a Haystack

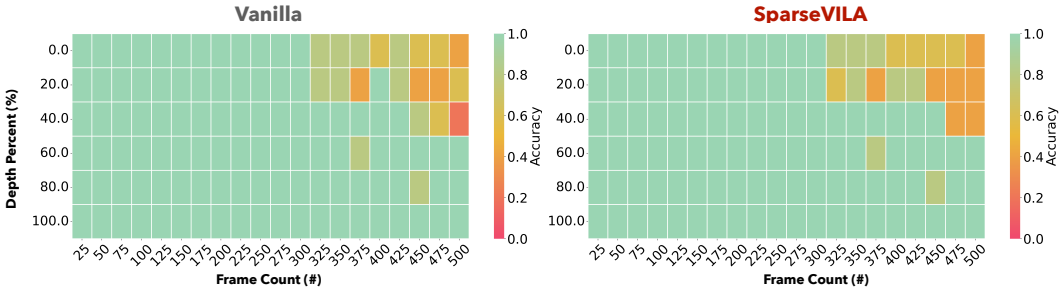

Figure 4: Long Context Visual Needle in a Haystack based on LongVILA [31]

In this section, we further evaluate the long-context retrieval performance with Visual Needle in a Haystack (V-NIAH) benchmark [32, 31]. We compare the long-context retrieval capacity of our method with the vanilla implementation on LongVILA [31] in Figure 4. SparseVILA-R1 maintains comparable performance with the baseline, indicating that the sparse selection of tokens can continue to support long context retrieval performance. We leverage a 90% decoding sparsity, translating to a 2× decoding speedup.