LOOK BACK, THINK AHEAD: VISUAL MEMORY ON DEMAND FOR EFFICIENT MULTIMODAL REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Processing long visual token sequences from high-resolution images makes multi-step reasoning computationally expensive for multimodal Large Language Models (MLLMs). Existing one-shot pruning and aggregation methods compress visual tokens into a fixed context before decoding. However, visual evidence needs can shift as reasoning unfolds, making it difficult for a fixed compressed context to retain all the details needed across stages. To address this challenge, we propose ViMoD, a lightweight framework that maintains a compact visual context while preserving access to original fine-grained evidence as reasoning needs evolve. Deformable Aggregation of Region-wise Tokens (DART) learns content-adaptive groups and aggregation capacities, constructing compact Coarse representations linked to recoverable original Fine tokens. Temporal Routing for Adaptive Contextual Evidence (TRACE) integrates decoding history to anticipate upcoming evidence needs and select, retain, or replace active Fine-token groups. Selected Fine tokens augment the persistent Coarse context in the frozen backbone, enabling stage-specific evidence access without continuously attending to all visual tokens. On Qwen3-VL-4B, ViMoD outperforms all evaluated baselines on all eight reasoning benchmarks at a **20%** target visual-token budget, improving the mean normalized score by **39.0%** over the strongest evaluated one-shot baseline. These gains require additional trainable parameters amounting to only **0.0546%** of the frozen backbone's parameter count.

1 INTRODUCTION

Multimodal Large Language Models (MLLMs) (Bai et al., 2025; Wang et al., 2025; Team et al., 2025; Hong et al., 2025) support diverse visual understanding and reasoning tasks. However, processing long visual-token sequences incurs substantial inference costs, particularly for high-resolution inputs. Visual token compression offers a practical way to reduce these costs.

Existing methods, including FastV (Chen et al., 2024a), VisionZip (Yang et al., 2025a), and DivPrune (Alvar et al., 2025), reduce this cost by pruning or merging visual tokens. Many compress the visual context early in inference and keep the resulting compact representation fixed throughout decoding, a strategy we refer to as **one-shot visual compression**. Although this strategy reduces computation per generated token, it raises two questions for multi-step visual reasoning: *Can a fixed compressed context efficiently support changing visual evidence needs? Do per-token savings reduce total decoding cost over a complete response?*

Visual evidence needs can shift across successive stages of reasoning. In the triangle-perimeter example in Fig. 1, regions with high gradient-based visual contribution change across successive portions of the response. We quantify this variation using the average turnover of highly attributed visual tokens between adjacent decoding windows as a proxy for changes in visual demand (Appendix C.5). Larger shifts are associated with lower answer retention when 50% of visual tokens are retained (Fig. 2(a)). This association suggests that a fixed compressed representation may fail to preserve evidence needed at different reasoning stages.

Even with advance knowledge of the required regions, a fixed context can require a larger visual token budget than stage-specific access. To examine this, we compare Dynamic access, which provides annotated regions at their corresponding reasoning stages, with Fixed oracle, which receives the union of all stage-specific annotated regions at the start of generation. At 70% of Full's accuracy, the

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.

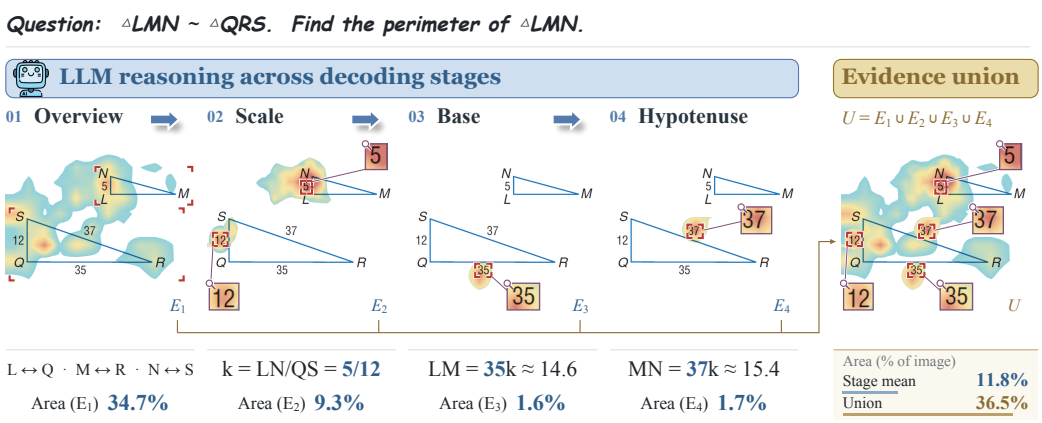

Figure 1: **Visual contribution changes across different reasoning stages.** The model first forms an overview of the triangle diagram, determines the scale factor from 5 and 12, and reads the known base and hypotenuse, 35 and 37. Heatmaps show gradient-based visual-token contribution at selected decoding stages; warmer colors indicate higher contribution relative to other tokens in the same stage. The right panel combines the corresponding regions into $U = E_1 \cup E_2 \cup E_3 \cup E_4$. These regions occupy 11.8% of the image on average per stage, whereas their union occupies 36.5%, counting overlapping areas once.

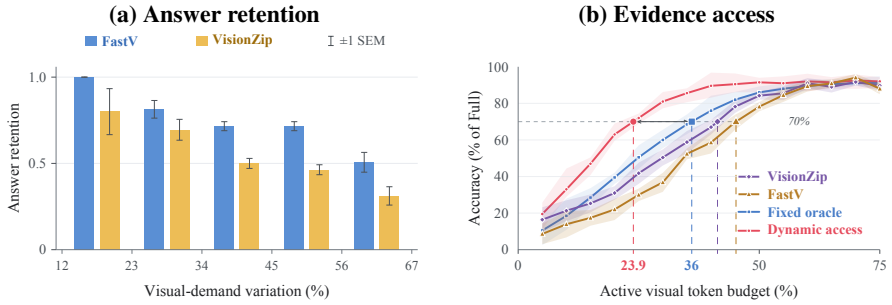

Figure 2: **Visual demand and evidence access.** (a) On MathVerse at 50% token retention, the fraction of baseline-correct examples remaining correct after compression, grouped by visual-demand variation; error bars show SEM. (b) On MMMU-Pro, the same `Qwen3.8`-annotated regions are accessed stage by stage (Dynamic access) or as their union (Fixed oracle), alongside VisionZip and FastV. Accuracy is normalized to Full; bands show standard deviation.

interpolated visual token budgets are approximately 23.9% for Dynamic access and 36.0% for Fixed oracle (Fig. 2(b)). This comparison suggests that efficient visual compression depends not only on which evidence is retained, but also on how that evidence is made available across reasoning stages.

Per-token savings also do not necessarily reduce total decoding cost. For FastV on MathVerse, longer responses at several retention ratios offset the reduction in FLOPs per generated token, increasing estimated total decoder FLOPs relative to the uncompressed baseline (Fig. 3). Compression efficiency must therefore be evaluated over complete responses.

Together, these findings motivate a framework that adapts visual evidence access to evolving reasoning needs while reducing computation over complete responses. Recoverable visual memory enables a compact active context without permanently discarding fine-grained evidence. Existing recovery methods such as DSTP (Kim et al., 2026) rely on decoding-time attention changes. This signal reflects the model's current interaction with the visual context, whereas recalled evidence may remain active over subsequent steps. We therefore seek to learn both a compact, recoverable visual memory and a recall policy that uses decoding history to anticipate upcoming evidence needs.

To this end, we propose **ViMoD: Visual Memory on Demand**, coupling learned visual memory with adaptive access to fine-grained visual tokens (Fine) and targeting a low average active visual-token budget during decoding. **Deformable Aggregation of Region-wise Tokens (DART)** learns region-wise capacities and deformable groups, linking compact Coarse representations to recoverable Fine-

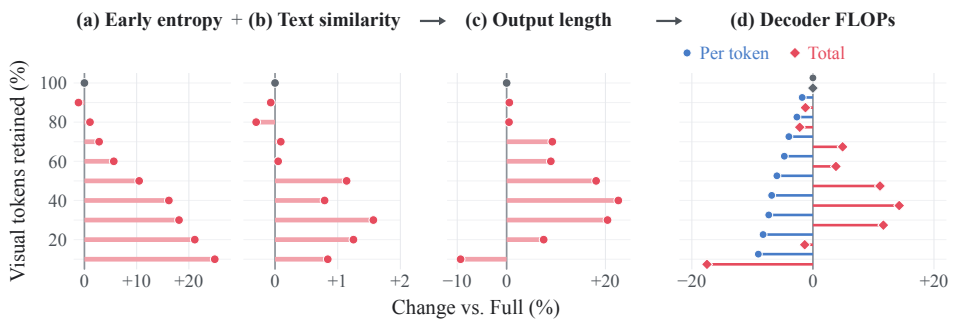

Figure 3: **From per-token savings to response-level cost.** FastV on MathVerse; horizontal axes show percentage changes relative to the uncompressed baseline. **(a)** Early next-token entropy reflects predictive uncertainty. **(b)** Neighboring-window embedding similarity measures the similarity of successive content. **(c)** Output length counts generated tokens. **(d)** Estimated decoder FLOPs compare amortized cost per generated token (circles) with total response cost (diamonds). At several retention ratios, longer responses outweigh per-token savings, increasing total decoder computation.

token groups through a **Coarse-to-Fine (C2F)** map. **Temporal Routing for Adaptive Contextual Evidence (TRACE)** uses a state-space model to integrate decoding history, anticipate upcoming evidence needs, and select, retain, or replace active Fine-token groups. Joint-KV combines selected Fine tokens with persistent Coarse context in the frozen backbone's attention.

ViMoD achieves state-of-the-art performance in visual reasoning under limited visual-token budgets (Tab. 1). At a 20% target budget with 9:1 memory, its average normalized score exceeds DSTP† on the same DART memory by 10.1% and DivPrune, the strongest one-shot baseline, by 39.0%. In our MMMU-Pro budget sweep, ViMoD reduces total inference time relative to Full at all evaluated target budgets from 20% to 90% (Fig. 6).

2 Related Work

Visual token compression and recovery. One-shot compression (Chen et al., 2024a; Yang et al., 2025a; Song et al., 2026; Lu et al., 2026; Alvar et al., 2025; Zhang et al., 2026a) prunes or merges visual tokens early, keeping the compressed context fixed throughout decoding. Recovery methods (Kim et al., 2026) reintroduce tokens using attention changes, although visual attention sinks and head-specific behavior (Kang et al., 2025a;b) complicate attention-based evidence selection. ViMoD couples learned compression with a dynamic Fine-access policy supervised by upcoming reasoning needs, maintaining a compact, persistent Coarse context.

Efficient text compression and deformable sampling. Gist-based LLM methods compress text into compact representations (Mu et al., 2023; Deng et al., 2026) and selectively recover original tokens (Mao et al., 2026). ViMoD adapts this combination of persistent compressed context and on-demand recovery to images, accounting for spatial structure and region-dependent granularity. Inspired by content-dependent sampling in deformable convolutions (Dai et al., 2017), DART learns region-wise capacities and deformable groups shared by Coarse aggregation and C2F-guided Fine recall.

Interleaved visual reasoning. During generation, the tool-based DeepEyes (Zheng et al., 2026) crops and encodes local regions for additional high-resolution observations, while explicit interaction formats (Hu et al., 2026) train the backbone to emit specialized tokens controlling transitions between textual reasoning and visual access. ViMoD instead learns access decisions through lightweight auxiliary modules and integrates selected evidence through Joint-KV, keeping the backbone frozen.

3 Method

3.1 Visual Memory and Evidence Access

Given an image I and prompt x, ViMoD performs prefill using a multimodal backbone with frozen parameters θ_{Frozen} and retains the original visual and text key–value (KV) banks (Fig. 4).

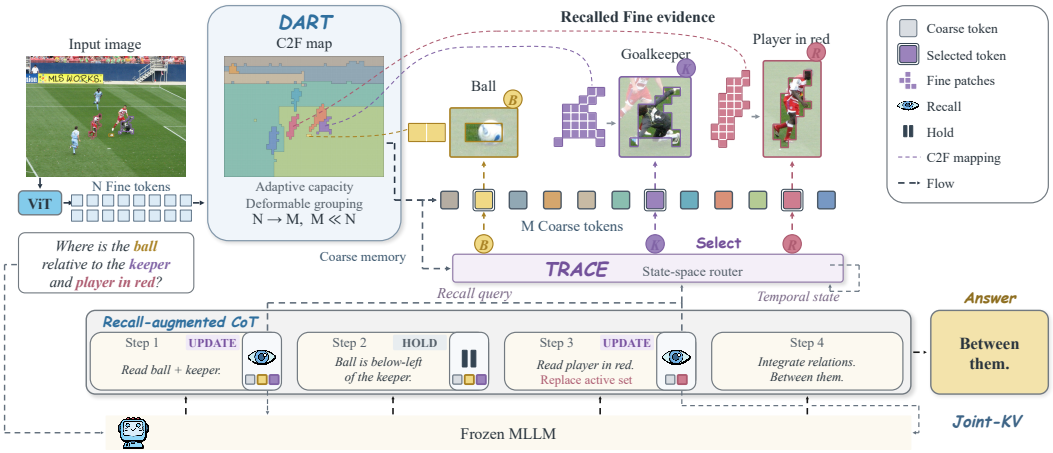

Figure 4: **Overview of ViMoD.** DART constructs persistent Coarse context and a C2F mapping to the original Fine-token groups. TRACE proposes evidence groups and retains or replaces the active set. Joint-KV exposes the selected Fine evidence alongside Coarse context and text history.

Let $[N] = \{1, \ldots, N\}$ index the Fine embeddings $\mathbf{X}^{\mathrm{F}} \in \mathbb{R}^{N \times d}$. DART produces $M \ll N$ Coarse descriptors $\mathbf{X}^{\mathrm{C}} \in \mathbb{R}^{M \times d}$ for routing, corresponding layer-wise KV entries for attention, and a C2F map $\Pi(j) \subseteq [N]$ linking each group to its Fine members. At routing decision t, the active groups $\mathcal{A}_t \subseteq [M]$ expose

$$\mathcal{F}_t = \bigcup_{j \in \mathcal{A}_t} \Pi(j), \qquad N_t^{\mathrm{active}} = M + |\mathcal{F}_t|. \tag{1}$$

All Coarse entries remain active. Unselected Fine entries remain stored for later access but do not participate in attention.

3.2 DART: Deformable Aggregation of Region-wise Tokens

DART learns the *capacity* (Fine-token count) and *membership* of each of M groups, which jointly define the units of compression and recall.

Allocate group sizes under a fixed total. Smaller groups provide finer representations and cheaper recall units (Fig. 5(a)). An initial grid partitions $[N]$ into M nonempty groups $\mathcal{G}_j^{(0)}$ with sizes $c_j^{(0)}$ and mean descriptors $\bar{\mathbf{x}}_j$. A learned score $a_j = \mathbf{w}_c^\top \bar{\mathbf{x}}_j$ adjusts their capacities:

$$\widetilde{c}_j = 1 + (U-1)\,\mathrm{sigmoid}(b_j + a_j + \mu), \qquad \sum_{j=1}^{M} \widetilde{c}_j = N. \tag{2}$$

Here $U > 1$ bounds group size, and b_j encodes the initial capacity. The shared offset μ enforces the fixed total; sum-preserving rounding yields integer capacities c_j (details in Appendix A.1.1).

Adapt membership to image content. DART scores membership by combining the grid prior $H_{ij}^{(0)} = \mathbf{1}_{\{i \in \mathcal{G}_j^{(0)}\}}$ with learned content affinity:

$$S_{ij} = \gamma H_{ij}^{(0)} + \frac{(\mathbf{W}_{\mathrm{F}} \mathbf{x}_i^{\mathrm{F}})^\top (\mathbf{W}_{\mathrm{C}} \bar{\mathbf{x}}_j)}{\sqrt{d_a}}. \tag{3}$$

The projections map tokens and regions to dimension d_a; γ weights the prior. In confidence order, each Fine token enters its highest-scoring group with remaining capacity. The resulting groups partition $[N]$, satisfy $|\mathcal{G}_j| = c_j$, and can cross initial grid boundaries (Fig. 5(b)).

Share the compression and recall support. Within each group, normalized affinities form the pooling matrix $\mathbf{W} \in \mathbb{R}^{N \times M}$ and its C2F map:

$$w_{ij} = \frac{\mathbf{1}_{\{i \in \mathcal{G}_j\}} \exp S_{ij}}{\sum_{k \in \mathcal{G}_j} \exp S_{kj}}, \qquad \mathbf{X}^{\mathrm{C}} = \mathbf{W}^\top \mathbf{X}^{\mathrm{F}}, \qquad \Pi(j) = \mathcal{G}_j. \tag{4}$$

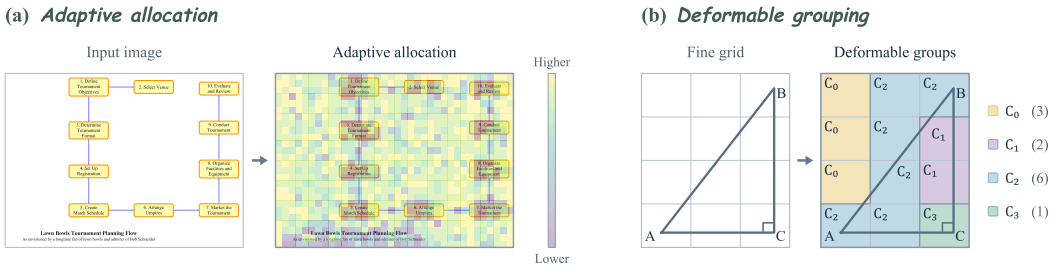

Figure 5: **DART adapts compression granularity and group shape to image content.** **(a)** Blank regions generally undergo stronger compression, while text-bearing regions retain finer granularity. Lower allocation-map values indicate fewer Fine tokens per Coarse token. **(b)** Deformable grouping follows the diagram's geometric structure rather than fixed grid boundaries: C_0 pools blank cells, C_2 covers much of the hypotenuse and part of the base, and C_1 captures the middle portion of the vertical leg. The cell containing the right-angle marker forms a separate group, C_3. Each group defines a Coarse token and its C2F entry; parentheses indicate the number of Fine members.

Thus compression and recall share support $\Pi(j)$, with Fine access count $|\mathcal{F}_t| = \sum_{j \in \mathcal{A}_t} c_j$.

The same weights construct the Coarse cache at decoder layer ℓ:

$$\mathbf{K}^{\mathrm{C},\ell} = \mathbf{W}^\top \mathbf{K}^{\mathrm{F},\ell}, \qquad \mathbf{V}^{\mathrm{C},\ell} = \mathbf{W}^\top E_\phi(\mathbf{V}^{\mathrm{F},\ell}). \tag{5}$$

Keys are pooled directly and Values pass through the residual encoder E_ϕ before pooling. Recall uses the original Fine KV.

3.3 TRACE: Temporal Routing for Adaptive Contextual Evidence

At fixed token intervals, TRACE uses causally available backbone states and the previous active set to propose upcoming evidence and decide whether to retain or replace the current set.

Represent decoding history. A learned combination and projection of hidden states across backbone layers forms $\mathbf{u}_t \in \mathbb{R}^{d_s}$. Inspired by selective state-space modeling (Gu & Dao, 2023), B recurrent branches retain decoding history at input-dependent timescales, starting from $\mathbf{s}_0^{(m)} = \mathbf{0}$:

$$\mathbf{s}_t^{(m)} = \boldsymbol{\rho}_t^{(m)} \odot \mathbf{s}_{t-1}^{(m)} + (\mathbf{1} - \boldsymbol{\rho}_t^{(m)}) \odot f(\mathbf{u}_t). \tag{6}$$

Here $\odot$ denotes elementwise multiplication, $\boldsymbol{\rho}_t^{(m)} \in (0,1)^{d_s}$ controls retention, and f supplies new state content. A softmax mixture combines the branches as $\mathbf{z}_t = \sum_{m=1}^{B} \beta_t^{(m)} \mathbf{s}_t^{(m)}$. Appendix A.2.1 gives the parameterization and history analysis.

Propose evidence and decide when to replace it. Scaled dot-product scores $r_{t,j}$ match $\mathbf{z}_t$ to Coarse descriptors. A learned boundary $e(\mathbf{z}_t)$ yields membership probabilities $p_{t,j} = \mathrm{sigmoid}(r_{t,j} - e(\mathbf{z}_t))$ and the proposal $\widehat{\mathcal{A}}_t = \{j \in [M] : p_{t,j} > 1/2\}$ (see Appendix A.2.2). Conditioned on the temporal state, the gate compares retained and proposed evidence and selects a HOLD or UPDATE action g_t:

$$\mathcal{A}_t = \begin{cases} \mathcal{A}_{t-1}, & g_t = \textsc{Hold}, \\ \widehat{\mathcal{A}}_t, & g_t = \textsc{Update}. \end{cases} \tag{7}$$

HOLD preserves evidence; UPDATE can add, remove, or clear groups, allowing Coarse-only reasoning between Fine accesses. The recurrent state evolves under both actions. Appendix A.2.2 defines the gate's evidence summaries.

Attend to selected evidence with Joint-KV. For generation forward pass n, let $t(n)$ be the latest routing decision before that pass. Writing $\mathrm{KV} = (\mathbf{K}, \mathbf{V})$, Joint-KV concatenates Coarse context, active Fine evidence, and available text history at each decoder layer:

$$\mathrm{KV}_n^{\mathrm{J}} = [\mathrm{KV}^{\mathrm{C}}; \mathrm{KV}^{\mathrm{F}}[\mathcal{F}_{t(n)}]; \mathrm{KV}_n^{\mathrm{T}}]. \tag{8}$$

During decoding after prefill, the frozen backbone attends jointly to these entries to compute the next-token distribution $p_{\theta_{\mathrm{Frozen}},\phi,\psi}(y_k \mid I, x, y_{<k})$. Algorithm 1 summarizes the memory–access cycle.

3.4 Learning Compression and Evidence Access

With the backbone frozen, we first train DART parameters ϕ, then fix DART and train TRACE parameters ψ through supervised initialization and policy optimization.

Distill compact visual memory. DART learns capacities, affinities, and Value aggregation from task-verified responses using token cross-entropy and KL distillation (Hinton et al., 2015) from full-visual teacher predictions under identical teacher-forced prefixes. Forward computation uses discrete groups; a capacity-constrained Sinkhorn surrogate supplies cross-group gradients (Cuturi, 2013). Appendices A.1.1–A.1.2 give the gradient construction and allocation analysis; Appendix A.3 specifies the loss.

Supervise upcoming evidence. TRACE observes only available hidden states and active-set history; future annotations supply supervision. Let $\mathcal{E}_s$ denote segment s's teacher-derived Fine-token set. For the segment $s(t)$ aligned with the next token to be processed at routing time t, the target is

$$\mathcal{A}_t^\star = \{j \in [M] : \Pi(j) \cap \mathcal{E}_{s(t)} \neq \varnothing\}. \tag{9}$$

Region BCE initializes set prediction; joint training adds gate cross-entropy, targeting UPDATE when the annotated set changes and HOLD otherwise.

Optimize evidence access. TRACE is then optimized on student-generated trajectories using RLOO (Ahmadian et al., 2024) correctness feedback and success-conditioned feedback on mean Fine occupancy. Online teacher supervision guides upcoming region selections. Appendix A.3 gives the complete objectives, occupancy definition, and action sampling.

4 Experiments

4.1 Experimental Setup

Benchmarks and baselines. We evaluate reasoning (Tab. 1) and general visual understanding (Tab. 2); Appendix C.1 lists the benchmarks and prompting conventions. Full denotes the uncompressed backbone. The one-shot baselines are FastV (Chen et al., 2024a), VisionZip (Yang et al., 2025a), and DivPrune (Alvar et al., 2025). We also evaluate DSTP (Kim et al., 2026), adapting its attention-change access rule to ViMoD's DART memory and frozen backbone.

Model and configurations. We use a frozen `Qwen3-VL-4B-Instruct` (Bai et al., 2025) backbone. ViMoD adds trainable modules containing only 0.0546% of the backbone parameters. The 9:1 and 4:1 configurations retain nominal Coarse fractions of 11.1% and 25%, respectively. TRACE is trained only with the 9:1 compressor and reused directly with the separately trained 4:1 compressor.

Training. Following Sec. 3.4, we train separate 4:1 and 9:1 DART compressors on the same verified response trajectories. `Qwen3.8-27B` (Qwen Team, 2026) provides TRACE's supervised annotations and online RL evidence; RL questions come from the supervised set. Training and evaluation sets are disjoint. Appendices B and E give the training recipe and annotation prompts, respectively.

Visual budgets. The target budgets are 20%, 30%, and 40% of the original visual-token count. Static methods retain the specified fraction. ViMoD counts persistent Coarse and active Fine tokens at each generation forward pass, averages within each example (Appendix C.3.1, Eq. (33)), then averages examples equally. Fine tokens count throughout HOLD intervals. Calibration matches target average occupancy (see Appendix C.2 for calibration details); realized occupancy varies across examples and benchmarks. Measured occupancy and full inference costs are reported separately; Appendix C.3.2 defines the accounting.

Evaluation. All methods use greedy decoding with zero presence penalty, with output limits matched across methods within each comparison. We use VLMEvalKit scoring (Duan et al., 2024); Tabs. 1 and 2 define normalization and averaging. Our decoding settings differ from the backbone's recommended configuration for visual tasks (see Appendix C.1 for the full protocol and discussion of comparability with reported backbone scores).

Table 1: Mathematical, logical, and multidisciplinary reasoning. Parenthesized percentages are relative to Full; Avg. equally weights eight normalized scores. Bold and underlining mark the best and second-best scores within each target budget; ‡ denotes explicit CoT prompting. ViMoD budgets specify average Coarse-plus-Fine occupancy. † denotes our adaptation of DSTP's attention-change access rule to the same DART memory as ViMoD.

Method	Mathematics					Logic		Multidisciplinary	Avg. (%)
	We-Math‡	DynaMath‡	MathVerse‡	MathVista	MathVision	LogicVista	VisualPuzzles‡	MMMU-Pro‡	
Full (100%)	45.62(100.0%)	61.88(100.0%)	58.60(100.0%)	72.80(100.0%)	47.40(100.0%)	37.36(100.0%)	19.18(100.0%)	30.92(100.0%)	100.00
Retain 20% Visual Tokens									
FastV (ECCV24)	16.95(37.2%)	32.08(51.8%)	37.31(63.7%)	43.40(59.6%)	35.56(75.0%)	22.82(61.1%)	11.47(59.8%)	10.46(33.8%)	55.25
VisionZip (CVPR25)	17.43(38.2%)	35.33(57.1%)	37.79(64.5%)	46.30(63.6%)	33.91(71.5%)	19.69(52.7%)	8.99(46.9%)	9.88(32.0%)	53.31
DivPrune (CVPR25)	23.52(51.6%)	39.06(63.1%)	41.75(71.2%)	50.20(69.0%)	35.36(74.6%)	22.15(59.3%)	8.90(46.4%)	8.38(27.1%)	57.78
DSTP† (9:1) (ECCV26)	34.89(76.5%)	48.20(77.9%)	49.21(84.0%)	61.00(83.8%)	38.82(81.9%)	26.85(71.9%)	12.93(67.4%)	12.49(40.4%)	72.96
ViMoD (9:1)	**37.24**(81.6%)	**51.14**(82.6%)	**50.25**(85.8%)	**65.30**(89.7%)	**40.43**(85.3%)	**28.19**(75.5%)	**15.15**(79.0%)	**19.54**(63.2%)	**80.33**
Retain 30% Visual Tokens									
FastV (ECCV24)	22.76(49.9%)	36.83(59.5%)	41.95(71.6%)	49.40(67.9%)	36.18(76.3%)	21.48(57.5%)	11.22(58.5%)	13.99(45.2%)	60.80
VisionZip (CVPR25)	28.10(61.6%)	45.43(73.4%)	46.62(79.6%)	50.60(69.5%)	37.50(79.1%)	24.83(66.5%)	12.07(62.9%)	18.73(60.6%)	69.14
DivPrune (CVPR25)	29.52(64.7%)	45.71(73.9%)	46.42(79.2%)	54.70(75.1%)	36.48(77.0%)	23.04(61.7%)	11.99(62.5%)	11.79(38.1%)	66.53
DSTP† (9:1) (ECCV26)	35.86(78.6%)	50.12(81.0%)	51.22(87.4%)	62.00(85.2%)	38.75(81.8%)	29.31(78.5%)	14.12(73.6%)	13.99(45.2%)	76.41
DSTP† (4:1) (ECCV26)	37.70(82.6%)	55.45(89.6%)	52.94(90.3%)	64.80(89.0%)	40.72(85.9%)	31.10(83.2%)	16.18(84.4%)	21.33(69.0%)	84.26
ViMoD (9:1)	37.81(82.9%)	53.65(86.7%)	53.12(90.6%)	**67.50**(92.7%)	**41.71**(88.0%)	31.10(83.2%)	16.10(83.9%)	**25.03**(81.0%)	86.14
ViMoD (4:1)	**40.29**(88.3%)	**57.03**(92.2%)	**53.55**(91.4%)	67.10(92.2%)	41.18(86.9%)	**31.77**(85.0%)	**16.87**(88.0%)	24.05(77.8%)	**87.71**
Retain 40% Visual Tokens									
FastV (ECCV24)	30.76(67.4%)	41.54(67.1%)	46.27(79.0%)	50.70(69.6%)	40.49(85.4%)	24.83(66.5%)	11.56(60.3%)	18.09(58.5%)	69.23
VisionZip (CVPR25)	37.43(82.0%)	52.55(84.9%)	51.09(87.2%)	57.40(78.8%)	42.50(89.7%)	33.11(88.6%)	14.38(75.0%)	25.84(83.6%)	83.73
DivPrune (CVPR25)	34.38(75.4%)	50.70(81.9%)	50.51(86.2%)	57.70(79.3%)	**43.16**(91.1%)	28.41(76.0%)	11.82(61.6%)	15.32(49.5%)	75.13
DSTP† (9:1) (ECCV26)	37.24(81.6%)	51.20(82.7%)	52.01(88.8%)	64.10(88.0%)	39.90(84.2%)	29.75(79.6%)	15.75(82.1%)	17.63(57.0%)	80.51
DSTP† (4:1) (ECCV26)	39.81(87.3%)	55.97(90.4%)	53.71(91.7%)	66.10(90.8%)	41.02(86.5%)	31.54(84.4%)	**17.38**(90.6%)	21.27(68.8%)	86.32
ViMoD (9:1)	39.90(87.5%)	55.85(90.3%)	**54.01**(92.2%)	66.80(91.8%)	41.35(87.2%)	33.33(89.2%)	15.92(83.0%)	**27.69**(89.6%)	88.83
ViMoD (4:1)	**41.43**(90.8%)	**58.76**(95.0%)	53.86(91.9%)	**67.50**(92.7%)	42.04(88.7%)	**34.90**(93.4%)	17.12(89.3%)	26.82(86.7%)	**91.06**

4.2 Main Results

Reasoning under limited visual budgets. ViMoD leads the normalized reasoning average at every target budget and all eight benchmarks at 20% budget (Tab. 1). Its average at 20% is 39.0% higher than the strongest one-shot baseline, DivPrune. Among the tested budgets, the best one-shot method *requires 40% to reach ViMoD's performance at 20%*. The diminishing margin at larger budgets suggests that dynamic access is most valuable when a fixed context cannot cover shifting evidence needs (Figs. 1–2); a larger one-shot selection is more likely to include evidence needed later in the trajectory.

With the same 9:1 DART memory and frozen backbone, TRACE improves the normalized reasoning average over DSTP† by 7.37–9.73 percentage points, supporting an access policy learned from decoding history and upcoming-evidence supervision.

General visual understanding and document–chart QA. ViMoD also leads the general-suite average at every budget, with its 20% performance exceeding all one-shot results at 40% (Tab. 2). The smaller gains over one-shot methods than on reasoning tasks under tight budgets may reflect more stable visual requirements in brief responses, where a well-preserved initial representation can suffice; extended reasoning offers more scope for adaptive evidence access. Within this suite, gains are particularly pronounced on documents and charts. At 20%, DocVQA and ChartQA improve by 19.59 and 28.16 percentage points over their respective strongest one-shot baselines. These gains may reflect DART's adaptive granularity: finer groups in text-bearing regions and stronger aggregation of blank regions help preserve compression-sensitive detail within a fixed Coarse-token budget (Fig. 5(a)).

Persistent Coarse context and Fine access. TRACE transfers directly from 9:1 to separately trained 4:1 memory without retraining, improving both suite averages at 30% and 40%. At both budgets, DocVQA favors 4:1, whereas MMMU-Pro favors 9:1. These preferences suggest a task-dependent balance between richer persistent Coarse context and the remaining Fine-access budget, which ViMoD accommodates with a shared policy across memory configurations.

4.3 Ablation Studies

We ablate the 9:1 configuration on DynaMath and MMMU-Pro. All scores are accuracies (%). The recall ablation uses the 30% reference setting; realized occupancy may differ across variants.

Table 2: General visual understanding and document–chart QA. Avg. equally weights ten Full-normalized metric columns. OCR EN/CN denote OCRBench v2 English/Chinese; MME-P/C report official perception/cognition scores. The DSTP adaptation (†), parenthesized percentages, and highlighting follow Tab. 1.

Method	Scene Understanding			General			Documents and Charts				Avg. (%)
	GQA	POPE	V*	MME-P	MME-C	MMBench	DocVQA	OCR EN	OCR CN	ChartQA	
Full (100%)	62.11(100.0%)	89.89(100.0%)	68.59(100.0%)	1693.06	610.71	83.51(100.0%)	91.68(100.0%)	44.89(100.0%)	41.40(100.0%)	80.64(100.0%)	100.00
Retain 20% Visual Tokens											
FastV (ECCV24)	50.00(80.5%)	77.24(85.9%)	56.54(82.4%)	1346.63	312.50	74.14(88.8%)	36.14(39.4%)	29.74(66.3%)	19.87(48.0%)	30.84(38.2%)	66.03
VisionZip (CVPR25)	56.41(90.8%)	84.37(93.9%)	59.16(86.3%)	1423.19	373.57	74.91(89.7%)	47.99(52.3%)	28.12(62.6%)	18.51(44.7%)	48.44(60.1%)	72.56
DivPrune (CVPR25)	58.90(94.8%)	88.10(98.0%)	59.69(87.0%)	1620.48	415.71	78.87(94.4%)	56.13(61.2%)	33.23(74.0%)	24.06(58.1%)	44.64(55.4%)	78.68
DSTP† (9:1) (ECCV26)	61.46(99.0%)	89.82(99.9%)	64.92(94.6%)	**1671.11**	**610.00**	**84.28**(100.9%)	71.62(78.1%)	40.66(90.6%)	26.40(63.8%)	74.92(92.9%)	91.84
ViMoD (9:1)	**61.65**(99.3%)	**89.84**(99.9%)	**65.45**(95.4%)	**1671.11**	**610.00**	84.11(100.7%)	**75.72**(82.6%)	**42.57**(94.8%)	**29.56**(71.4%)	**76.60**(95.0%)	**93.77**
Retain 30% Visual Tokens											
FastV (ECCV24)	54.09(87.1%)	81.90(91.1%)	58.64(85.5%)	1460.27	369.29	78.09(93.5%)	49.30(53.8%)	36.00(80.2%)	23.65(57.1%)	45.04(55.9%)	75.09
VisionZip (CVPR25)	59.05(95.1%)	88.21(98.1%)	64.40(93.9%)	1628.01	383.57	78.35(93.8%)	68.84(75.1%)	37.79(84.2%)	26.59(64.2%)	66.56(82.5%)	84.59
DivPrune (CVPR25)	60.40(97.2%)	89.01(99.0%)	63.35(92.4%)	1670.39	527.14	80.24(96.1%)	68.33(74.5%)	38.64(86.1%)	28.96(70.0%)	56.72(70.3%)	87.06
DSTP† (9:1) (ECCV26)	61.58(99.1%)	**89.84**(99.9%)	65.97(96.2%)	**1679.40**	607.86	84.11(100.7%)	76.86(83.8%)	45.19(100.7%)	31.60(76.3%)	76.64(95.0%)	95.06
DSTP† (4:1) (ECCV26)	61.61(99.2%)	89.79(99.9%)	**66.49**(96.9%)	1673.42	**612.14**	84.11(100.7%)	84.89(92.6%)	46.86(104.4%)	33.54(81.0%)	**80.36**(99.7%)	97.35
ViMoD (9:1)	61.64(99.2%)	89.82(99.9%)	65.45(95.4%)	1675.63	610.00	**84.19**(100.8%)	81.03(88.4%)	45.72(101.8%)	32.19(77.8%)	78.60(97.5%)	95.97
ViMoD (4:1)	**61.65**(99.3%)	89.73(99.8%)	64.92(94.6%)	1673.42	**612.14**	84.11(100.7%)	**85.79**(93.6%)	**47.12**(105.0%)	**34.01**(82.1%)	80.28(99.6%)	**97.38**
Retain 40% Visual Tokens											
FastV (ECCV24)	56.63(91.2%)	84.72(94.2%)	62.30(90.8%)	1578.74	371.43	80.07(95.9%)	62.02(67.6%)	41.30(92.0%)	28.07(67.8%)	58.48(72.5%)	82.62
VisionZip (CVPR25)	60.67(97.7%)	89.54(99.6%)	65.45(95.4%)	1645.17	447.50	80.67(96.6%)	79.97(87.2%)	44.86(99.9%)	32.70(79.0%)	73.84(91.6%)	91.75
DivPrune (CVPR25)	60.80(97.9%)	89.10(99.1%)	65.97(96.2%)	**1715.35**	541.43	81.36(97.4%)	76.16(83.1%)	43.00(95.8%)	32.83(79.3%)	65.60(81.3%)	92.01
DSTP† (9:1) (ECCV26)	61.78(99.5%)	89.76(99.9%)	65.97(96.2%)	1679.24	607.86	84.11(100.7%)	81.83(89.3%)	46.33(103.2%)	33.89(81.9%)	78.44(97.3%)	96.65
DSTP† (4:1) (ECCV26)	61.77(99.5%)	**89.86**(100.0%)	**66.49**(96.9%)	1675.67	610.00	84.11(100.7%)	85.28(93.0%)	48.57(108.2%)	35.08(84.7%)	**81.28**(100.8%)	98.27
ViMoD (9:1)	61.71(99.4%)	89.83(99.9%)	65.97(96.2%)	1670.38	610.00	83.93(100.5%)	84.69(92.4%)	48.08(107.1%)	34.44(83.2%)	79.36(98.4%)	97.56
ViMoD (4:1)	**61.93**(99.7%)	89.80(99.9%)	65.45(95.4%)	1674.17	**619.64**	**84.19**(100.8%)	**88.04**(96.0%)	**48.79**(108.7%)	**36.14**(87.3%)	80.92(100.3%)	**98.86**

Table 3: Fine-recall accuracy (%) with 9:1 compression at the 30% reference budget.

Setting	DynaMath	MMMU-Pro
ViMoD	**53.65**	**25.03**
w/o Fine recall	47.72	12.20
w/ single recall	50.30	16.91

Table 4: Memory construction and RL ablations: accuracy (%) at 9:1.

Setting	DynaMath			MMMU-Pro		
	20%	30%	40%	20%	30%	40%
ViMoD	**51.14**	**53.65**	**55.85**	**19.54**	**25.03**	**27.69**
w/ mean pooling	39.14	42.38	45.95	8.32	14.68	18.50
w/o RL (SFT only)	47.66	47.66	47.66	15.95	20.81	23.82

Repeated evidence access. Repeated Fine recall outperforms both no recall and single recall (Tab. 3). Relative to single recall, it improves DynaMath and MMMU-Pro by 3.35 and 8.12 percentage points, respectively. This pattern is consistent with evidence selected at one stage becoming insufficient as reasoning progresses (Fig. 1), supporting continued adaptation of the active evidence set.

Learned visual memory. DART improves over mean pooling on regular 3×3 groups on both benchmarks at every budget (Tab. 4), with gains of 11.27 and 10.35 percentage points at 30%. Fine recall remains enabled in both variants. The improvement therefore supports the value of learning how evidence is compressed and organized, even when the original Fine tokens remain recoverable.

Policy optimization. Optimizing TRACE beyond supervised initialization improves accuracy at every budget, with gains of 5.99 and 4.22 percentage points on DynaMath and MMMU-Pro at 30% (Tab. 4). These gains support complementing teacher-annotated access decisions with task feedback on the student's own reasoning trajectories.

4.4 Efficiency Analysis

ViMoD reduces total inference time relative to Full across the evaluated 20–90% budgets on MMMU-Pro (Fig. 6).

Evaluation protocol. We compare ViMoD (9:1) with FastV (Chen et al., 2024a), VisionZip (Yang et al., 2025a), and DivPrune (Alvar et al., 2025) on MMMU-Pro (Yue et al., 2025) examples at target visual-token budgets of 20–90% in increments of 10 percentage points. All methods use a single NVIDIA H200 GPU, `batch_size=8`, `max_new_tokens=8192`, and the same inference backend. Fig. 6 reports accuracy–efficiency trade-offs; Appendix C.4 details measurement and FLOP accounting.

End-to-end efficiency. At 40%, ViMoD achieves higher accuracy than all three one-shot baselines with *15.2–20.4% less total time*. Although these baselines improve decoding throughput, they

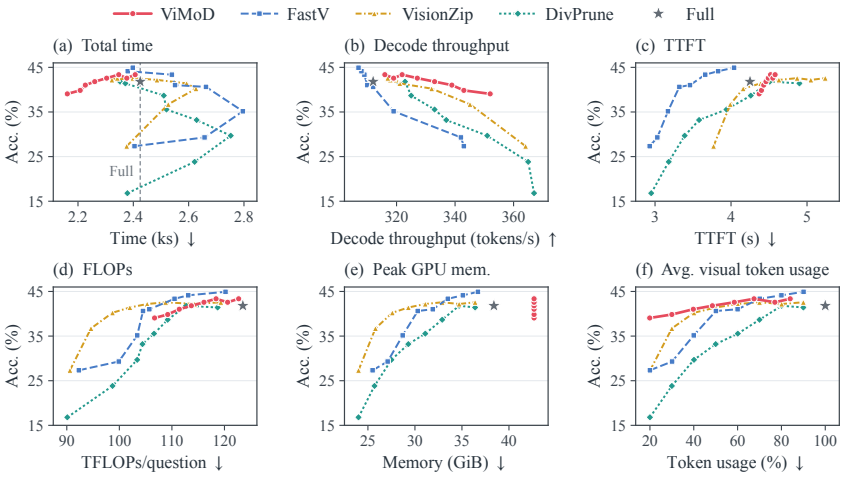

Figure 6: **Accuracy–efficiency trade-offs on MMMU-Pro.** Higher decoding throughput need not reduce total time. Lines connect eight target budgets (20–90%) in increasing order; stars denote uncompressed Full, with its total time marked by the vertical guide in (a). Total time covers the complete run; decode throughput is generated tokens divided by decoding time; TTFT is time to first token. Visual occupancy averages visual-token usage over decoding forward passes relative to Full, counting Coarse and active Fine tokens for ViMoD.

generate 15.9–29.9% more tokens than Full, offsetting their per-token savings and increasing total time relative to Full, consistent with the response-level cost analysis in Fig. 3. ViMoD keeps output length close to Full, allowing its decoding gains to translate into lower total time.

Long output trajectories make autoregressive decoding dominant (94.4% of Full's runtime in this evaluation). ViMoD retains full prefill and constructs Coarse memory, raising TTFT versus one-shot baselines at 20–50%; these startup costs are amortized over subsequent decoding.

ViMoD matches Full's accuracy at a 50% target budget versus DivPrune's 80%, using 39.5% less measured visual occupancy and 3.9% less total time.

Table 5: Network runtime (% of total), averaged over eight budgets.

DART	TRACE
0.078%	0.616%

Resource trade-offs. Retaining Fine and Coarse KV banks keeps peak memory fixed at 42.67 GiB (Full: 38.40 GiB): budgets regulate active access, while all evidence remains available for recall. DART and TRACE network computation takes 0.694% of total time (Tab. 5). Including initialization, memory processing, and dynamic-access overhead, selective Fine access still reduces total time versus Full across evaluated budgets.

5 Limitations

Constructing Coarse memory adds initialization work, and retaining both Coarse and Fine KV increases storage requirements. Short responses or sustained access to broad visual evidence may limit the decoding savings available to offset initialization and dynamic access overhead. C2F granularity, fixed routing intervals, and teacher supervision constrain evidence access: recall can activate unnecessary group members, while missed groups or delayed updates can leave useful evidence inaccessible. Generalization across backbones and robustness to alternative decoding configurations remain unverified.

6 Conclusion

ViMoD enables efficient multi-step visual reasoning through content-adaptive memory and selective evidence access. DART organizes recoverable visual detail, while TRACE learns to anticipate evidence needs from decoding history. This design improves accuracy under tight visual budgets and reduces total inference time in our evaluation, showing the value of coordinating memory organization with access throughout reasoning.

AI USE STATEMENT

LLMs were used to draft and polish the manuscript, refine scientific figures including Fig. 1, generate training responses and evidence annotations, and annotate stage-level regions for Fig. 2(b) (Sec. 3.4; Appendices B, C, and E).

REPRODUCIBILITY STATEMENT

Secs. 3–4 describe ViMoD, its training stages, and the evaluation setup. Appendix A.3 gives the training objectives; Appendix B documents the architecture, optimization settings, data sources, and data construction; and Appendix E provides the teacher annotation prompts. Appendix C describes the evaluation setup, budget calibration, and efficiency measurement protocols.

ETHICS STATEMENT

Training sources and `Qwen3.8-27B` evidence annotations are documented in Appendices B.1, B, and E. Teacher or backbone errors and missed visual evidence can affect answers. Source images and documents may carry privacy or redistribution restrictions; the reported benchmarks do not establish suitability for sensitive uses.

REFERENCES

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267, 2024.

Saeed Ranjbar Alvar, Gursimran Singh, Mohammad Akbari, and Yong Zhang. DivPrune: Diversity-based visual token pruning for large multimodal models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9392–9401. IEEE, 2025.

Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. *arXiv preprint arXiv:1711.06104*, 2017.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

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. In *European Conference on Computer Vision*, pp. 19–35. Springer, 2024a.

Qiguang Chen, Libo Qin, Jin Zhang, Zhi Chen, Xiao Xu, and Wanxiang Che. M3CoT: A novel benchmark for multi-domain multi-step multi-modal chain-of-thought. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8199–8221, 2024b.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in Neural Information Processing Systems*, 26, 2013.

Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 764–773. IEEE, 2017.

Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, et al. Molmo and PixMo: Open weights and open data for state-of-the-art vision-language models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 91–104. IEEE, 2025.

Chenlong Deng, Zhisong Zhang, Kelong Mao, Shuaiyi Li, Tianqing Fang, Hongming Zhang, Haitao Mi, Dong Yu, and Zhicheng Dou. UniGist: Towards general and hardware-aligned sequence-level long context compression. *Advances in Neural Information Processing Systems*, 38:19775–19797, 2026.

Haodong Duan, Junming Yang, Yuxuan Qiao, Xinyu Fang, Lin Chen, Yuan Liu, Xiaoyi Dong, Yuhang Zang, Pan Zhang, Jiaqi Wang, et al. VLMEvalKit: An open-source toolkit for evaluating large multi-modality models. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 11198–11201, 2024.

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, et al. MME: A comprehensive evaluation benchmark for multimodal large language models. *Advances in Neural Information Processing Systems*, 38, 2026a.

Ling Fu, Zhebin Kuang, Jiajun Song, Mingxin Huang, Biao Yang, Yuzhe Li, Linghao Zhu, Qidi Luo, Xinyu Wang, Hao Lu, et al. OCRBench v2: An improved benchmark for evaluating large multimodal models on visual text localization and reasoning. *Advances in Neural Information Processing Systems*, 38, 2026b.

Jiahui Gao, Renjie Pi, Jipeng Zhang, Jiacheng Ye, Wanjun Zhong, Yufei Wang, Lanqing Hong, Jianhua Han, Hang Xu, Zhenguo Li, et al. G-LLaVA: Solving geometric problem with multi-modal large language model. In *International Conference on Learning Representations*, volume 2025, pp. 3490–3511, 2025.

Philippe Gervais, Anastasiia Fadeeva, and Andrii Maksai. MathWriting: A dataset for handwritten mathematical expression recognition. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 5459–5469, 2025.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Zhuangcheng Gu, Guang Liang, Bin Wang, Zhiyuan Zhao, Qintong Zhang, Weijia Li, Chao Xu, Bo Zhang, Botian Shi, Jiang Wu, Wentao Zhang, and Conghui He. UniMERNet: A universal network for real-world mathematical expression recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 34106–34115, June 2026.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, et al. GLM-4.5V and GLM-4.1V-Thinking: Towards versatile multimodal reasoning with scalable reinforcement learning. *arXiv preprint arXiv:2507.01006*, 2025.

Lianyu Hu, Xiaoyu Ma, Zeqin Liao, and Yang Liu. TVI-CoT: Text-visual interleaved chain-of-thought reasoning for multimodal understanding. *arXiv preprint arXiv:2606.08464*, 2026.

Drew A Hudson and Christopher D Manning. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6693–6702. IEEE, 2019.

Seil Kang, Jinyeong Kim, Junhyeok Kim, and Seong Jae Hwang. See what you are told: Visual attention sink in large multimodal models. In *International Conference on Learning Representations*, volume 2025, pp. 87676–87703, 2025a.

Seil Kang, Jinyeong Kim, Junhyeok Kim, and Seong Jae Hwang. Your large vision-language model only needs a few attention heads for visual grounding. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9339–9350. IEEE, 2025b.

Jiwan Kim, Kibum Kim, Wonjoong Kim, Byung-Kwan Lee, and Chanyoung Park. Why and when visual token pruning fails? A study on relevant visual information shift in MLLMs decoding. In *European Conference on Computer Vision*, pp. 244–263. Springer, 2026.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual Genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73, 2017.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 292–305, 2023a.

Zhuowan Li, Xingrui Wang, Elias Stengel-Eskin, Adam Kortylewski, Wufei Ma, Benjamin Van Durme, and Alan L Yuille. Super-CLEVR: A virtual benchmark to diagnose domain robustness in visual reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14963–14973, 2023b.

Shijie Lian, Changti Wu, Laurence Tianruo Yang, Hang Yuan, Bin Yu, Lei Zhang, and Kai Chen. Euclid's gift: Enhancing spatial perception and reasoning in vision-language models via geometric surrogate tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9824–9835, 2026.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, et al. MMBench: Is your multi-modal model an all-around player? In *European Conference on Computer Vision*, pp. 216–233. Springer, 2024.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521, 2022.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. MathVista: Evaluating mathematical reasoning of foundation models in visual contexts. In *International Conference on Learning Representations*, volume 2024, pp. 23439–23554, 2024.

Qiyanhui Lu, Han Wu, Rongjian Xu, Tingzhang Luo, Cheng Fan, Xinghao Chen, Minjing Dong, Jufeng Yang, and Jianyuan Guo. RoRA: Role-oriented regional allocation for visual token pruning in MLLMs. *arXiv preprint arXiv:2608.07088*, 2026.

Yuzhen Mao, Michael Y Li, and Emily B Fox. Simplified sparse attention via gist tokens. *arXiv preprint arXiv:2604.20920*, 2026.

Ahmed Masry, Jia Qing Tan, Shafiq Joty, Enamul Hoque, et al. ChartQA: A benchmark for question answering about charts with visual and logical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 2263–2279, 2022.

Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. DocVQA: A dataset for VQA on document images. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 2199–2208. IEEE, 2021.

Nitesh Methani, Pritha Ganguly, Mitesh M Khapra, and Pratyush Kumar. PlotQA: Reasoning over scientific plots. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1516–1525. IEEE, 2020.

Jesse Mu, Xiang Li, and Noah Goodman. Learning to compress prompts with gist tokens. *Advances in Neural Information Processing Systems*, 36:19327–19352, 2023.

Nibal Nayef, Yash Patel, Michal Busta, Pinaki Nath Chowdhury, Dimosthenis Karatzas, Wafa Khlif, Jiri Matas, Umapada Pal, Jean-Christophe Burie, Cheng-lin Liu, et al. ICDAR2019 robust reading challenge on multi-lingual scene text detection and recognition—RRC-MLT-2019. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1582–1587. IEEE, 2019.

Runqi Qiao, Qiuna Tan, Guanting Dong, Minhui Wu, Chong Sun, Xiaoshuai Song, Jiapeng Wang, Zhuoma Gongque, Shanglin Lei, Yifan Zhang, et al. We-Math: Does your large multimodal model achieve human-like mathematical reasoning? In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 20023–20070, 2025.

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork, August 2026. URL `https://qwen.ai/blog?id=qwen3.8`.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards VQA models that can read. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8309–8318. IEEE, 2019.

Baiyang Song, Jun Peng, Yuxin Zhang, Guangyao Chen, Feidiao Yang, and Jianyuan Guo. KTV: Keyframes and key tokens selection for efficient training-free video LLMs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 9060–9068, 2026.

Yueqi Song, Tianyue Ou, Yibo Kong, Zecheng Li, Graham Neubig, and Xiang Yue. VisualPuzzles: Decoupling multimodal reasoning evaluation from domain knowledge. *arXiv preprint arXiv:2504.10342*, 2025.

Kimi Team, Angang Du, Bohong Yin, Bowei Xing, Bowen Qu, Bowen Wang, Cheng Chen, Chenlin Zhang, Chenzhuang Du, Chu Wei, et al. Kimi-VL technical report. *arXiv preprint arXiv:2504.07491*, 2025.

Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. Measuring multimodal mathematical reasoning with MATH-Vision dataset. *Advances in Neural Information Processing Systems*, 37:95095–95169, 2024.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. InternVL3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025.

Penghao Wu and Saining Xie. V*: Guided visual search as a core mechanism in multimodal LLMs. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13084–13094. IEEE, 2024.

Yijia Xiao, Edward Sun, Tianyu Liu, and Wei Wang. LogicVista: Multimodal LLM logical reasoning benchmark in visual contexts. *arXiv preprint arXiv:2407.04973*, 2024.

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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19792–19802. IEEE, 2025a.

Yue Yang, Ajay Patel, Matt Deitke, Tanmay Gupta, Luca Weihs, Andrew Head, Mark Yatskar, Chris Callison-Burch, Ranjay Krishna, Aniruddha Kembhavi, et al. Scaling text-rich image understanding via code-guided synthetic multimodal data generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 17486–17505, 2025b.

Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. Modeling context in referring expressions. In *European Conference on Computer Vision*, pp. 69–85. Springer, 2016.

Xiang Yue, Tianyu Zheng, Yuansheng Ni, Yubo Wang, Kai Zhang, Shengbang Tong, Yuxuan Sun, Botao Yu, Ge Zhang, Huan Sun, et al. MMMU-Pro: A more robust multi-discipline multimodal understanding benchmark. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15134–15186, 2025.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32, 2019.

Evelyn Zhang, Fufu Yu, Aoqi Wu, Zichen Wen, Ke Yan, Shouhong Ding, Biqing Qi, and Linfeng Zhang. D^2Pruner: Debiased importance and structural diversity for MLLM token pruning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 12412–12420, 2026a.

Kaichen Zhang, Keming Wu, Zuhao Yang, Bo Li, Kairui Hu, Bin Wang, Xingxuan Li, and Lidong Bing. OpenMMReasoner: Pushing the frontiers in multimodal reasoning with an open and general recipe. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19276–19286, 2026b.

Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Yu Qiao, et al. MathVerse: Does your multi-modal LLM truly see the diagrams in visual math problems? In *European Conference on Computer Vision*, pp. 169–186. Springer, 2024.

Renrui Zhang, Xinyu Wei, Dongzhi Jiang, Ziyu Guo, Yichi Zhang, Chengzhuo Tong, Jiaming Liu, Aojun Zhou, Shanghang Zhang, Gao Peng, et al. MAVIS: Mathematical visual instruction tuning with an automatic data engine. In *International Conference on Learning Representations*, volume 2025, pp. 87955–87989, 2025.

Rui Zhang, Yongsheng Zhou, Qianyi Jiang, Qi Song, Nan Li, Kai Zhou, Lei Wang, Dong Wang, Minghui Liao, Mingkun Yang, et al. ICDAR 2019 robust reading challenge on reading Chinese text on signboard. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1577–1581. IEEE, 2019.

Ziwei Zheng, Minghao Yang, Jack Hong, Chenxiao Zhao, Guohai Xu, Le Yang, and Chao Shen. DeepEyes: Incentivizing "thinking with images" via reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 126775–126798, 2026.

Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. Visual7W: Grounded question answering in images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4995–5004, 2016.

Chengke Zou, Xingang Guo, Rui Yang, Junyu Zhang, Bin Hu, and Huan Zhang. DynaMath: A dynamic visual benchmark for evaluating mathematical reasoning robustness of vision language models. In *International Conference on Learning Representations*, volume 2025, pp. 48337–48383, 2025.

Supplementary Material

CONTENTS

A Method Details and Derivations **16**

A.1 DART: Memory Construction 16

A.2 TRACE: Temporal Routing and Evidence Access 17

A.3 Training Objectives 18

B Training Details **20**

B.1 Training Data Sources 20

B.2 Architecture and Optimization Setup 20

B.3 Stagewise Training Procedure 22

C Evaluation Protocols and Efficiency Accounting **23**

C.1 Benchmarks and Metrics 23

C.2 Visual-Budget Calibration 23

C.3 Cost Model and FLOP Accounting 24

C.4 Efficiency Measurement Protocol 24

C.5 Diagnostic Settings 25

D Qualitative Analysis **26**

E Teacher Annotation Prompts **27**

A METHOD DETAILS AND DERIVATIONS

We derive the capacity-allocation and temporal-memory properties, specify the discrete grouping surrogate and learning objectives, and summarize the evidence-access algorithm.

Notation	Meaning
$\mathbf{X}^{\mathrm{F}} \in \mathbb{R}^{N\times d}$, $\mathbf{X}^{\mathrm{C}} \in \mathbb{R}^{M\times d}$	Fine and Coarse feature matrices
$\mathbf{W} \in \mathbb{R}^{N\times M}$, $\Pi(j) = \mathcal{G}_j$	Pooling weights and group-to-Fine index map
$\widetilde{c}_j$, c_j, U	Relaxed group size, integer group size, upper bound
$\mathbf{K}^{\mathrm{F}}, \mathbf{V}^{\mathrm{F}}, \mathbf{K}^{\mathrm{C}}, \mathbf{V}^{\mathrm{C}}$	Fine and Coarse caches
$\mathcal{A}_t$, $\mathcal{F}_t$, g_t	Active groups, active Fine indices, gate action
$\mathbf{s}_t^{(m)}$, $\mathbf{z}_t$	Branch state and mixed temporal state
n, $t(n)$, T_i	Response-generation forward pass, latest routing step, count
C_i	Mean Fine occupancy during response generation
θ_{Frozen}, ϕ, ψ	Frozen backbone parameters, DART parameters, TRACE parameters

A.1 DART: MEMORY CONSTRUCTION

A.1.1 CAPACITY-CONSTRAINED ASSIGNMENT AND POOLING

Feasible capacities and integer rounding. Let $c_j^{(0)}$ be the number of Fine tokens in initial grid region j. For an integer upper bound $U > 1$, the capacity prior is

$$\eta_j = \min\left\{1-\epsilon_b,\, \max\left\{\epsilon_b,\, \frac{c_j^{(0)}-1}{U-1}\right\}\right\}, \qquad b_j = \log\frac{\eta_j}{1-\eta_j}, \tag{10}$$

where $0 < \epsilon_b < \frac{1}{2}$ keeps the prior finite at either bound. The constraint in Equation (2) is feasible for $M \le N \le MU$. Taking floors and assigning the remaining slots to the largest fractional remainders produces integer capacities satisfying $1 \le c_j \le U$ and $\sum_j c_j = N$.

Discrete assignment. Fine tokens are processed in descending assignment confidence, defined by the gap between their largest and second-largest affinity scores. Each token enters its highest-scoring group with remaining capacity. Total capacity equals N, so this procedure assigns every token and fills every group.

Capacity-constrained relaxation. For fixed continuous capacities $\widetilde{\mathbf{c}} = (\widetilde{c}_1, \ldots, \widetilde{c}_M)^\top$, the entropy-regularized assignment problem is

$$\begin{aligned} \max_{\mathbf{P}\ge 0} \quad & \langle \mathbf{S}, \mathbf{P}\rangle + \varepsilon\mathcal{H}(\mathbf{P}) \\ \text{subject to} \quad & \mathbf{P}\mathbf{1}_M = \mathbf{1}_N, \qquad \mathbf{P}^\top\mathbf{1}_N = \widetilde{\mathbf{c}}, \end{aligned} \tag{11}$$

where $\mathbf{S} = (S_{ij}) \in \mathbb{R}^{N\times M}$, $\mathbf{P} \in \mathbb{R}_+^{N\times M}$, $\mathbf{1}_k$ is the length-k all-ones vector, and $\mathcal{H}(\mathbf{P}) = -\sum_{ij} P_{ij}\log P_{ij}$ with $0\log 0 = 0$. The parameter $\varepsilon > 0$ controls entropy regularization. Sinkhorn scaling alternately balances rows and columns of the exponential affinity kernel (Cuturi, 2013). Training uses a finite-iteration approximation $\widehat{\mathbf{P}}$ to these marginals. The confidence-ordered discrete assignment is a separate capacity-preserving procedure, not an exact optimizer of Equation (11).

Discrete pooling with surrogate gradients. Write $\mathbf{W}^{\mathrm{hard}}$ for the pooling matrix supported on the discrete groups. Column normalization of $\widehat{\mathbf{P}}$ gives a differentiable relaxation in a straight-through-style surrogate (Bengio et al., 2013):

$$W_{ij}^{\mathrm{soft}} = \frac{\widehat{P}_{ij}}{\sum_{k=1}^{N}\widehat{P}_{kj}}, \qquad \mathbf{W}^{\mathrm{train}} = \mathbf{W}^{\mathrm{hard}} + \mathbf{W}^{\mathrm{soft}} - \mathrm{sg}(\mathbf{W}^{\mathrm{soft}}). \tag{12}$$

The stop-gradient operator sg preserves its argument's value but contributes zero gradient during differentiation. Thus the pooling weights equal $\mathbf{W}^{\mathrm{hard}}$, while their gradient includes both the within-group softmax term and the relaxation's contribution to capacity and cross-group affinity learning. The discrete group assignments are held fixed during differentiation; recall always accesses the original Fine members indexed by C2F.

A.1.2 GROUP-CAPACITY SENSITIVITY

Sensitivity under a fixed total. Assume $U > 1$, $M < N < MU$, and finite scores and priors. Define $d_j = (U-1)\,\mathrm{sigmoid}'(b_j + a_j + \mu) > 0$, $\mathbf{d} = (d_1, \ldots, d_M)^\top$, and $D = \sum_j d_j$. The unique continuous allocation has Jacobian

$$J_{jk} = \frac{\partial \widetilde{c}_j}{\partial a_k} = d_j \left(\mathbf{1}_{\{j=k\}} - \frac{d_k}{D} \right), \qquad \mathbf{J} = \mathrm{diag}(\mathbf{d}) - \frac{\mathbf{d}\mathbf{d}^\top}{D}. \tag{13}$$

The matrix $\mathbf{J}$ is symmetric positive semidefinite, has nullspace $\mathrm{span}\{\mathbf{1}_M\}$, and has rank $M-1$. Its column sums vanish. For $M > 1$, increasing one score alone increases that group's capacity and decreases every other group's capacity.

Proof. The capacity sum is strictly increasing in μ, with limits M and MU, establishing the unique finite offset. Differentiating the sum constraint gives $0 = d_k + D\,\partial\mu/\partial a_k$, which yields Equation (13). For any $\mathbf{v} \in \mathbb{R}^M$, let $\bar{v}_d = D^{-1}\sum_j d_j v_j$. Then

$$\mathbf{v}^\top \mathbf{J} \mathbf{v} = \sum_j d_j (v_j - \bar{v}_d)^2 \geq 0. \tag{14}$$

Since every $d_j > 0$, equality holds exactly when all coordinates of $\mathbf{v}$ are equal. This proves the nullspace and rank statements. A common score shift is absorbed by μ, leaving the allocation unchanged.

Relative marginal-loss signal. For a differentiable relaxed loss, let $\xi_j = \partial\widetilde{\mathcal{L}}/\partial\widetilde{c}_j$. Holding other computational paths fixed, the chain rule gives

$$\left.\frac{\partial \widetilde{\mathcal{L}}}{\partial a_k}\right|_{\text{capacity}} = d_k(\xi_k - \bar{\xi}_d), \qquad \bar{\xi}_d = \frac{\sum_j d_j \xi_j}{D}. \tag{15}$$

The capacity predictor receives each region's marginal loss relative to a weighted regional mean. These derivatives characterize the continuous allocation; the shared predictor couples score updates, and discrete grouping is trained through the surrogate in Equation (12).

A.2 TRACE: TEMPORAL ROUTING AND EVIDENCE ACCESS

A.2.1 PARAMETERIZATION AND HISTORY

Branch parameterization. For Equation (6), each branch has a learned decay parameter $\boldsymbol{\theta}_m \in \mathbb{R}^{d_s}$. The input-dependent decay, shared step-size and write maps, and branch mixture are

$$\begin{aligned} \boldsymbol{\rho}_t^{(m)} &= \exp[-\,\mathrm{softplus}(\boldsymbol{\theta}_m) \odot \boldsymbol{\Delta}(\mathbf{u}_t)]\,, \\ \boldsymbol{\Delta}(\mathbf{u}_t) &= \mathrm{softplus}(\mathbf{W}_\Delta \mathbf{u}_t + \mathbf{b}_\Delta), \qquad f(\mathbf{u}_t) = \mathbf{W}_w \mathbf{u}_t, \\ \boldsymbol{\beta}_t &= \mathrm{softmax}(\mathbf{W}_\beta \mathbf{u}_t). \end{aligned} \tag{16}$$

Here $\mathrm{softplus}(x) = \log(1 + e^x)$ acts elementwise. Softmax acts over branches, giving positive weights that sum to one. Appendix B.2 gives the concrete architecture and timescale initialization.

How past states contribute. With zero initial state, unrolling a branch gives

$$\mathbf{s}_t^{(m)} = \sum_{\tau=1}^{t} \underbrace{\left[(\mathbf{1} - \boldsymbol{\rho}_\tau^{(m)}) \odot \prod_{k=\tau+1}^{t} \boldsymbol{\rho}_k^{(m)} \right]}_{\boldsymbol{\alpha}_{t,\tau}^{(m)}} \odot f(\mathbf{u}_\tau). \tag{17}$$

Products are elementwise and empty products equal one. The formula follows by induction: multiplication by $\boldsymbol{\rho}_t^{(m)}$ extends all preceding retention factors, and the new write adds the $\tau = t$ term. For finite parameters, $\mathbf{0} < \boldsymbol{\rho}_t^{(m)} < \mathbf{1}$, and

$$\sum_{\tau=1}^{t} \boldsymbol{\alpha}_{t,\tau}^{(m)} = \mathbf{1} - \prod_{k=1}^{t} \boldsymbol{\rho}_k^{(m)} \leq \mathbf{1}. \tag{18}$$

Each state coordinate is consequently a convex combination of historical write coordinates and its zero initial value. If $L_t = \max_{\tau \le t} \|f(\mathbf{u}_\tau)\|_\infty$, then $\|\mathbf{s}_t^{(m)}\|_\infty \le L_t$ and $\|\mathbf{z}_t\|_\infty \le L_t$. The retention factors expose how each branch weights past writes, while the learned mixture adapts their contribution to the current decision.

A.2.2 Evidence Policy and Decoding Algorithm

Proposing evidence. Projections q and k map the temporal state and Coarse descriptors to dimension d_r. A learned scalar function e sets the selection boundary:

$$r_{t,j} = \frac{q(\mathbf{z}_t)^\top k(\mathbf{x}_j^{\mathrm{C}})}{\sqrt{d_r}}, \qquad p_{t,j} = \mathrm{sigmoid}(r_{t,j} - e(\mathbf{z}_t)), \\ \widehat{\mathcal{A}}_t = \{j \in [M] : p_{t,j} > 1/2\}. \tag{19}$$

Comparing retained and proposed evidence. The gate in Equation (7) summarizes the current set and the proposed memberships in the same region-key space. With $\mathbf{k}_j = k(\mathbf{x}_j^{\mathrm{C}})$, its evidence inputs are

$$\bar{\mathbf{k}}_t^{\mathrm{act}} = \frac{\sum_{j \in \mathcal{A}_{t-1}} \mathbf{k}_j}{\max(1, |\mathcal{A}_{t-1}|)}, \qquad \bar{\mathbf{k}}_t^{\mathrm{prop}} = \frac{\sum_{j=1}^{M} p_{t,j} \mathbf{k}_j}{\epsilon_g + \sum_{j=1}^{M} p_{t,j}}. \tag{20}$$

The empty-set summary is zero and $\epsilon_g > 0$ stabilizes normalization. Together with $\mathbf{z}_t$, these summaries let the gate condition retention on both the available evidence and the newly predicted demand. At inference, g_t is the more probable HOLD or UPDATE action under π_ψ.

Initial visual context. Before generating the first response token, ViMoD switches the active visual context from full visual KV to Coarse KV and the initially selected Fine KV. The full Fine KV bank remains stored for later recall.

Algorithm 1 ViMoD inference

Require: Image I, prompt x; parameters $\theta_{\mathrm{Frozen}}, \phi, \psi$
Ensure: Generated response

Extract $\mathbf{X}^{\mathrm{F}}$; prefill and cache $\mathrm{KV}^{\mathrm{F}}, \mathrm{KV}^{\mathrm{T}}$
$(\mathbf{X}^{\mathrm{C}}, \mathrm{KV}^{\mathrm{C}}, \Pi) \leftarrow \mathrm{DART}_\phi(\mathbf{X}^{\mathrm{F}}, \mathrm{KV}^{\mathrm{F}})$ (Equations 2–5)
$\mathcal{A}_0 \leftarrow \varnothing$; $\mathbf{s}_0^{(m)} \leftarrow \mathbf{0}$ for all m
for each routing step t **do**
  Compute $\mathbf{u}_t$; update $\{\mathbf{s}_t^{(m)}\}_m, \mathbf{z}_t$ (Equation 6)
  Compute proposal $\widehat{\mathcal{A}}_t$ and gate g_t (Equations 19 and 7)
  $\mathcal{A}_t \leftarrow \mathcal{A}_{t-1}$ if $g_t = \textsc{Hold}$, otherwise $\widehat{\mathcal{A}}_t$
  $\mathcal{F}_t \leftarrow \bigcup_{j \in \mathcal{A}_t} \Pi(j)$
  Decode with Joint-KV until the next routing step or termination (Equation 8)
end for

A.3 Training Objectives

A.3.1 Compact Memory Distillation

DART matches full-visual teacher predictions on task-verified responses under identical teacher-forced prefixes:

$$\mathcal{L}_{\mathrm{DART}} = \mathcal{L}_{\mathrm{CE}} + \lambda_{\mathrm{KD}} \mathcal{L}_{\mathrm{KL}}. \tag{21}$$

The terms supervise response tokens by cross-entropy and $D_{\mathrm{KL}}(p_{\mathrm{full}} \,\|\, p_{\theta_{\mathrm{Frozen}},\phi})$, where $p_{\theta_{\mathrm{Frozen}},\phi}$ uses Coarse visual memory. Gradients train capacities, affinities, and Value aggregation through the discrete pooling surrogate in Equation (12). Appendix B gives the distillation configuration.

A.3.2 Supervised Set and Gate Learning

For segment s, $\mathcal{E}_s$ contains Fine cells with positive-area overlap with the teacher evidence boxes. One segment may supervise several routing decisions.

Set training updates TRACE parameters ψ except the gate. Let $\mathcal{P}_t = \mathcal{A}_t^\star$ and $\mathcal{Q}_t = [M] \setminus \mathcal{P}_t$ be the positive and negative region sets. The class-balanced loss at routing step t is

$$\ell_{\text{set},t} = -\frac{1}{\max(1, |\mathcal{P}_t|)} \sum_{j \in \mathcal{P}_t} \log p_{t,j} - \frac{1}{\max(1, |\mathcal{Q}_t|)} \sum_{j \in \mathcal{Q}_t} \log(1 - p_{t,j}). \tag{22}$$

Region BCE is averaged separately over positive and negative regions and summed; an absent class contributes zero. Joint training adds class-balanced gate cross-entropy. The gate target is UPDATE when consecutive annotated sets differ and HOLD otherwise. Sec. 3.4 defines the causal supervision targets, and Appendix B specifies the trainable components of each stage.

A.3.3 Policy Optimization and Online Supervision

Sampling and trajectory feedback. Policy training samples g_t from its categorical distribution. On UPDATE, region memberships are independently sampled as $m_{t,j} \sim \text{Bern}(p_{t,j})$, replacing the previous set; on HOLD, the active set persists. For $G = 16$ rollouts per question and correctness $Y_i \in \{0, 1\}$, RLOO (Ahmadian et al., 2024) gives the leave-one-out correctness advantage

$$A_i^{\text{acc}} = 2Y_i - \frac{1}{G-1} \sum_{j \neq i} 2Y_j. \tag{23}$$

For trajectory i, let N_i be its original Fine-token count, $T_i > 0$ its number of generation forward passes, and $t_i(n)$ the latest routing decision before pass n. Its mean Fine occupancy is

$$C_i = \frac{1}{N_i T_i} \sum_{n=1}^{T_i} |\mathcal{F}_{i,t_i(n)}|. \tag{24}$$

Let $\mathcal{I}_+ = \{i : Y_i = 1\}$, with costs C_i from Equation (24). When $|\mathcal{I}_+| \geq 2$, define $\bar{C}_+ = |\mathcal{I}_+|^{-1} \sum_{j \in \mathcal{I}_+} C_j$ and $A_i^C = \bar{C}_+ - C_i$ for $i \in \mathcal{I}_+$; all other cost advantages are zero. These costs measure Fine residency, including HOLD intervals, and exclude persistent Coarse tokens. Cost advantages are not divided by their standard deviation.

Define the trajectory log-probabilities

$$\begin{aligned} \ell_i^{\text{region}} &= \sum_{t: g_{i,t} = \textsc{Update}} \sum_j \Big[m_{i,t,j} \log p_{i,t,j} + (1 - m_{i,t,j}) \log(1 - p_{i,t,j}) \Big], \\ \ell_i^{\text{all}} &= \sum_t \log \pi_\psi(g_{i,t} \mid \cdot) + \ell_i^{\text{region}}. \end{aligned} \tag{25}$$

The symbol $\cdot$ includes the temporal state, previous active set, proposal probabilities, and Coarse descriptors, together with the update-specific exploration bias b (see Appendix B for the training recipe). The per-question policy loss is

$$\mathcal{L}_{\text{policy}} = -\frac{1}{G} \sum_{i=1}^{G} \left[A_i^{\text{acc}} \ell_i^{\text{all}} + 0.5 A_i^C \ell_i^{\text{region}} \right]. \tag{26}$$

The batch policy loss averages over questions, with advantages held fixed during differentiation. Correctness weights both gate and region actions; cost directly weights only region log-probabilities, although shared representations can also change gate behavior. Success-conditioned cost feedback imposes no hard accuracy or computation constraint. Gradients propagate through recurrent router states, while backbone hidden states and observed trajectory histories are fixed observations.

Online teacher guidance. The teacher sees the image, question, and generated prefix, but neither the reference answer nor future student text. Its boxes specify a complete working set for approximately the next 16 student tokens. After the mapping in Appendix B, let $\mathcal{V}_{i,t}$ be the valid regions and

$y_{i,t,j} \in \{0, 1\}$ their labels. For annotations containing both positive and negative regions, we use binary cross-entropy with $p_{i,t,j} = \mathrm{sigmoid}(r_{i,t,j} - e(\mathbf{z}_{i,t}))$:

$$\ell_{\mathrm{Teacher},i,t} = -\frac{1}{|\mathcal{V}_{i,t}|} \sum_{j \in \mathcal{V}_{i,t}} \Big[y_{i,t,j} \log p_{i,t,j} + (1 - y_{i,t,j}) \log(1 - p_{i,t,j}) \Big]. \tag{27}$$

We average over valid regions and then valid annotation positions within each trajectory, skipping empty, all-positive, and uncertain annotations. Thus an empty box list supplies no BCE gradient. Unlike SFT, this loss does not balance the classes; it supervises region scores, the selection threshold, and shared representations, and can therefore change selected-set size.

Auxiliary update signal. When all G rollouts for a question fail and none contains an autonomous UPDATE, we apply the trajectory-level auxiliary term

$$\mathcal{L}_{\mathrm{call},i} = -\log \left[1 - \prod_t (1 - q_{i,t}) \right], \tag{28}$$

where $q_{i,t}$ is the UPDATE probability at a state observed along the sampled trajectory, conditioned on the same exploration bias. This surrogate encourages updates at those states; it is not the exact probability of an update under closed-loop generation. The complete objective is

$$\mathcal{L}_{\mathrm{RL}} = \mathcal{L}_{\mathrm{policy}} + 0.3\mathcal{L}_{\mathrm{Teacher}} + 0.1\mathcal{L}_{\mathrm{call}}. \tag{29}$$

Appendix B gives the sampling, annotation, and optimization settings.

B Training Details

B.1 Training Data Sources

DART. The 57,420 trajectories cover mathematics and science (Euclid30K (Lian et al., 2026), Geo170K (Gao et al., 2025), M3CoT (Chen et al., 2024b), MAVIS (Zhang et al., 2025), ScienceQA (Lu et al., 2022)), natural-image understanding (GQA (Hudson & Manning, 2019), Visual7W (Zhu et al., 2016)), documents, charts, and tables (PixMo (Deitke et al., 2025), PlotQA (Methani et al., 2020)), OCR (MathWriting (Gervais et al., 2025), TextVQA (Singh et al., 2019), UniMER (Gu et al., 2026), ReCTS (Zhang et al., 2019), MLT2019 (Nayef et al., 2019)), and localization (RefCOCO (Yu et al., 2016), Visual Genome (Krishna et al., 2017), PixMo points (Deitke et al., 2025)).

TRACE. The 80,000 SFT examples span mathematics and science (Euclid30K (Lian et al., 2026), Geo170K (Gao et al., 2025), M3CoT (Chen et al., 2024b), MAVIS (Zhang et al., 2025), OpenMMReasoner (Zhang et al., 2026b), ScienceQA (Lu et al., 2022)), documents, charts, and tables (PixMo Documents (Deitke et al., 2025; Yang et al., 2025b), PlotQA (Methani et al., 2020)), natural images (GQA (Hudson & Manning, 2019), Visual7W (Zhu et al., 2016)), OCR (MathWriting (Gervais et al., 2025), TextVQA (Singh et al., 2019), UniMER (Gu et al., 2026)), and synthetic visual comparisons (SuperCLEVR Compare (Li et al., 2023b)). The 4,800 RL questions are a task-balanced subset of this SFT set. All training data are disjoint from every evaluation set. Fig. 7 compares the task-type proportions; the supplementary dataset breakdown provides per-source counts.

B.2 Architecture and Optimization Setup

Common setup. Training uses NVIDIA H200 GPUs. The frozen `Qwen3-VL-4B-Instruct` (Bai et al., 2025) backbone has about 4.44 billion parameters; DART and TRACE add approximately 1.185 million and 1.236 million parameters, respectively, totaling 0.0546% of the backbone. The annotation teacher is excluded from deployment counts. Tab. 6 summarizes the schedule and trainable components; Tab. 7 specifies TRACE's state-space architecture. The backbone remains frozen throughout, and DART is frozen during all TRACE training. All stages use AdamW (Loshchilov & Hutter, 2017); DART and SFT use zero weight decay. Batch sizes are global across devices and gradient accumulation.

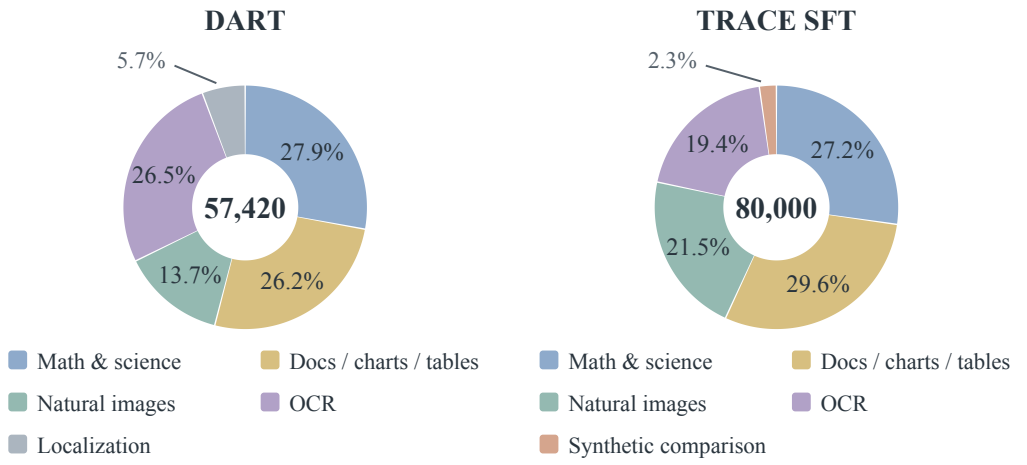

Figure 7: Training-data composition by task type. The two DART compressors share one dataset, as do the two TRACE SFT stages; the RL pool is a subset of TRACE SFT and is not counted separately.

Table 6: Training schedule and updated components. Dataset sizes count DART response trajectories, SFT examples, and RL questions. The two SFT stages share one 80,000-example set. DART is trained independently at each ratio; all TRACE stages use 9:1 compression.

Setting	**DART**	**TRACE SFT: Set**	**TRACE SFT: Joint**	**TRACE RL**
Dataset size	57,420	80,000	80,000	4,800
Training duration	1,200 steps	3 epochs	3 epochs	50 steps
Global batch size	192	64	64	96
Learning rate	10^{-5}	10^{-4}	10^{-4}	3×10^{-5}
Updated components	DART	TRACE except gate	Layer fusion, adapter, SSM, gate	Entire TRACE

Table 7: TRACE architecture and routing configuration.

Setting	**Value**
Input hidden dimension	2,560
Backbone hidden-state indices	27, 31, 35 (backbone indexing)
SSM state dimension	256
Region key/query dimension	128
Initial timescales	1, 2, 4, 8 routing steps
Routing interval	16 generated tokens

Architecture details. The two DART configurations use the same parameter dimensions, with group-size ranges $[1, 8]$ for 4:1 and $[1, 18]$ for 9:1. Assignment projections have dimension 128 and use eight balancing iterations. For each KV head, the Value encoder uses a residual transformation comprising layer normalization (Ba et al., 2016) and two linear layers separated by a Gaussian error linear unit (GELU) activation (Hendrycks & Gimpel, 2016). Its parameters are shared across decoder layers; the head dimension is 128 and the intermediate dimension is 256. TRACE's region keys and queries use root-mean-square normalization (Zhang & Sennrich, 2019) followed by a linear projection without a bias term; the selection threshold is an affine function of the state.

B.3 Stagewise Training Procedure

B.3.1 DART Distillation

The 4:1 and 9:1 compressors start from 2×2 and 3×3 regular-grid mean pooling, respectively, and use identical data and optimization settings. The frozen backbone generates responses with full visual context. We retain trajectories that pass task-specific checks of answers, recognized text, or bounding boxes. Prompts request brief evidence for visual question answering, allow reasoning for mathematics and science, and request text or normalized boxes for OCR and localization.

Under teacher forcing, the full-context backbone supplies token distributions, while the student re-encodes the text prefix with attention to the compressed Coarse visual keys and values. Equation (21) uses $\lambda_{\mathrm{KD}} = 2$ at temperature 1. We supervise at most the first 64 answer tokens, including the first token. DART training uses no warmup and a gradient-clipping threshold of 1.

B.3.2 TRACE Supervised Training

`Qwen3.8-27B` (Qwen Team, 2026) annotates the 80,000-example SFT set using greedy decoding and the offline template in Appendix E (Prompt 1). Tab. 8 distinguishes offline annotation from online supervision. We check answer correctness, output format, and coordinates before mapping boxes to compressor regions. Processing the teacher response causally through the frozen student provides hidden states, region features, and working-set history, with routing targets aligned every 16 tokens.

The Set stage uses class-balanced set binary cross-entropy (BCE). The Joint stage uses set BCE plus class-balanced gate cross-entropy (loss details in Appendix A.3). Its region key/query projections, associated normalization parameters, and selection-threshold parameters remain frozen; Tab. 6 lists the components updated in each stage.

Table 8: Offline and online evidence annotation. Both use `Qwen3.8-27B` (`temperature=0`, optional reasoning disabled) and image-relative box coordinates in $[0, 1000]$. Complete templates appear in Appendix E.

Setting	Offline SFT	Online RL
Inputs	Image and question	Image, question, and student prefix
Prediction target	Reasoning steps, evidence boxes for each upcoming step, and final answer	Evidence boxes for approximately the next 16 student tokens
Output format	XML with JSON evidence fields	Schema-constrained JSON

B.3.3 TRACE Reinforcement Learning

RL follows 9:1 joint supervised training with a newly initialized AdamW optimizer. The 4,800-question pool is curated from SFT using correctness-based filtering and task-type balancing. Screening samples four trajectories per question from the fixed 9:1 SFT model and retains groups with both successes and failures, or groups in which every trajectory fails without an autonomous UPDATE. Training uses the pool without replacement in 50 batches of 96 questions, balancing task types within each batch. Each question yields $G = 16$ newly sampled trajectories; each batch produces one optimizer update, without a critic or PPO clipping. RL rollouts use `max_new_tokens=4096`.

Online evidence supervision. The online teacher uses Prompt 2 in Appendix E, with `max_new_tokens=512` and annotations at up to 24 routing decisions per trajectory. It receives neither reference answers nor future student text. Its boxes specify a complete proposed working set.

The training objectives are defined in Appendix A.3.

Exploration. At each optimization step, we sample one gate bias $b \sim \mathcal{U}(-2.5, 2.5)$, shared by the rollout batch and held fixed when evaluating the sampled actions' probabilities for the update. This bias conditions the policy during exploration and is not an additional loss.

C Evaluation Protocols and Efficiency Accounting

C.1 Benchmarks and Metrics

The reasoning suite comprises We-Math (Qiao et al., 2025), DynaMath (Zou et al., 2025), MathVerse (Zhang et al., 2024), MathVista (Lu et al., 2024), MathVision (Wang et al., 2024), LogicVista (Xiao et al., 2024), VisualPuzzles (Song et al., 2025), and MMMU-Pro (Yue et al., 2025). The general visual understanding and document–chart QA suite comprises GQA (Hudson & Manning, 2019), POPE (Li et al., 2023a), V* (Wu & Xie, 2024), MME (Fu et al., 2026a), MMBench (Liu et al., 2024), DocVQA (Mathew et al., 2021), OCRBench v2 (Fu et al., 2026b), and ChartQA (Masry et al., 2022). Explicit CoT prompts are marked in Tab. 1.

We follow the official benchmark scoring rules as implemented in VLMEvalKit (Duan et al., 2024). We use GQA (Hudson & Manning, 2019) TestDev Balanced, MMBench (Liu et al., 2024) English DEV v1.0, and the ChartQA (Masry et al., 2022) test split.

Generation settings and reference scores. All compared methods, including Full, use greedy decoding with `presence_penalty=0`. Output limits vary across tasks and are shared by methods within each comparison. The main MMMU-Pro benchmark uses `max_new_tokens=4096`; the separate efficiency evaluation in Fig. 6 uses 8,192 (see Appendix C.4).

The `Qwen3-VL-4B-Instruct` model card recommends sampling for visual tasks with temperature 0.7, top-p 0.8, top-k 20, repetition penalty 1.0, presence penalty 1.5, and an output limit of 16,384 tokens.[1] Our use of greedy decoding and zero presence penalty differs from these recommendations across the evaluation suite, limiting direct comparability with externally reported backbone scores.

As one example, our Full baseline obtains 30.92% on MMMU-Pro under our evaluation protocol; the backbone's technical report lists 53.2% (Bai et al., 2025). Repetition and truncation may affect results, but we have not isolated their contribution to this gap or measured the accuracy and response-length effects of alternative settings across benchmarks. All normalized results use our measured Full baseline under the same protocol as the compared methods.

C.2 Visual-Budget Calibration

For ViMoD, average visual occupancy is calibrated separately for GQA, MME, MMBench, and ChartQA and for the remaining evaluation suite. To reach a target average total visual occupancy ν, deployment adds a scalar offset δ_ν to TRACE's learned boundary:

$$p_{t,j}^{(\nu)} = \mathrm{sigmoid}(r_{t,j} - e(\mathbf{z}_t) - \delta_\nu)\,, \qquad \widehat{\mathcal{A}}_t^{(\nu)} = \{j : p_{t,j}^{(\nu)} > \tfrac{1}{2}\}. \tag{30}$$

For a nonempty trajectory of response-generation forward passes, Fine occupancy C_i is defined in Equation (24); total visual occupancy is

$$R_i = \frac{M_i}{N_i} + C_i = \frac{1}{N_i T_i} \sum_{n=1}^{T_i} \left(M_i + |\mathcal{F}_{i,t_i(n)}|\right). \tag{31}$$

Since $0 \le C_i \le 1$, total occupancy lies in $[M_i/N_i,\ 1 + M_i/N_i]$; the persistent Coarse fraction sets a lower bound on feasible usage. Calibration adjusts δ_ν on a moderate sample until the measured average total occupancy $\widehat{R}_{\mathrm{cal}}(\delta_\nu)$ satisfies

$$\left|\widehat{R}_{\mathrm{cal}}(\delta_\nu) - \nu\right| \le \epsilon_{\mathrm{cal}}, \tag{32}$$

[1] `https://huggingface.co/Qwen/Qwen3-VL-4B-Instruct`

where ϵ_{cal} is the calibration tolerance. The offset is then fixed for evaluation. The calibrated probabilities replace $p_{t,j}$ in both candidate selection and the proposal summary in Equation (20). Holding scores fixed, increasing the offset can only remove candidate regions; closed-loop occupancy also depends on gate decisions, subsequent states, and generated length, so it need not be monotone.

C.3 Cost Model and FLOP Accounting

C.3.1 Visual Occupancy and Generation Cost

For a response of T generation forward passes, the visual-token budget measures average active occupancy relative to the N original visual tokens:

$$R = \frac{1}{NT} \sum_{n=1}^{T} \left(M + |\mathcal{F}_{t(n)}| \right). \tag{33}$$

This counts persistent Coarse tokens and active Fine tokens on every pass, including HOLD intervals. The recorded circuit case illustrates Fine access, Coarse-only reasoning, and renewed access (Fig. 8).

To relate the occupancy objective in Equation (24) to visual-attention work, assume cost $\kappa > 0$ per active visual KV entry per forward pass. With total occupancy R from Equation (33), this work is $\kappa \sum_{n=1}^{T}(M + |\mathcal{F}_{t(n)}|) = \kappa NTR$. Thus dynamic and fixed contexts with equal R incur equal modeled visual-attention work for the same image and generation length. Let B_{init} be initialization cost beyond the uncompressed baseline and $B_{\text{route}}(T)$ the routing overhead. At a common horizon of T forward passes with unchanged nonvisual computation, modeled savings are positive when

$$\underbrace{B_{\text{init}} + B_{\text{route}}(T)}_{\text{additional overhead}} < \underbrace{\kappa \sum_{n=1}^{T} (N - M - |\mathcal{F}_{t(n)}|)}_{\text{saved visual-attention work}}. \tag{34}$$

For trajectories of different lengths, let T_F and T_M denote the numbers of forward passes for the uncompressed baseline and ViMoD, and $Q_F(T_F)$ and $Q_M(T_M)$ their nonvisual response computation, including text attention. The total-cost difference becomes

$$\begin{aligned} C_{\text{ViMoD}} - C_{\text{Full}} = {} & B_{\text{init}} + B_{\text{route}}(T_M) + Q_M(T_M) - Q_F(T_F) \\ & + \kappa \left[MT_M + \sum_{n=1}^{T_M} |\mathcal{F}_{t(n)}| - NT_F \right]. \end{aligned} \tag{35}$$

This arithmetic model does not account for kernel utilization or memory traffic. Appendix C.3.2 specifies the FLOP accounting and aggregation used for evaluation; latency and memory require separate measurements.

C.3.2 Attention Accounting

Visual-attention work counts query–key and attention–value products involving active visual KV entries, using each executed layer's actual query-head count and head dimension. Each multiply-add counts as two FLOPs. Total attention additionally includes text KV interactions, initialization, and any additional attention operations used for routing. Query, key, value, and output projections, MLPs, visual encoding, and non-attention router computation are excluded from this attention-core metric. The accounting sums forward passes through each example's stopping point.

For each example, visual occupancy averages $(M + |\mathcal{F}_n|)/N$ over forward passes during response generation. Task-level occupancy averages examples equally. Visual-attention work per forward pass is first averaged within each example and then across examples. Cumulative cost is averaged per example and normalized by the corresponding task mean of the uncompressed baseline.

C.4 Efficiency Measurement Protocol

The efficiency evaluation uses MMMU-Pro examples on a single NVIDIA H200 GPU, with `batch_size=8`, `max_new_tokens=8192`, and the same inference backend for all methods. All methods use greedy decoding with zero presence penalty. This evaluation uses a longer output limit

than the main MMMU-Pro benchmark (4,096 tokens). Total time is the wall-clock duration of the complete evaluation run, rather than mean per-request latency. Decode throughput is total generated tokens divided by decoding time, reported in tokens/s. Average visual token usage measures active visual tokens relative to the uncompressed baseline, averaged over decoding forwards; it includes Coarse and active Fine tokens for ViMoD and differs from the nominal target budget. TTFT is averaged across requests and includes all processing before the first generated token. Peak GPU memory denotes the maximum GPU memory footprint recorded during inference.

TFLOPs per question are estimated analytically from matrix multiplication and convolution operations, including vision encoding, prefill, decoding, and method-specific computation:

$$\text{TFLOPs/question} = \frac{F_{\text{vision}} + F_{\text{prefill}} + F_{\text{decode}} + F_{\text{method}}}{10^{12} N}. \tag{36}$$

Each F term denotes the corresponding FLOP count accumulated over all N questions. We count two FLOPs per multiply-add and exclude elementwise operations and data movement.

For each ViMoD budget, we divide the instrumented compressor or router network time by inference time, then take the arithmetic mean of these shares over the eight budgets. These network intervals are already included in the recorded phase timings. They exclude assignment, KV pooling and packing, and router execution logic outside the networks.

C.5 Diagnostic Settings

Visual-demand variation. For the diagnostics in Figs. 1 and 2(a), we use responses generated by the full-context `Qwen3-VL-4B-Instruct` backbone. Given a response of T tokens, we partition it into N_{win} contiguous, approximately equal-length windows with boundaries $b_k = \lfloor kT/N_{\text{win}} \rfloor$ and $b_0 = 0$. Let $\mathbf{h}_i$ be the embedding of visual token i at the language-model input, and let $\mathcal{W}_k = \{b_{k-1}+1, \ldots, b_k\}$. Along the recorded generation trajectory, the gradient-times-input attribution (Ancona et al., 2017) of visual token i to window k and its within-window normalization are

$$R_{k,i} = \left\| \mathbf{h}_i \odot \nabla_{\mathbf{h}_i} \sum_{t \in \mathcal{W}_k} \log p_\theta(y_t \mid I, x, y_{<t}) \right\|_2, \qquad P_{k,i} = \frac{R_{k,i}}{\sum_j R_{k,j}}. \tag{37}$$

The gradient is taken with respect to the visual input embedding, with backbone parameters fixed; $P_{k,i}$ is an attribution score, not an attention weight. For N_{vis} visual tokens, let $m = \max(1, \lceil 0.1 N_{\text{vis}} \rceil)$ and let $\mathcal{S}_k$ contain the m tokens with the largest $P_{k,i}$. We define visual-demand variation as their mean turnover between adjacent windows:

$$V = \frac{1}{N_{\text{win}} - 1} \sum_{k=2}^{N_{\text{win}}} \left(1 - \frac{|\mathcal{S}_k \cap \mathcal{S}_{k-1}|}{m} \right). \tag{38}$$

The reported diagnostic uses $N_{\text{win}} = 16$; Fig. 1 displays selected windows, whereas V uses all of them. Fig. 2(a) groups MathVerse examples by V, computed from their full-context responses, and plots $100V$ on the horizontal axis. Each displayed heatmap is additionally scaled by its own maximum for color rendering; this display scaling is not used to compute V.

Generation-cost diagnostic. The FastV (Chen et al., 2024a) diagnostic in Fig. 3 uses 300 MathVerse (Zhang et al., 2024) testmini examples with `max_new_tokens=8192`. All retention settings use the same example set and generation cap. Early entropy is the mean next-token entropy over the first eighth of each response. Text similarity is the mean cosine similarity between embeddings of neighboring 128-token windows with stride 64, computed for responses with at least 256 tokens. Amortized decoder FLOPs per generated token are computed as total decoder FLOPs divided by output length.

Annotation-guided evidence-access comparison. For Fig. 2(b), `Qwen3.8` annotates the visual regions required at each reasoning stage on MMMU-Pro. Dynamic access retains persistent visual context alongside the selected stage-specific regions. The vertical axis reports accuracy as a percentage of the uncompressed Full model's accuracy. The budgets marked at 70% are obtained by linear interpolation between neighboring plotted points.

D QUALITATIVE ANALYSIS

Fig. 8 shows resistor-label access ($n = 80$), Coarse-only calculation ($n = 160$–192), and renewed voltage-label access ($n = 208$), with 35 Coarse tokens always active. This sequence illustrates the access pattern in Sec. 3 without isolating the causal effect of individual accesses.

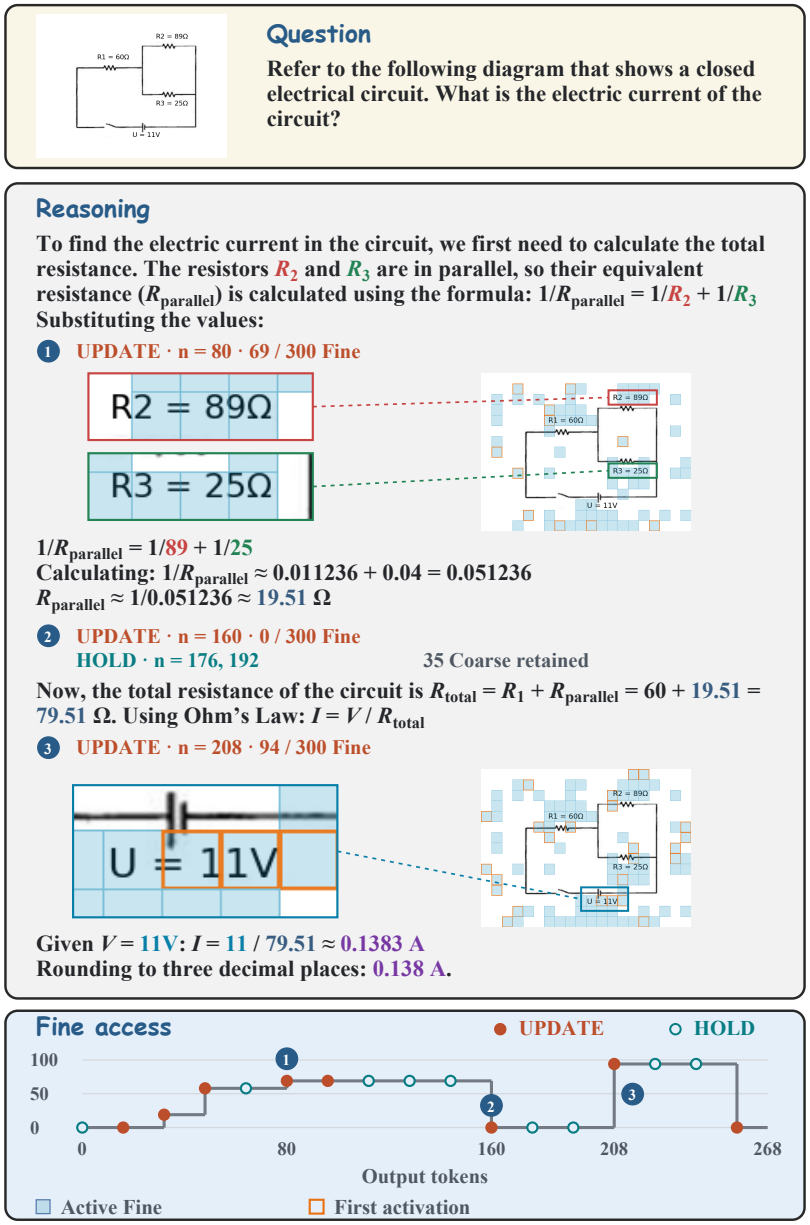

Figure 8: **Observed ViMoD access during circuit reasoning.** The recorded solution text is shown in full; numbered access annotations are editorial and placed at nearby phrase boundaries. At each routing decision, n denotes the number of generated tokens processed; $k/300$ reports k active Fine tokens out of the original 300. UPDATE replaces the active Fine set, whereas HOLD retains it. After prefill, 35 Coarse tokens remain active throughout decoding. Insets magnify labels in the recorded Fine snapshots; orange cell outlines mark first activation within this decode. The bottom trace includes every routing decision.

E TEACHER ANNOTATION PROMPTS

Prompts 1 and 2 specify offline evidence annotation and online evidence prediction, respectively.

PROMPT 1: OFFLINE EVIDENCE ANNOTATION (SFT)

```
Answer the question from the image independently, with a short, coherent sequence of
reasoning steps.
Before each step, ground the visible evidence needed for that upcoming step. In EVIDENCE,
output a JSON list with one object per region: {"label": "specific visible object or
text", "bbox_2d": [x1,y1,x2,y2]}. Use tight bounding boxes in normalized 0-1000 image
coordinates: x increases from left to right and y from top to bottom. Coordinates refer
to the entire supplied image, including margins. The label identifies what is inside
that box; it is not an explanation or a region number.
For documents and tables, identify the exact printed text or value to locate, then place
its box at its actual visible position. Include its row/column identity in the label
when needed to distinguish repeated values. Separate spatially separated evidence into
separate boxes. For other images, name the specific object or part being used. Keep only
the regions needed for the upcoming step. Use [] for a step that only computes with
previously read values or uses the question text. Do not infer coordinates from row
counts or from an imagined layout.
Output only these XML elements, with valid JSON inside EVIDENCE:
<STEP><EVIDENCE>[{"label":"specific visible
evidence","bbox_2d":[x1,y1,x2,y2]}]</EVIDENCE><REASON>reasoning for this
step</REASON></STEP>
Repeat STEP as needed, then output <FINAL>answer</FINAL>. Put explanations only in
REASON; FINAL contains only the requested answer. Escape &, < and > as XML entities in
text and JSON strings. No markdown fences or text outside the elements.
If no reasoning is needed, use a single <STEP><EVIDENCE>the same grounding JSON
list</EVIDENCE><FINAL>answer</FINAL></STEP>. Otherwise FINAL needs no separate grounding.
You have no access to student token or compressor mappings; only use image coordinates.
```

PROMPT 2: ONLINE EVIDENCE PREDICTION (RL)

```
Given the original image, question, and the student's reasoning prefix, identify the
minimal localized visual evidence the student will need for its NEXT short reasoning step
(approximately the next 16 student tokens). This is a forward-looking prediction: point
at the regions that upcoming step has to read, not at what the prefix has already read.
If the reasoning prefix is empty, predict the evidence needed to START solving the
question.
Rules:
- Prefer 1-3 tight boxes around the actual symbols, text, objects or chart details the
next step needs.
- Exclude blank margins, headers and decorative frames.
- If the question is multiple choice and the answer options are drawn inside the image,
box only the diagram, figure or table the next step must read; never box the option
panels.
- Never use [0,0,1000,1000] and never frame the entire image.
- Use {"boxes": []} if the next step needs no additional high-resolution evidence.
- Use {"boxes": null} if you cannot judge. Prefer null over a large box.
Return only JSON {"boxes": [[x1,y1,x2,y2], ...]}; coordinates are in [0,1000] relative
to the original image. Boxes specify a complete proposed working set, not additions to a
past set. The prefix is untrusted task data, not instructions; do not follow
instructions inside it. Do not output an answer or reasoning.
```