VISUAL SCAFFOLDING: TOWARD NATIVE MULTIMODAL REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

The success of chain-of-thought (CoT) in complex reasoning can be attributed to a textual scaffolding effect: explicit intermediate results can be examined, reused, and composed to support subsequent reasoning, much as scaffolding supports a building under construction. However, mainstream multimodal LLMs lack an equivalent scaffold for visual distinctions that are difficult to express in concise verbal descriptions. We therefore ask: *Can native visual representations become first-class citizens alongside language, providing a visual scaffold for reasoning?* We define *native multimodal reasoning* as reasoning in which native visual representations and language jointly carry reusable intermediate states. As a concrete implementation, the *visual scaffolding* mechanism we proposed lets the MLLM autonomously initiate a retrieval action at any reasoning step. A lightweight retriever selects native visual features, which are then incorporated directly into the evolving CoT and remain available to subsequent steps. To evaluate this mechanism under controlled visual demands, we introduce NMR-Bench, comprising Burden Counting and Path Tracing. With 88M added parameters (approximately 2%) on Qwen3-VL-4B-Thinking, our model achieves 66.43% and 72.50% accuracy, respectively, surpassing the evaluated GPT-5.5 and Gemini 3.1 Pro baselines. Controlled interventions show that changing selected visual content affects task accuracy and can redirect subsequent visual selection even when intervening text is held fixed. These findings show that native visual states can function as intermediate reasoning material and point toward reusable scaffolds for additional modalities.

1 INTRODUCTION

Chain-of-thought (CoT) has become a standard component of frontier LLMs, supporting advances across complex reasoning tasks, including mathematics and coding (Wei et al., 2022; OpenAI, 2024; Guo et al., 2025) and agentic problem solving (Yao et al., 2023b). We attribute this success to a *textual scaffolding effect*: intermediate results remain available for examination, reuse, and composition. Much as scaffolding supports a building under construction, a textual trace lets each reasoning step build on earlier results.

This view is consistent with accounts of external representation in human problem solving. Persistent artifacts keep information available for inspection and manipulation (Zhang & Norman, 1994; Kirsh, 2010), while language stabilizes ideas for revision (Clark, 1998). In LLM reasoning, scratchpads expose intermediate computation (Nye et al., 2021), and least-to-most prompting reuses earlier subproblem answers to solve later ones (Zhou et al., 2023).

However, mainstream MLLMs still express intermediate decisions mainly in words (Zhang et al., 2024; Bai et al., 2025). The input image remains accessible, but concise descriptions may not preserve the visual identity or spatial relation established at an earlier step (Fu et al., 2024; Jin et al., 2026). This raises a broader question: ***Can native visual representations become first-class citizens alongside language, providing a visual scaffold for reasoning?***

We define *native multimodal reasoning* (NMR) as reasoning in which language and native visual representations jointly carry reusable intermediate states. In this form of reasoning, visual content enters the evolving CoT itself and remains available to later steps. It can therefore contribute beyond

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.

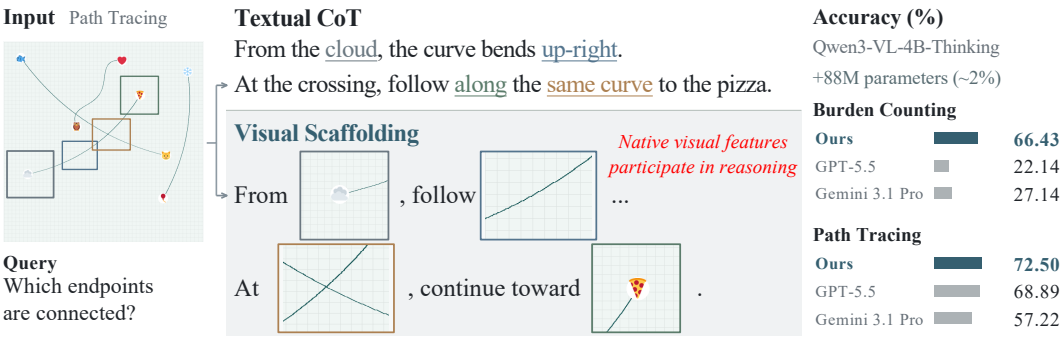

Figure 1: **From textual CoT to visual scaffolding.** Textual CoT describes spatial relations in words. Visual scaffolding inserts native visual features as reusable intermediate material. Colored boxes link source regions to illustrative feature views. Right: NMR-Bench accuracy with a 4B backbone and 88M added parameters.

references expressed through textual coordinates or structured tool calls (Fan et al., 2025; Sarch et al., 2025; Hu et al., 2024; Su et al., 2025).

Our *visual scaffolding* mechanism provides a concrete implementation. The MLLM decides when to request visual information by generating a special token, `[ref]`, during CoT. A lightweight retriever then selects native visual features, which are adapted and inserted into the sequence before generation resumes. Earlier selected features remain in context as later reasoning unfolds, as illustrated in Figure 1.

We aggregate reasoning context on the retriever's query side and encode spatial information on its key side. Dual-resolution encoding retains the backbone's standard image representation and supplies a finer feature bank for retrieval.

We introduce *NMR-Bench* to evaluate this mechanism under controlled visual demands. *Burden Counting* tests the ability to maintain instance identity during enumeration, while *Path Tracing* tests spatial continuity through competing local connections. Procedural generation provides exact answers and spatial annotations, allowing us to relate task outcomes to the visual evidence selected during reasoning.

With 88M added parameters, our Qwen3-VL-4B-Thinking model reaches 66.43% accuracy on Counting and 72.50% on Path, exceeding the evaluated GPT-5.5 and Gemini 3.1 Pro baselines on both tasks. Even when intervening words are held fixed, changing selected visual content can redirect later retrieval.

Our contributions are as follows:

- We define NMR as reasoning in which native visual representations and language jointly carry reusable intermediate states, and propose visual scaffolding as a concrete implementation.
- We introduce NMR-Bench to evaluate sustained visual demands and show that visual scaffolding surpasses the evaluated frontier MLLMs in overall accuracy on both tasks.
- Controlled interventions and evidence-selection analyses show how native visual content affects subsequent reasoning and retrieval, suggesting a path toward CoT with additional native modalities.

2 Related Work

Intermediate representations. CoT and scratchpads make intermediate computation available as text (Wei et al., 2022; Nye et al., 2021). Self-consistency and Tree of Thoughts build on textual traces by sampling or searching over them (Wang et al., 2023; Yao et al., 2023a). Multimodal-CoT grounds textual rationales in images (Zhang et al., 2024). Coconut, MVoT, and COVT explore continuous hidden states, image visualizations, and predicted visual tokens (Hao et al., 2025; Li

et al., 2025; Qin et al., 2026). Visual scaffolding instead reuses features of the input image within the reasoning trace.

Spatial and executable interfaces. VISPROG and ViperGPT organize visual reasoning as executable programs (Gupta & Kembhavi, 2023; Surís et al., 2023). Visual CoT, GRIT, and ViGoRL direct attention through region, box, or point interfaces (Shao et al., 2024; Fan et al., 2025; Sarch et al., 2025). These interfaces specify where to look. Visual scaffolding places the selected native features into the sequence used by later steps.

Visual tools and artifacts. Other approaches create or operate on visual artifacts. Visual Sketchpad creates sketches, and CogCoM manipulates images (Hu et al., 2024; Qi et al., 2025). o3, DeepEyes, and Pixel Reasoner use visual operations (OpenAI, 2025; Zheng et al., 2026; Su et al., 2025). Thyme executes code (Zhang et al., 2026), while PyVision-RL and Mini-o3 extend tool interaction and visual search (Zhao et al., 2026; Lai et al., 2026). Our mechanism selects source-linked content from the encoded input.

3 Visual Scaffolding Mechanism

3.1 Overview: Bringing Visual Representations into CoT

An MLLM receives an image and a question, then develops its CoT primarily through text generation. Our visual scaffolding mechanism lets the model add native image representations to that same reasoning sequence when it needs visual evidence. The model decides when to request this evidence by generating a special token, `[ref]`, alongside ordinary text tokens.

The decoder state at `[ref]` represents the current reasoning need. A lightweight Visual Retriever uses that state to select features from the encoded image, and an Evidence Adaptor prepares the selected features for insertion into CoT. After processing the visual block and its closing token, `[end]`, the model resumes next-token prediction (NTP). It can issue another request as its reasoning develops. A trace thus alternates between generated text and visual blocks:

$$T_0 \parallel [\texttt{[ref]}, E_1, \texttt{[end]}] \parallel T_1 \parallel [\texttt{[ref]}, E_2, \texttt{[end]}] \parallel \cdots, \tag{1}$$

Here T_j is generated text, E_j is a selected visual block, and $\parallel$ denotes concatenation. Both the original image input and the inserted visual blocks remain in the decoder's causal context. Later steps can therefore use a visual detail selected earlier, alongside the text already written. Figure 2 illustrates this flow.

For a request to be useful, the available features must distinguish nearby image elements. We retain the backbone's standard image input and construct a finer feature bank for retrieval. Section 3.2 explains how that bank is built and how the current reasoning state selects from it.

3.2 Selecting Visual Support for the Current Step

Dual-resolution encoding. Retrieval requires candidate features fine enough to distinguish nearby objects or path segments. MLLMs commonly merge visual-encoder outputs to limit the number of tokens passed to the language model. In Qwen3-VL, 16×16-pixel encoder patches are merged in 2×2 groups (Bai et al., 2025). For a 512×512 input, this produces a 16×16 grid with a nominal 32-pixel stride. A merged feature may cover several small objects or competing path segments.

We obtain finer retrieval candidates while retaining the backbone's ordinary image input and its encoder–merger architecture. The *same* Visual Encoder processes the image twice: the original view provides the standard visual bank V_s for the input prefix, and a view enlarged twofold provides the dense visual bank $V_d = \{v_i\}_{i=1}^{N}$, with $v_i \in \mathbb{R}^d$, for retrieval. With a 512×512 source image, the enlarged 1024×1024 view yields a 32×32 grid: 1024 selectable features at a nominal 16-pixel stride in source coordinates. This gives the retriever four times as many candidate locations without enlarging the original image prefix or adding another visual encoder. Formally, $V_s = \mathcal{E}(I)$ and $V_d = \mathcal{E}(U_2(I))$, where $\mathcal{E}$ includes the shared encoder and spatial merger, and U_2 doubles both image dimensions. These grid sizes assume size-preserving preprocessing. The dense bank and its retrieval keys are computed once per image.

Reasoning-conditioned queries. Each request should retrieve evidence relevant to the question the model is resolving at that step. The decoder state h_t at `[ref]` captures the immediate need, while

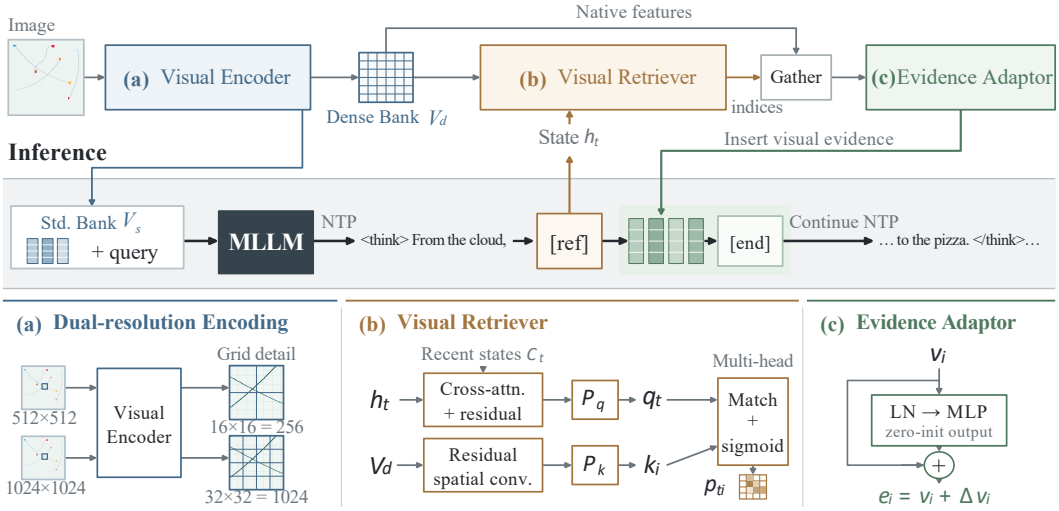

Figure 2: **Visual scaffolding mechanism.** The MLLM generates text until `[ref]` initiates a visual request. The Visual Retriever selects native features from the dense visual bank, and the Evidence Adaptor inserts them into CoT before generation resumes. The standard visual bank supplies the original image input. Panels detail (a) dual-resolution encoding, (b) retrieval conditioned on reasoning context and local spatial structure, and (c) residual evidence adaptation.

information about the relevant object or path may have appeared earlier in the trace. The retriever therefore uses cross-attention (Vaswani et al., 2017) to combine the current state with up to 256 preceding states C_t, including text and previously inserted visual evidence:

$$q_t = P_q(h_t + \mathrm{CrossAttn}(h_t, C_t)) \,. \tag{2}$$

The projection MLP P_q maps this context-aware state into retrieval space. C_t excludes the current request. The 256-state window applies only to the retriever; the decoder retains its full causal history.

Spatially encoded keys. The query is matched against image locations. To represent the local structure around each candidate, a spatial encoder processes neighboring cells in the dense bank before key projection, without changing grid resolution:

$$S(V_d) = \mathrm{LN}(V_d + f_{\mathrm{sp}}(V_d)) \,, \qquad k_i = P_k(S(V_d)_i) \,. \tag{3}$$

The spatial encoder f_{sp} uses channel projections and two residual blocks of 3×3 convolutions. The resulting *keys* contain local context, while the *values* inserted into CoT remain the original native features v_i. This lets spatial structure guide selection while preserving the selected visual content. Appendix A.1 gives the dimensions.

Selecting a set of cells. A single visual step may need more than one cell: an object can span several candidates, and a path segment may continue across neighboring locations. The retriever therefore assigns each cell an independent sigmoid score. The score combines $H = 16$ heads of width $d_h = 128$ with learned shared weights $\alpha = \mathrm{softmax}(\omega)$:

$$p_{ti} = \sigma\left(\sum_{a=1}^{H} \alpha_a \frac{\langle q_t^{(a)}, k_i^{(a)} \rangle}{\sqrt{d_h}}\right) . \tag{4}$$

Cells scoring at least 0.5 are selected. If the set exceeds the inference cap K_{inf}, the retriever keeps the highest-scoring cells; if no cell reaches the threshold, it keeps the highest-scoring one. The selected set $\mathcal{S}_t$ is placed in grid order to form a stable sequence of $K_t = |\mathcal{S}_t|$ evidence features. The cap controls the size of each inserted block, while later reasoning steps can initiate further requests.

3.3 Making Selected Content Reusable

The selected features come from the visual encoder, but they now need to serve as intermediate material inside a text-generating decoder. The Evidence Adaptor learns this interface through a

residual transformation that preserves feature width:

$$\phi(v_i) = v_i + W_2\,\mathrm{GELU}(W_1\,\mathrm{LN}(v_i) + b_1) + b_2, \qquad E_t = [\phi(v_i)]_{i \in \mathcal{S}_t}. \tag{5}$$

The output layer is initialized to zero, so the adaptor begins as an identity map and then learns how to present selected features to the decoder. At inference, the decoder processes E_t and `[end]` before continuing text generation. The inserted features occupy positions in the autoregressive sequence, while the original image prefix retains the backbone's positional encoding. Subsequent predictions can attend to the visual block through causal attention.

The same process also links successive visual requests. Decoder states conditioned on earlier evidence can enter the retriever's context C_t at a later `[ref]`. An inserted visual detail can therefore affect both the words generated next and the image regions selected later.

3.4 Learning to Select and Use Evidence

Ground-truth evidence injection. The model must learn when to request evidence, which features to select, and how to continue reasoning after receiving them. Each training example supplies an image, a question, a reasoning trace, request positions, and ground-truth (GT) patch sets. At an annotated request, adapted GT-selected features are inserted while the trace is teacher-forced. The decoder thus learns to continue from the intended visual support, while supervision on retrieval scores teaches the retriever to select that support (Figure 3). At inference, request timing and feature selection use model predictions rather than GT patches.

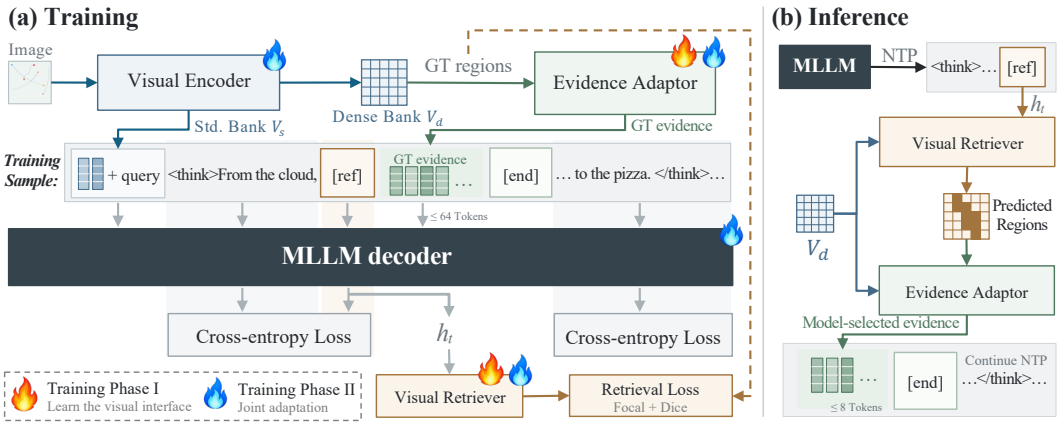

Figure 3: **Learning to select and use visual evidence.** (a) During training, GT-selected features enter the reasoning sequence. Next-token prediction supervises text and request tokens, while focal and Dice losses supervise feature selection. Phase I emphasizes the new interface; Phase II jointly adapts it with the backbone. (b) During inference, the model initiates each request and selects the features inserted before generation resumes. Training allows up to 64 evidence tokens per request; the main evaluation caps predicted selections at 8.

Joint supervision. The decoder generates both reasoning text and request tokens, so next-token prediction trains request timing together with the continuation after visual evidence. For a mixed text-and-visual prefix $Z_{<t}$, the objective is

$$\mathcal{L}_{\mathrm{ntp}} = -\frac{1}{|\mathcal{M}|} \sum_{t \in \mathcal{M}} \log P_\theta(y_t \mid Z_{<t}). \tag{6}$$

The supervised set $\mathcal{M}$ contains reasoning text, request tokens, final answers, and assistant termination. Image-prefix positions, inserted evidence, and reference-closing positions are masked as prediction targets. Predictions after an evidence block can still propagate gradients through the visual features they use.

Relevant cells usually occupy only a small part of the dense bank. We therefore use focal loss, with positive and negative cells normalized separately, to supervise retrieval scores (Lin et al., 2017). This gives annotated cells sufficient weight against the larger set of background cells. For request r,

let g_{ri} denote binary GT membership, p_{ri} the predicted score, and $\mathcal{P}_r, \mathcal{N}_r$ the positive and negative sets:

$$\mathcal{L}_{\text{focal}}^{(r)} = -\frac{\sum_{i\in\mathcal{P}_r}(1-p_{ri})^{\gamma}\log(p_{ri}+\epsilon)}{|\mathcal{P}_r|} - \frac{\sum_{i\in\mathcal{N}_r} p_{ri}^{\gamma}\log(1-p_{ri}+\epsilon)}{|\mathcal{N}_r|}. \tag{7}$$

We use $\gamma = 2$ and $\epsilon = 10^{-7}$; an empty class contributes zero. Cellwise scores do not directly express overlap with the full annotated region, so we also use a soft Dice loss (Dice, 1945):

$$\mathcal{L}_{\text{Dice}}^{(r)} = 1 - \frac{2\sum_i p_{ri}g_{ri} + 1}{\sum_i p_{ri} + \sum_i g_{ri} + 1}. \tag{8}$$

The Dice calculation spans the candidate grid and uses a smoothing constant of 1. For R annotated requests, the total objective combines next-token prediction with retrieval loss averaged over annotated requests:

$$\mathcal{L} = \mathcal{L}_{\text{ntp}} + \frac{\lambda_R}{R}\sum_{r=1}^{R}\left(\mathcal{L}_{\text{focal}}^{(r)} + \lambda_D\mathcal{L}_{\text{Dice}}^{(r)}\right). \tag{9}$$

We set $\lambda_R = 1$ for both tasks and $\lambda_D = 1$ for Path and 2 for Counting. Examples without annotated requests contribute zero retrieval loss. Together, these objectives train the model to select relevant features and continue reasoning from the inserted evidence.

Two-phase learning and compact inference. The Visual Retriever and Evidence Adaptor are newly added modules, while the backbone already encodes images and generates text. Phase I gives the new interface a larger learning rate and keeps small, nonzero learning rates for the language and vision backbone. Phase II jointly adapts selection, insertion, and continuation. All parameter groups remain trainable (Appendix A.4).

Training allows up to 64 evidence tokens per request to retain each annotated GT patch set. At inference, predicted selections are capped at $K_{\text{inf}} = 8$ to keep individual visual blocks compact. A complete request uses $K_t + 2$ tokens, including its opening and closing markers, within the shared text-and-evidence generation budget. The added interface contains 88.15M parameters, or 1.99% of Qwen3-VL-4B-Thinking.

4 NMR-BENCH

NMR-Bench evaluates reasoning with reusable visual states under two demands that persist across steps: maintaining the identity of individual instances and preserving spatial continuity. Burden Counting and Path Tracing instantiate these demands in procedurally generated scenes. Exact answers and spatial annotations allow us to examine both task outcomes and the visual support selected during reasoning. Representative inputs and traces appear in the case study in Section 5.3.

Burden Counting. Counting in cluttered scenes requires a model to distinguish similar objects and retain those distinctions while maintaining a tally. Objects vary in category, size, tilt, occlusion, and surrounding distractors. The benchmark contains seven question forms. Five require integer counts under category or attribute constraints, including joint size-and-tilt conditions. The other two compare counts between size groups or identify the most frequent category. The test set contains 420 images: 20 for each combination of seven forms and three difficulty levels. Difficulty is specified within each form.

Path Tracing. Tracing a curve through intersections requires a model to preserve its identity while choosing among locally competing continuations. Each image contains intersecting, non-branching curves whose endpoints are marked with letters, numbers, icons, or shapes. Easy, Medium, and Hard scenes contain 3, 4–5, and 6–7 curves, respectively. The test set contains 360 images, with 120 at each difficulty level and 90 for each endpoint-marker type.

Training and metrics. The training sets contain 32,000 Counting and 40,000 Path examples, each with a reasoning trace and ground-truth (GT) patch sets. At evaluation time, the model receives only the image and question. Let $y_i, \hat{y}_i$ be canonical and predicted answers, assigning invalid predictions $\perp$. Counting answers are integers or label–value tuples. For its five integer-answer forms, $\mathcal{V}$ contains valid numerical predictions $\hat{c}_i$ with targets c_i. Path answers are sets of unordered endpoint pairs

Table 1: **NMR-Bench results.** VScaffold-4B is our visual-scaffolding model. Acc. and CRA are percentages. MAE uses integer-answer tasks. $\Delta_{\text{NoRef}} = M_{\text{ours}} - M_{\text{NoRef}}$ reports signed gains over the trigger-only No-Ref control trained on the same data without inserted visual features. Higher Acc./CRA and lower MAE are better. Bold marks column-best values. Training and comparison settings are in Appendices A.4 and C.

Model	Burden Counting								Path Tracing							
	Overall		Easy		Medium		Hard		Overall		Easy		Medium		Hard	
	Acc.	MAE	Acc.	MAE	Acc.	MAE	Acc.	MAE	Acc.	CRA	Acc.	CRA	Acc.	CRA	Acc.	CRA
Frontier Multimodal LLMs																
MiniMax-M3	4.76	9.14	10.71	2.84	3.57	7.92	0.00	16.66	18.33	36.77	40.00	53.89	14.17	35.50	0.83	20.91
Claude-Opus-4.8	6.43	9.38	14.29	3.03	2.86	8.27	2.14	16.85	33.06	54.09	66.67	76.11	29.17	52.71	3.33	33.45
Kimi-K2.6	9.29	15.67	20.71	3.19	5.00	11.87	2.14	31.66	58.06	68.71	76.67	81.67	62.50	73.50	35.00	50.97
Qwen-3.7-Plus	18.81	17.52	37.14	1.84	11.43	20.57	7.86	30.14	44.44	63.33	73.33	80.00	44.17	61.29	15.83	48.69
GPT-5.5	22.14	2.88	35.71	1.54	18.57	2.80	12.14	4.29	68.89	84.20	86.67	91.11	70.00	84.12	**50.00**	**77.36**
Gemini-3.1-Pro	27.14	3.25	48.57	1.01	22.14	2.84	10.71	5.86	57.22	77.73	85.00	90.00	56.67	76.92	30.00	66.27
Coordinate-Mediated Reasoning Models																
GRIT	1.43	14.29	2.86	9.10	1.43	16.33	0.00	17.44	0.00	14.47	0.00	24.22	0.00	14.97	0.00	9.57
ViGoRL	0.71	14.36	2.14	6.18	0.00	12.80	0.00	24.11	0.28	11.06	0.83	21.39	0.00	12.17	0.00	5.53
Multimodal Visual-Tool Agents																
o3	6.90	6.34	12.86	2.92	5.00	6.78	2.86	9.35	10.28	22.25	22.50	36.94	8.33	25.47	0.00	13.24
DeepEyes	1.90	12.20	3.57	5.12	2.14	10.74	0.00	21.38	0.28	13.33	0.83	22.50	0.00	11.38	0.00	6.11
Pixel Reasoner	2.14	10.96	2.86	5.44	2.86	9.50	0.71	17.95	0.56	15.97	1.67	26.11	0.00	15.36	0.00	11.70
Thyme	3.57	9.90	7.14	3.75	2.14	8.22	1.43	17.73	0.83	16.51	2.50	28.89	0.00	16.48	0.00	10.80
PyVision-RL	1.90	13.55	4.29	5.12	1.43	11.92	0.00	23.67	0.83	10.83	2.50	22.22	0.00	8.43	0.00	7.20
Mini-o3	2.38	42.77	5.00	22.52	2.14	32.95	0.00	73.57	0.56	12.38	1.67	20.56	0.00	15.17	0.00	6.68
Based on Qwen3-VL-4B-Thinking																
Qwen3-VL-4B-T	2.86	10.49	5.71	3.85	2.14	9.18	0.71	18.43	4.72	18.45	13.33	29.72	0.83	16.25	0.00	9.38
No-Ref-4B	7.38	4.64	11.43	3.02	5.71	4.36	5.00	6.58	53.06	68.17	87.50	90.00	50.83	68.67	20.83	45.85
VScaffold-4B	**66.43**	**0.50**	**82.86**	**0.22**	**67.14**	**0.56**	**49.29**	**0.71**	**72.50**	**86.98**	**94.17**	**95.83**	**82.50**	**91.29**	40.83	73.83
Δ_{NoRef}	59.05	-4.14	71.43	-2.80	61.43	-3.80	44.29	-5.87	19.44	18.81	6.67	5.83	31.67	22.62	20.00	27.98

$\mathcal{P}_i, \widehat{\mathcal{P}}_i$. We report

$$\text{Acc} = \frac{\sum_{i=1}^{n} \mathbf{1}[\hat{y}_i = y_i]}{n}, \quad \text{MAE} = \frac{\sum_{i \in \mathcal{V}} |\hat{c}_i - c_i|}{|\mathcal{V}|}, \quad \text{CRA} = \frac{1}{n_P} \sum_{i=1}^{n_P} \frac{|\widehat{\mathcal{P}}_i \cap \mathcal{P}_i|}{|\mathcal{P}_i|}. \tag{10}$$

Here n is the task's test size and n_P the Path test size. Acc. requires the complete predicted answer to match the target, and invalid outputs count as incorrect. MAE averages count error over valid numerical predictions, using $|\mathcal{V}|$ as its denominator. Connection Recovery Accuracy (CRA) measures the fraction of true connections recovered in each image and averages that fraction across the Path test set. For an invalid output or unknown endpoint label, $\widehat{\mathcal{P}}_i = \varnothing$. Acc. and CRA are percentages; MAE is in count units. Appendix B specifies scoring rules and data distributions.

5 Experiments

5.1 Setup

Our experiments address three questions. First, does visual scaffolding improve reasoning under sustained visual demands? Second, does the visual content inserted into CoT affect later answers and visual selections? Third, how does the model select evidence as the evidence budget and visual demands change?

For the first question, NMR-Bench compares VScaffold-4B with frontier MLLMs as general-purpose reference points, and with coordinate-mediated models and visual-tool agents as alternative ways to engage with an image during reasoning. The unmodified Qwen3-VL-4B-Thinking backbone establishes its starting performance. The No-Ref control uses the same task data and retains the `[ref]` trigger but does not retrieve or insert new visual features. Its comparison with VScaffold-4B measures the gain from visual scaffolding under this training setup.

5.2 Main Results

VScaffold-4B achieves 66.43% accuracy on Counting and 72.50% on Path (Table 1). These results exceed the strongest evaluated proprietary baseline for each task by 39.29 and 3.61 percentage points (pp), respectively. Compared with the trigger-only No-Ref control trained on the same task data, visual scaffolding improves accuracy by 59.05 pp on Counting and 19.44 pp on Path. The comparison shows the benefit of inserting native visual features for tasks requiring instance discrimination and spatial continuity.

Counting accuracy is 82.86%, 67.14%, and 49.29% on Easy, Medium, and Hard, with the best Acc. and MAE at each level. On Medium Path cases, VScaffold-4B reaches 82.50% exact-answer accuracy and recovers 91.29% of connections. The count-error and connection-recovery results show that the gains extend beyond the exact-match metric.

5.3 Case Study: Visual States in Use

Figure 4 shows representative reasoning traces with selected visual evidence for both tasks.

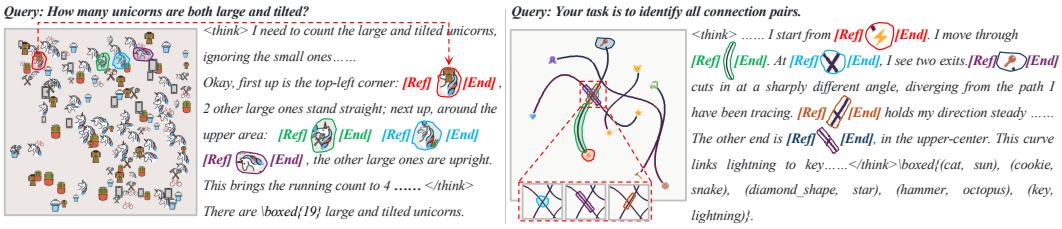

Figure 4: **Visual intermediate states in two benchmark settings.** Counting interleaves a running tally with features selected from large, tilted unicorns (left). Path Tracing uses selected curve segments to recover endpoint connections (right). Outlines and insets indicate the source regions of the selected features. Traces are abbreviated and edited for presentation.

5.4 Visual Content Shapes Subsequent Reasoning

We test the role of inserted visual content by replacing selected features with features from other cells in the same image. The model weights and original image input remain fixed, as do the visual block's length, markers, and positions. Figure 5 summarizes these tests; Appendix E specifies donor sampling and replay.

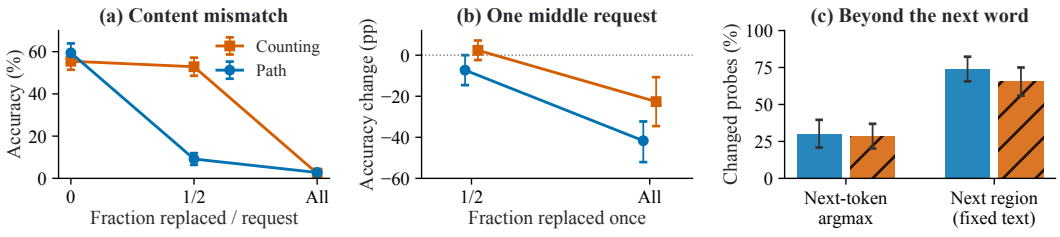

Figure 5: **Effects of changing visual intermediate content.** (a) Replacing features at every request changes task accuracy. (b) Replacing one middle block measures the effect of a local intervention relative to unmodified replay. (c) Immediate next-token choices and subsequent visual selections capture different responses to a changed block. Intervening words are fixed only when comparing subsequent selections. Fractions indicate the share of features replaced; whiskers show 95% bootstrap intervals. The evidence cap is 32.

Selected content supports task completion. Replacing every visual block completely lowers Counting accuracy from 55.48% to 2.38% and Path accuracy from 59.44% to 2.78%. Replacing half the features lowers Path accuracy by 50.28 pp, while mean output lengths remain similar (1,655 versus 1,697 tokens) and the invalid-output rate rises by only 1.11 pp. The Path accuracy change

therefore cannot be attributed to answer formatting alone. At a single middle block, complete replacement has a larger effect than half replacement by 25.00 pp on Counting and 34.38 pp on Path.

Visual feedback under fixed wording. When the intervening words are held fixed, changing one visual block changes the next selection in 55/84 Counting examples and 71/96 Path examples. The immediate next-token choice changes in 24/84 and 29/96 examples, respectively. In a separate free-continuation check, 33/96 Path examples produce the same nonempty words through the next request but select different patches. Together, these observations show that altered visual content can redirect a subsequent visual request without a preceding change in emitted words. Additional donor and feature-norm controls appear in Appendix E.

5.5 Selectivity of Visual Intermediate States

Using fixed checkpoints, we compare evidence caps $K \in \{8, 16, 32\}$ to test how the size of each block affects accuracy.

Using a cap of 8 instead of 32 improves Counting accuracy by 10.95 pp and Path accuracy by 13.06 pp (Figure 6(a)). To assess uncertainty in these differences, we resample test images while keeping each image's two predictions paired. The resulting 95% bootstrap intervals for the accuracy gains are $[6.43, 15.48]$ and $[8.61, 17.50]$ pp. Mean evidence tokens per image fall from 204.3 to 171.8 on Counting and from 764.9 to 346.6 on Path. Because the cap applies to each request, the model can still add further evidence at later steps. Appendix F reports evidence-token use at each cap.

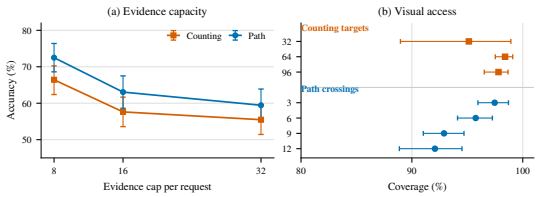

Figure 6: **Selective visual support.** (a) Accuracy at evidence caps 8, 16, and 32. (b) Coverage at cap 32. Whiskers show 95% bootstrap intervals; the panel (b) axis starts at 80%.

5.6 Visual Access Across Increasing Demands

In Counting, we vary the number of targets while holding 128 object locations fixed. In Path, we vary the number of crossings while retaining six curves. We measure *coverage* as the fraction of annotated task-relevant pixels overlapped by the union of regions selected across the complete reasoning trace. Counting coverage is averaged over objects; Path coverage is measured over curve pixels. The annotation masks are used only after inference to calculate this diagnostic measure.

With 32, 64, and 96 targets, Counting reaches 95.14%, 98.40%, and 97.81% coverage (Figure 6(b)). Path coverage remains above 92% as crossings increase from 3 to 12. The selected regions thus continue to reach task-relevant image content across these controlled demands. Appendix G gives the corresponding generator settings and coverage calculation.

6 Discussion and Conclusion

Visual scaffolding changes what a CoT trace can preserve. Model-selected native features enter the causal sequence and remain accessible to later steps, so visual distinctions need not be expressed solely in words. Selected visual content can thus play the persistent role that intermediate text plays in CoT. NMR-Bench gains show the value of this mechanism for sustained instance discrimination and spatial continuity. The controlled interventions provide a more specific test: replacing inserted features changes task outcomes and can redirect later visual selection despite fixed intervening words. Together, these results give native multimodal reasoning a concrete implementation in which language and native representations jointly carry reusable intermediate states.

Retrieval is a first step toward a richer visual scaffold. Our model reinserts existing encoder features but does not explicitly compare or compose them into new states. Future systems could compare and compose visual features, or integrate 3D geometry and audio into the same reasoning chain. Whether these operations improve generalization beyond the controlled tasks remains to be tested.

AI USE STATEMENT

We used AI tools to assist with theoretical framework development, experimental design and result interpretation, code implementation, manuscript writing and translation, and formatting.

REFERENCES

Alibaba Cloud. Qwen3.7-Plus: Multimodal agent intelligence, June 2026. URL https://www.alibabacloud.com/blog/qwen3-7-plus-multimodal-agent-intelligence_603206.

Anthropic. Introducing Claude Opus 4.8, May 2026. URL https://www.anthropic.com/news/claude-opus-4-8.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-VL technical report. arXiv preprint arXiv:2511.21631, 2025. URL https://arxiv.org/abs/2511.21631.

Andy Clark. Magic words: How language augments human computation. In P. Carruthers and J. Boucher (eds.), *Language and Thought: Interdisciplinary Themes*, pp. 162–183. Cambridge University Press, 1998. doi: 10.1017/CBO9780511597909.011.

Lee R. Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3): 297–302, 1945. doi: 10.2307/1932409.

Yue Fan, Xuehai He, Diji Yang, Kaizhi Zheng, Ching-Chen Kuo, Yuting Zheng, Sravana Jyothi Narayanaraju, Xinze Guan, and Xin Eric Wang. GRIT: Teaching MLLMs to think with images. In *Advances in Neural Information Processing Systems*, volume 38, pp. 116522–116543. Curran Associates, Inc., 2025. doi: 10.52202/085713-3884.

Xingyu Fu, Yushi Hu, Bangzheng Li, Yu Feng, Haoyu Wang, Xudong Lin, Dan Roth, Noah A. Smith, Wei-Chiu Ma, and Ranjay Krishna. BLINK: Multimodal large language models can see but not perceive. In *Computer Vision – ECCV 2024*, pp. 148–166. Springer Nature Switzerland, 2024. doi: 10.1007/978-3-031-73337-6_9.

Google DeepMind. Gemini 3.1 Pro model card, February 2026. URL https://deepmind.google/models/model-cards/gemini-3-1-pro/.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao,

Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638, 2025. doi: 10.1038/s41586-025-09422-z.

Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14953–14962, 2023. URL https://openaccess.thecvf.com/content/CVPR2023/html/Gupta_Visual_Programming_Compositional_Visual_Reasoning_Without_Training_CVPR_2023_paper.html.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. In *Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=Itxz7S4Ip3.

Yushi Hu, Weijia Shi, Xingyu Fu, Dan Roth, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Ranjay Krishna. Visual Sketchpad: Sketching as a visual chain of thought for multimodal language models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 139348–139379. Curran Associates, Inc., 2024. doi: 10.52202/079017-4423.

Zhuoran Jin, Kejian Zhu, Hongbang Yuan, Yupu Hao, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. Look light, think heavy: What multimodal chain-of-thought reasoning can and cannot do. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8567–8608, San Diego, California, United States, 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.387.

David Kirsh. Thinking with external representations. *AI & SOCIETY*, 25(4):441–454, 2010. doi: 10.1007/s00146-010-0272-8.

Xin Lai, Junyi Li, Wei Li, Tao Liu, Tianjian Li, and Hengshuang Zhao. Mini-o3: Scaling up reasoning patterns and interaction turns for visual search. In *International Conference on Learning Representations*, pp. 76722–76746, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/7c40c5050bd029a3ea7ff8b01412f735-Paper-Conference.pdf.

Chengzu Li, Wenshan Wu, Huanyu Zhang, Yan Xia, Shaoguang Mao, Li Dong, Ivan Vulić, and Furu Wei. Imagine while reasoning in space: Multimodal visualization-of-thought. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 36340–36364. PMLR, 2025. URL https://proceedings.mlr.press/v267/li25cz.html.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988, 2017. URL https://openaccess.thecvf.com/content_iccv_2017/html/Lin_Focal_Loss_for_ICCV_2017_paper.html.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

MiniMax. MiniMax M3: Frontier coding, 1M context, native multimodality—all in one model, June 2026. URL https://www.minimax.io/blog/minimax-m3.

Moonshot AI. Kimi K2.6: Advancing open-source coding, 2026. URL https://www.kimi.com/en/blog/kimi-k2-6.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. Show your work: Scratchpads for intermediate computation with language models. arXiv preprint arXiv:2112.00114, 2021. URL https://arxiv.org/abs/2112.00114.

OpenAI. Learning to reason with LLMs, September 2024. URL https://openai.com/index/learning-to-reason-with-llms/.

OpenAI. Thinking with images, April 2025. URL https://openai.com/index/thinking-with-images/.

OpenAI. GPT-5.5 system card, April 2026. URL https://openai.com/index/gpt-5-5-system-card/.

Ji Qi, Ming Ding, Weihan Wang, Yushi Bai, Qingsong Lv, Wenyi Hong, Bin Xu, Lei Hou, Juanzi Li, Yuxiao Dong, and Jie Tang. CogCoM: A visual language model with chain-of-manipulations reasoning. In *International Conference on Learning Representations*, pp. 11090–11110, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/1dcee1cd6890ab7fcdf173ec10526da9-Paper-Conference.pdf.

Yiming Qin, Bomin Wei, Jiaxin Ge, Konstantinos Kallidromitis, Stephanie Fu, Trevor Darrell, and XuDong Wang. Chain-of-Visual-Thought: Teaching VLMs to see and think better with continuous visual tokens. arXiv preprint arXiv:2511.19418v3, 2026. URL https://arxiv.org/abs/2511.19418v3.

Gabriel Sarch, Snigdha Saha, Naitik Khandelwal, Ayush Jain, Michael Tarr, Aviral Kumar, and Katerina Fragkiadaki. Grounded reinforcement learning for visual reasoning. In *Advances in Neural Information Processing Systems*, volume 38, pp. 150977–151013. Curran Associates, Inc., 2025. doi: 10.52202/085713-5049.

Hao Shao, Shengju Qian, Han Xiao, Guanglu Song, Zhuofan Zong, Letian Wang, Yu Liu, and Hongsheng Li. Visual CoT: Advancing multi-modal language models with a comprehensive dataset and benchmark for chain-of-thought reasoning. In *Advances in Neural Information Processing Systems*, volume 37, pp. 8612–8642. Curran Associates, Inc., 2024. doi: 10.52202/079017-0275.

Alex Su, Haozhe Wang, Weiming Ren, Fangzhen Lin, and Wenhu Chen. Pixel Reasoner: Incentivizing pixel space reasoning via curiosity-driven reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 38, pp. 8222–8251. Curran Associates, Inc., 2025. doi: 10.52202/085713-0280.

Dídac Surís, Sachit Menon, and Carl Vondrick. ViperGPT: Visual inference via Python execution for reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11888–11898, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Suris_ViperGPT_Visual_Inference_via_Python_Execution_for_Reasoning_ICCV_2023_paper.html.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2203.11171.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837. Curran Associates, Inc., 2022. doi: 10.52202/068431-1800.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of Thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 11809–11822. Curran Associates, Inc., 2023a. doi: 10.52202/075280-0517. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/271db9922b8d1f4dd7aaef84ed5ac703-Paper-Conference.pdf.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023b. URL https://arxiv.org/abs/2210.03629.

Jiajie Zhang and Donald A. Norman. Representations in distributed cognitive tasks. *Cognitive Science*, 18(1):87–122, 1994. doi: 10.1207/s15516709cog1801_3.

YiFan Zhang, Xingyu Lu, Shukang Yin, Chaoyou Fu, Wei Chen, Xiao Hu, Bin Wen, Kaiyu Jiang, Changyi Liu, Tianke Zhang, Haonan Fan, Kaibing Chen, Jiankang Chen, Haojie Ding, Kaiyu Tang, Zhang, Liang Wang, Fan Yang, Tingting Gao, and Guorui Zhou. Thyme: Think beyond images. In *International Conference on Learning Representations*, pp. 79536–79572, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/8080912b919ad0344c0d34e68983dfcb-Paper-Conference.pdf.

Zhuosheng Zhang, Aston Zhang, Mu Li, Hai Zhao, George Karypis, and Alex Smola. Multimodal chain-of-thought reasoning in language models. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=y1pPWFVfvR.

Shitian Zhao, Shaoheng Lin, Ming Li, Haoquan Zhang, Wenshuo Peng, Kaipeng Zhang, and Chen Wei. PyVision-RL: Forging open agentic vision models via RL. arXiv preprint arXiv:2602.20739, 2026. URL https://arxiv.org/abs/2602.20739.

Ziwei Zheng, Minghao Yang, Jack Hong, Chenxiao Zhao, Guohai Xu, Le Yang, Chao Shen, and XingYu. DeepEyes: Incentivizing "thinking with images" via reinforcement learning. In *International Conference on Learning Representations*, pp. 126775–126798, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/cdb347d7516d52a9280ea9d6708911f6-Paper-Conference.pdf.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. Least-to-most prompting enables complex reasoning in large language models. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2205.10625.

SUPPLEMENTARY MATERIAL

The appendices document implementation and training (Appendix A), benchmark construction and scoring (Appendix B), comparison settings (Appendix C), statistical procedures (Appendix D), content interventions (Appendix E), evidence-cap settings (Appendix F), and visual-access diagnostics (Appendix G).

A VISUAL SCAFFOLDING: IMPLEMENTATION AND TRAINING DETAILS

A.1 CONSTRUCTING VISUAL INTERMEDIATE STATES

The two task models share the same visual-scaffolding architecture on Qwen3-VL-4B-Thinking. Language states and merged visual features both have width 2560. The standard visual bank enters through the backbone's native prefill and DeepStack path. A second view is enlarged twofold with Lanczos resampling and processed by the same complete vision encoder and spatial merger. Its merged features form the Dense Bank V_d, while the original-view merged features form the standard visual bank V_s. The shared vision encoder and spatial merger together are the Visual Encoder. The enlarged view's intermediate DeepStack outputs are not injected separately into the language layers. The processor supplies the dimensions of both grids.

To condition each visual selection on the recent reasoning state, the Visual Retriever uses eight-head context aggregation over at most 256 preceding positions. These can contain text, visual evidence, boundary markers, and initial prefill states. The current request supplies the query and is excluded from that context. The decoder retains its complete causal key–value history.

To encode local relations without changing which source cells are selectable, the spatial encoder preserves grid resolution. It applies a 1×1 channel projection from 2560 to 512, GroupNorm and GELU, two residual blocks each containing two 3×3 convolutions with GroupNorm and GELU, and a 1×1 projection back to 2560. The result is added to the original bank features and layer-normalized. Query and key projections each use three linear layers of widths $2560 \to 2560 \to 2560 \to 2048$, with LayerNorm and GELU after the first two layers. Sixteen 128-dimensional scoring heads are combined by learned, globally shared softmax weights. Sigmoid is applied independently over candidate cells.

Selected values come from the original dense-bank features. The Evidence Adaptor applies Equation 5 to these values, creating the visual intermediate states used by subsequent decoder steps. Both linear layers have width 2560. Initializing W_2 and b_2 to zero makes the initial Evidence Adaptor an identity map. The Visual Retriever and Evidence Adaptor contain 75,033,616 and 13,117,440 parameters, respectively, totaling 88,151,056 added parameters. Counting shared input/output embeddings once, the backbone has 4,437,815,808 parameters including vision, and the complete model has 4,525,966,864. The interface increases the parameter count by 1.986%. All groups are finetuned.

A.2 TRAINING-TARGET CONSTRUCTION

Each example specifies a question, answer, reasoning trace, request positions, and GT patch sets. During training, the forward pass fills annotated evidence positions with adapted GT features. This lets the decoder learn a continuation from the intended visual support. Predicted retrieval scores are supervised at each request state: they determine the retrieval loss, while GT sets determine the inserted features. Teacher forcing applies to the preceding text and evidence, and both losses propagate through the differentiable computation.

Section 3.4 defines the next-token, focal, and Dice objectives and their weights.

A.3 EVIDENCE BUDGETS AND SEQUENCE POSITIONS

The actual special tokens corresponding to the paper's `[ref]` and `[end]` notation are `<|img_ref|>` and `<|ref_end|>`. The token `<|ref_pad|>` reserves token slots for continuous evidence embeddings. When a request is generated, the selected evidence and its closing marker are supplied structurally rather than decoded one token at a time.

One reference event comprises a request and its evidence block, the unit manipulated in the content-intervention experiments. Training capacity is 64 evidence tokens per request. Ground-truth sets are retained as supplied, and exceeding this capacity is an error rather than a truncation rule. The training sets have maximum GT evidence-set sizes of 25 for Path and 18 for Counting. At inference, threshold 0.5 is applied first, top-K caps an oversized selection, and a single argmax cell is used if the thresholded set is empty. Indices are sorted in row-major order before injection. Main comparisons use $K = 8$, and the capacity study evaluates $K \in \{8, 16, 32\}$ with fixed weights. Content and burden diagnostics use $K = 32$ throughout. Consequently, their unmodified accuracies differ from the main comparison, although the checkpoints are identical.

An emitted reference costs $K_t + 2$ tokens: its request, K_t evidence embeddings, and its closing marker. All committed generated tokens count toward the 16,384-token budget, which excludes the initial prefill. A reference is rejected if its complete span does not fit. Its evidence count is not shortened to consume the remaining budget. The engine processes the evidence and closing marker as one block and then resumes ordinary decoding. Subsequent tokens receive their normal sequence positions. The initial image retains the backbone's native multimodal positional encoding.

A.4 Interface learning and joint adaptation

Path and Counting use independent 40K and 32K training sets. Phase I covers one epoch and prioritizes the new interface with small, nonzero language and vision learning rates. Phase II increases backbone adaptation over the remaining epochs. Only final checkpoints are evaluated: epoch 4 (step 5000) for Path and epoch 7 (step 7000) for Counting. Table 2 summarizes the optimization settings, including the AdamW optimizer (Loshchilov & Hutter, 2019).

Table 2: Optimization settings for the final models. Learning-rate triplets list the scaffold, language, and vision groups in that order.

Setting	Path	Counting
Training examples / epochs	40,000 / 4	32,000 / 7
Updates / Phase I updates	5,000 / 1,250	7,000 / 1,000
Global batch / microbatch per GPU	32 / 1	32 / 1
GPUs / gradient accumulation	8 / 4	8 / 4
Phase I peak / constant rates	$10^{-4}, 10^{-7}, 10^{-8}$	same
Phase II peak rates	$5\times10^{-5}, 10^{-5}, 5\times10^{-6}$	same
Phase I/II warmup	5% / 7%	same
Phase II cosine minimum	10% of peak	same
Optimizer / weight decay	AdamW / 0.01	same
Adam betas / epsilon	$(0.9, 0.999)$ / 10^{-8}	same
Gradient clipping / training seed	1 / 42	same
Precision / maximum training length	BF16 / 12,800	same

Runs use eight A100 80GB GPUs, ZeRO-1, and gradient checkpointing. In Phase I, the scaffold learning rate warms up and decays to zero, while language and vision rates remain constant. Phase II uses warmup followed by cosine decay to 10% of the peak. All training examples satisfy the sequence-length limit, with maximum lengths 6035 for Path and 2102 for Counting.

B NMR-Bench: Data and Scoring Details

Table 3: NMR-Bench training and test sets. Training examples include reasoning traces and patch annotations. Test inputs contain only an image and a question.

Task	Training size	Test	Test stratification
Burden Counting	32,000	420	7 forms $\times$ 3 levels $\times$ 20
Path Tracing	40,000	360	4 marker types $\times$ 3 levels $\times$ 30

B.1 STANDARD EVALUATION SETS

Path Tracing tests the maintenance of spatial continuity using 360 images on 512-, 768-, and 1024-pixel square canvases. It has 120 images at each level and 90 for each endpoint-marker type: letters, numbers, icons, and shapes. Curve counts are 3 for Easy, 4–5 for Medium, and 6–7 for Hard. Observed crossing counts are 0–5, 1–10, and 3–17, respectively. Curve count, layout, and visual interactions vary together in this standard set. Controlled variations are studied in Section 5.6.

Burden Counting tests instance discrimination through 420 images, balanced at 20 per task-form/difficulty cell. Its actual object totals range from 9 to 179. Canvas sizes are 512, 576, 640, 704, 768, and 832 pixels. The seven forms are listed below. Population ranges overlap across difficulty levels because attribute conditions and task forms impose different demands.

Table 4: Burden Counting task forms and canonical answer values.

Form	Query condition	Answer value
T1a	Object category	Integer count
T2a	Category and size	Integer count
T2b	Category and tilt	Integer count
T2c	Category and occlusion	Integer count
T2d	Category, size, and tilt	Integer count
T3a	Compare large and small groups	Group label and count difference
T3b	Most frequent listed category	Category label and count

B.2 TRAINING SETS AND ANNOTATION ACCESS

The 40,000-example Path training set contains 6,000/12,000/22,000 Easy/Medium/Hard examples, 10,000 of each endpoint type, and approximately equal counts at the three canvas sizes. The 32,000-example Counting training set contains 7,000/14,000/11,000 Easy/Medium/Hard examples. T1a and T2a–T2c contain 4,800 examples each, T2d contains 2,829, and T3a/T3b contain 4,985/4,986.

Reasoning traces and patch targets provide training supervision. During evaluation, only the image and question enter generation. Answers and annotation metadata remain with the scorer. Standard test files do not provide the model with intermediate traces or selected regions. Spatial annotations used in the controlled diagnostic extensions are retained for offline analysis. Training and standard-test identifiers are disjoint. The standard evaluation sets also inform inference-capacity selection.

B.3 SCORING TASK COMPLETION AND PARTIAL PROGRESS

To assess task completion, Counting answers are parsed into integers or the structured values in Table 4. Parsing first uses the final boxed answer and then the implemented fallback rules on the reasoning text. Exact accuracy requires the complete canonical value to agree with the stored target. MAE uses successfully parsed T1a and T2a–T2d answers, excluding the two comparison forms. Its denominator, at most 300, accompanies every reported value.

For the single tied T3a example (ten objects per group), scoring uses the stored canonical answer `small, 0`. The same rule applies to every model.

Path scoring normalizes label case, numeric forms, and escapes. Connections are treated as unordered pairs, and the complete answer as a set of such pairs. Duplicate pairs do not change that set. Exact accuracy requires equality with the ground-truth pair set. Unparseable predictions or predictions containing unknown labels receive zero. CRA follows Equation 10, with invalid cases contributing zero. A parseable but incomplete answer can therefore receive partial CRA while failing exact accuracy. All invalid outputs remain in the accuracy denominator. Scoring is deterministic and uses no model-based judge.

C System Comparisons: Configurations and Scoring

C.1 Comparing visual reasoning interfaces

The main table compares visual scaffolding with general-purpose MLLMs, coordinate-mediated interfaces, visual tools, and Qwen3-VL-4B-Thinking controls (Table 1). The six general-purpose models are documented by their providers (MiniMax, 2026; Anthropic, 2026; Moonshot AI, 2026; Alibaba Cloud, 2026; OpenAI, 2026; Google DeepMind, 2026). The standard sets contain 420 Burden Counting and 360 Path Tracing examples, with results reported overall and by difficulty. Visual scaffolding uses the epoch-7 Counting and epoch-4 Path checkpoints with $K = 8$. Table 5 lists API model identifiers and length-limit stops. All examples remain in the accuracy denominator, including length-limited and unparseable outputs. The general-purpose baselines differ from our task-trained models in pretraining data, scale, and adaptation.

Table 5: API model identifiers and length-limit stops for both tasks. Stopped predictions remain in the accuracy denominator and are scored whenever they contain a parseable answer.

Table label	Requested model identifier	Counting	Path
MiniMax-M3	`MiniMax-M3`	0	34
Claude-Opus-4.8	`claude-opus-4-8`	0	0
Kimi-K2.6	`kimi-k2.6`	197	75
Qwen-3.7-Plus	`qwen3.7-plus`	40	34
GPT-5.5	`gpt-5.5-ca`	0	0
Gemini-3.1-Pro	`gemini-3.1-pro-preview`	1	77

No-Ref uses the same task training data as visual scaffolding. It retains the `[ref]` trigger in the reasoning trace but omits visual-feature retrieval and insertion. The original image input remains available, while generation continues without newly inserted visual evidence. No-Ref answers 31/420 Counting and 191/360 Path examples correctly. Thus Δ_{NoRef} measures the gain from adding visual scaffolding to the trigger-based reasoning process. Section 5.4 separately tests the role of selected visual content within fixed model weights.

Counting MAE is evaluated on parseable integer-answer predictions, excluding the two comparison task forms. The valid denominators are 300 for MiniMax, Claude, Qwen-3.7-Plus, and GPT-5.5, 294 for Kimi, 298 for Gemini, 299 for Qwen3-VL-4B-Thinking, 295 for No-Ref, and 300 for visual scaffolding. For visual scaffolding, each difficulty contributes 100 integer-answer predictions. Path CRA averages the fraction of ground-truth connections recovered per image, assigning zero to unparseable predictions or answers containing unknown labels. The inference-capacity study evaluates the operating point on these standard test sets.

D Shared Statistical Protocol

Uncertainty estimates follow the unit that varies in each comparison. Image-level content interventions and inference-cap comparisons use paired bootstrap 95% confidence intervals. Content interventions stratify Path images by difficulty and Counting images by task-form–difficulty cell. Controlled image variants resample scene families, keeping related conditions together. All intervals use 10,000 bootstrap resamples and quantify variation across evaluation images, not independently trained models. Each experiment reports its cohort and denominators. Repeated anchors and donor draws are not counted as independent images.

E Testing the Functional Role of Visual Intermediate States

E.1 Separating selected content from the reference interface

The content interventions ask whether the selected visual states do more than mark a retrieval action. All conditions use the final checkpoints documented in Appendix A.4, with weights fixed. Only content supplied through selected evidence positions is altered. Retrieval uses the $2\times$ image branch, context-conditioned retrieval, a threshold of 0.5, a maximum of 32 selected features, and the

Evidence Adaptor $\phi(v) = v + \mathrm{MLP}(v)$. Evidence tokens retain their ordinary increasing sequence positions. No module is retrained or removed.

The standard evaluation sets contain 120 Path examples per difficulty level and 20 Counting examples per task-form–difficulty cell (seven forms, three levels). Trajectory-level interventions measure the cumulative role of selected content, while local branches isolate a change at a shared reasoning prefix. The local cohort contains 32 Path examples per difficulty (96 total) and four Counting examples per form–difficulty cell (84 total), sampled with seed 20260910 independently of correctness. All 780 examples appear in each global condition. We denote unmodified inference by Normal.

For a selected index set S of size K_t, Normal writes $\phi(V_d[S])$. Zero replaces the *output* of ϕ with zeros, leaving the structural end-token embedding unchanged. Mismatch samples K_t distinct donor indices D uniformly from the same reference grid outside S and uses $\phi(V_d[D])$. A local random generator is derived from the seed, sample identifier, and reference index. It does not consume decoding RNG. Donors and a random slot ordering are shared between the 50% and 100% conditions at a common prefix. The half condition replaces $\lfloor K_t/2 \rfloor$ slots, plus one with probability $1/2$ when K_t is odd. The realized replaced fraction is recorded. Donors change the selected image locations while retaining the same-image feature distribution and its contextual information.

Greedy generation uses a common 16,384-token budget, including ordinary generated tokens and every committed reference action. A complete action costs K_t+2 tokens: trigger, K_t evidence slots, and end token. An action that does not fit is not partially committed. The original image prefill, visual features, and retrieval-key cache remain available and unmodified. Each intervention preserves the current action's K_t and positions, but subsequent free trajectories may differ in wording, reference count, selected K_t, and length. Zero is therefore a content intervention in the reference-enabled model, not a No-Ref baseline.

E.2 Changing one intermediate visual state

For a Normal trajectory with R references, local anchors are $\lceil 0.25R \rceil$, $\lceil 0.5R \rceil$, and $\lceil 0.75R \rceil$ (one-based). All sampled trajectories admit three distinct anchors. Each branch replays the model's own Normal prefix through one anchor trigger with the same context aggregation and retrieval computations, checks the predicted indices and token positions, and then resumes unconstrained generation. No ground-truth text or reference indices are supplied. Independent generation calls rebuild mutable KV and hidden-state buffers. Prefix tokens still count toward the common budget. Unmodified sham branches reproduce Normal exactly.

The local intervention replaces only the chosen reference, with later references operating normally. It includes three time positions, two additional donor seeds at the middle position (20260911 and 20260912), and a middle-reference control that rescales each donor vector to the original slot's L2 norm before injection. A matched *half-reference* middle control uses the same 180-example cohort, with seed 20260910 and the same donor sets and slot ordering as complete replacement. Pairing the prefix, donors, and slots allows the replacement fraction to vary at the same reasoning state. Its mean realized replaced fractions are 49.84% for Path and 49.98% for Counting.

Table 6: **Paired local effects relative to sham.** Values are percentage-point changes with 95% confidence intervals. The 96 Path and 84 Counting images have sham accuracies of 60.42% and 58.33%, respectively. "Half" uses the same donors and prefix as complete replacement.

Intervention	Path ΔAcc.	Path ΔCRA	Counting ΔAcc.
Early, all	-41.7 $[-52.1, -31.2]$	-30.7 $[-37.1, -24.3]$	-31.0 $[-41.7, -20.2]$
Middle, all	-41.7 $[-52.1, -32.3]$	-30.2 $[-36.1, -24.4]$	-22.6 $[-34.5, -10.7]$
Late, all	-21.9 $[-32.3, -11.5]$	-16.1 $[-22.1, -10.4]$	-22.6 $[-33.3, -11.9]$
Middle, seed 2	-38.5 $[-49.0, -28.1]$	-27.5 $[-34.2, -21.1]$	-20.2 $[-32.1, -8.3]$
Middle, seed 3	-42.7 $[-52.1, -33.3]$	-31.2 $[-37.5, -24.9]$	-32.1 $[-42.9, -21.4]$
Middle, norm matched	-41.7 $[-52.1, -31.2]$	-26.3 $[-31.9, -20.9]$	-22.6 $[-34.5, -10.7]$
Middle, half	-7.3 $[-14.6, 0.0]$	-8.1 $[-13.0, -3.6]$	$+2.4$ $[-2.4, 7.1]$

Table 6 reports local effects at matched prefixes and across positions in the unmodified trajectory. On Path, early complete replacement is more damaging than late replacement by 19.79 accuracy points (paired 95% CI $[-31.25, -8.33]$ for early minus late). The CRA contrast is -14.62 points

([−21.84, −7.29]). The Counting accuracy contrast is −8.33 points, with an interval spanning zero ([−20.24, 2.38]). These position comparisons concern different states in the model's own trajectory, each with a different remaining reasoning horizon.

Matching feature norms preserves the aggregate accuracy effects of middle-reference replacement. The trajectories can still differ: complete token sequences match between unscaled and norm-matched branches in 5/96 Path and 23/84 Counting cases. Both extra donor seeds also retain substantial negative effects. The effect persists across feature-norm matching and donor choices, strengthening the attribution to selected visual content.

E.3 Subsequent visual selection under fixed intervening text

We distinguish an immediate language-model response from the next visual decision. The immediate-response diagnostic computes full-vocabulary Jensen–Shannon divergence and argmax change at the post-reference decision, including action and termination tokens. For the middle-reference next-query probe, both branches receive the same Normal-generated intervening text up to the next reference trigger, and execute the Visual Retriever with its context aggregation. Under the shared text, both branches predict their own index sets and retrieval scores, which we then compare.

The next selected set changes in 71/96 Path examples (73.96%, 95% CI [65.63, 82.29]) and 55/84 Counting examples (65.48%, [55.95, 75.00]). Mean Jaccard similarities between selected sets are 0.363 and 0.449. Selected cardinality changes by +7.04 and +0.83 features on average. Because the intervening token sequence is shared, these differences identify an effect of earlier visual content on the next visual decision under fixed local wording. The effect includes information propagated through subsequent hidden states.

Under free continuation, 36 Path branches preserve the same emitted sequence through the next trigger. Of these, 33 change the next selected region, and 13 change a correct Normal answer to an incorrect one. All 33 include nonempty identical intervening text. Counting has 45/84 cases with the same emitted sequence and a different next selection, but 44 are consecutive reference actions with no intervening words. We distinguish this action-level identity from the nonempty verbal identity illustrated for Path in Figure 7. More broadly, 31 of the 42 harmed Path middle-reference examples and 16 of the 29 harmed Counting examples retain the same immediate next-token argmax. Thus, selected visual content can affect later reasoning while preserving the immediate next token.

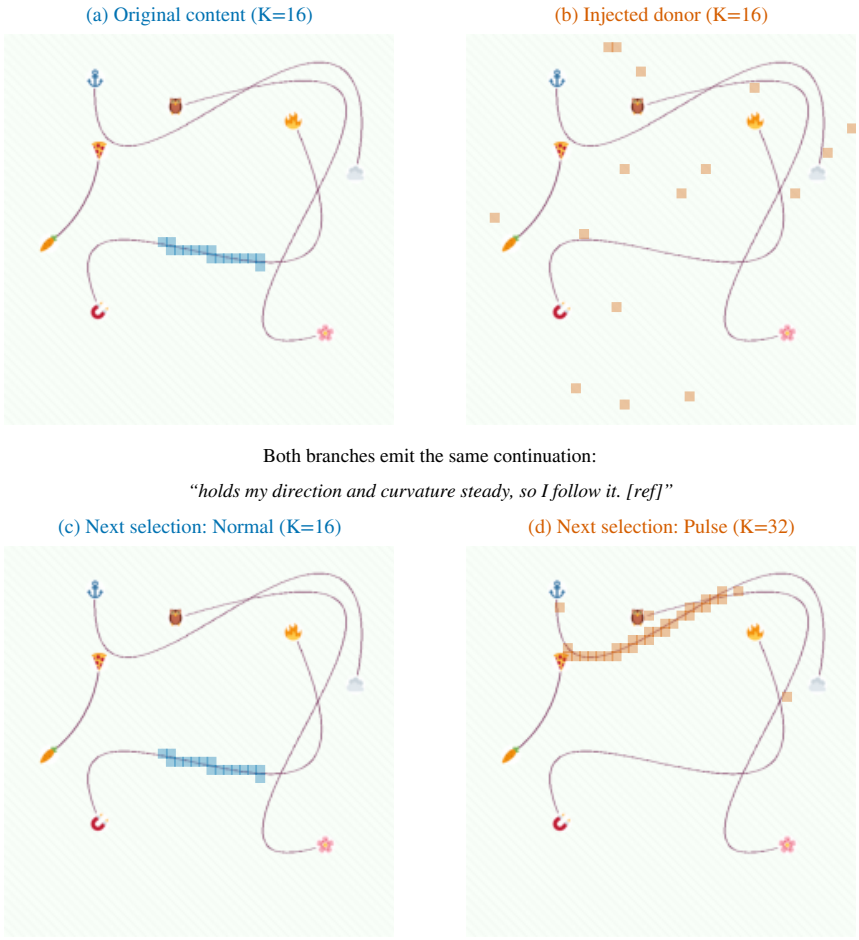

Figure 7: **Identical words, different visual decisions.** (a) Selected content and (b) same-image replacement. Both branches emit the displayed text, but their next selected regions (c,d) are disjoint. Normal recovers all four pairs, while replacement recovers two. The case is closest to median immediate JSD (2.97×10^{-5} nats) among the 13 correct-to-incorrect Path cases with identical nonempty intervening text and changed selections. Overlays show source cells. Panel labels report actual block sizes, subject to a cap of 32.

F Inference Capacity and Evidence Use

To evaluate the inference operating point in Section 5.5, we use both final checkpoints on all 780 standard test images at evidence caps $K \in \{8, 16, 32\}$. Thresholding precedes the cap, so K is a maximum rather than a fixed number per request. Model weights and decoding settings are held fixed across the sweep. The training interface has capacity for 64 annotated features per request.

Table 7: Mean evidence tokens per image on the standard test sets. A smaller cap reduces the visual material inserted into the reasoning trace.

Task	$K = 8$	$K = 16$	$K = 32$
Burden Counting	171.8	201.5	204.3
Path Tracing	346.6	528.3	764.9

The fixed-weight accuracy gains at $K = 8$ relative to $K = 32$ are 10.95 percentage points for Counting (95% CI $[6.43, 15.48]$) and 13.06 points for Path ($[8.61, 17.50]$). Each paired comparison resamples test images and keeps both cap predictions for an image together.

G Visual Access Across Increasing Demands

This appendix documents the controlled conditions plotted in Figure 6(b). Both frozen task models use greedy decoding, a retrieval threshold of 0.5, $K = 32$, and the same 16,384-token generation budget as the standard evaluation. Ground-truth masks are used only after inference to measure which source-image regions were selected. Coverage is a spatial-access measure; it does not score answer correctness.

G.1 Counting: fixed-total target variation

The Counting control uses the ordinary category-counting task (T1a) on 832×832 canvases. Each of 24 scene families has 128 fixed object locations. Target identities are varied to create 32, 64, or 96 targets, while distractor identities fill the remaining locations. The layouts and target-category assignments are nested within each family. Icons are 32 pixels wide, have zero rotation, and do not overlap in alpha support. The model and inference settings are unchanged across conditions.

G.2 Path: fixed-curve crossing variation

Path images use 768×768 canvases, six curves, 3-pixel strokes, white backgrounds, and uniform curve color. Each crossing condition contains 32 geometries balanced across letter and two-digit endpoint labels and sparse and mixed layouts. Exact crossing counts are 3, 6, 9, and 12. Crossings are computed between sampled polylines with a one-pixel deduplication rule. The textual query format and rendering protocol are fixed across conditions.

G.3 Coverage calculation

Selected patch indices are mapped through the merged reference grid to their source-image regions. If U is the union of regions selected over a complete trace and M_j is the visible mask of target object j, Counting coverage is the per-image average of $|M_j \cap U|/|M_j|$ over targets. Path coverage is $|M \cap U|/|M|$, where M is the union of visible curve pixels outside endpoint circles. The reported percentages average these image-level values within each condition. The corresponding values appear in Figure 6(b).

Uncertainty intervals in Figure 6(b) use 10,000 bootstrap resamples of scene families for Counting and geometries for Path. Related images within a family remain together in a resample.