WHEN MORE SEEING DOES NOT HELP: UNDERSTANDING AND MITIGATING HALLUCINATIONS IN THINK-WITH-IMAGES MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Think-with-Images has emerged as a promising paradigm for multimodal reasoning. By acquiring visual evidence through tool calls (*e.g.,* cropping and zooming), these models are widely expected to ground their reasoning in the image and mitigate visual hallucinations. In this paper, we revisit this paradigm and find that more tool invocations fail to consistently reduce hallucinations. Our analyses uncover a pronounced *visual recency bias*: the model over-relies on the recent visual observation while neglecting salient evidence from preceding tool calls, leading to hallucinations. Motivated by these findings, we propose **R**ebalancing **A**ttention over **V**isual **E**vidence (RAVE), a lightweight, training-free strategy that recycles excessive attention from visual tokens in the latest tool output and reallocates it to historical visual contexts. Extensive experiments across six models and fifteen benchmarks demonstrate that RAVE not only mitigates hallucinations but also enhances multimodal perception and reasoning capabilities. These results show that reliable multimodal reasoning hinges not on invoking more tools, but on how to effectively integrate them along the reasoning chain. The code will be released.

1 INTRODUCTION

Building on the success of chain-of-thought (CoT) in large language models (*e.g.,* OpenAI-o1/o3 (Jaech et al., 2024) and DeepSeek-R1 (Guo et al., 2025)), multimodal large language models (MLLMs) increasingly rely on this reasoning paradigm to solve complex visual tasks such as multimodal mathematical reasoning (Lu et al., 2024) and visual spatial planning (Chen et al., 2025). While CoT can improve model performance by decomposing a problem into intermediate textual steps, the resulting long reasoning chains also cause models to drift away from the initial image, leading to severe visual hallucinations (Xu et al., 2026a; Tian et al., 2026). To better ground the reasoning process in the image, *Think-with-Images* (Su et al., 2025) has emerged as a new paradigm that interleaves textual reasoning with visual operations (*e.g.,* cropping (Su et al., 2026) and zooming (Hong et al., 2026)). By iteratively leveraging query-relevant visual observations, these models are widely expected to improve visual grounding and mitigate hallucinations (Zheng et al., 2026).

In this paper, we revisit the prevailing assumption that the *Think-with-Images* paradigm can effectively mitigate hallucinations. As shown in Fig. 1, we uncover a counter-intuitive phenomenon: *more visual tool calls do not lead to fewer hallucinations.* From the figure, initial tool calls are helpful, but the reduction in hallucinations quickly saturates with increasing invocation number and eventually reverses into more unfaithful responses (further experimental details are provided in Appendix A.1). Notably, with more tool calls (*e.g.*, 4 and 5+), hallucination ratios even exceed those without any tool calls, despite the additional visual evidence available to the model. This phenomenon raises a key question: **why do multiple tool invocations fail to consistently reduce hallucinations?**

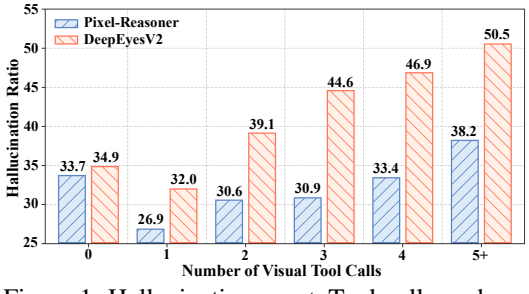


Figure 1: Hallucinations *w.r.t.* Tool-call numbers.

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.

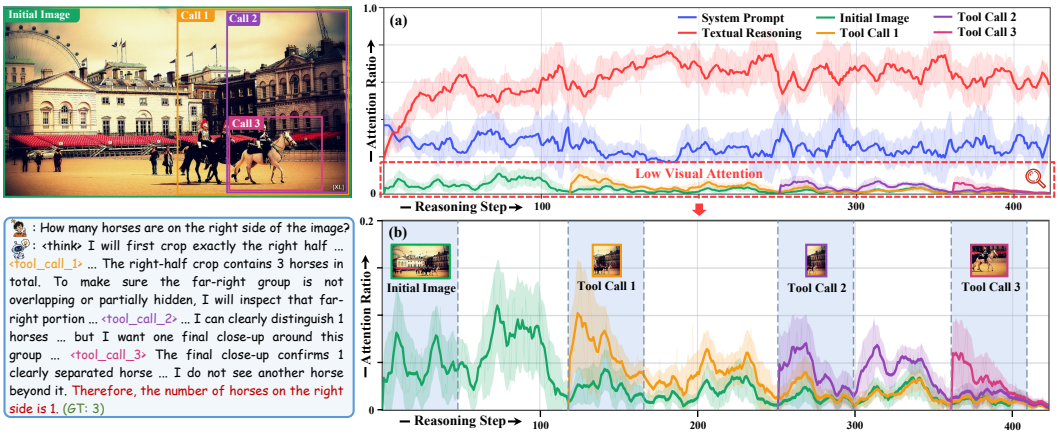


Figure 2: Attention distribution throughout the tool-augmented reasoning chain. (a) Despite multiple tool invocations, visual observations receive substantially less attention than textual context. (b) After each tool call, visual attention shifts sharply toward the newly acquired image, causing preceding visual evidence to be overlooked and leading to hallucinations.

To answer this question, we first examine how models allocate attention to cumulative visual contexts throughout the reasoning trajectory.[1] As shown in Fig. 2 (a), despite receiving explicit visual evidence from multiple tool calls, the model allocates little attention to these intermediate visual observations during subsequent reasoning. This deficiency naturally motivates increasing the attention assigned to visual contexts. Yet, simply increasing visual attention mass fails to effectively mitigate hallucinations (§2.1). Fine-grained analyses reveal that the key lies not in the attention quantity, but in how attention is distributed across different visual observations. As illustrated in Fig. 2 (b), the model shifts its focus toward the newly cropped or zoomed image after tool invocations, while attention to earlier visual evidence rapidly diminishes. We term this phenomenon *visual recency bias* and empirically show its prevalence across *Think-with-Images* models (§2.2).

To further investigate this bias, we visualize token-level attention patterns over all visual contexts. As illustrated in Fig. 3, the model's attention is heavily concentrated on a small subset of visual tokens, mainly from the latest observation (*tool call 3*). Meanwhile, salient cues from earlier observations, such as the three horses in *tool call 1*, are largely overlooked. Unlike visual sink tokens in traditional MLLMs (Kang et al., 2025b; An et al., 2025), these high-attention tokens are not fixed; instead, they dynamically shift toward the latest visual observation, revealing a tool-induced bias rather than model-inherent properties (more discussion in Appendix C). This attention imbalance disrupts the integration of semantic information across visual observations, thereby leading to more hallucinations (§2.3).



Figure 3: Token-level attention visualization.

Based on the above insights, we propose **R**ebalancing **A**ttention over **V**isual **E**vidence (RAVE), a training-free strategy that mitigates *visual recency bias* and facilitates balanced information integration across visual observations. RAVE first collects excessive attention assigned to over-attended tokens in the latest tool output. Then, the recycled attention is allocated to tokens in earlier visual contexts based on their semantic diversity and attention sufficiency, thereby enhancing the contribution of complementary visual evidence. Benefiting from the above design, our strategy is lightweight and training-free, allowing direct inference-time intervention without modifying model parameters. Extensive experiments across six models and fifteen benchmarks demonstrate that RAVE not only mitigates hallucinations but also enhances general perception and reasoning capabilities. In summary, our contributions are listed as follows:

[1] In this paper, we focus on image manipulation tools (*e.g.,* cropping and zooming). Quantitative analyses in Appendix B.1 show that these tool types account for over 99.5% of overall tool usage on perception tasks.

- We systematically analyze hallucinations in *Think-with-Images* models and identify *visual recency bias* as a key factor underlying the diminishing benefits of deeper tool use.
- We propose RAVE, a lightweight and training-free attention reallocation strategy that mitigates *visual recency bias* by rebalancing attention across visual observations.
- Extensive experiments across six models and fifteen benchmarks demonstrate that RAVE consistently mitigates hallucinations and also improves general perception and reasoning capabilities.

2 Understanding Hallucinations in Think-with-Images Models

In this section, we conduct case studies on the widely used hallucination benchmarks, including POPE (Li et al., 2023), HallusionBench (Guan et al., 2024) and AMBER (Wang et al., 2023) to investigate why iterative visual tool use *does not* reliably suppress hallucinations. Meanwhile, we consider three different models, *i.e.,* Pixel-Reasoner (Su et al., 2026), DeepEyesV2 (Hong et al., 2026), and PyVision-Image (Zhao et al., 2026). Note that their default system prompts are used to elicit the native tool-use behaviors. For attention analysis, we use the attention from textual tokens to the visual tokens along the tool-use trajectory. Further preliminaries are provided in Appendix D.1. With this setup, we answer the following research questions: (i) Can simply increasing visual attention mass effectively mitigate hallucinations? (§2.1); (ii) If attention quantity is not the key factor, does the problem instead lie in how attention is allocated across different visual contexts? (§2.2); (iii) If so, how does this attention pattern increase visual hallucinations? (§2.3).

2.1 Finding 1: Increasing Visual Attention Does Not Mitigate Hallucinations

To answer the first research question, we conduct a controlled attention amplification experiment. Specifically, we apply temperature coefficients $T > 1$ to amplify the attention assigned to all visual tokens. As shown in Fig. 4 , increasing visual attention *does not* alleviate hallucinations. Moreover, substantial amplification (*i.e.,* $T = 3.0$) can even slightly increase hallucinations, *e.g.,* 36.2 *vs.* 35.1 for DeepEyesV2 (Hong et al., 2026). These results suggest that insufficient visual attention *is not* the primary cause of hallucination. We therefore turn from the *magnitude* of visual attention to its *allocation* across accumulated observations. In particular, we investigate whether attention is unevenly distributed across different visual observations, leading to the underutilization of information from earlier contexts.

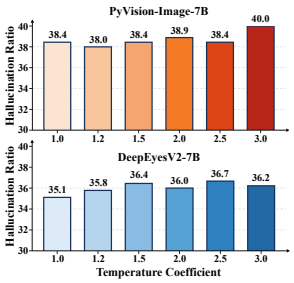


Figure 4: Effect of visual attention amplification.

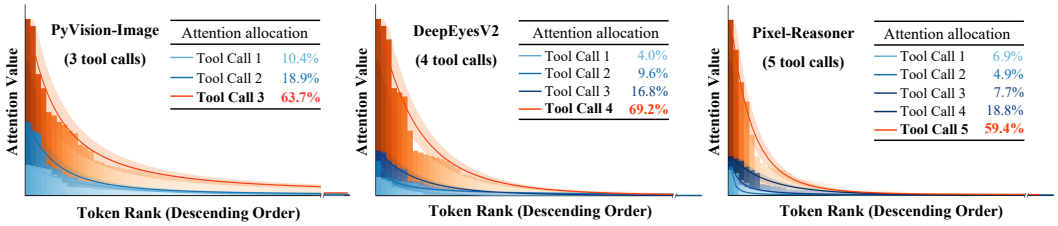


Figure 5: Quantitative analysis of visual attention patterns across tool invocations. Attention is skewed toward the recent tool output and concentrated on a small subset of visual tokens.

2.2 Finding 2: Visual Attention Exhibits a Strong Recency Bias

Building on the above analysis, we next investigate how attention is distributed over the cumulative visual contexts along the reasoning chain. Note that we focus on reasoning trajectories with up to five calls, while excluding those with six or more calls due to their extremely low frequency, *i.e.,* accounting for less than 0.85% of all samples. More discussions of tool-call numbers can be found in Appendix B.2. As shown in Fig. 5, visual attention is highly skewed toward the recent tool call (*e.g.,* for DeepEyesV2, the *tool call 4* receives 69.2% of the total attention). Please refer to Appendix D.2 for more examples. Meanwhile, within the latest tool call, a small fraction of tokens dominates the visual attention (*i.e.,* the sharp decay in the attention curve of Fig. 5). Together, these results show an evident *visual recency bias: Think-with-Images models disproportionately attend to*

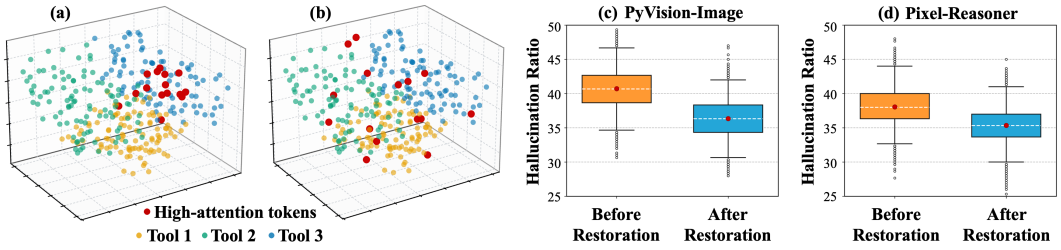


Figure 6: (a) Visual tokens from different tool calls occupy diverse regions of the representation space, but high-attention tokens are concentrated in a limited region and primarily correspond to the latest tool output, *i.e., visual recency bias*. (b) Reallocating attention to a more diverse set of visual tokens via farthest-point sampling broadens semantic coverage. (c–d) Restoring semantic coverage consistently reduces the hallucination ratio for PyVision-Image and Pixel-Reasoner, respectively.

the most recent tool observation, concentrating attention on only a small subset of its visual tokens. This skewed attention allocation may leave complementary information from earlier visual contexts underutilized, motivating us to further investigate its impact on cross-call semantic integration.

2.3 Finding 3: Visual Recency Bias Impedes Semantic Coverage and Ultimately Leads to Hallucinations

The above findings reveal a pronounced *visual recency bias* in the model's attention distribution. We next examine how this bias leads to hallucinations via its impact on cross-call semantic coverage.

Recency Bias Impedes Semantic Coverage. To characterize cross-call semantic coverage, we visualize the hidden representations of visual tokens from different tool calls and highlight those receiving greater attention scores. As shown in Fig. 6 (a), visual tokens from different tool calls span diverse regions of the representation space. However, high-attention tokens cover only a small portion of this space and are mainly associated with more recent tool observations (*e.g., tool 3*). As a result, complementary information from earlier calls is underutilized, leading to limited semantic coverage. More examples illustrating this phenomenon can be found in the Appendix D.3.

Restoring Semantic Coverage Mitigates Hallucination. To assess whether limited semantic coverage contributes to visual hallucinations, we perform a targeted intervention by reallocating attention toward a more diverse set of visual tokens. To construct this set, we use farthest-point sampling (Eldar et al., 1997) to promote diversity among the selected tokens. As shown in Fig. 6 (b), these tokens span a broader portion of the representation space. We then compare the hallucination ratios before and after this restoration. As shown in Fig. 6 (c) and (d), restoring semantic coverage consistently reduces the hallucination ratio for both PyVision-Image (Zhao et al., 2026) and Pixel-Reasoner (Su et al., 2026). These results suggest that insufficient semantic coverage is an important factor contributing to visual hallucinations. When complementary evidence from earlier visual contexts is underutilized, the model operates on incomplete semantic information and becomes more susceptible to hallucination. Conversely, restoring semantic coverage allows the model to incorporate a broader range of visual evidence, leading to more faithful responses.

3 Mitigating Hallucinations in Think-with-Images Models

Our analyses in Sec. 2 suggest that hallucinations arise from attention misallocation, where a small subset of over-attended tokens induces a pronounced *visual recency bias*. Motivated by this insight, we propose **R**ebalancing **A**ttention over **V**isual **E**vidence (RAVE), a training-free strategy that redistributes the misallocated visual attention to facilitate better integration of semantic information. Figure 7 provides an overview of the proposed strategy. Next, we introduce RAVE in detail.

3.1 Recycling Excessive Attention from the Latest Visual Tool

At the current decoding step, let a_i denote the attention value assigned to the i-th visual token of the latest tool output $\mathcal{V}_t$. To adaptively identify the high-attended tokens without relying on a fixed ratio or threshold, we exploit the shape of the ranked attention distribution. Specifically, we fit a smooth curve to the visual attention values sorted in descending order (like in Fig. 5) and identify

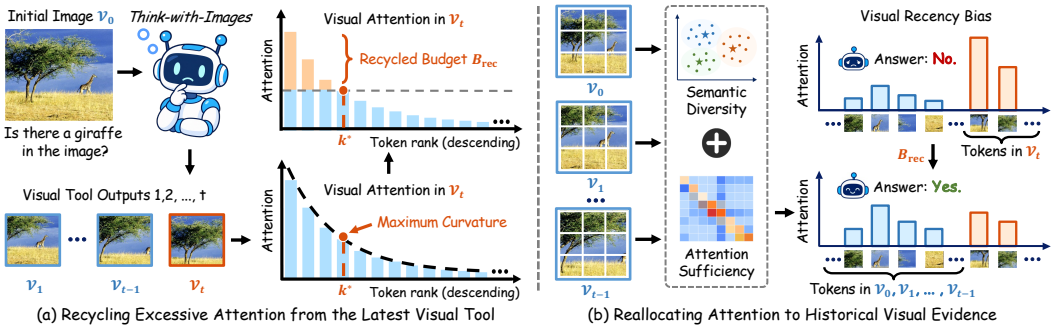


Figure 7: Overview of RAVE. (a) Given the latest visual tool output $\mathcal{V}_t$, RAVE identifies over-attended tokens via an adaptive cutoff derived from the maximum curvature of the ranked attention distribution and recycles their excess attention into a budget B_{rec}. (b) The recycled budget is reallocated to historical visual tokens in $\mathcal{V}_0, \ldots, \mathcal{V}_{t-1}$ according to semantic diversity and attention sufficiency, promoting complementary yet under-attended evidence to mitigate visual hallucinations.

its maximum-curvature point (Satopaa et al., 2011), which characterizes the transition from the dominant high-attention region to the remaining long tail (Kang et al., 2025a). Denoting the rank of this transition point by k^*, we define the adaptive cutoff as $\tau = a_{(k^*)}$, where $a_{(k)}$ is the k-th largest attention value. Tokens with $a_i > \tau$ are then identified as over-attended.

With the above designs, the threshold τ can be dynamically determined by the attention distribution of the current tool output, rather than manually set to a predefined value. The robustness of this adaptive criterion is further validated in Appendix E.1. Based on this criterion, the indices of over-attended tokens are given by $\mathcal{S} = \{i \mid a_i > \tau\}$. Instead of completely removing the visual attention assigned to tokens in $\mathcal{S}$, we truncate their attention values at τ and recycle only the excess attention mass, yielding a recycled budget $B_{\text{rec}} = \sum_{i \in \mathcal{S}} (a_i - \tau)$. This operation preserves salient evidence in the latest observation while releasing the attention mass above τ that would otherwise dominate and suppress evidence from earlier observations.

3.2 Reallocating Attention to Historical Visual Evidence

To compensate for historical visual evidence suppressed by the *visual recency bias*, we allocate the recycled attention budget B_{rec} to tokens in earlier visual contexts. Based on the analyses in Sec. 2.3, our allocation jointly considers *semantic diversity* and *attention sufficiency*, prioritizing preceding visual tokens that provide complementary information yet receive insufficient attention.

Semantic Diversity. To identify historical visual tokens that can complement the currently dominant evidence and broaden the overall semantic coverage, we measure their semantic distance to the over-attended tokens in $\mathcal{S}$. Inspired by the max–min criterion of farthest-point sampling (Eldar et al., 1997), we characterize each prior token by its distance to the nearest token in $\mathcal{S}$. Let $\mathcal{H}$ index all tokens from preceding visual contexts, including the initial image and prior tool outputs, and let $\mathbf{z}_j$ denote the hidden representation of token $j \in \mathcal{H}$. We define $d(\cdot, \cdot) = 1 - \cos(\cdot, \cdot)$ as the cosine distance. Given the attention-dominant token set $\mathcal{S}$, we compute the semantic distance of a historical token as $\min_{i \in \mathcal{S}} d(\mathbf{z}_j, \mathbf{z}_i)$. Here, a larger value indicates lower semantic similarity to the tokens in $\mathcal{S}$, suggesting that the historical token can provide more complementary visual evidence.

Since the above computations only measure the distance to $\mathcal{S}$ while ignoring pairwise similarity among historical tokens, semantically overlapping tokens can still receive high scores and be selected redundantly (*e.g.,* the same region in the initial image and tool invocations). To further promote diversity, we perform farthest-point selection in a greedy manner. We initialize $\mathcal{S}^{(0)} = \mathcal{S}$. At the r-th step, we greedily select the historical token farthest from $\mathcal{S}^{(r-1)}$ and update the set as

$$j_r^\star = \arg\max_j \min_{i \in \mathcal{S}^{(r-1)}} d(\mathbf{z}_j, \mathbf{z}_i), \qquad \mathcal{S}^{(r)} = \mathcal{S}^{(r-1)} \cup \{j_r^\star\}. \tag{1}$$

As selected historical tokens are iteratively added to $\mathcal{S}^{(r)}$, we update the diversity score at each step as $\delta_{j_r^\star} = \min_{i \in \mathcal{S}^{(r-1)}} d(\mathbf{z}_{j_r^\star}, \mathbf{z}_i)$. This score measures the minimum semantic distance of a preceding visual token to both the attention-dominant tokens in $\mathcal{S}$ and previously selected tokens. Therefore, it can serve as a reliable measure of semantic diversity for subsequent attention reallocation.

Visual Attention Reallocation. To prioritize visual tokens that provide complementary information while receiving insufficient attention, we jointly consider *semantic diversity* and *attention sufficiency*. For attention sufficiency, we consider an inverse-attention score $\eta_j = 1 - a_j$, so that less-attended tokens receive higher priority. Here, a_j denotes the visual attention assigned to the preceding visual token $j \in \mathcal{H}$. We then combine attention sufficiency with semantic diversity using adaptive scaling:

$$c_j = \gamma \eta_j + \delta_j, \qquad \gamma = \frac{\mathbb{E}_{j \in \mathcal{H}}[\delta_j]}{\mathbb{E}_{j \in \mathcal{H}}[\eta_j]}, \tag{2}$$

where c_j denotes the composite score used to guide attention reallocation for token j, and γ adaptively balances the magnitudes of semantic diversity and attention sufficiency. To avoid favoring observations with more visual tokens, we further normalize c_j by the sequence length and obtain $\bar{c}_j$. The recycled budget B_{rec} is then redistributed over the historical tokens as

$$a'_j = a_j + B_{\text{rec}} \frac{\bar{c}_j}{\sum_{i \in \mathcal{H}} \bar{c}_i}, \qquad j \in \mathcal{H}. \tag{3}$$

This allocation favors prior visual tokens that are semantically complementary and under-attended. Since the allocation weights sum to one, B_{rec} is reassigned while keeping the visual attention mass unchanged. In practice, RAVE uses attention averaged across all heads and focuses the intervention in the middle layers. We empirically find this configuration to yield stable performance across different models, and further layer-wise and head-wise ablations are provided in Appendix G.2.

3.3 Overview and Analyses of RAVE

We provide an overview of RAVE in Fig. 7, and detailed steps of RAVE are provided in Algorithm 1 of Appendix E.2. Moreover, theoretical analyses in Appendix E.3 establish that RAVE (1) preserves the total visual attention mass while inducing the minimum ℓ_1 change required by the cutoff, (2) provides a constant-factor semantic cover of historical visual tokens, and (3) guarantees a lower bound on the attention gain of each historical token. In practice, we apply RAVE for only k reasoning steps following each tool invocation, as *visual recency bias* is most pronounced during these steps (see Fig. 2 (b) and Appendix D.2). This targeted intervention induces marginal latency overhead, and experimental validations are provided in Fig. 8 (d) and Appendix G.3. Besides, we also empirically study the choice of k and find that this value is model-agnostic, avoiding labor-intensive tuning for new model adaptations (see Fig. 8 (a) and Sec. 4.3).

4 Experiments

4.1 Experimental Setup

Benchmarks and Metrics. For hallucination evaluation, we assess RAVE on five widely used benchmarks: AMBER (Wang et al., 2023), HallusionBench (Guan et al., 2024), POPE (Li et al., 2023), MMHal-Bench (Sun et al., 2024), and CHAIR (Rohrbach et al., 2018a). In addition, we evaluate perception and reasoning capabilities on a diverse suite of benchmarks, including HR-4K/8K (Wang et al., 2025a), MMStar (Chen et al., 2024b), V* (Wu & Xie, 2024), OCR-Bench (Liu et al., 2024b), ChartQA (Masry et al., 2022), RealWorldQA (AI, 2024), MME-RealWorld-Lite (Zhang et al., 2025), MMEvalPro (Huang et al., 2025), and MathVista (Lu et al., 2024). Further details on the benchmarks and evaluation metrics are provided in Appendix F.1.

Models and Baselines. We evaluate RAVE across six representative *Think-with-Images* models, namely Pixel-Reasoner-7B (Su et al., 2026), DeepEyesV2-7B (Hong et al., 2026), PyVision-Image-7B (Zhao et al., 2026), and Qwen-3-VL-2B/4B/30B (Bai et al., 2025). To demonstrate the effectiveness of our method, we compare RAVE against state-of-the-art hallucination mitigation methods, including VAR (Kang et al., 2025b), MemVR (Zou et al., 2024), AGE (Ren et al., 2026), and LEAD (Xu et al., 2026b). Further details on these models and baselines are provided in Appendix F.2 and Appendix F.3. Implementation details can be found in Appendix F.4.

4.2 Quantitative Evaluations

Results on Hallucination Benchmarks. In Table 1, we evaluate RAVE on three *Think-with-Images* models across five hallucination benchmarks. The results show that RAVE consistently

Table 1: Performance comparisons across five hallucination benchmarks. The best result for each backbone is highlighted in **bold**. ↑ (↓) indicates that higher (lower) is better.

Methods	HalBench	POPE-R		POPE-A		POPE-P		MMHal		CHAIR		AMBER
	aAcc↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Hal↓	Score↑	C_S ↓	C_I ↓	Acc↑
DeepEyesV2-7B	63.33	91.10	92.20	83.83	85.47	86.37	87.66	15.62	4.20	17.26	5.81	84.50
+ VAR [ICLR'25]	64.22	90.17	91.53	82.87	84.89	85.43	87.09	16.67	4.17	18.68	6.63	84.93
+ MemVR [ICML'25]	62.00	90.90	91.96	83.30	85.09	85.83	87.36	15.62	4.34	16.68	5.98	85.41
+ AGE [ICLR'26]	63.78	91.20	92.19	83.90	85.48	86.30	87.55	19.79	3.93	16.11	6.03	85.52
+ LEAD [CVPR'26]	63.56	90.30	92.05	83.00	85.25	85.63	87.52	15.62	4.11	18.30	6.72	84.72
+ RAVE (Ours)	**67.56**	**92.40**	**93.51**	**85.90**	**86.14**	**89.03**	**88.80**	**14.24**	**4.81**	**14.74**	**5.19**	**86.48**
PyVision-Image-7B	66.44	90.90	90.82	83.00	84.15	85.70	86.37	28.13	3.91	19.30	7.04	87.86
+ VAR [ICLR'25]	66.00	90.37	90.31	83.07	84.18	85.47	86.04	33.33	3.77	20.36	7.24	87.60
+ MemVR [ICML'25]	64.89	90.70	90.74	82.77	84.10	85.50	86.12	38.54	3.50	19.22	7.33	87.64
+ AGE [ICLR'26]	65.11	91.23	91.13	84.10	84.94	86.40	86.70	32.29	3.72	20.54	7.41	87.70
+ LEAD [CVPR'26]	66.89	91.37	91.24	83.23	84.37	86.53	87.03	28.12	3.92	19.86	6.59	87.26
+ RAVE (Ours)	**68.67**	**92.93**	**92.81**	**85.73**	**85.66**	**87.97**	**87.91**	**27.08**	**4.23**	**17.66**	**6.09**	**89.03**
Pixel-Reasoner-7B	67.11	91.10	90.64	87.77	87.41	89.33	88.80	28.13	3.79	11.24	5.43	86.48
+ VAR [ICLR'25]	67.56	89.93	89.25	87.13	86.66	88.70	88.19	35.42	3.59	11.70	5.71	85.76
+ MemVR [ICML'25]	67.56	91.03	90.46	87.47	87.06	89.70	88.55	27.08	3.82	10.70	4.94	86.22
+ AGE [ICLR'26]	68.67	87.37	86.11	85.60	84.45	86.57	85.34	31.25	3.66	12.18	7.21	83.84
+ LEAD [CVPR'26]	66.67	90.80	90.12	87.13	86.75	88.73	88.15	27.43	3.83	12.42	5.10	85.92
+ RAVE (Ours)	**69.33**	**93.23**	**91.78**	**89.37**	**88.97**	**91.73**	**89.21**	**23.96**	**3.91**	**9.10**	**4.11**	**88.70**

Table 2: Quantitative evaluations on visual perception and mathematical reasoning benchmarks.

Methods	HR-4K	HR-8K	MMStar	MMEval_{Pro}	OCR	ChartQA	RWQA	MME-RW	V*	MathVista
DeepEyesV2-7B	77.90	73.80	64.07	77.14	83.30	88.40	61.05	53.88	81.80	65.80
+ RAVE (Ours)	**78.50**	**74.47**	**65.40**	**79.02**	**84.40**	**88.87**	**65.23**	**54.51**	**84.29**	**67.50**
PyVision-Image-7B	78.10	74.30	65.60	75.44	82.00	86.80	66.67	54.09	88.70	70.51
+ RAVE (Ours)	**79.50**	**75.40**	**66.93**	**77.21**	**83.00**	**87.20**	**67.58**	**54.40**	**89.53**	**71.80**
Pixel-Reasoner-7B	75.66	68.85	62.93	77.29	85.80	84.32	68.13	51.58	84.30	70.05
+ RAVE (Ours)	**76.56**	**70.81**	**64.53**	**78.24**	**86.80**	**86.68**	**69.41**	**52.74**	**85.34**	**71.20**

outperforms prior hallucination mitigation methods. On DeepEyesV2-7B, RAVE improves HallusionBench aAcc (*i.e.,* all accuracy) from 63.33 to 67.56, while outperforming the previous SOTA method, *i.e.,* VAR, by 3.34 points. Moreover, similar gains over previous methods are observed on Pixel-Reasoner-7B and PyVision-Image-7B, where RAVE reduces the hallucination rates of MMHal-Bench by 3.12 and 1.04 points, and CHAIR_S by 1.60 and 1.56 points, respectively. These results demonstrate that RAVE can effectively mitigate hallucinations across different models and benchmarks. We also provide more results on the Qwen-3-VL series (including 2B/4B/30B) in Table 12 of Appendix G.4.

Results on Perception and Reasoning Benchmarks. To explore whether RAVE can also improve the general perceptual and reasoning capabilities of *Think-with-Images* models, we further conduct experiments on ten multimodal benchmarks. As shown in Table 2, RAVE achieves consistent improvements across three backbone models and ten benchmarks. For example, RAVE improves the performance of DeepEyesV2-7B on RealWorldQA (*i.e.,* RWQA) by 4.18 points, and further yields gains of 1.77 points for PyVision-Image-7B on MMEval_{Pro} and 1.96 points for Pixel-Reasoner-7B on HR-8K. These results demonstrate that RAVE not only mitigates hallucinations but also reliably improves the general visual perception and reasoning capabilities of *Think-with-Images* models.

4.3 Ablation Study

Effects of Intervention Length. We first investigate the effect of intervention length by applying RAVE for k decoding steps after each tool invocation, where $k \in \{8, 16, 32, 64, \text{All}\}$. As shown in Fig. 8 (a), increasing the intervention length from 8 to 32 tokens effectively improves performance, with 32 tokens achieving the best results on both DeepEyesV2 and PyVision-Image. Extending the intervention further provides no additional benefit, whereas applying RAVE throughout the entire reasoning trajectory results in a noticeable performance degradation (*i.e.,* "All" in the figure). These results suggest that a moderate intervention length is preferable. Therefore, we set $k = 32$ as the default intervention length.

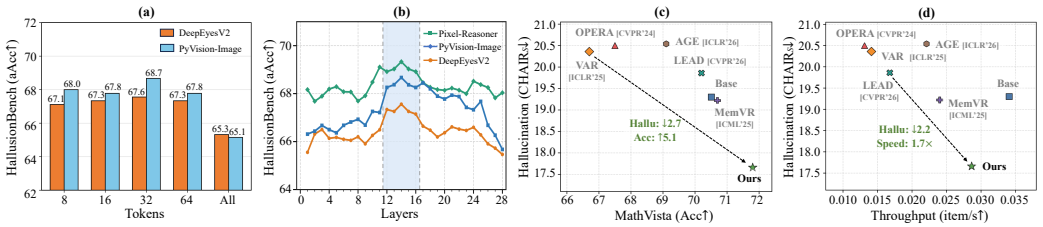


Figure 8: Analysis of RAVE. (a) Effects of intervention length k. (b) Effects of intervention layer. (c) Trade-off between reasoning and hallucination. (d) Balance between performance and efficiency.

Effects of Intervention Layer. We further investigate the effect of the intervention layer. As shown in Fig. 8 (b), RAVE achieves the optimal performance when intervening in middle layers (*i.e.,* layer 12–16). Notably, this pattern is consistently observed across different models. These results align with prior work (Jiang et al., 2025b), which suggests that middle layers are important for visual processing. Therefore, we focus on applying RAVE in these middle layers. Moreover, RAVE uses head-averaged attention for intervention, which we empirically find to yield stable performance across different models. More detailed layer- and head-wise ablations are provided in Appendix G.2.

Effects of Each Component. In Table 3, we ablate three key designs of our RAVE, *i.e.,* identifying over-attended tokens, handling excessive attention, and guiding attention redistribution. Here, Pixel-Reasoner-7B is used as the backbone. For token identification, maximum curvature achieves the best performance, reaching 69.33 on HallusionBench and 3.91 on MMHalBench. This suggests that adaptive token identification based on curvature provides an effective signal for capturing abnormal attention concentration. For handling excessive attention, attention recycling outperforms TAF and OPERA by reusing over-concentrated attention instead of merely suppressing it, leading to gains of +3.77 and +1.13 on the two hallucination benchmarks. For attention reallocation, semantic diversity achieves the best performance, outperforming both the "Initial Image" and PAI strategies by directing attention toward more diverse and complementary visual evidence. These results further validate the effectiveness of our designs in mitigating visual hallucinations.

Table 3: Ablation study of each component in RAVE. "Initial Image" denotes reallocating excessive attention only to the initial image.

Methods	HallusionBench	MMHalBench
Identify Over-attended Tokens:		
SparseVLM (Zhang et al., 2024)	68.00 (+0.00)	2.88 (+0.00)
FastV (Chen et al., 2024a)	68.67 (+0.67)	3.09 (+0.21)
Maximum Curvature (Ours)	**69.33** (+1.33)	**3.91** (+1.03)
Handle Excessive Attention:		
TAF (Ouyang et al., 2026)	65.56 (+0.00)	2.78 (+0.00)
OPERA (Huang et al., 2024)	67.33 (+1.77)	2.95 (+0.17)
Attention Recycling (Ours)	**69.33** (+3.77)	**3.91** (+1.13)
Guide Attention Reallocation:		
Initial Image	67.11 (+0.00)	3.07 (+0.00)
PAI (Liu et al., 2024a)	68.67 (+1.56)	3.16 (+0.09)
Semantic Diversity (Ours)	**69.33** (+2.22)	**3.91** (+0.84)

4.4 More Analyses

Trade-off between Hallucinations and Reasoning. To investigate whether RAVE mitigates hallucinations at the expense of reasoning capability, we evaluate the reasoning and hallucination performance of PyVision-Image-7B on MathVista and CHAIR, respectively. In Fig. 8 (c), it can be observed that our method achieves a favorable trade-off between the two aspects. For example, compared to VAR, RAVE not only reduces hallucinations by 2.7 points but also improves the MathVista accuracy by 5.1 points. These results demonstrate that our method can effectively mitigate hallucinations while also boosting the model's reasoning capabilities.

Balance between Efficiency and Effectiveness. In Fig. 8 (d), we compare the inference throughput of RAVE with several hallucination mitigation approaches. PyVision-Image-7B is used as the base model. The results show that our method achieves a better balance between hallucination reduction and inference efficiency. In particular, compared to LEAD, RAVE not only reduces hallucinations by 2.2 points but is also 1.7× faster in terms of inference throughput. These results indicate that our method can effectively suppress hallucinations while maintaining high inference efficiency.

Qualitative Analysis. In Fig. 9, we qualitatively analyze RAVE on DeepEyesV2-7B. The base model suffers from *visual recency bias*, overemphasizing the latest visual observation while neglecting evidence from previous tool calls, which leads to a hallucinated answer. In contrast, our method effectively integrates evidence across different visual observations, mitigating *visual recency bias* and producing a faithful and accurate answer. More examples can be found in Appendix H.

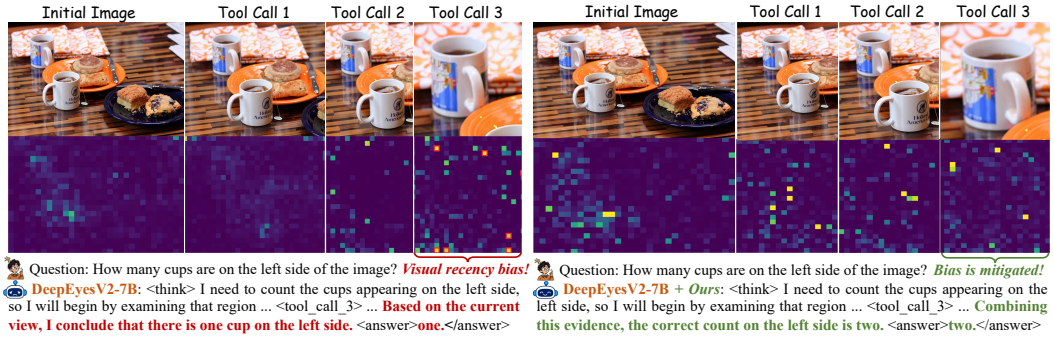


Figure 9: Qualitative analysis. (Left) The original model exhibits *visual recency bias*, leading to a hallucinated answer. (Right) After applying RAVE, the bias is mitigated, yielding the faithful answer.

Analysis on Tool-call Depth. We examine RAVE across different tool-call depths on HallusionBench, grouping samples by the tool invocation counts. As shown in Table 4, RAVE outperforms previous methods, with gains over the base model increasing from +1.45 (two calls) to +4.49 (four or more calls). These results suggest that the benefits of RAVE become more pronounced as tool-use depth increases. More results can be found in Appendix B.3.

Table 4: Performance comparisons with different tool-call numbers. Experiments are conducted on HallusionBench using PyVision-Image-7B.

Num	Samples	Base	OPERA	MemVR	**RAVE (Ours)**
1	5	100.00	20.00	80.00	**100.00** (+0.00)
2	138	80.43	77.54	77.54	**81.88** (+1.45)
3	218	61.47	61.01	61.47	**63.30** (+1.83)
4+	89	55.06	58.43	52.81	**59.55** (+4.49)

5 Related Work

Multimodal Chain-of-Thought Reasoning. Multimodal chain-of-thought (MCoT) extends textual CoT reasoning (Wei et al., 2022) to visual tasks. Prior MCoT approaches largely follow a *Think-about-Images* paradigm, where models condition on a fixed visual input and perform intermediate reasoning in the textual format (Zhang et al., 2023). Recently, models such as DeepEyesV2 (Hong et al., 2026) and PyVision-RL (Zhao et al., 2026) shift toward the *Think-with-Images* paradigm by interleaving textual reasoning with visual tool calls. Instead of reasoning over a fixed visual input, these models actively acquire new visual observations throughout the reasoning process, a capability widely expected to improve visual grounding and mitigate hallucinations (Su et al., 2026; Zheng et al., 2026). In this paper, we revisit this prevailing assumption and uncover a counter-intuitive phenomenon: increasing visual evidence fails to consistently reduce hallucinations.

Visual Hallucinations in Multimodal Reasoning. Visual hallucination (Rohrbach et al., 2018b) is a long-standing challenge in MLLMs. Numerous works have explored mitigation strategies based on contrastive decoding (Leng et al., 2024; Suo et al., 2025), attention calibration (Liu et al., 2024a; Huang et al., 2024), and representation editing (Jiang et al., 2025a; Suo et al., 2026). Recent studies show that hallucination also persists in MCoT reasoning models. Xu et al. (2026a) find that longer text reasoning weakens visual attention, thereby exacerbating hallucinations. Ma et al. (2026) and Xu et al. (2026b) further show that hallucinations are closely associated with textual reasoning steps that exhibit high uncertainty about the visual evidence. Despite the progress, these works primarily study hallucinations in text-based MCoT reasoning, following the *Think-about-Images* paradigm (Wang et al., 2025b). In contrast, we focus on the *Think-with-Images* setting, where models iteratively acquire and integrate new visual evidence through tool interactions.

6 Conclusion

In this work, we investigate hallucinations in *Think-with-Images* models and reveal a counter-intuitive phenomenon: more visual tool calls (*e.g.,* cropping and zooming) do not lead to fewer hallucinations. Our analyses identify *visual recency bias* as a key factor underlying this failure mode, *i.e.,* attention overly concentrated on the recent tool output while neglecting earlier visual evidence. Based on these findings, we propose **R**ebalancing **A**ttention over **V**isual **E**vidence (RAVE), a lightweight and training-free strategy that rebalances attention across visual observations. Extensive experiments across six models and fifteen benchmarks demonstrate that RAVE consistently mitigates hallucinations while also improving general perception and reasoning capabilities.

AI USE STATEMENT

In this work, we used generative AI tools to correct grammar, improve clarity and readability. We also used generative AI tools to assist with classifying sample difficulty in our case study, and all resulting classifications were manually verified by the authors (see Appendix A.2 for details). The generative AI tools were not involved in the central scientific components of this study, such as formulating the research questions, designing the methodology, analysing the results, or producing the conclusions. The authors take full responsibility for the final content of this paper.

REFERENCES

X AI. Grok-1.5 vision preview. *https://x.ai/news/grok-1.5v*, 2024.

Wenbin An, Feng Tian, Sicong Leng, Jiahao Nie, Haonan Lin, QianYing Wang, Ping Chen, Xiaoqin Zhang, and Shijian Lu. Mitigating object hallucinations in large vision-language models with assembly of global and local attention. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 29915–29926. IEEE, 2025.

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.

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.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, et al. Are we on the right way for evaluating large vision-language models? *Advances in Neural Information Processing Systems*, 37:27056–27087, 2024b.

Shiqi Chen, Tongyao Zhu, Ruochen Zhou, Jinghan Zhang, Siyang Gao, Juan Carlos Niebles, Mor Geva, Junxian He, Jiajun Wu, and Manling Li. Why is spatial reasoning hard for vlms? an attention mechanism perspective on focus areas. *arXiv preprint arXiv:2503.01773*, 2025.

Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.

Yuval Eldar, Michael Lindenbaum, Moshe Porat, and Yehoshua Y Zeevi. The farthest point strategy for progressive image sampling. *IEEE transactions on image processing*, 6(9):1305–1315, 1997.

Teofilo F Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical computer science*, 38:293–306, 1985.

Tianrui Guan, Fuxiao Liu, Xiyang Wu, Ruiqi Xian, Zongxia Li, Xiaoyu Liu, Xijun Wang, Lichang Chen, Furong Huang, Yaser Yacoob, et al. Hallusionbench: an advanced diagnostic suite for entangled language hallucination and visual illusion in large vision-language models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14375–14385. IEEE, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Jack Hong, Chenxiao Zhao, ChengLin Zhu, Weiheng Lu, and Guohai Xu. Deepeyesv2: Toward agentic multimodal model. In *International Conference on Learning Representations*, volume 2026, pp. 114851–114872, 2026.

Jinsheng Huang, Liang Chen, Taian Guo, Fu Zeng, Yusheng Zhao, Bohan Wu, Ye Yuan, Haozhe Zhao, Zhihui Guo, Yichi Zhang, et al. Mmevalpro: Calibrating multimodal benchmarks towards trustworthy and efficient evaluation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 4805–4822, 2025.

Qidong Huang, Xiaoyi Dong, Pan Zhang, Bin Wang, Conghui He, Jiaqi Wang, Dahua Lin, Weiming Zhang, and Nenghai Yu. Opera: Alleviating hallucination in multi-modal large language models via over-trust penalty and retrospection-allocation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13418–13427. IEEE, 2024.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Nick Jiang, Anish Kachinthaya, Suzanne Petryk, and Yossi Gandelsman. Interpreting and editing vision-language representations to mitigate hallucinations. In *International Conference on Learning Representations*, volume 2025, pp. 63582–63605, 2025a.

Zhangqi Jiang, Junkai Chen, Beier Zhu, Tingjin Luo, Yankun Shen, and Xu Yang. Devils in middle layers of large vision-language models: Interpreting, detecting and mitigating object hallucinations via attention lens. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 25004–25014. IEEE, 2025b.

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, 2025a.

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, 2025b.

Sicong Leng, Hang Zhang, Guanzheng Chen, Xin Li, Shijian Lu, Chunyan Miao, and Lidong Bing. Mitigating object hallucinations in large vision-language models through visual contrastive decoding. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13872–13882. IEEE, 2024.

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, 2023.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

Shi Liu, Kecheng Zheng, and Wei Chen. Paying more attention to image: A training-free method for alleviating hallucination in lvlms. In *European Conference on Computer Vision*, pp. 125–140. Springer, 2024a.

Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xu-Cheng Yin, Cheng-Lin Liu, Lianwen Jin, and Xiang Bai. Ocrbench: on the hidden mystery of ocr in large multimodal models. *Science China Information Sciences*, 67(12):220102, 2024b.

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.

Xinyu Lyu, Shuailong Wang, Beitao Chen, Jingkuan Song, Lianli Gao, et al. Flexac: Towards flexible control of associative reasoning in multimodal large language models. *Advances in Neural Information Processing Systems*, 38:2473–2514, 2026.

Ji Ma, Wei Suo, Peng Wang, and Yanning Zhang. Understanding and mitigating hallucinations in multimodal chain-of-thought models. *arXiv preprint arXiv:2603.27201*, 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.

Shuyi Ouyang, Hongyi Wang, Gongfan Fang, Xinyin Ma, Lanfen Lin, and Xinchao Wang. Taming the phantom: Token-asymmetric filtering for hallucination mitigation in large vision-language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 8206–8214, 2026.

Hairui Ren, Zixuan Wang, Yibo Yang, He Zhao, Fan Tang, Dandan Guo, and Yi Chang. Imitating the truth: Attention-aware truth-guided enhancement for hallucination mitigation in large vision-language models. In *International Conference on Learning Representations*, volume 2026, pp. 43994–44020, 2026.

Anna Rohrbach, Lisa Anne Hendricks, Kaylee Burns, Trevor Darrell, and Kate Saenko. Object hallucination in image captioning. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 4035–4045, 2018a.

Anna Rohrbach, Lisa Anne Hendricks, Kaylee Burns, Trevor Darrell, and Kate Saenko. Object hallucination in image captioning. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 4035–4045, 2018b.

Ville Satopaa, Jeannie Albrecht, David Irwin, and Barath Raghavan. Finding a" kneedle" in a haystack: Detecting knee points in system behavior. In *2011 31st international conference on distributed computing systems workshops*, pp. 166–171. IEEE, 2011.

Alex Su, Haozhe Wang, Weiming Ren, Fangzhen Lin, and Wenhu Chen. Pixel reasoner: Incentivizing pixel space reasoning via curiosity-driven reinforcement learning. *Advances in Neural Information Processing Systems*, 38:8222–8251, 2026.

Zhaochen Su, Peng Xia, Hangyu Guo, Zhenhua Liu, Yan Ma, Xiaoye Qu, Jiaqi Liu, Yanshu Li, Kaide Zeng, Zhengyuan Yang, et al. Thinking with images for multimodal reasoning: Foundations, methods, and future frontiers. *arXiv preprint arXiv:2506.23918*, 2025.

Zhiqing Sun, Sheng Shen, Shengcao Cao, Haotian Liu, Chunyuan Li, Yikang Shen, Chuang Gan, Liangyan Gui, Yu-Xiong Wang, Yiming Yang, et al. Aligning large multimodal models with factually augmented rlhf. In *Findings of the association for computational linguistics: ACL 2024*, pp. 13088–13110, 2024.

Wei Suo, Lijun Zhang, Mengyang Sun, Lin Yuanbo Wu, Peng Wang, and Yanning Zhang. Octopus: Alleviating hallucination via dynamic contrastive decoding. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 29904–29914. IEEE, 2025.

Wei Suo, Hanzu Zhang, Lijun Zhang, Ji Ma, Peng Wang, and Yanning Zhang. Hallucination-aware intermediate representation edit in large vision-language models. *arXiv preprint arXiv:2603.29405*, 2026.

Xinyu Tian, Shu Zou, Zhaoyuan Yang, Mengqi He, Fabian Waschkowski, Lukas Wesemann, Peter Tu, and Jing Zhang. More thought, less accuracy? on the dual nature of reasoning in vision-language models. In *International Conference on Learning Representations*, volume 2026, pp. 133928–133965, 2026.

Junyang Wang, Yuhang Wang, Guohai Xu, Jing Zhang, Yukai Gu, Haitao Jia, Jiaqi Wang, Haiyang Xu, Ming Yan, Ji Zhang, et al. Amber: An llm-free multi-dimensional benchmark for mllms hallucination evaluation. *arXiv preprint arXiv:2311.07397*, 2023.

Wenbin Wang, Liang Ding, Minyan Zeng, Xiabin Zhou, Li Shen, Yong Luo, Wei Yu, and Dacheng Tao. Divide, conquer and combine: A training-free framework for high-resolution image perception in multimodal large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 7907–7915, 2025a.

Yaoting Wang, Shengqiong Wu, Yuecheng Zhang, Shuicheng Yan, Ziwei Liu, Jiebo Luo, and Hao Fei. Multimodal chain-of-thought reasoning: A comprehensive survey. *arXiv preprint arXiv:2503.12605*, 2025b.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

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.

Zhongxing Xu, Chengzhi Liu, Qingyue Wei, Juncheng Wu, James Zou, Xin Wang, Yuyin Zhou, and Sheng Liu. More thinking, less seeing? assessing amplified hallucination in multimodal reasoning models. *Advances in Neural Information Processing Systems*, 38:82878–82905, 2026a.

Zhongxing Xu, Zhonghua Wang, Zhe Qian, Dachuan Shi, Feilong Tang, Ming Hu, Shiyan Su, Xiaocheng Zou, Wei Feng, Dwarikanath Mahapatra, et al. Thinking in uncertainty: Mitigating hallucinations in mlrms with latent entropy-aware decoding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11064–11075, 2026b.

Le Yang, Ziwei Zheng, Boxu Chen, Zhengyu Zhao, Chenhao Lin, and Chao Shen. Nullu: Mitigating object hallucinations in large vision-language models via halluspace projection. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14635–14645. IEEE, 2025.

YiFan Zhang, Huanyu Zhang, Haochen Tian, Chaoyou Fu, Shuangqing Zhang, Junfei Wu, Feng Li, Kun Wang, Qingsong Wen, Zhang Zhang, et al. Mme-realworld: Could your multimodal llm challenge high-resolution real-world scenarios that are difficult for humans? In *International Conference on Learning Representations*, volume 2025, pp. 89655–89701, 2025.

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

Zhuosheng Zhang, Aston Zhang, Mu Li, Hai Zhao, George Karypis, and Alex Smola. Multimodal chain-of-thought reasoning in language models. *arXiv preprint arXiv:2302.00923*, 2023.

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.

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.

Xin Zou, Yizhou Wang, Yibo Yan, Yuanhuiyi Lyu, Kening Zheng, Sirui Huang, Junkai Chen, Peijie Jiang, Jia Liu, Chang Tang, et al. Look twice before you answer: Memory-space visual retracing for hallucination mitigation in multimodal large language models. *arXiv preprint arXiv:2410.03577*, 2024.

CONTENTS OF SUPPLEMENTARY MATERIAL

A More Experimental Details and Analyses for Case Study **15**

A.1 Further Experimental Details for Case Study 15

A.2 Further Analyses on Sample Difficulty 15

B Analyses of Visual Tool Use **16**

B.1 Tool Type Statistics 16

B.2 Tool Invocation Depth Statistics 16

B.3 Hallucinations across Tool Invocation Depth 17

C Distinguishing Visual Recency Bias from Attention Sinks **17**

D Understanding Hallucinations in Think-with-Images Models **18**

D.1 Preliminary 18

D.2 More Qualitative and Quantitative Results on Visual Recency Bias 18

D.3 More Results on Semantic Coverage 19

E Mitigating Hallucinations in Think-with-Images Models **19**

E.1 Robustness of Maximum Curvature 19

E.2 Algorithm of `RAVE` 20

E.3 Theoretical Analysis of `RAVE` 20

F More Experimental Settings **22**

F.1 Benchmarks and Metrics 22

F.2 Think-with-Images Models 24

F.3 Baseline Methods 24

F.4 Experimental Details 25

G Additional Experimental Results and Analyses **25**

G.1 Further Ablation Study of Each Component 25

G.2 Intervention Layer and Attention Heads 26

G.3 Latency Analyses 27

G.4 Evaluations on Qwen-3-VL-series 27

H Additional Qualitative Examples **27**

I Limitations and Future Work **28**

A More Experimental Details and Analyses for Case Study

A.1 Further Experimental Details for Case Study

In Fig. 1 of the main paper, we conduct controlled experiments on HallusionBench (Guan et al., 2024) to study how visual tool-call depth affects hallucination. For each model, we retain its released system prompt and native tool interface, and append an exact-depth instruction to the user query. We evaluate the same set of HallusionBench examples independently at each prescribed tool-call depth, thereby controlling for differences in sample difficulty. A tool call is counted only if it successfully returns a valid, non-empty visual observation. We then report the hallucination ratio separately for each controlled tool-call depth.

A.2 Further Analyses on Sample Difficulty

To examine whether the trend observed in Fig. 1 is due to sample difficulty rather than tool-induced bias, we stratify the same HallusionBench samples (Guan et al., 2024) into easy, medium, and hard subsets. This analysis tests whether samples receiving more tool calls are inherently more difficult and therefore more prone to hallucination. We use GPT-5.5 as the difficulty judge, yielding easy, medium, and hard subsets of 125, 183, and 142 samples, respectively. Three graduate-level researchers then independently review the resulting partition, with Cohen's κ (Cohen, 1960) values of 0.81, 0.85, and 0.84, respectively. The high agreement supports the reliability of the difficulty partition. After obtaining these difficulty annotations, we conduct independent experiments with three different random seeds to evaluate whether the relationship between tool-call frequency and hallucination persists within each difficulty subset. As shown in Fig. 10, the main trend persists across all difficulty groups: a small tool-invocation count reduces hallucination, whereas a larger number leads to increased hallucination. This consistency across difficulty levels suggests that the increased hallucination ratio with tool-call number is not simply driven by difficult samples receiving more tool calls.

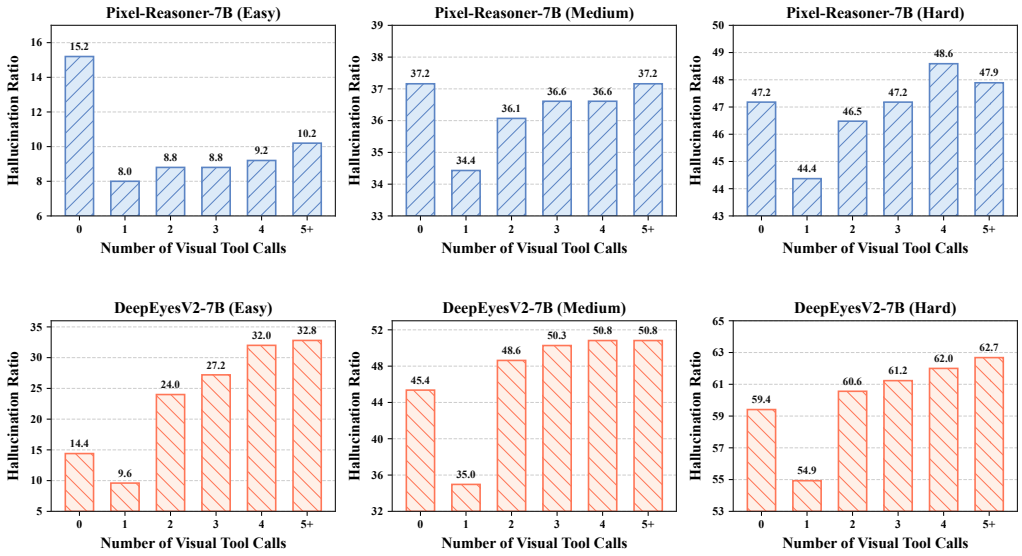


Figure 10: Hallucination ratio across sample difficulty. Results are shown for Pixel-Reasoner-7B (top) and DeepEyesV2-7B (bottom) on HallusionBench, divided into easy, medium, and hard subsets. Across difficulty levels, hallucination decreases with a single visual tool call but generally increases as additional tool calls are invoked.

B Analyses of Visual Tool Use

B.1 Tool Type Statistics

This paper focuses on hallucinations induced by image manipulation tools, particularly cropping and zooming. To characterize how these tools are used in perception-related queries, we analyze the native tool-use patterns of three models (*i.e.,* Pixel-Reasoner-7B (Su et al., 2026), DeepEyesV2-7B (Hong et al., 2026), and PyVision-Image-7B (Zhao et al., 2026)) across four hallucination benchmarks (*i.e.,* HallusionBench (Guan et al., 2024), AMBER (Wang et al., 2023), POPE (Li et al., 2023), and MMHal-Bench (Sun et al., 2024)). We count each successful tool invocation and categorize it according to the underlying visual operation. For models that perform visual manipulation through `Python`, we categorize the resulting visual operations as follows: *crop* for extracting a subregion, *zoom* for redisplaying or rescaling the full image, *region grounding* for adding visual overlays to indicate localized regions, and *other* for tool types that do not fall into the above categories. Figure 11 shows the tool-type composition across models and benchmarks. As shown in the figure, cropping and zooming dominate tool usage across all settings, while other visual operations are rarely invoked. Across all evaluated models, cropping and zooming account for an average of **99.6%** of tool calls. This finding provides empirical validations for our focus on cropping and zooming in the main paper, as they are the most commonly used image manipulation operations in perception-related queries.

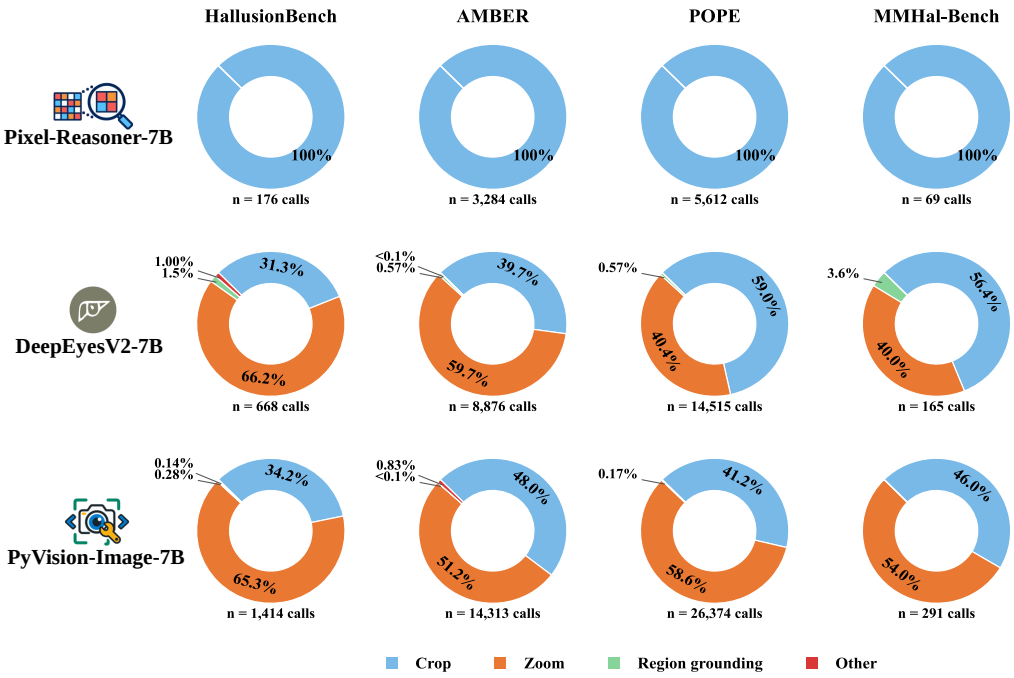


Figure 11: Statistics of visual tool types across three models and four hallucination benchmarks. Each donut shows the percentage of tool calls for different tool types, with n denoting the total number of calls. Cropping and zooming dominate visual tool usage, accounting for 99.6% of tool calls on average.

B.2 Tool Invocation Depth Statistics

In the main paper, we focus our analysis on reasoning trajectories with up to five tool calls. We further examine the empirical distribution of tool invocation depths across three models, *i.e.,* Pixel-Reasoner-7B (Su et al., 2026), DeepEyesV2-7B (Hong et al., 2026), and PyVision-Image-7B (Zhao et al., 2026), to assess the prevalence of longer trajectories. Table 5 reports the proportion of trajectories at each depth on HallusionBench (Guan et al., 2024), POPE (Li et al., 2023), RealWorldQA (AI,

2024), and MME-RealWorld-Lite (Zhang et al., 2025). To account for the different benchmark sizes, we pool all evaluated examples and report sample-weighted proportions rather than an unweighted average of four benchmark-level percentages. This analysis includes all three POPE splits (9,000 examples in total). As shown in Table 5, trajectories with six or more calls are rare across all three backbones, with an average frequency of only 0.84%, supporting our focus on trajectories with at most five calls.

Table 5: Tool invocation depth statistics. Each entry reports the percentage of trajectories at a given tool-call depth, micro-averaged over the pooled examples from HallusionBench, all three POPE splits, RealWorldQA, and MME-RealWorld-Lite. The "6+ calls" column denotes trajectories with six or more tool calls, and Avg. reports the mean across the three backbones.

Model	0 calls	1 call	2 calls	3 calls	4 calls	5 calls	6+ calls
Pixel-Reasoner-7B	33.86	65.05	0.95	0.13	0.00	0.01	0.00
DeepEyesV2-7B	18.25	55.39	12.86	7.38	3.63	1.42	1.08
PyVision-Image-7B	0.21	0.17	25.43	49.80	19.05	3.88	1.45
Avg.	**17.44**	**40.20**	**13.08**	**19.10**	**7.56**	**1.77**	**0.84**

B.3 Hallucinations across Tool Invocation Depth

Table 6 provides additional results across more models and tool-call depths. From the table, the largest gain is consistently observed at the highest reported tool-call depth, further supporting the effectiveness of RAVE in reducing hallucinations during longer tool-use trajectories.

Table 6: Hallucinations across different tool-call depths on additional models.

Benchmark	Call	N	Base	OPERA		MemVR		RAVE (ours)	
				Score	Δ	Score	Δ	Score	Δ
DeepEyesV2-7B (Hong et al., 2026)									
HallusionBench	1	83	56.63	57.83	+1.20	50.60	-6.02	62.65	**+6.02**
	2	90	61.11	62.22	**+1.11**	60.00	-1.11	56.67	-4.44
	3	46	41.30	41.30	+0.00	41.30	+0.00	56.52	**+15.22**
	4+	56	37.50	42.86	+5.36	46.43	+8.93	58.93	**+21.43**
PyVision-Image-7B (Zhao et al., 2026)									
HallusionBench	1	5	100.00	20.00	-80.00	80.00	-20.00	100.00	**+0.00**
	2	138	80.43	77.54	-2.90	77.54	-2.90	81.88	**+1.45**
	3	218	61.47	61.01	-0.46	61.47	+0.00	63.30	**+1.83**
	4+	89	55.06	58.43	+3.37	52.81	-2.25	59.55	**+4.49**

C Distinguishing Visual Recency Bias from Attention Sinks

VAR (Kang et al., 2025b) investigates visual attention sinks in conventional MLLMs, where a small set of semantically uninformative tokens consistently receives disproportionately high attention. VAR attributes this phenomenon to massive activations along a few fixed hidden-state dimensions inherited from the base language model. Although the high-attention tokens associated with *visual recency bias* may appear similar to VAR's visual sinks because they likewise dominate the attention distribution, the two are fundamentally different in how they arise. VAR's visual sink tokens form a largely fixed, model-intrinsic set of high-attention tokens that is invariant to the user query or the model's current decoding step. In contrast, the high-attention tokens induced by *visual recency bias* dynamically shift toward the latest visual observation according to the current decoding step's attention distribution, yielding a query- and step-dependent set that varies across queries and tool calls, as shown in Fig. 12 and Fig. 3.

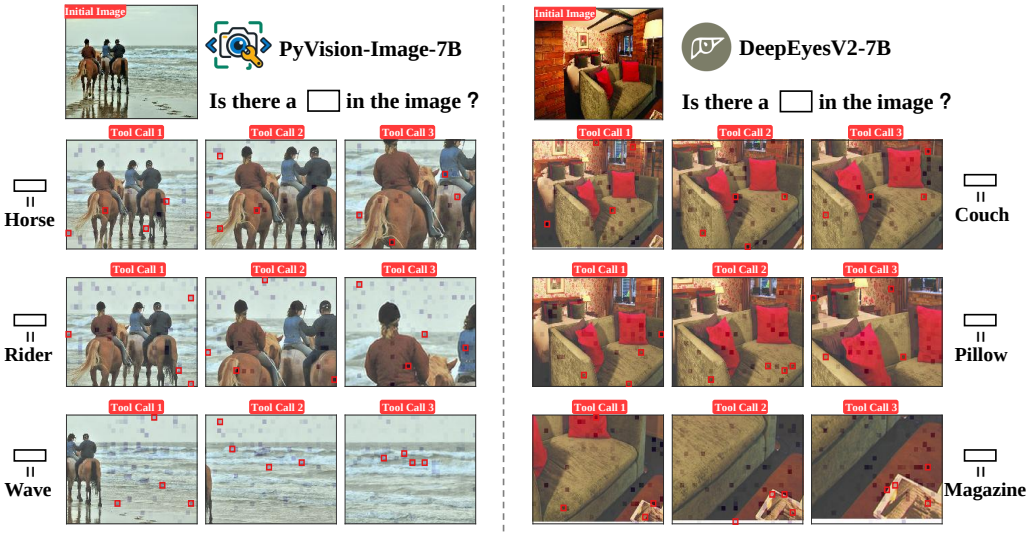


Figure 12: High-attention tokens differ from fixed visual sinks. For the same visual context, changing the queried noun shifts the selected high-attention tokens toward different image regions, indicating that they are query-conditioned rather than fixed model-intrinsic sinks.

D Understanding Hallucinations in Think-with-Images Models

D.1 Preliminary

Multi-turn Visual Tool Reasoning. We consider a multimodal reasoning model $\mathcal{M}_\theta$ that interleaves textual reasoning with iterative visual tool use under the *Think-with-Images* paradigm. Given an original image and a textual query q, the model may invoke visual tools (*e.g.,* cropping and zooming) to acquire additional visual evidence during reasoning. Let $\mathcal{V}_0$ denote the visual tokens of the original image. At the t-th tool-use turn, the model produces textual reasoning r_t and invokes a visual tool, which returns a new visual observation represented by visual tokens $\mathcal{V}_t$. The complete multimodal reasoning trajectory is denoted as

$$\xi = \left\{\mathcal{V}_0,\, q,\, (r_t, \mathcal{V}_t)_{t=1}^{T},\, y\right\}, \tag{4}$$

where y denotes the final answer. For simplicity, we omit the explicit tool actions a_t from the notation. After the t-th tool invocation, the visual evidence accumulated along the trajectory is $\mathcal{V}_{0:t} = \{\mathcal{V}_0, \mathcal{V}_1, \ldots, \mathcal{V}_t\}$. As more visual observations are introduced through successive tool calls, faithful reasoning requires the model to preserve and integrate evidence across the entire visual trajectory rather than relying only on the newly involved observation. This motivates our central analysis of how visual evidence is attended and integrated across multiple tool invocations.

D.2 More Qualitative and Quantitative Results on Visual Recency Bias

We provide additional qualitative and quantitative evidence for the *visual recency bias* observed in the main paper. (1) *Qualitatively*, Fig. 20 – 22 show representative trajectories with 2, 4, and 5 tool calls from PyVision-Image-7B (Zhao et al., 2026), complementing the 3-call example in Fig. 2. In these cases, relevant evidence appears in earlier observations, while attention shifts toward the most recent tool output. (2) *Quantitatively*, Fig. 13 extends the analysis to Pixel-Reasoner-7B (Su et al., 2026), DeepEyesV2-7B (Hong et al., 2026), and PyVision-Image-7B (Zhao et al., 2026) across trajectories with two to five tool calls. Across models and call depths, the latest tool observation consistently receives a disproportionate share of visual attention, which is further concentrated on a small subset of its tokens. Together, these results show that *visual recency bias* persists across model backbones and trajectory lengths.

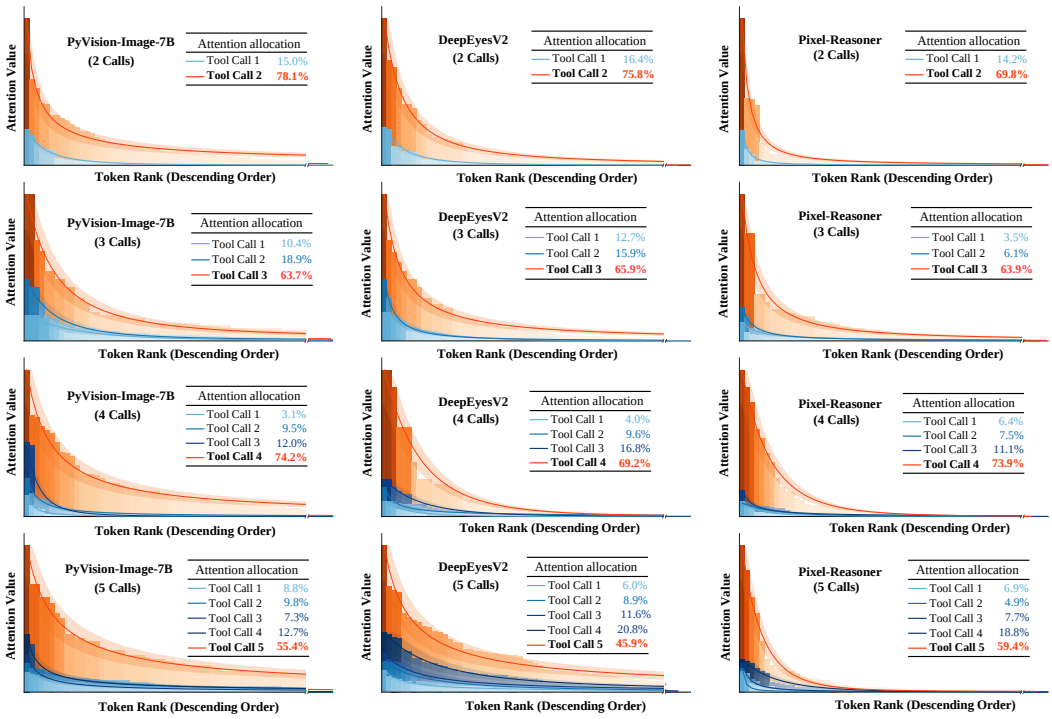


Figure 13: *Visual recency bias* across models and tool-call depths. Columns correspond to different model backbones, while rows correspond to different numbers of tool calls (2, 3, 4, and 5). Curves show visual-token attention ranked in descending order, and percentages indicate the attention allocated to each observation. Across settings, the most recent observation consistently receives the largest share of visual attention, showing the evident *visual recency bias*.

D.3 More Results on Semantic Coverage

To complement the semantic coverage analysis in Fig. 6 of the main paper, we further examine semantic coverage across different tool-call depths. Fig. 14 presents representative examples with 2, 3, 4, and 5 visual-tool calls. Within each panel, the left plot shows the high-attention tokens before restoration, while the right plot shows those after restoration. Before restoration, high-attention tokens tend to be more closely associated with the most recent tool return, leaving earlier observations underrepresented. After restoration, high-attention tokens extend to all displayed source observations, including earlier tool returns. This broader semantic coverage improves the integration of complementary evidence from earlier observations.

E Mitigating Hallucinations in Think-with-Images Models

E.1 Robustness of Maximum Curvature

To further validate the robustness of this adaptive criterion, Table 7 compares maximum-curvature with two fixed attention-ranking strategies. In the table, Top-k selects the $k = 50$ visual tokens with the largest attention weights, while Top-p selects the top $p = 10\%$ of visual tokens ranked by attention. Maximum curvature consistently outperforms both alternatives across all four benchmarks, with a particularly large gain on HR-8K. These results validate the benefit of adapting the criterion to each attention distribution rather than using a fixed token count or token-retention ratio.

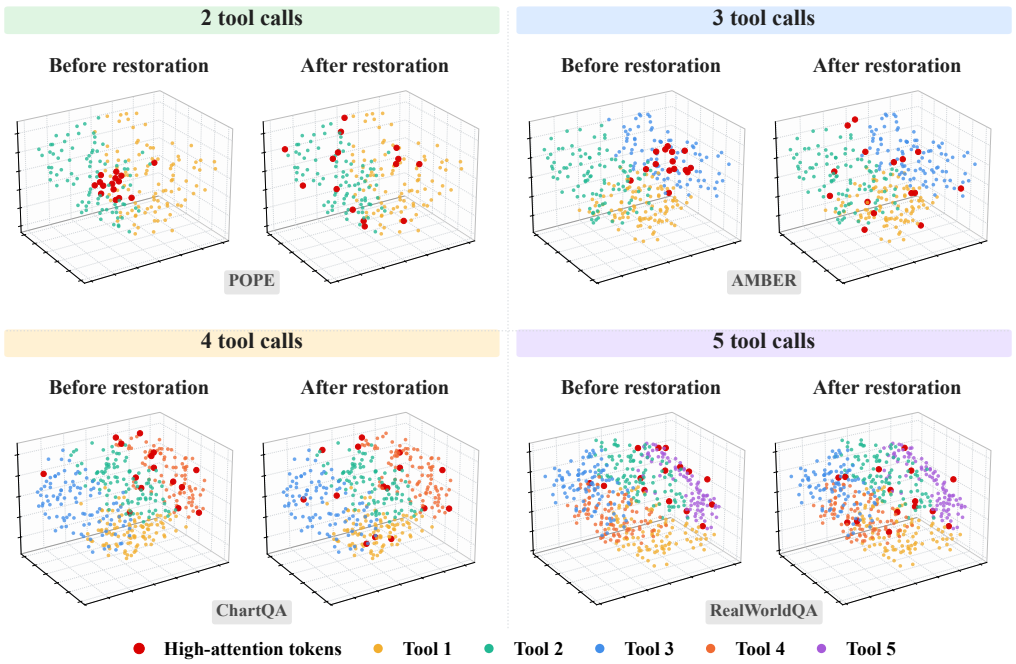


Figure 14: Semantic coverage across 2 to 5 tool calls. Before restoration, high-attention tokens are associated with recent tool returns and cover a limited region of the semantic space. After restoration, they extend to diverse regions across source observations, broadening semantic coverage.

Table 7: Comparison of token-identification strategies

Method	**HallusionBench** aAcc↑	**MMHal-Bench** Score↑	**HR-8K** Acc.↑	**MMStar** Acc.↑
Top-k	68.67 (+0.00)	3.82 (+0.00)	66.13 (+0.00)	64.33 (+0.00)
Top-p	68.89 (+0.22)	3.82 (+0.00)	66.00 (-0.13)	64.33 (+0.00)
Maximum Curvature (Ours)	**69.33** (+0.66)	**3.91** (+0.09)	**70.81** (+4.68)	**64.53** (+0.20)

E.2 Algorithm of RAVE

We provide detailed steps of RAVE in Algorithm 1.

E.3 Theoretical Analysis of RAVE

Setup. At a fixed decoding position, layer, and head, let a be a nonnegative attention row on a finite index set $\mathcal{I}$, with $\sum_{i\in\mathcal{I}} a_i = 1$. Let $\mathcal{V}_t$ and $\mathcal{H}$ index the latest and historical visual tokens, respectively, with $\mathcal{V}_t \cap \mathcal{H} = \varnothing$ and $|\mathcal{H}| \geq 1$. For the cutoff $\tau > 0$, recall that

$$\mathcal{S} = \{i \in \mathcal{V}_t : a_i > \tau\}, \qquad B_{\text{rec}} = \sum_{i\in\mathcal{S}} (a_i - \tau). \tag{5}$$

We consider $B_{\text{rec}} > 0$; otherwise, the attention row is unchanged. Using the length-normalized scores $\bar{c}_j$ from the main paper, RAVE applies

$$\begin{aligned} a'_i &= \min\{a_i, \tau\}, && i \in \mathcal{V}_t, \\ a'_j &= a_j + B_{\text{rec}} \frac{\bar{c}_j}{\sum_{k\in\mathcal{H}} \bar{c}_k}, && j \in \mathcal{H}, \end{aligned} \tag{6}$$

and leaves non-visual entries unchanged.

Lemma 1 (Attention recycling). *For any scores $\bar{c}_j \geq 0$ with $\sum_{j\in\mathcal{H}} \bar{c}_j > 0$, the update in Eq. 6 is nonnegative and normalized, and*

$$\sum_{i\in\mathcal{V}_t\cup\mathcal{H}} a'_i = \sum_{i\in\mathcal{V}_t\cup\mathcal{H}} a_i, \qquad \|a' - a\|_1 = 2B_{\mathrm{rec}}. \tag{7}$$

For every visual token, $a_i \geq \tau$ implies $a'_i \geq \tau$. Moreover, a' minimizes the ℓ_1 change among nonnegative, normalized rows capped by τ on $\mathcal{V}_t$, with nonvisual entries fixed.

Proof. Clipping removes B_{rec} from $\mathcal{V}_t$, and reallocation adds the same amount to $\mathcal{H}$. This proves nonnegativity, normalization, visual-attention conservation, and the ℓ_1 identity. Latest-call tokens with $a_i \geq \tau$ retain attention τ, while historical attention does not decrease.

For any feasible alternative $\hat{a}$, equal total mass and the cap on $\mathcal{V}_t$ give

$$\frac{1}{2}\|\hat{a} - a\|_1 = \sum_{i\in\mathcal{I}} (a_i - \hat{a}_i)_+ \geq \sum_{i\in\mathcal{S}} (a_i - \tau) = B_{\mathrm{rec}}, \tag{8}$$

where $(u)_+ = \max\{u, 0\}$. RAVE attains this lower bound. □

Semantic diversity. Assume that the token representations $\boldsymbol{z}_i$ are nonzero. Starting from $\mathcal{S}^{(0)} = \mathcal{S}$, the exact farthest-point selection in the main text is

$$\begin{aligned} j_r^\star &\in \operatorname*{arg\,max}_{j\in\mathcal{H}\setminus\mathcal{S}^{(r-1)}} \min_{i\in\mathcal{S}^{(r-1)}} d(\boldsymbol{z}_j, \boldsymbol{z}_i), \\ \mathcal{S}^{(r)} &= \mathcal{S}^{(r-1)} \cup \{j_r^\star\}, \end{aligned} \tag{9}$$

where $d(\boldsymbol{z}_i, \boldsymbol{z}_j) = 1 - \cos(\boldsymbol{z}_i, \boldsymbol{z}_j)$. For the analysis, define

$$\rho(i,j) = \left\| \frac{\boldsymbol{z}_i}{\|\boldsymbol{z}_i\|_2} - \frac{\boldsymbol{z}_j}{\|\boldsymbol{z}_j\|_2} \right\|_2 = \sqrt{2d(\boldsymbol{z}_i, \boldsymbol{z}_j)}. \tag{10}$$

The selection order is identical under d and ρ with the same tie-breaking. The allocation scores remain those computed using d in the main paper. Define

$$\begin{aligned} R_K &= \max_{j\in\mathcal{H}} \min_{i\in\mathcal{S}^{(K)}} \rho(j,i), \\ R_K^\star &= \min_{\substack{\mathcal{A}\subseteq\mathcal{H} \\ |\mathcal{A}|\leq K}} \max_{j\in\mathcal{H}} \min_{i\in\mathcal{S}\cup\mathcal{A}} \rho(j,i). \end{aligned} \tag{11}$$

The optimum retains the same initial set $\mathcal{S}$ and adds at most K historical tokens.

Attention reallocation. Let $\delta_j \geq 0$ be the diversity scores used for reallocation, with $\sum_{j\in\mathcal{H}} \delta_j > 0$, and let $\eta_j = 1 - a_j$. As in the main text,

$$c_j = \gamma\eta_j + \delta_j, \qquad \gamma = \frac{\mathbb{E}_{j\in\mathcal{H}}[\delta_j]}{\mathbb{E}_{j\in\mathcal{H}}[\eta_j]}. \tag{12}$$

Here the expectations are uniform averages over $\mathcal{H}$. Write the sequence-length normalization as $\bar{c}_j = \lambda_j c_j$, where $0 < \lambda_{\min} \leq \lambda_j \leq \lambda_{\max} < \infty$, and let $\kappa = \lambda_{\max}/\lambda_{\min}$.

Theorem 1 (Semantic coverage and attention gain). *Under the above assumptions, every executed prefix $\mathcal{S}^{(K)}$ with $1 \leq K \leq |\mathcal{H}|$ satisfies*

$$R_K \leq 2R_K^\star, \qquad a'_j - a_j > \frac{B_{\mathrm{rec}}\tau}{2\kappa|\mathcal{H}|}, \quad j \in \mathcal{H}. \tag{13}$$

Proof. Semantic coverage. Set $R_0 = \max_{j\in\mathcal{H}} \min_{i\in\mathcal{S}} \rho(j,i)$. The sets $\mathcal{S}^{(r)}$ are nested, so R_r is nonincreasing. Following the farthest-first argument (Gonzalez, 1985), suppose $R_K > 2R_K^\star$ and select one more historical token $j_{K+1}^\star$. This is possible because $R_K > 0$. For every $1 \leq r \leq K+1$,

$$\min_{i\in\mathcal{S}^{(r-1)}} \rho(j_r^\star, i) = R_{r-1} \geq R_K. \tag{14}$$

Thus, the $K+1$ selected tokens are pairwise at distance at least R_K, and each is at distance at least R_K from $\mathcal{S}$. None can be covered by $\mathcal{S}$ within radius $R_K^\star$. An optimal solution must therefore cover them using at most K historical centers, so two share a center. Their distance is at most $2R_K^\star$ by the triangle inequality, contradicting their separation.

Attention gain. The definition of γ gives $\gamma \sum_{j\in\mathcal{H}} \eta_j = \sum_{j\in\mathcal{H}} \delta_j$, hence $\sum_{j\in\mathcal{H}} c_j = 2\gamma \sum_{j\in\mathcal{H}} \eta_j$; moreover, $\gamma > 0$ because $\sum_{j\in\mathcal{H}} \delta_j > 0$. Since $B_{\text{rec}} > 0$, some $i \in \mathcal{S}$ satisfies $a_i > \tau$. For each $j \in \mathcal{H}$, normalization and disjointness imply $\eta_j = 1 - a_j \geq a_i > \tau$. Also, $\sum_{j\in\mathcal{H}} \eta_j \leq |\mathcal{H}|$. Using $c_j \geq \gamma\eta_j$ (as $\delta_j \geq 0$) and the bounds on λ_j, we obtain

$$\frac{\bar{c}_j}{\sum_{k\in\mathcal{H}} \bar{c}_k} \geq \frac{\lambda_{\min}\gamma\eta_j}{2\gamma\lambda_{\max}\sum_{k\in\mathcal{H}} \eta_k} = \frac{\eta_j}{2\kappa\sum_{k\in\mathcal{H}} \eta_k} > \frac{\tau}{2\kappa|\mathcal{H}|}. \tag{15}$$

By the reallocation rule, $a'_j - a_j = B_{\text{rec}}\, \bar{c}_j / \sum_{k\in\mathcal{H}} \bar{c}_k$, so multiplying equation 15 by B_{rec} proves the gain bound. □

Implication. The above results formalize two properties of RAVE. First, attention recycling redistributes only the excess latest-call attention, preserving the total visual attention mass while introducing the minimum ℓ_1 change required by the cutoff. Second, the farthest-point selection provides a constant-factor semantic cover of the historical visual tokens, while the balanced allocation gives every historical token a lower-bounded attention increase. Together, RAVE recycles the excess attention to historical visual evidence without increasing the total visual attention budget, and selects a semantically diverse subset of this evidence.

Algorithm 1 Detailed Steps of RAVE

Input: Visual attention $\mathbf{a}$ at a decoding step after the latest visual context $\mathcal{V}_t$ ($t \geq 1$); historical visual tokens $\mathcal{H} = \bigcup_{m<t} \mathcal{V}_m$ (initial image and prior tool outputs); hidden representations $\{\mathbf{z}_j\}_{j\in\mathcal{V}_t\cup\mathcal{H}}$; cosine distance $d(\cdot,\cdot) = 1 - \cos(\cdot,\cdot)$.
Output: Rebalanced visual attention $\mathbf{a}'$.

$\mathbf{a}' \leftarrow \mathbf{a}$
▷ Recycle excessive attention from the latest visual evidence
Sort $\{a_i\}_{i\in\mathcal{V}_t}$ as $a_{(1)} \geq \cdots \geq a_{(|\mathcal{V}_t|)}$ and fit a smooth curve $\tilde{a}$
$k^\star \leftarrow$ maximum-curvature point of $\tilde{a}$; $\quad \tau \leftarrow a_{(k^\star)}$
$\mathcal{S} \leftarrow \{i \in \mathcal{V}_t \mid a_i > \tau\}$; $\quad B_{\text{rec}} \leftarrow \sum_{i\in\mathcal{S}} (a_i - \tau)$
if $B_{\text{rec}} = 0$ **then**
  return $\mathbf{a}'$ ▷ no excess attention to recycle
end if
$a'_i \leftarrow \tau,\ \forall i \in \mathcal{S}$ ▷ truncate at τ, keeping salient evidence
▷Recycle excessive attention from the latest visual evidence
$\mathcal{S}^{(0)} \leftarrow \mathcal{S}$
for $r = 1, \ldots, |\mathcal{H}|$ **do**
  $j_r^\star \leftarrow \arg\max_{j\in\mathcal{H}\setminus\mathcal{S}^{(r-1)}} \min_{i\in\mathcal{S}^{(r-1)}} d(\mathbf{z}_j, \mathbf{z}_i)$; $\quad \delta_{j_r^\star} \leftarrow \min_{i\in\mathcal{S}^{(r-1)}} d(\mathbf{z}_{j_r^\star}, \mathbf{z}_i)$
  $\mathcal{S}^{(r)} \leftarrow \mathcal{S}^{(r-1)} \cup \{j_r^\star\}$
end for
▷ Reallocate the recycled budget over historical visual evidence
$\eta_j \leftarrow 1 - a_j$, $\quad \gamma \leftarrow \dfrac{\mathbb{E}_{j\in\mathcal{H}}[\delta_j]}{\mathbb{E}_{j\in\mathcal{H}}[\eta_j]}$ ▷ attention sufficiency and adaptive scaling
$c_j \leftarrow \gamma\eta_j + \delta_j$; $\quad \bar{c}_j \leftarrow c_j / |\mathcal{V}_{m(j)}|$ ▷ $m(j)$: tool observation m that contains token j
$a'_j \leftarrow a_j + B_{\text{rec}} \dfrac{\bar{c}_j}{\sum_{i\in\mathcal{H}} \bar{c}_i},\ \forall j \in \mathcal{H}$ ▷ total visual attention mass unchanged
return $\mathbf{a}'$

F More Experimental Settings

F.1 Benchmarks and Metrics

HallusionBench (Guan et al., 2024). HallusionBench evaluates both language hallucination and visual illusion using paired image-context controls. It contains 346 images, 1,129 questions, and 455 control pairs covering visual-dependent and visual-supplement questions. After excluding samples

without available visual inputs, 951 examples remain, from which we randomly select a fixed subset $\mathcal{V}$ of 450 examples with category–subcategory proportional sampling and a fixed seed. The same subset is used across all models, methods, and call-depth analyses. For each $(I, q) \in \mathcal{V}$, evaluator outputs are mapped to binary correctness $b_M(I, q) \in \{0, 1\}$, and we report All Accuracy (aAcc):

$$\mathrm{aAcc}(M) = \frac{1}{|\mathcal{V}|} \sum_{(I,q)\in\mathcal{V}} b_M(I, q). \tag{16}$$

POPE (Li et al., 2023). POPE evaluates object hallucination using yes/no questions of the form "Is there a [object] in the image?" Each split contains 3,000 questions from 500 MSCOCO validation images. Positive questions target annotated objects, while negative objects are selected using Random, Popular, or Adversarial strategies of increasing difficulty. Following previous works (Yang et al., 2025; Lyu et al., 2026), we randomly sample 500 images from the MSCOCO (Lin et al., 2014) validation set to conduct experiments. We report Accuracy and F1 on all three splits.

MMHal-Bench (Sun et al., 2024). MMHal-Bench contains 96 image-question pairs from OpenImages, covering 12 object topics and eight question types. Following the official protocol, a judge scores each response from 0 to 6 based on factual grounding and informativeness, with scores below 3 indicating hallucination. Each model and method is evaluated over three runs. For each run with $N = 96$ responses, we report:

$$\mathrm{Score} = \frac{1}{N}\sum_{i=1}^{N} s_i, \qquad \mathrm{HallRate} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}[s_i < 3]. \tag{17}$$

CHAIR (Rohrbach et al., 2018a). CHAIR evaluates object hallucination in free-form image descriptions by matching generated object mentions against MSCOCO captions and segmentation annotations, with synonym mapping for the 80 object classes. We report instance-level CHAIR_I, the fraction of object mentions that are hallucinated, and sentence-level CHAIR_S, the fraction of responses containing at least one hallucinated object:

$$\mathrm{CHAIR}_I = \frac{|H_{\mathrm{obj}}|}{|M_{\mathrm{obj}}|}, \qquad \mathrm{CHAIR}_S = \frac{|H_{\mathrm{sent}}|}{|N_{\mathrm{sent}}|}. \tag{18}$$

AMBER (Wang et al., 2023). AMBER is an LLM-free hallucination benchmark with 1,004 densely annotated images spanning 337 object categories and covering existence, attribute, and relation hallucinations. We evaluate its discriminative setting on a fixed subset of 5,000 examples.

HR-4K/HR-8K (Wang et al., 2025a). HR-Bench evaluates whether MLLMs can exploit native high-resolution inputs. It contains 200 image-question samples covering fine-grained single- and cross-instance perception. HR-8K uses the original 8K images, while HR-4K uses object-centered crops of the same samples. Under the official cyclic-permutation protocol, each split yields 800 evaluation instances. Its detail-sensitive questions are particularly suited to evaluating visual agents that use cropping or zooming to recover local evidence.

MMStar (Chen et al., 2024b). MMStar contains 1,500 human-curated multiple-choice questions spanning six core capabilities and 18 evaluation axes. Its examples are designed to require visual information and reduce text-only shortcuts and training-data leakage, emphasizing genuine multimodal reasoning.

V*Bench (Wu & Xie, 2024). V*Bench contains 191 high-resolution, visually crowded examples covering attribute and relative-position queries. Each question requires locating a small target object before reasoning about its properties or spatial relations, making it well suited to evaluating iterative visual inspection.

OCRBench (Liu et al., 2024b). OCRBench combines 29 datasets covering text recognition, scene- and document-based VQA, key information extraction, and handwritten mathematical expression recognition, testing both text perception and reasoning over recognized content.

ChartQA (Masry et al., 2022). ChartQA evaluates visual and logical reasoning over charts with 9.6K human-authored and 23.1K automatically generated questions. It requires identifying relevant chart elements and performing operations such as comparison and aggregation.

RealWorldQA (AI, 2024). RealWorldQA contains 765 images from everyday scenes, each paired with a question and verifiable answer. It evaluates spatial and physical understanding through tasks involving relative position, distance, counting, object size, and traffic scenes.

MME-RealWorld-Lite (Zhang et al., 2025). MME-RealWorld evaluates challenging high-resolution scenes from practical applications, with 13,366 images and 29,429 human-annotated question-answer pairs across 43 subtasks and five scenarios. We use the official Lite subset to retain its focus on fine-grained perception and real-world reasoning at lower evaluation cost.

MMEvalPro (Huang et al., 2025). MMEvalPro contains 2,138 groups of three questions, totaling 6,414 questions. Each original multiple-choice question is paired with a perception question and a knowledge anchor, enabling diagnosis of cases where a model answers correctly without the required visual evidence or prerequisite knowledge.

MathVista (Lu et al., 2024). MathVista evaluates mathematical reasoning in visual contexts using 6,141 examples from 31 datasets. It spans diagrams, charts, plots, tables, and scientific figures, requiring models to combine visual understanding with arithmetic, algebraic, geometric, logical, and statistical reasoning.

F.2 Think-with-Images Models

Pixel-Reasoner-7B (Su et al., 2026). Pixel-Reasoner-7B is a vision-language model that integrates textual reasoning with visual operations. While reasoning in text, it can selectively zoom in on images to gather additional visual evidence, enabling more informed and accurate answer generation.

DeepEyesV2-7B (Hong et al., 2026). DeepEyesV2-7B is an agentic multimodal model that integrates image manipulation, code execution, and search into its reasoning process. It dynamically selects tools based on the task complexity and can compose multiple operations, with each resulting observation informing subsequent reasoning and tool selection.

PyVision-Image-7B (Zhao et al., 2026). PyVision-Image-7B is an open vision agent that writes and executes Python programs to inspect and manipulate images during reasoning. Instead of relying on a fixed set of predefined visual operations, it can dynamically construct task-specific operations and iteratively gather visual evidence over multiple rounds.

Qwen3-VL series (Bai et al., 2025). Qwen3-VL series (2B/4B/30B) are general-purpose vision-language models designed for understanding images, videos, and text, with strong capabilities in visual perception, multimodal reasoning, and agentic interaction.

F.3 Baseline Methods

OPERA (Huang et al., 2024). OPERA mitigates hallucination by penalizing beam-search logits when attention over-focuses on summary tokens, while its retrospection mechanism rolls decoding back and reselects beam candidates without additional training or external knowledge.

VAR (Kang et al., 2025b). VAR mitigates visual attention sinks by redirecting attention from visually irrelevant sink tokens to non-sink visual tokens in selected image-centric heads, without additional training, models, or inference steps.

MemVR (Zou et al., 2024). MemVR mitigates hallucination by detecting decoder uncertainty and reinjecting visual tokens as key–value memory through the feed-forward network, restoring visual evidence without backbone retraining or additional inference passes.

AGE (Ren et al., 2026). AGE mitigates hallucination by using attention differences between truthful and hallucinated tokens to guide layer-wise interventions, enhancing visual or textual attention as needed without updating model parameters.

LEAD (Xu et al., 2026b). LEAD adaptively switches between latent and discrete decoding based on token entropy, using probability-weighted continuous embeddings and a pretrained visual anchor under high uncertainty before reverting to discrete decoding as entropy decreases.

F.4 Experimental Details

Table 8: Model decoding and inference configurations. Output and Context denote the maximum numbers of output and context tokens. Pixels denotes the maximum image pixel budget.

	Decoding			Inference configuration				
Backbone	**Temp.**	**Top-p**	**Top-k**	**Dtype**	**Output**	**Context**	**Pixels**	**Trajectory limit**
Pixel-Reasoner-7B	0.1	1.0	0 (off)	BF16	2,048	18,192	4,014,080	10 tool calls; 16 images
DeepEyesV2-7B	0.1	1.0	0 (off)	FP16	20,480	128,000	12,845,056	10 agent attempts
PyVision-Image-7B	0.1	1.0	20	FP16	16,000	128,000	12,845,056	50 images; 10 Python calls
Qwen3-VL	0.1	0.8	20	FP16	20,480	128,000	12,845,056	10 agent attempts

Decoding Settings and Inference Configurations. Detailed decoding settings are provided in Table 8. Main experiments use eight NVIDIA RTX A6000 GPUs. Additionally, Table 8 reports numerical precision and model-specific limits on output length, context length, image size, and agent trajectories.

Backbones and tool interfaces. We evaluate three 7B think-with-images backbones on the fifteen benchmarks (Sec. F.1): Pixel-Reasoner-7B (Su et al., 2026), DeepEyesV2-7B (Hong et al., 2026), and PyVision-Image-7B (Zhao et al., 2026). They instantiate two complementary tool-use paradigms: Pixel-Reasoner-7B calls `crop_image_normalized` (and `select_frames` for video inputs) via structured function calling, whereas DeepEyesV2-7B and PyVision-Image-7B generate executable Python code for image manipulation in their released agent environments. For each checkpoint, we preserve the original system prompt, tool definitions, and tool-selection policy. `RAVE` only modifies attention within these native agent loops, without introducing a new tool interface or external planner. Each successful image-producing call appends a new visual observation, while the original image and prior tool outputs remain in the causal context. We further evaluate instruction-tuned Qwen3-VL models (Bai et al., 2025) at 2B, 4B, and 30B scales under the same bounded agent setup; results are reported in Table 12. Full system prompts and tool instructions are provided in Fig. 16–19.

G Additional Experimental Results and Analyses

G.1 Further Ablation Study of Each Component

In the main paper, we ablate the three components of `RAVE` on HallusionBench and MMHal-Bench (Table 3). Here, we extend the analysis to HR-8K and MMStar to examine whether their contributions remain consistent on perception-oriented benchmarks. We use Pixel-Reasoner-7B (Su et al., 2026) as the backbone. Recall that `RAVE` comprises three components: maximum curvature for identifying over-attended tokens, attention recycling for redistributing their excess attention, and semantic diversity for redirecting it toward complementary historical evidence. From Table 9, we observe that our three design choices consistently outperform their respective alternatives on the newly evaluated HR-8K and MMStar benchmarks. In particular, maximum curvature improves over the reference baseline by 4.18 and 1.89 points, attention recycling by 6.06 and 0.99 points, and semantic diversity by 2.06 and 1.20 points on HR-8K and MMStar, respectively. These results demonstrate the consistent effectiveness of each component on perception-oriented benchmarks.

Table 9: Component-wise comparison of RAVE on four benchmarks.

Method	HallusionBench aAcc↑	MMHal-Bench Score↑	HR-8K Acc.↑	MMStar Acc.↑
Identify Over-Attended Tokens:				
SparseVLM (Zhang et al., 2024)	68.00 (+0.00)	2.88 (+0.00)	66.63 (+0.00)	62.61 (+0.00)
FastV (Chen et al., 2024a)	68.67 (+0.67)	3.09 (+0.21)	67.63 (+1.00)	61.33 (-1.28)
Maximum Curvature (Ours)	**69.33** (+1.33)	**3.91** (+1.03)	**70.81** (+4.18)	**64.53** (+1.92)
Handle Excessive Attention:				
TAF (Ouyang et al., 2026)	65.56 (+0.00)	2.78 (+0.00)	64.75 (+0.00)	63.51 (+0.00)
OPERA (Huang et al., 2024)	67.33 (+1.77)	2.95 (+0.17)	65.47 (+0.72)	62.17 (-1.34)
Attention Recycling (Ours)	**69.33** (+3.77)	**3.91** (+1.13)	**70.81** (+6.06)	**64.53** (+1.02)
Guide Attention Reallocation:				
Initial Image	67.11 (+0.00)	3.07 (+0.00)	68.75 (+0.00)	63.30 (+0.00)
PAI (Liu et al., 2024a)	68.67 (+1.56)	3.16 (+0.09)	66.63 (-2.12)	62.42 (-0.88)
Semantic Diversity (Ours)	**69.33** (+2.22)	**3.91** (+0.84)	**70.81** (+2.06)	**64.53** (+1.23)

G.2 Intervention Layer and Attention Heads

Intervention Layer. In Fig. 8 of the main paper, we examine the effect of the intervention layer on HallusionBench (Guan et al., 2024). Here, we provide results on additional benchmarks, including MMHal-Bench (Sun et al., 2024), HR-8K (Wang et al., 2025a), and MathVista (Lu et al., 2024), covering hallucination, high-resolution perception, and multimodal reasoning. As shown in Fig. 15, the best performance is consistently observed in the middle layers across all three backbones. Layers 12–16 achieve better accuracy while maintaining a low hallucination rate. In contrast, interventions at shallower layers offer only limited improvements, while those at deeper layers tend to yield diminishing returns or even degrade performance. These results suggest focusing intervention on the middle layers, which consistently perform well across benchmarks and backbones.

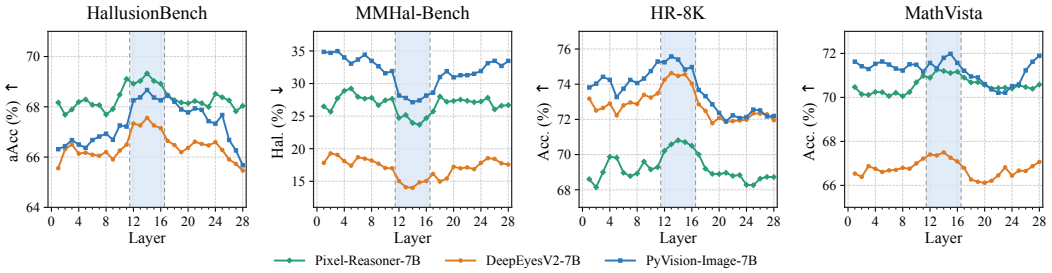


Figure 15: Layer-wise performance of RAVE across three 7B backbones. We vary the intervention layer across four benchmarks. It can be observed that the middle layers consistently yield competitive performance across backbones.

Attention Heads. In this paper, we use head-averaged attention to guide the intervention. To examine whether the choice of attention heads affects intervention effectiveness, we use Pixel-Reasoner-7B as the backbone and compare several head-selection strategies. Specifically, we rank attention heads according to their average attention to the latest visual observation and perform the intervention on the top 25%, 50%, and 75% of heads. For the 50% setting, we further compare interventions using the top-ranked, bottom-ranked, and randomly selected heads. As shown in Table 10, the performance remains relatively stable across different head-selection strategies and proportions, suggesting that the effectiveness of the intervention does not critically depend on identifying a particular subset of attention

Table 10: Ablation of attention-head selection.

Head subset	HallusionBench aAcc↑	MME-RW Lite Acc.↑	MathVista Acc.↑
Top 25%	69.11	52.50	71.20
Top 50%	69.11	52.39	71.20
Top 75%	69.11	52.39	71.20
Bottom 50%	68.89	50.50	71.20
Random 50%	69.33	52.39	71.10
All heads (averaged)	**69.33**	**52.74**	**71.20**

heads. This robustness also indicates that the relevant attention signal is broadly captured by head-level aggregation rather than being highly sensitive to a specific selection strategy. We therefore adopt simple head averaging in our main experiments, avoiding the need for an additional head-selection procedure and its associated hyperparameters.

G.3 Latency Analyses

Here we analyze the computational overhead of RAVE against two representative training-free methods, VAR (Kang et al., 2025b) and OPERA (Huang et al., 2024). As shown in Table 11, RAVE achieves the highest aAcc across all three backbones while consistently incurring lower latency and memory overhead than VAR and OPERA.

Table 11: Latency and memory comparison across three backbones on HallusionBench. All results are measured on a single RTX A6000 with batch size 1.

Model	Method	aAcc↑	Latency/sample (s)↓	Latency vs. Base (%)↓	Avg memory (GB)↓
Pixel-Reasoner-7B (Su et al., 2026)	Base	67.11	4.81	100.00	23.89
	OPERA	68.22	10.51	218.50	45.07
	VAR	67.56	9.71	201.90	37.07
	RAVE (Ours)	**69.33**	**4.93**	**102.49**	**25.76**
DeepEyesV2-7B (Hong et al., 2026)	Base	63.33	13.91	100.00	20.85
	OPERA	63.78	23.24	167.05	45.13
	VAR	64.22	22.65	162.85	26.54
	RAVE (Ours)	**67.56**	**19.01**	**136.70**	**20.86**
PyVision-Image-7B (Zhao et al., 2026)	Base	66.44	31.09	100.00	37.24
	OPERA	65.11	76.76	246.88	47.37
	VAR	66.00	62.83	202.10	41.38
	RAVE (Ours)	**68.67**	**41.72**	**134.17**	**39.52**

G.4 Evaluations on Qwen-3-VL-series

Table 12: Evaluation of RAVE across different Qwen3-VL model scales, demonstrating consistent improvements from 2B to 30B parameters.

Model	Method	HallusionBench	MMHal-Bench		HR-8K	MathVista
		aAcc↑	Hal.↓	Score↑	Acc.↑	Acc.↑
Qwen3-VL-2B	Base	64.22	7.29	3.14	37.38	58.90
	+ RAVE (Ours)	**65.78**	**6.25**	**3.15**	**39.50**	**60.60**
Qwen3-VL-4B	Base	69.78	28.13	3.30	51.13	71.00
	+ RAVE (Ours)	**70.00**	**25.00**	**3.49**	**54.13**	**72.00**
Qwen3-VL-30B	Base	69.56	19.79	3.79	59.00	76.50
	+ RAVE (Ours)	**72.00**	**17.71**	**3.90**	**62.25**	**76.70**

We evaluate the scalability of RAVE on Qwen3-VL models ranging from 2B to 30B parameters (Bai et al., 2025), including the 2B, 4B, and 30B variants. We use the same evaluation settings across scales, adjusting only the intervention layer. As shown in Table 12, RAVE consistently mitigates hallucinations while improving general perception and reasoning performance across all model scales. Specifically, on Qwen3-VL-2B, RAVE improves MathVista accuracy by up to 1.70 points; on Qwen3-VL-4B, it reduces the MMHal-Bench hallucination rate by 3.13 points; and on Qwen3-VL-30B, it improves HallusionBench aAcc and HR-8K accuracy by 2.44 and 3.25 points, respectively. These results demonstrate that RAVE can be effectively integrated into the Qwen3-VL series and remains robust across different model scales.

H Additional Qualitative Examples

Figure 23 – 24 present additional examples demonstrating that RAVE effectively mitigates hallucinations produced by the base models, including PyVision-Image-7B and DeepEyesV2-7B. Figure 25

illustrates a failure case of RAVE on DeepEyesV2-7B. Answering this question accurately requires fine-grained perception of the subtle visual evidence around the driver, as well as reasoning about the driver's current posture and leveraging relevant commonsense knowledge, making the example particularly challenging.

I LIMITATIONS AND FUTURE WORK

Limitation. RAVE requires access to the model's internal attention maps to quantify and mitigate *visual recency bias*. Consequently, it cannot be directly applied to proprietary models that do not expose their internal attention states (*e.g.,* GPT-series models). **Future Work.** A natural direction for future work is to incorporate *visual recency bias* directly into model training. In particular, the bias signals introduced in this work can be formulated as process-level rewards or penalties, discouraging excessive reliance on recent tool observations while encouraging the consistent integration of relevant visual evidence across tool calls. This training paradigm enables models to learn more balanced visual evidence aggregation intrinsically, thereby reducing their susceptibility to *visual recency bias* and mitigating hallucinations.

SYSTEM PROMPT

Pixel-Reasoner-7B

You are a helpful assistant.

You may call one or more functions to assist with the user query.

You are provided with function signatures within <tools></tools> XML tags:
<tools> {"type": "function", "function": {"name": "crop_image_normalized", "description": "Zoom in on the image based on the bounding box coordinates. It is useful when the object or text in the image is too small to be seen.", "parameters": {"type": "object", "properties": {"bbox_2d": {"type": "array", "description": "coordinates for bounding box of the area you want to zoom in. Values should be within [0.0,1.0].", "items": {"type": "number"}}, "target_image": {"type": "number", "description": "The index of the image to crop. Index from 1 to the number of images. Choose 1 to operate on original image."}}, "required": ["bbox_2d", "target_image"]}}}
{"type": "function", "function": {"name": "select_frames", "description": "Select frames from a video.", "parameters": {"type": "object", "properties": {"target_frames": {"type": "array", "description": "List of frame indices to select from the video (no more than 8 frames in total).", "items": {"type": "integer", "description": "Frame index from 1 to 16."}}}, "required": ["target_frames"]}}}
</tools>

For each function call, return a json object with function name and arguments within <tool_call></tool_call> XML tags:
<tool_call>
{"name": <function-name>, "arguments": <args-json-object>}
</tool_call>

Guidelines: Understand the given visual information and the user query. Determine if it is beneficial to employ the given visual operations (tools). For a video, we can look closer by `select_frames`. For an image, we can look closer by `crop_image`. Reason with the visual information step by step, and put your final answer within \boxed{}.
```

Figure 16: System prompt for Pixel-Reasoner-7B.
```

SYSTEM PROMPT

DeepEyesV2-7B

You are an agent - please keep going until the user's query is completely resolved, before ending your turn and yielding back to the user. Only terminate your turn when you are sure that the problem is solved.

Solve the following problem step by step. In your reasoning process, if the answer cannot be determined, you can write Python code in a Jupyter Notebook to process the image and extract more information from it. The stdout and stderr content, along with the images generated by `plt.show()` will be returned to better assist with the user query.
You MUST use the python tool to analyze or transform images whenever it could improve your understanding. This includes but is not limited to zooming in, rotating, adjusting contrast, computing statistics, or isolating features.
You MUST plan extensively before each function call, and reflect extensively on the outcomes of the previous function calls. DO NOT do this entire process by making function calls only, as this can impair your ability to solve the problem and think insightfully.
For all the provided images, in order, the i-th image has already been read into the global variable "image_i" using the "PIL.Image.open()" function. For example, the first image can be accessed as "image_1". When writing Python code, you can directly use these variables without needing to read them again.

Your python code should be enclosed within <code> </code> tag.
Example for calling Python code in Jupyter Notebook:
<code>
```python
\# python code here
```
</code>

Note:
1. python can be called to analyze the image. python will respond with the output of the execution or time out after 300.0 seconds.
2. Like jupyter notebook, you can use Python code to process the input image and use `plt.show()` to visualize processed images in your code.
3. All python code are running in the same jupyter notebook kernel, which means the functions and variables are automatically stored after code execution.
4. You program should always returns in finite time. Do not write infinite loop in your code.
5. Writing file to disk is not allowed.
````

Figure 17: System prompt for DeepEyesV2-7B.
````

SYSTEM PROMPT **PyVision-Image-7B**

You are a helpful assistant.

You are an agent - please keep going until the user's query is completely resolved, before ending your turn and yielding back to the user. Only terminate your turn when you are sure that the problem is solved. Solve the following problem step by step. You now have the ability to selectively write executable Python code to enhance your reasoning process. The Python code will be executed by an external sandbox. You MUST plan extensively before each function call, and reflect extensively on the outcomes of the previous function calls. DO NOT do this entire process by making function calls only, as this can impair your ability to solve the problem and think insightfully.

For all the provided images, in order, the i-th image has already been read into the global variable `image_clue_i` using the PIL.Image.open() function. When writing Python code, you can directly use these variables without needing to read them again. Since you are dealing with the vision-related question answering task, you MUST use the python tool (e.g., matplotlib library) to analyze or transform images whenever it could improve your understanding or aid your reasoning. This includes but is not limited to zooming in, rotating, adjusting contrast, computing statistics, or isolating features. Note that when you use matplotlib to visualize data or further process images, you need to use plt.show() to display these images; there is no need to save them. Do not use image processing libraries like cv2 or PIL. If you want to check the value of a variable, you MUST use print() to check it.

The output (wrapped in `<interpreter>output_str</interpreter>`) can be returned to aid your reasoning and help you arrive at the final answer. The Python code should be complete scripts, including necessary imports.
Each code snippet is wrapped with `<code>
```python
code snippet
```
</code>`.
The last part of your response should be in the following format:
<answer>
\boxed{{'The final answer goes here.'}}
</answer>

Image Width: {width}; Image Height: {height}

User Question:
{query}
Remember to place the final answer in the last part using the format:
<answer>
\boxed{{'The final answer goes here.'}}
</answer>
````

Figure 18: System prompt for PyVision-Image-7B.
````

SYSTEM PROMPT **Qwen3-VL**

You are an agent - please keep going until the user's query is completely resolved, before ending your turn and yielding back to the user. Only terminate your turn when you are sure that the problem is solved.

Solve the following problem step by step. In your reasoning process, if the answer cannot be determined, you can write Python code in a Jupyter Notebook to process the image and extract more information from it. The stdout and stderr content, along with the images generated by `plt.show()` will be returned to better assist with the user query. You MUST use the python tool to analyze or transform images whenever it could improve your understanding. This includes but is not limited to zooming in, rotating, adjusting contrast, computing statistics, or isolating features. You MUST plan extensively before each function call, and reflect extensively on the outcomes of the previous function calls. DO NOT do this entire process by making function calls only, as this can impair your ability to solve the problem and think insightfully. For all the provided images, in order, the i-th image has already been read into the global variable "image_i" using the "PIL.Image.open()" function. For example, the first image can be accessed as "image_1". When writing Python code, you can directly use these variables without needing to read them again.

Your python code should be enclosed within <code> </code> tag.
Example for calling Python code in Jupyter Notebook:
<code>
```python
# python code here
```
</code>
Note:
1. python can be called to analyze the image. python will respond with the output of the execution or time out after 300.0 seconds.
2. Like jupyter notebook, you can use Python code to process the input image and use `plt.show()` to visualize processed images in your code.
3. All python code are running in the same jupyter notebook kernel, which means the functions and variables are automatically stored after code execution.
4. You program should always returns in finite time. Do not write infinite loop in your code.
5. Writing file to disk is not allowed.

{question}
Reason step by step in <think> </think> tags. Use Python in <code> </code> tags to inspect the image when useful. After considering all evidence, finish with exactly one separate line 'Final answer: <your concise final answer>'. For a yes/no question, write 'Final answer: Yes' only if the proposition asked in the question is supported by the visual evidence; otherwise write 'Final answer: No'. The final answer must agree with the visual reasoning.
````

Figure 19: System prompt for the Qwen3-VL series.
````

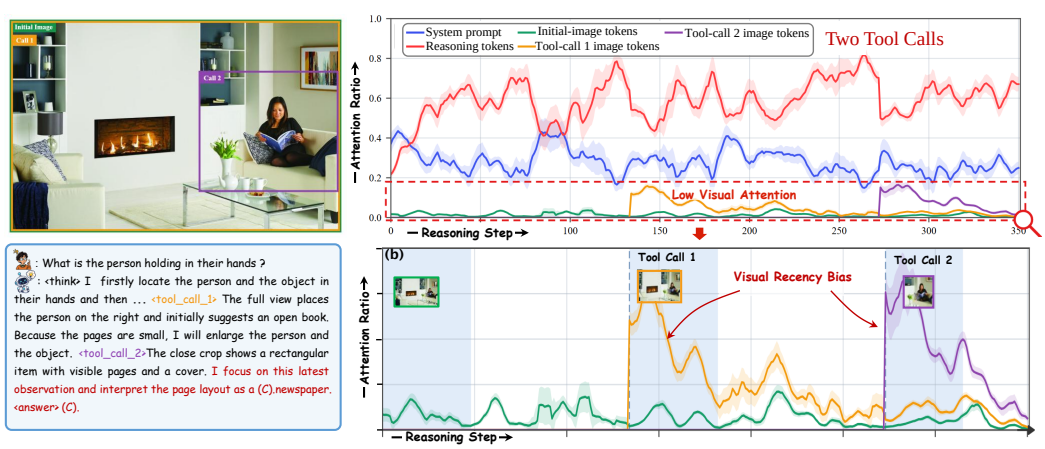


Figure 20: Qualitative illustration of a reasoning trajectory with two tool invocations. The first visual observation supports the correct interpretation of the object as a book, whereas the second crop dominates subsequent reasoning and leads to the incorrect prediction of a newspaper.

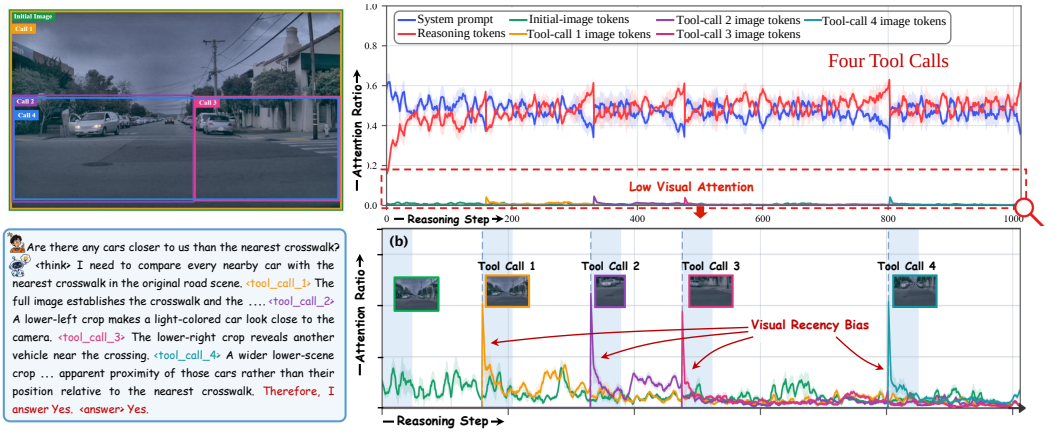


Figure 21: Qualitative illustration of a four-call reasoning trajectory for spatial reasoning. The model examines the crosswalk and nearby vehicles across successive crops, but the most recent observations dominate the comparison, leading to an incorrect final answer.

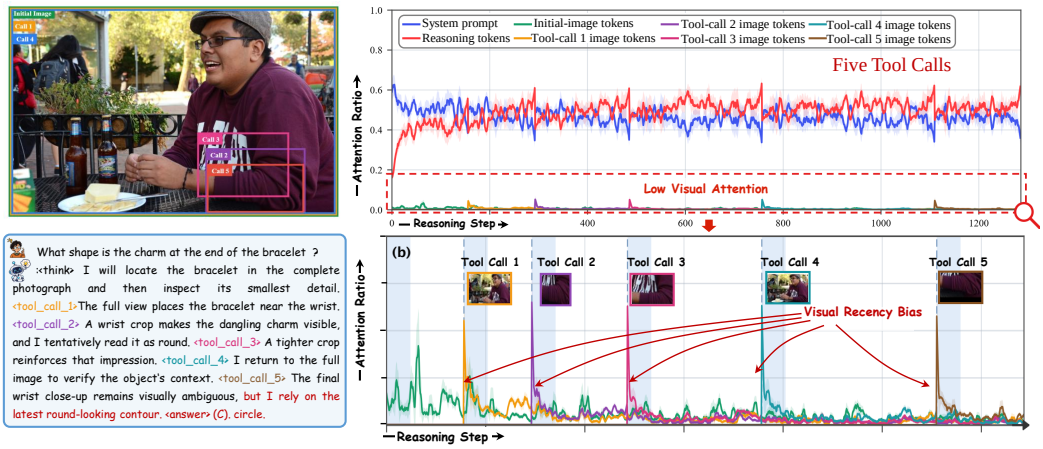
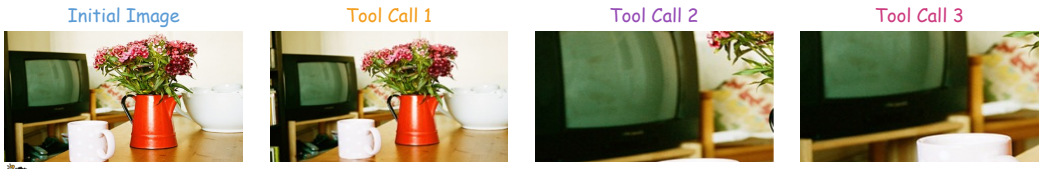
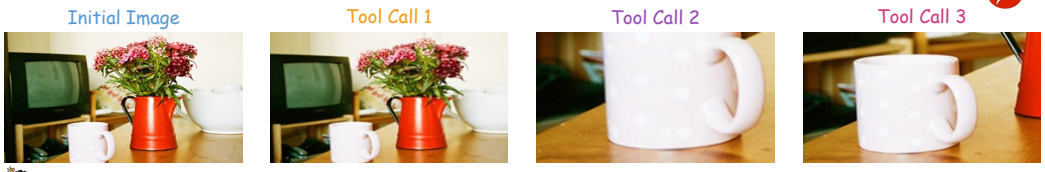


Figure 22: Qualitative illustration of a five-call reasoning trajectory for fine-grained attribute recognition. The model correctly localizes the bracelet, but further close-up observations shift attention toward the latest crop, resulting in the incorrect prediction of a circular charm.



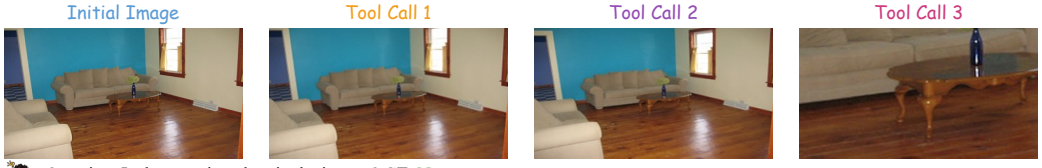
Question: Is there a remote in the image? GT: Yes.

PyVision-Image-7B: <think> Let's plan the steps: 1. Visually inspect the image for a remote control. 2. If needed, zoom in or enhance the relevant areas to confirm the presence of a remote ... <tool_call_1> Upon visual inspection, I do not see any clear evidence of a remote control in the image. However, to be certain ... <tool_call_2> After zooming in, I can still not see any clear remote control in the image. The objects visible are a TV, mugs, but no remote control is present. To be thorough, I'll zoom in ... <tool_call_3> After zooming in, it is still not possible to see any remote control in the image. The objects visible are a TV, mugs, a flower pot, and some shelves, but no remote control is present. Let me reflect: In a typical setting, a remote control would be visible near the ... <answer>NO</answer>



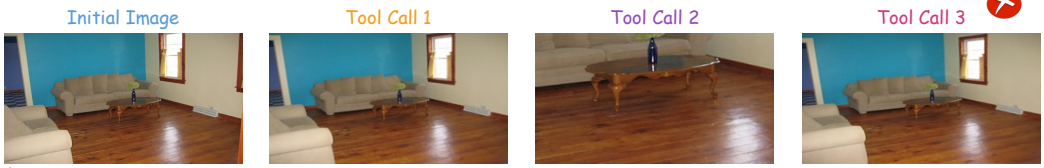
Question: Is there a remote in the image? GT: Yes.

PyVision-Image-7B + Ours: <think> Let's plan the steps: 1. Visually inspect the image to see if there is a remote control visible. 2. If needed, zoom in or enhance the relevant area to confirm ... <tool_call_1> Upon visual inspection, there appears to be a green object near the bottom left of the TV, which could be a remote control. To be sure, let's zoom in on that area to check for a ... <tool_call_2> Upon zooming in, I can see a green object near the bottom left of the TV, which appears to be a remote control. Let's confirm this by zooming ... <tool_call_3> After zooming in, it is clear that there is a green object near the bottom left of the TV. This matches the typical appearance of a TV remote. Therefore, the answer to the question "Is there a remote in the image?" is yes. <answer>YES</answer>



Question: Is there a wine glass in the image ? GT: No.

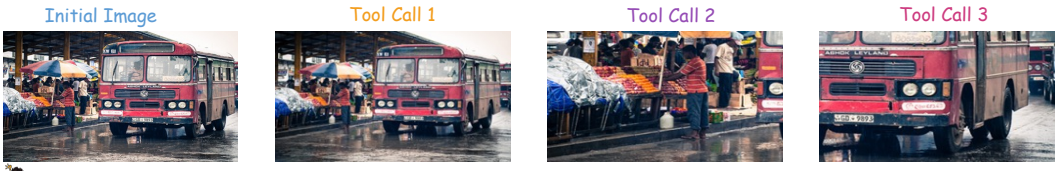
PyVision-Image-7B: <think> Let's break down the steps: 1. Visually inspect the image to see if there is a wine glass present. 2. Use matplotlib to zoom in and enhance area around ... <tool_call1> Upon initial inspection, it appears there is a blue glass object on the coffee table, which could be a wine glass. To be certain, Let's zoom ... <tool call 2> After zooming in, I can clearly see a blue glass object on the coffee table. Its shape and position suggest that it is a wine glass ... <tool_call_3> Upon the closer inspection, the blue glass object on the coffee table has the shape and appearance of a wine glass, with a narrow stem and rounded bowl. This is consistent with the typical design of a wine glass. Therefore, the answer to the question "Is there a wine <answer> YES</answer>



Question: Is there a wine glass in the image ? GT: No.

PyVision-Image-7B + Ours: <think> Let's break down the steps: 1. Visually inspect the image to check for the presence of a wine glass. 2. If needed, zoom in or enhance the image to confirm ... <tool_call_1> Upon visual inspection of the displayed image, I see a blue bottle on the coffee table, but I do not immediately see a wine glass. To be sure, let's zoom in on the coffee table area ... <tool_call_2> After zooming in, I can clearly see a blue bottle on the coffee table, but I do not see any wine glass. The bottle appears to be a wine bottle ... <tool_call_3> After zooming out, there is no visible wine glass in the room. The only glass-like object is a bottle, which is not a wine glass. Let me summarize and answer the question: - There is a blue ... <answer>NO</answer>

Figure 23: More qualitative examples.



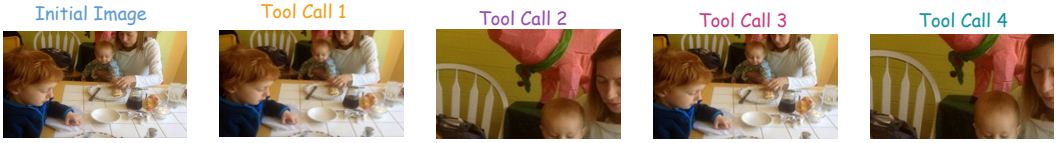
Question: Is there a handbag in the image ? GT: No.

PyVision-Image-7B: <think> Let's break down the steps to answer the question: 1. Carefully examine the image for any visible handbag. 2. Use matplotlib to zoom in or enhance the image if needed ... <tool_call_1> Upon initial inspection, it appears that there is a handbag visible on the left side of the image. To be sure, let's zoom in on the left area of the image to ... ... <tool_call_2> Upon zooming in, it is clear that there is a handbag visible near the vendor on the left side of the image ... <tool_call_3> After zooming in on the right area, I confirm that there are no handbags visible in the right section of the image. The left side, however, shows a handbag near the vendor. Thus, the answer is clear: there is a handbag in the image. <answer>YES</answer>



Question: Is there a handbag in the image ? GT: No.

PyVision-Image-7B + Ours: <think> Let's break down the steps: 1. Visually inspect the image to look for any handbag. 2. If needed, zoom in or enhance the image to check for any subtle details ... <tool_call_1> Upon initial inspection, I do not see any obvious handbag in the image. However, to be thorough, let's zoom in on the area around ... <tool_call_2> After zooming in, I can still see the market area and the people, but I do not observe any handbag being carried by any of the individuals. ... <tool_call_3> After zooming in, I can clearly see the people and the market area. There are no visible handbags. The only visible bags are the umbrellas, which are not handbags. Given the careful inspection, I can confidently say there is no handbag in the ... <answer>NO</answer>



Question: Is there a potted plant in the image ? GT: No.

DeepEyesV2-7B: <think> Let's plan the steps: 1. Visually inspect the image for any potted plant. 2. Zoom in or adjust the contrast ... <tool_call_1> From the displayed image, I notice a pink object in the background, which appears to be a potted plant with green leaves. To be ... ... <tool_call_2> Upon zooming in, I can clearly see a pink object with green leaves in the background, which resembles a potted plant. However, to be certain ... <tool_call_3> After reviewing the full image, I can confirm that there is a pink object with green leaves in the background, which looks like a potted plant ... <tool_call_4> This object resembles a potted plant, and it is distinct from the other items on the table. Therefore, the answer to the question ... <answer>YES</answer>



Question: Is there a potted plant in the image ? GT: No.

DeepEyesV2-7B + Ours: <think> Let's plan the steps: 1. Visually inspect the image to check for any potted plants. 2. If needed, zoom in ... <tool_call_1> Upon initial inspection, I do not see any potted plant in the image. However, to be thorough, let's zoom in on the background and ... <tool_call_2> After zooming in, I still do not see any potted plant in the image. The background shows a pink stuffed animal and some bags, but ... <tool_call_3> After zooming in on the left side of the image, I still do not see any potted plant. The only visible plant-like object is the pink ... ... <tool_call_4> After zooming in on the right side of the image, I still do not see any potted plant. The table has food items, a pitcher, and some containers, but no plants are visible. Given the ... <answer>NO</answer>

Figure 24: More qualitative examples.

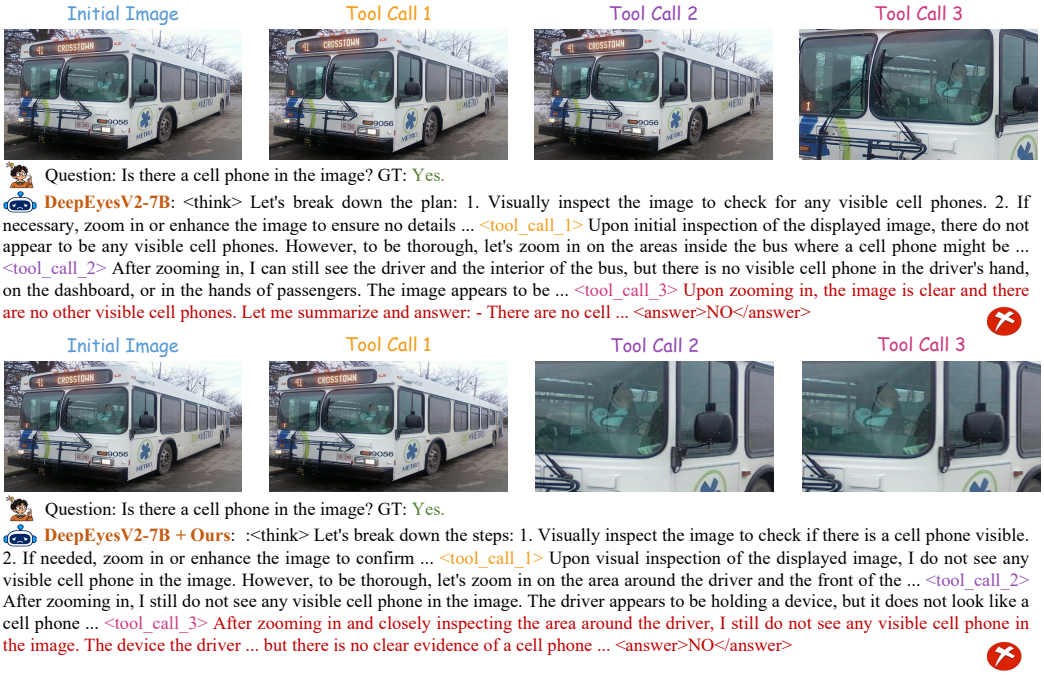


Figure 25: Failure case. Answering this question accurately requires fine-grained perception of the subtle visual evidence around the driver, as well as reasoning about the driver's current posture and leveraging relevant commonsense knowledge, making the example particularly challenging.