

v1: Learning to Point Visual Tokens for Multimodal Grounded Reasoning

Jiwan Chung^{1*} Junhyeok Kim^{1*} Siyeol Kim¹
 Jaeyoung Lee¹ Min Soo Kim¹ Youngjae Yu²

¹Yonsei University ²Seoul National University
 jiwan.chung.research@gmail.com junhyeok@yonsei.ac.kr

Abstract

When thinking with images, humans rarely rely on a single glance: they revisit visual evidence while reasoning. In contrast, most Multimodal Language Models encode an image once to key-value cache and then reason purely in text, making it hard to re-ground intermediate steps. We empirically confirm this: as reasoning chains lengthen, models progressively lose focus on relevant regions. We introduce **v1**, a lightweight extension for active visual referencing via point-and-copy: the model selects relevant image patches and copies their embeddings back into the reasoning stream. Crucially, our point-and-copy mechanism retrieves patches using their semantic representations as keys, ensuring perceptual evidence remains aligned with the reasoning space. To train this behavior, we build **v1g**, a dataset of 300K multimodal reasoning traces with interleaved grounding annotations. Across multimodal mathematical reasoning benchmarks, **v1** consistently outperforms comparable baselines. We release our code, model, and data.

1 Introduction

Multimodal mathematical problem solving often requires repeated access to diagrams: intermediate steps depend on localized cues (e.g., angles, tangency points, symmetries) that are revisited as reasoning unfolds. Cognitive studies report similar visual revisitation (Cox, 1999; Brun et al., 2016; Chu et al., 2017; Kozhevnikov et al., 2002).

Recent advances in Multimodal Large Language Models (MLLMs) (Liu et al., 2023; Bai et al., 2025; Chen et al., 2025) have shifted focus toward complex multi-step reasoning (Xu et al., 2025; Yao et al., 2024; Sun et al., 2025; Huang et al., 2025) rather than direct recognition. Mathematical reasoning (Lu et al., 2024; Zhang et al., 2024a; Wang et al., 2024a) is a standard benchmark for this capability; it requires long-horizon grounding and unambiguous solutions, making it a natural testbed for grounded multimodal inference.

However, current MLLMs process images only once at the start and, due to causal masking, thereafter reason mainly over the frozen key-value cache of visual embeddings. This limits their ability to actively revisit visual context as inference unfolds. In practice, this constraint manifests as two forms of *visual grounding decay*. First, attention to all image tokens steadily weakens as reasoning chains extend. Second, even the relative weight on relevant tokens declines, reducing the model’s ability to focus on the most informative regions (Section 3). These effects highlight the need for mechanisms that let models explicitly re-access visual information to keep reasoning grounded in the input.

We propose **v1**¹, a simple yet effective extension that equips MLLMs with a *point-and-copy* mechanism for dynamically referencing input visual tokens during multimodal reasoning

^{*}Equal contribution.

¹Code: <https://github.com/jun297/v1>. Model: <https://huggingface.co/kjunh/v1-7B>. Data (**v1g**): <https://huggingface.co/datasets/kjunh/v1g>. Project page: <https://jun297.github.io/v1/>.

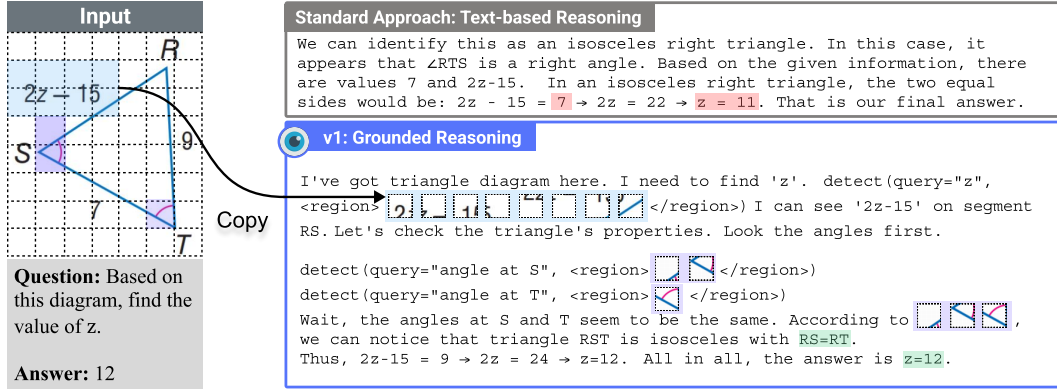


Figure 1: Pure text-based reasoning vs. v1 during inference. v1 can actively re-access visual context by point-and-copying relevant image regions throughout reasoning process.

(Figure 1). Specifically, we augment the model with an additional pointing head that outputs a probability distribution over the input image token positions, alongside the standard vocabulary logits. When an image token is selected, its embedding is copied and injected as the next-step input, allowing the model to dynamically retrieve and reuse visual information during generation.

Our approach is readily compatible with popular MLLM architectures (Liu et al., 2023; Wang et al., 2024b; Chen et al., 2025) that operate on continuous image embeddings. Unlike methods that attempt to generate new image tokens (Chameleon Team, 2024) which require full-scale pretraining for image generation capacity, our method simply reuses existing input embeddings through pointing and copying. The only additional parameters are lightweight linear heads, incurring minimal computational overhead.

To train v1, we construct v1g, a dataset of 300K multimodal mathematical reasoning paths with interleaved grounding annotations, where each reasoning step is explicitly linked to a corresponding image region. The construction pipeline comprises three stages: (1) oversampling diverse reasoning traces from an MLLM, (2) extracting visual queries and retrieval steps from the traces using an LLM-guided decomposition process, and (3) grounding each visual reference by associating it with a bounding box in the input image. The pipeline is fully automated, leveraging the generative and interpretive capabilities of LLMs to produce high-quality, grounded reasoning trajectories at scale.

We evaluate v1 on three established multimodal mathematical reasoning benchmarks: MathVista (Lu et al., 2024), MathVision (Wang et al., 2024a), and MathVerse (Zhang et al., 2024a), following prior work (Yao et al., 2024; Sun et al., 2025; Huang et al., 2025). v1 demonstrates strong performance across all benchmarks, outperforming existing models of comparable scale and approaching the capabilities of much larger models, particularly on tasks requiring precise visual grounding and iterative reference to localized regions. These results suggest that dynamic access of visual input at inference time can improve multimodal mathematical reasoning capabilities.

Our contributions are:

- **v1 model:** a lightweight MLLM extension that helps mitigate visual grounding decay through a point-and-copy mechanism, enabling dynamic visual reference.
- **v1g dataset:** a large-scale training set with 300K multimodal mathematical reasoning traces and fine-grained visual grounding.
- **Empirical findings:** extensive experiments and ablations on multimodal mathematical reasoning benchmarks, showing that dynamic visual reference and the point-and-copy design both mitigate visual grounding decay and lead to better multimodal reasoning.

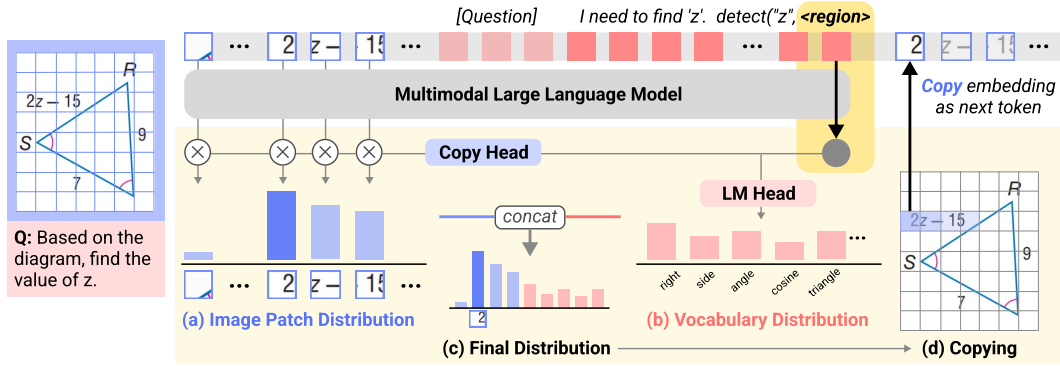


Figure 2: **Inference process of v1.** At each step, the MLLM encodes the multimodal context and generation history into token representations. For the last token (e.g., "<region>"), (a) a copy head projects its representation and computes logits against image patch embeddings, (b) a language head produces logits over the vocabulary, and (c) the two are concatenated to form the final distribution. If a patch is chosen, its embedding is copied as the next token input, enabling v1 to reference image regions one patch at a time.

2 Related Work

2.1 Reasoning in Large Language Models

Reasoning in text-only large language models. Recent advances have significantly improved reasoning in text-only LLMs. OpenAI’s o1 model (Jaech et al., 2024) had achieved state-of-the-art performance on mathematical benchmarks (Lightman et al., 2023; Cobbe et al., 2021), motivating follow-up work that induces stronger reasoning through reinforcement learning and reflective Chain-of-Thought (Guo et al., 2025), as well as inference-time scaling (Muennighoff et al., 2025). While effective in text-only settings, extending these approaches to multimodal reasoning remains challenging.

Reasoning in multimodal large language models. Multimodal reasoning poses challenges beyond text-only inference, requiring both visual perception and the integration of visual inputs into multi-step reasoning. Early approaches (Liu et al., 2023; Chen et al., 2024; Zhang et al., 2024b; Yang et al., 2023) typically convert images into textual descriptions for downstream tasks. More recently, inspired by Chain-of-Thought prompting in LLMs, models such as LLaVA-CoT (Xu et al., 2025), TVC (Sun et al., 2025), and Mulberry (Yao et al., 2024), among others (Huang et al., 2025; Deng et al., 2025; Meng et al., 2025; Wu et al., 2025; Wang et al., 2025b), extend CoT reasoning to multimodal settings and achieve strong results on benchmarks such as MathVista (Lu et al., 2024) and MathVision (Wang et al., 2024a). However, these models treat visual inputs as fixed context and perform reasoning entirely in the text space, lacking explicit mechanisms to re-access visual representations.

2.2 Implementing Visual Reference

A growing body of work equips MLLMs with mechanisms to interact with images more explicitly during reasoning, beyond treating the visual input as a single, static context. Existing approaches span a range of design choices, including predicting structured spatial references (e.g., boxes or points) to retrieve localized evidence (Gupta & Kembhavi, 2023; Wu & Xie, 2023; Chung et al., 2025; Jiang et al., 2025; Liu et al., 2024b; Gao et al., 2025), generating intermediate visual artifacts (e.g., sketches or diagrams) to externalize intermediate states (Hu et al., 2024; Borazjanizadeh et al., 2025; Li et al., 2025; Ma et al., 2025), and hybrid tool-use pipelines that interleave perception and reasoning steps (Gupta & Kembhavi, 2023; Hu et al., 2024).

Our work is most closely related to methods that enable *visual re-access* during decoding. We extend the pointer-generator idea (See et al., 2017) from selective copying in text to

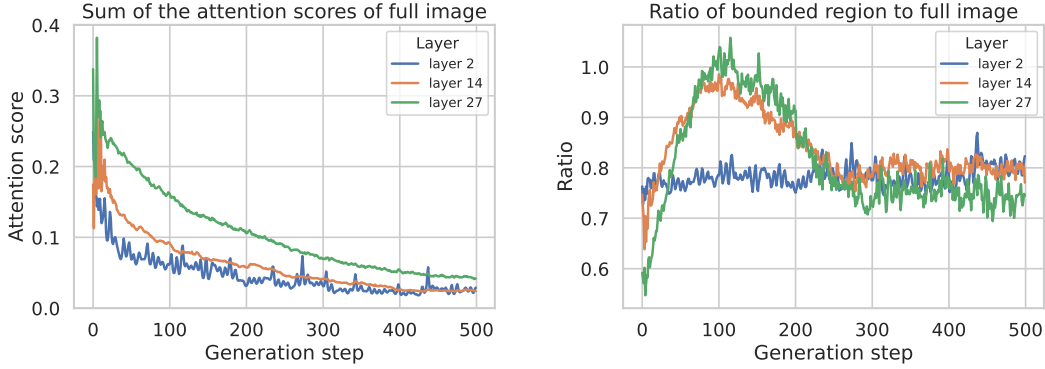


Figure 3: **Left:** Cumulative attention across all visual tokens, showing a gradual decrease in overall attention to the input image tokens. **Right:** Attention dynamics during reasoning, showing that semantically important visual regions receive disproportionately low attention, suggesting inefficient grounding.

selectively reusing visual token embeddings, yielding an interpretable and lightweight mechanism to directly point at the visual evidence.

3 Visual Grounding Decays During Reasoning

To examine how visual attention evolves at each generation step, we use RefCOCO (Kazemzadeh et al., 2014), a visual reference expression generation benchmark where the task is to create a caption that uniquely identifies a target region specified by a bounding box. Although RefCOCO generally involves shorter generations than mathematical reasoning, it provides a controlled setting for measuring attention dynamics over visual regions. We analyze the TVC-7B model (Sun et al., 2025) on the RefCOCO testA split, examining attention from the most recent token to all image tokens across early, middle, and late transformer layers (layers 2, 14, and 27).

Figure 3 left tracks total attention to image tokens and shows a steady decline across decoding steps, indicating a reduced reliance on visual grounding as generation proceeds. Figure 3 right measures attention to the target region by computing the ratio of mean attention on bounding-box image tokens to that over all image tokens. While layers 14 and 27 initially place greater emphasis on the target region, attention converges to a ratio of ~ 0.8 by mid-generation, suggesting weakened relative focus on salient visual tokens.

These results suggest *visual grounding decay*, where attention to relevant visual content diminishes during extended generation. This limitation is especially relevant for multimodal reasoning tasks where intermediate steps may benefit from revisiting visual evidence, thereby motivating architectures with dynamic visual access during inference.

4 Method

At inference time, **v1** operates as a single, self-contained MLLM: all textual reasoning, visual localization, and point-and-copy operations are performed by the same fine-tuned backbone without invoking external models or tools.

4.1 Preliminary: Pointing for Language Generation

Formulation. We formulate a conditional next-token prediction objective, as commonly adopted in modern multimodal large language models (MLLMs). Given a sequence of continuous input representations c (e.g. embedded text tokens or visual features) the model is trained to autoregressively predict the discrete next token x_t conditioned on the input c

and previously generated tokens $x_{<t}$:

$$p(x_1, \dots, x_T \mid c) = \prod_{t=1}^T p(x_t \mid c, x_1, \dots, x_{t-1})$$

The continuous input sequence c may include a heterogeneous mixture of modality-specific features, such as embedded discrete text tokens or continuous visual embeddings produced by image encoders (e.g. CLIP (Radford et al., 2021)). This general formulation covers a wide range of multimodal architectures such as LLaVA (Liu et al., 2023) and Qwen-VL (Bai et al., 2025), which use continuous input image representations.

Pointing. For visually grounded reasoning, rather than generating new visual tokens, we instead teach the model to *point* to its position within the input image sequence it already understands, thereby referencing it explicitly.

The pointing mechanism we examine was first introduced by the pointer-generator network (See et al., 2017) in text summarization research. In the pointer-generator network, the input context sequence c also consists of discrete tokens within the vocabulary space V , unlike our setup. The model dynamically mixes two distributions at each decoding step t : (1) a generation distribution over the vocabulary $P_{\text{gen}}(x_t)$, and (2) a copy distribution $P_{\text{ptr}}(x_t)$ over input tokens. The final output probability is given by a gated mixture:

$$\begin{aligned} p(x_t \mid c, x_{<t}) &= \lambda_t P_{\text{gen}}(x_t) + (1 - \lambda_t) P_{\text{ptr}}(x_t), \\ \lambda_t &\equiv \lambda(x_t \mid c, x_{<t}), \end{aligned} \quad (1)$$

where $\lambda \in [0, 1]$ is a learnable scalar gate that controls the trade-off between generating a new token and copying one from the input.

The pointer distribution is obtained via attention over the encoder representations:

$$\begin{aligned} \alpha_t^{(k)} &= \frac{\exp(\text{score}(h_t, c_k))}{\sum_{k'} \exp(\text{score}(h_t, c_{k'}))}, \\ P_{\text{ptr}}(x_t = w) &= \sum_{k: w_k = w} \alpha_t^{(k)}. \end{aligned} \quad (2)$$

where h_t is the decoder hidden state at step t , w_k the token at position k , and score denotes a standard attention scoring function (e.g., dot-product or additive). We generalize this formulation beyond the original implementation to arbitrary autoregressive language models for explanatory purposes.

Discrete targets. The above formulation constrains the pointing targets to be within the discrete vocabulary space V . This prevents application to general MLLMs as the multimodal inputs often consist of continuous feature sequences (Liu et al., 2023; Bai et al., 2025).

4.2 v1: Pointing for Multimodal Grounded Reasoning

To overcome these limitations, we introduce **v1**, a lightweight extension to autoregressive MLLMs that enables explicit visual grounding by pointing to continuous input embeddings. **v1** augments the vocabulary with pointer tokens that allow the model to either generate text or copy visual content during inference. All reasoning and grounding are handled within a single fine-tuned backbone (e.g., Qwen2.5-VL), without external modules or auxiliary grounding networks. This design supports unified inference over discrete and continuous modalities without modifying the core architecture (Figure 2).

Pointing to continuous inputs. The gated mixture formulation of See et al. (2017) is unfit for continuous inputs as image embeddings, as such inputs lack discrete mappings to vocabulary tokens V . To enable pointing in this setting, we extend the output space to include references to *positions* in the continuous input. Specifically, we define the augmented output space as $\bar{V} = V \cup C$, where $C = \{c_1, c_2, \dots, c_K\}$ denotes the set of continuous input

Model	Size	Reasoning Only	MathVista mini	MathVision mini	MathVision full	MathVerse mini	Average mini	Average full
Qwen2-VL (Wang et al., 2024b)	7B	✗	60.9	-	16.3	24.6	-	20.5
Qwen2-VL (Wang et al., 2024b)	72B	✗	69.7	-	26.6	36.2	-	31.4
Qwen2.5-VL (Bai et al., 2025)	7B	✗	67.8	23.6	-	44.5	45.3	-
Qwen2.5-VL (Bai et al., 2025)	72B	✗	74.8	39.8	-	57.6	57.4	-
InternVL2.5 (Chen et al., 2025)	8B	✗	64.4	22.0	19.7	39.5	41.9	29.6
InternVL2.5 (Chen et al., 2025)	78B	✗	72.3	34.9	32.2	51.7	53.0	42.0
GPT-4o (Hurst et al., 2024)	-	✗	63.8	-	30.4	50.2	-	40.3
LLaVa-CoT (Xu et al., 2025)	11B	✓	54.8	16.3	-	33.9	35.0	-
Mulberry (Yao et al., 2024)	7B	✓	63.1	-	-	39.6	-	-
TVC (Sun et al., 2025)	7B	✓	68.1	-	22.7	38.9	-	30.8
TVC (Sun et al., 2025)	72B	✓	72.2	-	41.9	48.8	-	45.4
QVQ-72B-preview (Qwen Team, 2024)	72B	✓	71.4	35.9	-	41.5	49.6	-
Base (Qwen2.5-VL)	7B	✗	67.8	23.6	-	44.5	45.3	-
Text-Only (TVC)	7B	✓	68.1	-	22.7	38.9	-	30.8
Ours	7B	✓	68.6	34.5	28.1	48.6	50.6	38.4
↳ Inference w/o Pointing	7B	✓	60.0	25.3	23.7	33.6	39.6	28.7

Table 1: **Results on multimodal mathematical reasoning tasks.** We report MathVision on both the mini and full subsets to include additional baselines. Average (mini) is the mean over MathVista (mini), MathVision (mini), and MathVerse (mini); Average (full) is the mean over MathVision (full) and MathVerse (mini), as MathVista has no publicly scorable full split. **Bold** denotes the best result among 7B-scale models; larger models (72B/78B) and GPT-4o are reported as reference points rather than size-matched competitors. The final block repeats the 7B base and text-only rows alongside **v1** to make the controlled comparison directly readable. Some scores are absent in the references (“-”).

vectors (e.g. embeddings of the input image patches). This formulation allows the model to generate either a vocabulary token or a pointer to a specific continuous input. We denote a pointer to input vector c_k as $\langle \text{ptr} : c_k \rangle$, which is treated as a discrete token during decoding.

At each decoding step t , the model computes two distributions: (1) a generation distribution over the vocabulary V , producing logits $\text{logit}_{\text{gen}} \in \mathbb{R}^{|V|}$, and (2) a pointing distribution over the input positions C , producing logits $\text{logit}_{\text{ptr}} \in \mathbb{R}^K$. The final output logits are defined as:

$$\text{logit}_t = [\text{logit}_{\text{gen}} \parallel \text{logit}_{\text{ptr}}] \in \mathbb{R}^{|V|+K}$$

where $[\cdot \parallel \cdot]$ denotes concatenation. Pointing logits are computed by attending over the image sequence:

$$\text{logit}_{\text{ptr}}^{(k)} = \frac{L_q(h_t) \cdot L_k(c_k)^\top}{\sqrt{D}}$$

where h_t is the decoder hidden state at step t , L_q and L_k are learned linear projections, and the scaling factor $\sqrt{D}$ follows standard attention practice. We omit the gating module λ as the logit types are defined over disjoint spaces and do not require interpolation.

During inference, if the model selects an index in V , the next token x_t is emitted as the corresponding vocabulary token. If the model selects an index $k \in C$, the token is represented as a pointer $x_t = \langle \text{ptr} : c_k \rangle$. On the subsequent decoding step, the input embedding at position t is replaced with the continuous vector c_k , enabling the model to attend directly to the referenced content.

4.3 Annotating Visually-grounded Reasoning Data

To train **v1**, we require fine-grained multimodal reasoning traces in which each step is grounded to specific visual evidence. To this end, we construct **v1g**, a dataset of 300K multimodal reasoning paths with interleaved grounding annotations. Each trajectory includes a sequence of reasoning steps, where textual inferences are explicitly linked to corresponding image regions.

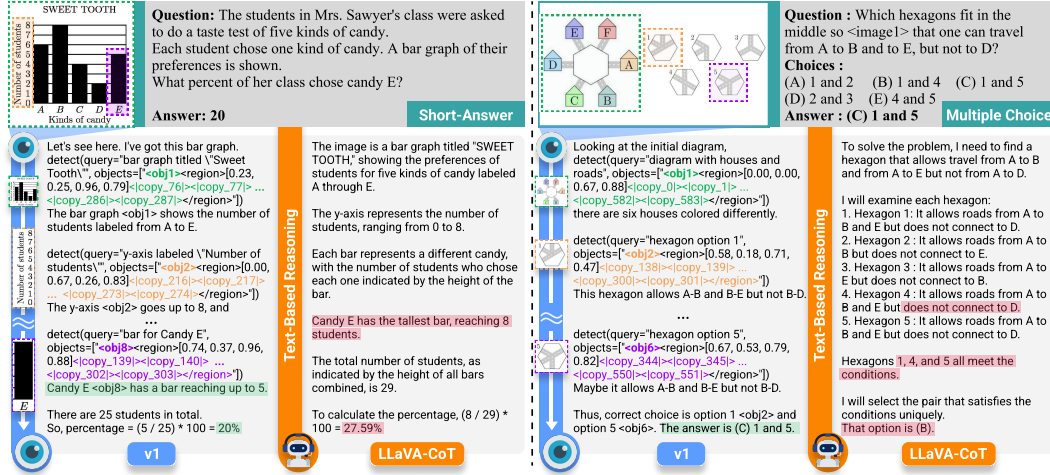


Figure 4: **Qualitative comparison on MathVision.** v1’s dynamic grounding helps to solve both bar graph and spatial reasoning tasks, while LLaVA-CoT misinterprets visual context.

We generate the dataset using a fully automated three-stage pipeline: (1) oversampling textual reasoning paths from a pretrained MLLM, (2) decomposing each path into discrete visual queries and retrieval steps via an LLM, and (3) grounding each visual reference by aligning it with a bounding box in the input image (see Section D and Figure 7 for details).

Constructing base reasoning traces. As a seed to our grounded corpus, we adopt the training set of TVC (Sun et al., 2025), which consists of reasoning traces generated from the QvQ model (Qwen Team, 2024).

Decomposing reasoning traces into visual reference steps. We use a strong off-the-shelf LLM, Gemini-2.0-flash (Google, 2025), to extract visual grounding cues from text-based reasoning traces. The model collects and rewrites each visual reference as a *detect* call, which takes a short natural-language description and returns the corresponding image region. Retrieved regions are cached and assigned symbolic identifiers such as <objX> in order of appearance. The LLM also produces a key-value list of visual components, where each key is a unique descriptive reference used in later steps. We guide this process with domain-specific few-shot prompts, described in Appendix D. We then filter out invalid outputs, including mismatches between references and retrieved objects, non-unique labels, insufficient object count (≤ 2), and ill-formed reasoning. After filtering, about 82% of samples are retained.

Grounding visual references to image regions. Visual grounding remains difficult in multimodal reasoning, especially for abstract or unconventional regions such as charts, geometry, and medical scans. Existing grounding models often perform poorly in these settings (Steiner et al., 2024; Xiao et al., 2024), particularly when the target is abstract or symbolic, such as an angle or geometric relation.

To exploit the implicit visual grounding behavior in MLLMs, we build on Qwen2.5-VL’s visual grounding capacity (Bai et al., 2025). However, rather than relying on its coordinate generation interface, we estimate the model’s visual focus using a *relative attention mechanism* inspired by Zhang et al. (2025), in order to better handle non-natural domains and symbolic cues. The corresponding algorithm is detailed in Appendix F. We validate the quality of this pipeline through human evaluation in Appendix E. This produces a curated training set of ~300K grounded examples.

5 Implementation Details

Preprocessing. Given interleaved images, text, and bounding box annotations, we first convert each image into a flattened sequence of patches following the backbone’s patchification scheme (e.g., Qwen2.5-VL (Bai et al., 2025)). Each bounding box is then mapped to a sequence of pointer tokens (e.g., $\langle \text{ptr4} \rangle, \dots, \langle \text{ptr32} \rangle$) that index the enclosed patches. These tokens are added to the tokenizer vocabulary without changing the original embedding table or generation head. Instead, during preprocessing, their embeddings are replaced by the corresponding image patch embeddings before entering the transformer. The resulting input is a unified sequence of text tokens, pointer tokens, and image patch embeddings.

Model. We instantiate **v1** with Qwen2.5-VL as the MLLM backbone and add only *two lightweight linear layers*: a pointing query head $L_q \in \mathbb{R}^{D \times D}$ and a pointing key head $L_k \in \mathbb{R}^{D \times D}$, where D is the latent dimensionality. Both are initialized as identity matrices scaled by $1/\sqrt{D}$, so their initial effect on the output distribution is minimal. This is well suited to our setting because the pretrained backbone already provides meaningful generative likelihoods P_{gen} , while the pointing module selects at most one position per timestep from P_{ptr} , allowing smooth early training without catastrophic forgetting.

Training. Given the dual-nature output space comprising a generative vocabulary V and a pointing reference set K , we incorporate z-loss regularization to stabilize the softmax partition function, following Chameleon Team (2024). Specifically, we regularize the log-partition function $Z = \sum_j e^{x_j}$ in the softmax $\sigma(x)_i = e^{x_i}/Z$ by introducing a z-loss term $\mathcal{L}_z = \lambda \log \bar{Z}$, where $\lambda = 10^{-5}$. To reduce computational overhead, we approximate Z using a top- $k = 40$ partition function $\bar{Z} = \sum_{j \in \text{TopK}(x)} e^{x_j}$. This approximation enables efficient and numerically stable training in large-output-space settings. Further details are in Section A.

Inference. At each decoding step x_t , **v1** uses two extra caches: (1) key features $L_k(c)$ at image patch positions to compute pointing logits $\text{logit}_{\text{ptr}}$, and (2) the corresponding image patch features for copying. We implement these by extending the HuggingFace Transformers key-value cache (Wolf et al., 2020). The overhead is small, since only a subset of tokens from a single layer is stored.

The added model cost is also minimal: **v1** introduces only one linear projection over image embeddings, negligible relative to the language head. During inference, copied visual tokens increase sequence length, but with revisit suppression they typically stay below 60% of the text token count, yielding an average total length of about $1.6 \times$ text-only reasoning. No external models or executors are used.

6 Experiments

6.1 Downstream Evaluation

Setup. We use three representative multimodal mathematical reasoning benchmarks: MathVista (mini) (Lu et al., 2024), MathVision (mini/full) (Wang et al., 2024a), and MathVerse (mini) (Zhang et al., 2024a). Following prior work (Duan et al., 2024), we compute accuracy with an LLM judge (Gemini-2.0-Flash) that matches predicted and reference answers.

We compare our method against both general-purpose and reasoning-specialized MLLMs. General MLLMs include Qwen2-VL (Wang et al., 2024b) and Qwen2.5-VL (Bai et al., 2025) at both 7B and 72B scales, as well as InternVL2.5 (Chen et al., 2025) at 8B and 78B. We also include GPT-4o (Hurst et al., 2024) as a high-performing proprietary baseline. For reasoning-oriented models, we evaluate LLaVa-CoT-11B (Xu et al., 2025), Mulberry-7B (Yao et al., 2024), TVC-7B and 72B (Sun et al., 2025), and QVQ-72B-preview (Qwen Team, 2024).

Results. Quantitative results are presented in Table 1. Our approach yields substantial performance improvements over baseline models. Among 7B-scale models, **v1** with full

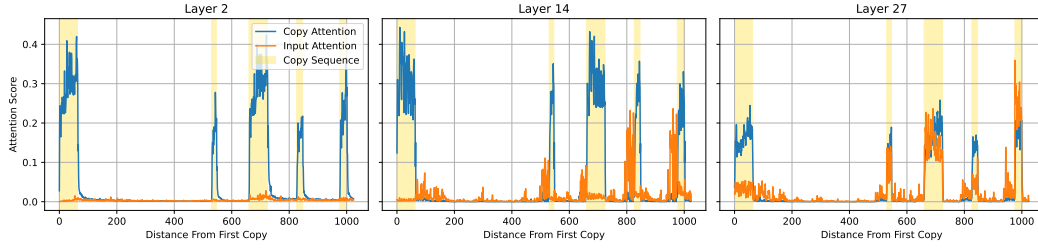


Figure 5: **Comparison of attention to copy tokens vs. original visual tokens.** Layer-wise sum of attention scores directed to copy tokens and their corresponding original visual input tokens from a **v1**. Copy token intervals are shown in yellow.

pointing capability outperforms both general-purpose and reasoning-specialized baselines. Notably, despite its smaller size, our 7B model narrows the performance gap with several 72B-scale models. The gains are particularly pronounced on MathVision, a benchmark known for its stronger demand for correct grounding.

6.2 Further Analysis

Qualitative results. Figure 4 compares our method with LLaVA-CoT (Xu et al., 2025) on MathVision short-answer and multiple-choice examples. Our **v1** exhibits explicit visual grounding via pointer-based detection and selective copying of relevant image regions. In the bar graph example, **v1** correctly identifies the bar for Candy E and computes the correct percentage, while LLaVA-CoT misidentifies the tallest bar. In the hexagon pathfinding task, **v1** reasons over spatial connectivity by attending to structural differences among options, whereas LLaVA-CoT fails to eliminate invalid candidates. These examples illustrate how active visual reference supports more precise and interpretable grounded reasoning than text-only chain-of-thought methods.

Ablation study. We report an ablation study in Table 2 to isolate the effect of the point-and-copy mechanism. All variants share the same Qwen2.5-VL-7B backbone: *Backbone* applies no task-specific fine-tuning, while *Ours* (*no pointing*) uses the same training setup as **v1** but disables pointing at inference. On MathVision (testmini), **v1** improves over the backbone from 23.6 to 34.5 (+10.9), whereas disabling pointing reduces the score to 25.3 (+1.7), indicating that point-and-copy at inference is a key driver of the gains.

Variant	Tr.	Inf.	Score	Δ
Backbone	✗	✗	23.6	–
(no pointing)	✓	✗	25.3	+1.7
Ours	✓	✓	34.5	+10.9

Table 2: **Ablation on MathVision mini.** *Train*: grounded-reasoning training. *Infer*: point-and-copy at inference. Δ : gain relative to the backbone.

For context, TVC (Sun et al., 2025) serves as a text-only reasoning baseline; our training data extends its reasoning traces with additional visual-grounding annotations.

How does **v1 utilize pointed visual regions?** We analyze how **v1** uses visual regions retrieved via the point-and-copy mechanism. As shown in Figure 5, we compare *Input Attention* (to original visual tokens) and *Copy Attention* (to copied tokens) during generation after the first copy operation.

We observe a structured sequence of behaviors. Before copying, attention to original image tokens increases, indicating a localization step in which relevant regions are identified. Immediately after copying, intermediate layers (e.g., layers 2 and 14) exhibit dominant copy attention, reflecting focused post-retrieval processing of the retrieved region. When averaged across layers, copied tokens consistently receive higher attention than original image tokens, suggesting that copied regions serve as stable and accessible references. In

higher layers (e.g., layer 27), attention to input and copied tokens becomes more balanced, which may correspond to a late-stage integration of retrieved visual information into the broader reasoning context.

7 Conclusion

We introduced **v1**, a lightweight extension that enables MLLMs to actively revisit input images via a point-and-copy mechanism, and **v1g**, a dataset of 300K multimodal reasoning traces with fine-grained visual grounding. Experiments on established multimodal mathematical reasoning benchmarks show that **v1** improves performance, particularly on tasks requiring grounded, multi-step visual reasoning. We hope this work encourages further exploration of dynamic visual access as a core component of multimodal reasoning.

Acknowledgements

This work was partly supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korean government (MSIT) (No. RS-2021-II211343, Artificial Intelligence Graduate School Program (Seoul National University), No. RS-2026-25522885, Development of a World Foundation Model for Training and Development of Physical AI Systems, No. RS-2026-25512061, Development of Large Action Model-Based Autonomous Digital Twin Operation Technology for Proactive Problem Solving), the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (No. RS-2024-00354218, No. RS-2026-25561904), and the Technology Innovation Program (RS-2025-25456760, Development of a humanoid robot specialized in chemical processes based on AI foundation model) funded by the Ministry of Trade, Industry and Resources (MOTIR, Korea). We express special thanks to KAIT GPU project. The ICT at Seoul National University provides research facilities for this study.

References

- Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibao Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025. URL <https://arxiv.org/abs/2502.13923>.
- Nasim Borazjanizadeh, Roei Herzig, Eduard Oks, Trevor Darrell, Rogerio Feris, and Leonid Karlinsky. Visualizing thought: Conceptual diagrams enable robust planning in llms, 2025. URL <https://arxiv.org/abs/2503.11790>.
- Juliette Brun, Pascal Le Masson, and Benoît Weil. Designing with sketches: the generative effects of knowledge preordering. *Design Science*, 2, 2016. URL <https://api.semanticscholar.org/CorpusID:17302973>.
- Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*, 2024.
- Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14455–14465, June 2024.
- Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, Lixin Gu, Xuehui Wang, Qingyun Li, Yimin Ren, Zixuan Chen, Jiapeng Luo, Jiahao Wang, Tan Jiang, Bo Wang, Conghui He, Botian Shi, Xingcheng Zhang, Han Lv, Yi Wang, Wenqi Shao, Pei Chu, Zhongying Tu, Tong He, Zhiyong Wu, Huipeng Deng, Jiaye Ge, Kai Chen, Kaipeng Zhang, Limin Wang, Min Dou, Lewei Lu, Xizhou Zhu, Tong Lu, Dahua Lin, Yu Qiao, Jifeng Dai, and Wenhao Wang.

- Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling, 2025. URL <https://arxiv.org/abs/2412.05271>.
- Junyi Chu, Emily R. Fyfe, and Bethany Rittle-Johnson. Diagrams benefit symbolic problem-solving. *British Journal of Educational Psychology*, 87:273–287, 2017. URL <https://api.semanticscholar.org/CorpusID:14563301>.
- Jiwan Chung, Saejin Kim, Yongrae Jo, Jaewoo Park, Dongjun Min, and Youngjae Yu. Teaching metric distance to discrete autoregressive language models, 2025. URL <https://arxiv.org/abs/2503.02379>.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.
- Richard J. Cox. Representation construction, externalised cognition and individual differences. *Learning and Instruction*, 9:343–363, 1999. URL <https://api.semanticscholar.org/CorpusID:143780266>.
- Yihe Deng, Hritik Bansal, Fan Yin, Nanyun Peng, Wei Wang, and Kai-Wei Chang. Open-vlthinker: An early exploration to complex vision-language reasoning via iterative self-improvement, 2025. URL <https://arxiv.org/abs/2503.17352>.
- Haodong Duan, Junming Yang, Yuxuan Qiao, Xinyu Fang, Lin Chen, Yuan Liu, Xiaoyi Dong, Yuhang Zang, Pan Zhang, Jiaqi Wang, et al. Vlmevalkit: An open-source toolkit for evaluating large multi-modality models. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 11198–11201, 2024.
- Jun Gao, Yongqi Li, Ziqiang Cao, and Wenjie Li. Interleaved-modal chain-of-thought. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19520–19529, 2025. URL <https://arxiv.org/abs/2411.19488>.
- Gemma Team. Gemma 3 technical report, 2025. URL <https://arxiv.org/abs/2503.19786>.
- Google. Gemini 2.0 flash (gemini-2.0-flash-001), February 2025. URL <https://cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/2-0-flash>.
- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.
- Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14953–14962, June 2023.
- Yunzhuo Hao, Jiawei Gu, Huichen Will Wang, Linjie Li, Zhengyuan Yang, Lijuan Wang, and Yu Cheng. Can mllms reason in multimodality? EMMA: An enhanced multimodal reasoning benchmark. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, Proceedings of Machine Learning Research, 2025. URL <https://arxiv.org/abs/2501.05444>.
- Yushi Hu, Weijia Shi, Xingyu Fu, Dan Roth, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Ranjay Krishna. Visual sketchpad: Sketching as a visual chain of thought for multimodal language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL <https://openreview.net/forum?id=GNSM1P5VR>.
- Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Yao Hu, and Shaohui Lin. Vision-r1: Incentivizing reasoning capability in multimodal large language models, 2025. URL <https://arxiv.org/abs/2503.06749>.
- Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 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.
- Qing Jiang, Junan Huo, Xingyu Chen, Yuda Xiong, Zhaoyang Zeng, Yihao Chen, Tianhe Ren, Junzhi Yu, and Lei Zhang. Detect anything via next point prediction, 2025. URL <https://arxiv.org/abs/2510.12798>.
- Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. ReferItGame: Referring to objects in photographs of natural scenes. In Alessandro Moschitti, Bo Pang, and Walter Daelemans (eds.), *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 787–798, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1086. URL <https://aclanthology.org/D14-1086>.
- Maria Kozhevnikov, Mary Hegarty, and Richard E. Mayer and. Revising the visualizer-verbalizer dimension: Evidence for two types of visualizers. *Cognition and Instruction*, 20(1):47–77, 2002. doi: 10.1207/S1532690XCI2001_3. URL https://doi.org/10.1207/S1532690XCI2001_3.
- Chengzu Li, Wenshan Wu, Huanyu Zhang, Yan Xia, Shaoguang Mao, Li Dong, Ivan Vulić, and Furu Wei. Imagine while reasoning in space: Multimodal visualization-of-thought, 2025. URL <https://arxiv.org/abs/2501.07542>.
- Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let’s verify step by step, 2023. URL <https://arxiv.org/abs/2305.20050>.
- Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=w0H2xGH1kw>.
- Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European conference on computer vision*, pp. 38–55. Springer, 2024a.
- Zuyan Liu, Yuhao Dong, Yongming Rao, Jie Zhou, and Jiwen Lu. Chain-of-spot: Interactive reasoning improves large vision-language models, 2024b. URL <https://arxiv.org/abs/2403.12966>.
- 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 *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=KUNzEQMWU7>.
- Tianren Ma, Lingxi Xie, Yunjie Tian, Boyu Yang, and Qixiang Ye. Clawmachine: Learning to fetch visual tokens for referential comprehension. In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=T0tk9dTYGG>.
- Fanqing Meng, Lingxiao Du, Zongkai Liu, Zhixiang Zhou, Quanfeng Lu, Daocheng Fu, Tiancheng Han, Botian Shi, Wenhai Wang, Junjun He, Kaipeng Zhang, Ping Luo, Yu Qiao, Qiaosheng Zhang, and Wenqi Shao. Mm-eureka: Exploring the frontiers of multimodal reasoning with rule-based reinforcement learning, 2025. URL <https://arxiv.org/abs/2503.07365>.
- Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling, 2025. URL <https://arxiv.org/abs/2501.19393>.
- Qwen Team. Qvq: To see the world with wisdom, December 2024. URL <https://qwenlm.github.io/blog/qvq-72b-preview/>.

- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. URL <https://arxiv.org/abs/2103.00020>.
- Abigail See, Peter J Liu, and Christopher D Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1073–1083, 2017.
- Haozhan Shen, Peng Liu, Jingcheng Li, Chunxin Fang, Yibo Ma, Jiajia Liao, Qiaoli Shen, Zilun Zhang, Kangjia Zhao, Qianqian Zhang, Ruochen Xu, and Tiancheng Zhao. Vlm-r1: A stable and generalizable r1-style large vision-language model, 2025. URL <https://arxiv.org/abs/2504.07615>.
- Andreas Steiner, André Susano Pinto, Michael Tschannen, Daniel Keysers, Xiao Wang, Yonatan Bitton, Alexey Gritsenko, Matthias Minderer, Anthony Sherbondy, Shangbang Long, et al. Paligemma 2: A family of versatile vlms for transfer. *arXiv preprint arXiv:2412.03555*, 2024.
- Hai-Long Sun, Zhun Sun, Houwen Peng, and Han-Jia Ye. Mitigating visual forgetting via take-along visual conditioning for multi-modal long cot reasoning, 2025. URL <https://arxiv.org/abs/2503.13360>.
- Haozhe Wang, Chao Qu, Zuming Huang, Wei Chu, Fangzhen Lin, and Wenhui Chen. VL-rethinker: Incentivizing self-reflection of vision-language models with reinforcement learning, 2025a. URL <https://arxiv.org/abs/2504.08837>.
- Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. Measuring multimodal mathematical reasoning with math-vision dataset. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024a. URL <https://openreview.net/forum?id=QWTCcxMpPA>.
- Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model’s perception of the world at any resolution, 2024b. URL <https://arxiv.org/abs/2409.12191>.
- Xiyao Wang, Zhengyuan Yang, Chao Feng, Hongjin Lu, Linjie Li, Chung-Ching Lin, Kevin Lin, Furong Huang, and Lijuan Wang. Sota with less: Mcts-guided sample selection for data-efficient visual reasoning self-improvement, 2025b. URL <https://arxiv.org/abs/2504.07934>.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. URL <https://www.aclweb.org/anthology/2020.emnlp-demos.6>.
- Penghao Wu and Saining Xie. V*: Guided visual search as a core mechanism in multimodal llms, 2023. URL <https://arxiv.org/abs/2312.14135>.
- Qiong Wu, Xiangcong Yang, Yiyi Zhou, Chenxin Fang, Baiyang Song, Xiaoshuai Sun, and Rongrong Ji. Grounded chain-of-thought for multimodal large language models, 2025. URL <https://arxiv.org/abs/2503.12799>.
- Jiaer Xia, Yuhang Zang, Peng Gao, Sharon Li, and Kaiyang Zhou. Visionary-r1: Mitigating shortcuts in visual reasoning with reinforcement learning, 2025. URL <https://arxiv.org/abs/2505.14677>.

- Bin Xiao, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yumao Lu, Michael Zeng, Ce Liu, and Lu Yuan. Florence-2: Advancing a unified representation for a variety of vision tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4818–4829, 2024.
- Guowei Xu, Peng Jin, Hao Li, Yibing Song, Lichao Sun, and Li Yuan. Llava-cot: Let vision language models reason step-by-step, 2025. URL <https://arxiv.org/abs/2411.10440>.
- Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Ehsan Azarnasab, Faisal Ahmed, Zicheng Liu, Ce Liu, Michael Zeng, and Lijuan Wang. Mm-react: Prompting chatgpt for multimodal reasoning and action. *arXiv preprint arXiv:2303.11381*, 2023.
- Huanjin Yao, Jiaxing Huang, Wenhao Wu, Jingyi Zhang, Yibo Wang, Shunyu Liu, Yingjie Wang, Yuxin Song, Haocheng Feng, Li Shen, and Dacheng Tao. Mulberry: Empowering mllm with o1-like reasoning and reflection via collective monte carlo tree search, 2024. URL <https://arxiv.org/abs/2412.18319>.
- Jiarui Zhang, Mahyar Khayatkhoei, Prateek Chhikara, and Filip Ilievski. MLLMs know where to look: Training-free perception of small visual details with multimodal LLMs. In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=DgaY5mDdmT>.
- Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Yu Qiao, et al. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems? In *European Conference on Computer Vision*, pp. 169–186. Springer, 2024a.
- Zhuosheng Zhang, Aston Zhang, Mu Li, hai zhao, George Karypis, and Alex Smola. Multi-modal chain-of-thought reasoning in language models. *Transactions on Machine Learning Research*, 2024b. ISSN 2835-8856. URL <https://openreview.net/forum?id=y1pPWFVfVR>.

Table 3: **Inference cost per sample.** Total tokens, copied visual tokens, wall-clock time, and throughput under HuggingFace `.generate` and batched vLLM. Dashes indicate quantities not separately instrumented under batched inference.

Runner	Model	Tokens	Copy tokens	Wall clock	Tokens/sec
HF <code>.generate</code>	TVC-7B	4,057	0	143.42 s	28.29
vLLM (batched)	TVC-7B	–	–	23.86 s	110.38
HF <code>.generate</code>	v1 -7B	4,087	990	154.04 s	26.53
vLLM (batched)	v1 -7B	–	–	25.65 s	103.24

Overview of the Appendix

This Appendix is structured as follows:

- Section A describes implementation details and resources used in the project;
- Section B reports additional experiments on generalization, cross-backbone transfer, test-time scaling, and training compute;
- Section C discusses limitations and future directions;
- Section D provides details of the data generation process;
- Section E reports human evaluation results validating the grounding quality of our training dataset (**v1g**) and our model (**v1**);
- Section F details pseudo-code on the visual grounding pipeline we utilized in the data generation process;
- Section G presents the use of Large Language Models (LLMs);
- Section H presents additional qualitative results.

A Implementation Details & Resources

Training Details All models are trained under uniform settings: a base learning rate of 3×10^{-5} , per-device batch size of 2, and gradient accumulation over 4 steps. We leverage DeepSpeed for distributed training across 8 NVIDIA A100 GPUs. Optimization uses AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.95$, and training is performed for 5 epochs. Training **v1** requires approximately 672 A100-equivalent GPU-hours (300K examples over 5 epochs on 8 NVIDIA A100 GPUs).

Inference Cost Table 3 reports measured inference cost per sample under HuggingFace `.generate` and batched vLLM inference. Point-and-copy adds roughly 990 copied visual tokens per sample, yet the wall-clock overhead relative to the text-only TVC-7B baseline is modest, about 7% under both runners. The $1.6\times$ sequence-length figure quoted in the main text is a conservative upper bound derived from training-data statistics rather than a measured runtime cost.

Training Duration Our training schedule of 5 epochs follows the setup used for the original text-only reasoning trace dataset, which we extend to the grounded reasoning setup (Sun et al., 2025). Because the reasoning traces contain substantially longer token sequences than typical MLLM data, shorter training runs produced unstable behaviors such as repetition and incomplete reasoning without a final answer. In contrast, the point-and-copy behavior required relatively little data and typically saturated within the first epoch, as indicated by an early plateau in copy-token accuracy. The longer schedule therefore reflects the requirements of the inherited reasoning-trace setup rather than the needs of the pointer mechanism itself.

Table 4: **Generalization beyond mathematics (EMMA-mini).** Accuracy (%) on the physics and coding subsets. Rows below the rule are 7B-scale models evaluated in our pipeline; rows above are larger reference models.

Model	Physics	Coding
GPT-4o	44.0	38.0
Qwen2-VL-72B	40.0	37.0
Qwen2.5-VL-7B	30.0	32.0
v1 w/o copying	31.0	28.0
v1	36.0	40.0

Table 5: **Cross-backbone transfer.** Applying point-and-copy to Gemma3-4B (**v1**-Gemma3) improves over the vanilla backbone across math benchmarks. Gemma3-4B numbers are the OpenCompass leaderboard reference.

Benchmark	Gemma3-4B	v1 -Gemma3	Δ
MathVision-mini	21.05	25.00	+3.95
MathVista-mini	46.30	60.40	+14.10
MathVerse-mini	29.54	37.80	+8.26

B Additional Experimental Results

This section reports supplementary experiments that probe how far the point-and-copy mechanism generalizes: beyond mathematics (Table 4), beyond a single backbone (Table 5), against sequential test-time scaling (Table 6), and relative to the training compute of reinforcement-learning alternatives (Table 7). We also analyze where the mechanism still fails (Figure 6).

Generalization beyond mathematics. Although **v1g** is built for mathematical grounded reasoning, the point-and-copy mechanism itself is domain-agnostic. We therefore evaluate on two non-math subsets, physics and coding, of the EMMA-mini multimodal reasoning benchmark (Hao et al., 2025). As shown in Table 4, **v1** improves over both the Qwen2.5-VL-7B backbone and the matched no-copy variant on both domains, suggesting the gain is connected to explicit visual re-access rather than to math-specific supervision alone.

Cross-backbone transfer. To test whether the gains depend on a particular backbone, we apply point-and-copy to Gemma3-4B (Gemma Team, 2025) while keeping the grounding supervision unchanged. Table 5 shows consistent improvements across all three math benchmarks, indicating that the benefit is driven by the mechanism rather than by backbone-specific attention behavior.

Comparison to sequential test-time scaling. A natural question is whether simply spending more decoding budget on a text-only model recovers the benefit of visual re-access. Because a trained **v1** cannot be scaled down at test time without truncating its answer, we apply sequential test-time scaling (Muennighoff et al., 2025) to the text-only TVC-SFT baseline instead. Table 6 shows that increasing the text-only budget from 8,192 to 12,800 tokens yields only a marginal gain and remains far below **v1** at the same budget.

Failure cases. Point-and-copy provides explicit visual re-access, but it does not eliminate reasoning errors. Inspecting incorrect MathVision predictions, we observe four recurring failure modes, shown in Figure 6. In *wrong-region copying* (a), the pointed region does not contain the queried object: the model asks for an 8×8 grid but points only at its top-left quadrant. In *under-copying* (b), the model points once at the entire image and then continues in text without re-accessing localized evidence. In *high-count copying* (c), the question requires tracking many distinct instances, and the model issues sixteen separate pointer

Table 6: **Sequential test-time scaling on MathVision-mini.** Text-only chain-of-thought at matched token budgets versus **v1**.

Model	Size	Token budget	Score
Text-only	7B	8,192	24.0
Text-only	7B	12,800	24.7
v1	7B	12,800	34.5

Table 7: **Training compute reference.** A100-equivalent GPU-hours for **v1** and for reinforcement-learning vision-language runs with reported or estimable compute. VL-Rethinker additionally uses a Forced-Rethinking SFT stage. GRPO entries are a compute reference, not a controlled baseline.

Method	Setting	Training data	A100-equiv. GPU-hrs
v1 (ours)	7B-VL SFT	300K $\times$ 5 epochs	672
Visionary-R1 (Xia et al., 2025)	3B-VL GRPO	272.6K	$\sim$ 1,500
VL-Rethinker (Wang et al., 2025a)	7B-VL GRPO	16K $\times$ 3 epochs	$\sim$ 2,000–2,300
VLM-R1 (Shen et al., 2025)	3B-VL GRPO	2 epochs	3,200–3,700+
VLM-R1 (Shen et al., 2025)	3B-VL GRPO	extended run	11,200–12,900+

calls across the answer options without resolving the count. In *ambiguous targets* (d), the referent cannot be delimited by a box at all, such as the reflected path of a ball on a billiard table. These cases indicate that point-and-copy improves access to visual evidence but does not by itself resolve localization ambiguity or long-horizon symbolic reasoning.

Training compute relative to RL-based alternatives. **v1** is trained with supervised fine-tuning rather than online policy optimization, so reinforcement-learning methods such as GRPO are not a like-for-like baseline: they change both the optimization objective and the source of supervision. They are nonetheless a useful compute reference. Wall-clock cost is rarely reported for VL-GRPO runs, so in Table 7 we compile runs with reported or estimable compute and normalize them to A100-equivalent GPU-hours using published BF16 H100/A100 throughput ratios of roughly $2.0\text{--}2.3\times$.

C Limitations and Future Work

This work focuses on demonstrating the effectiveness of active visual reference in structured multimodal reasoning via a simple point-and-copy mechanism. While **v1** shows strong performance in mathematical domains, several directions remain for broader applicability.

Beyond mathematical domains. Extending **v1** to other settings (*e.g.*, scientific diagrams, medical images, or visual commonsense) presents new challenges in representation and supervision. These domains often lack structured reasoning traces, making data collection more difficult. Since **v1g** relies on a pretrained text-only MLLM to seed reasoning, generalizing to less structured domains will require advances in decomposition, grounding, and alignment.

Weak supervision and reinforcement learning. Recent work in inference-time scaling and alignment has shown the promise of reward-based learning for reasoning. Incorporating such methods into **v1** may enable more flexible and efficient visual retrieval strategies without dense supervision. We leave this exploration to future work due to current resource constraints.

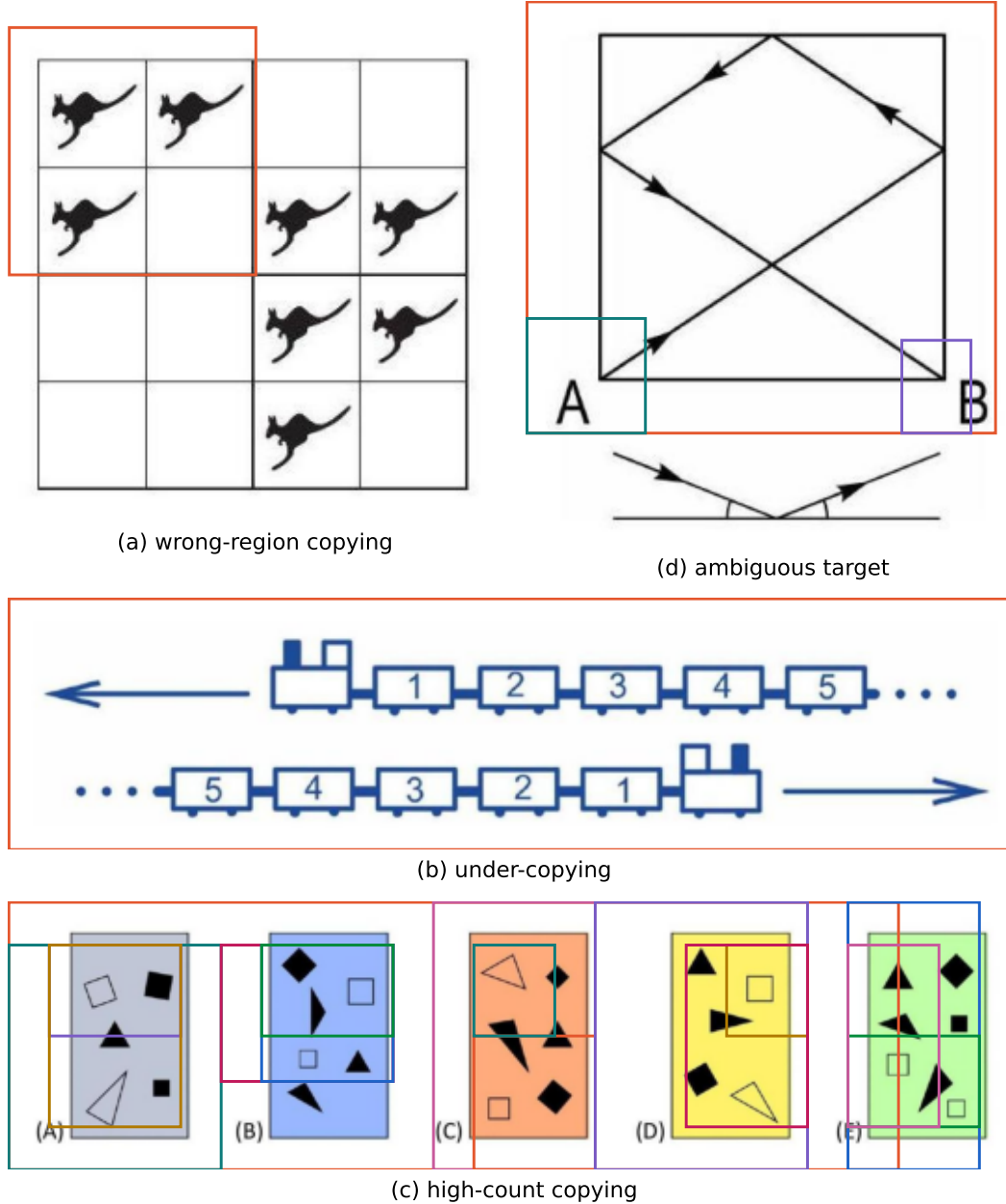


Figure 6: **Representative failure modes of v1 on MathVision.** Boxes show the regions selected by the model, colored by step. (a) The query names the full 8×8 grid but only its top-left quadrant is selected. (b) A single whole-image selection, after which reasoning proceeds without further visual access. (c) Sixteen pointer calls across answer options and their sub-shapes. (d) A reflected billiard path, which no axis-aligned box can isolate.

D Data Generation Details

Figure 7 illustrates the construction pipeline for our **v1g** dataset; each stage of this pipeline is described in detail in Section 4.3. The specific prompt template used to decompose text-based reasoning paths into visual queries (as outlined in our methodology in Section 4.3) is provided in Table 10.

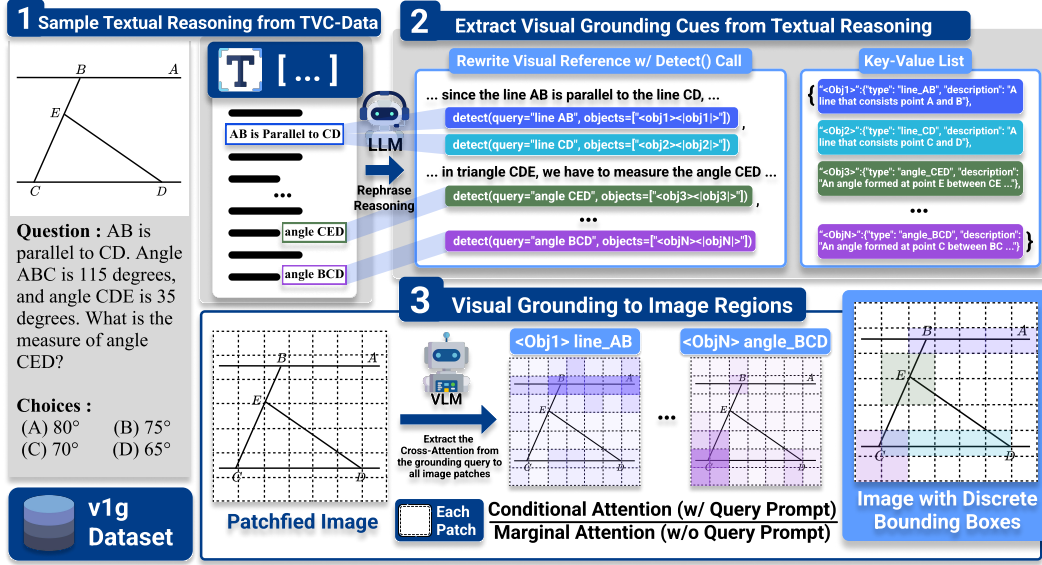


Figure 7: v1g dataset construction pipeline.

E Human Evaluation of Grounding Quality

E.1 Evaluation of v1g Dataset Quality

To validate the quality of our automatically generated visual grounding annotations in the v1g dataset, we conducted a human evaluation comparing our attention-based grounding approach against GroundingDINO (Liu et al., 2024a), a widely used open-set object detector.

Methodology. We randomly sampled 100 examples from the v1g dataset, each containing multiple bounding boxes. Three expert annotators independently evaluated each bounding box using binary classification on three criteria:

- **Correctness:** Whether the bounding box covers the intended object or region
- **Comprehensiveness:** Whether all relevant visual content is included within the box
- **Tightness:** Whether the box is well-fitted with minimal extraneous background

We report the average score across annotators, majority vote, and Fleiss’ κ to assess inter-annotator agreement. Agreement quality follows standard interpretations: Fair (0.21–0.40), Moderate (0.41–0.60), Substantial (0.61–0.80), and Almost Perfect (0.81–1.00).

Results. As shown in Table 8, our attention-based grounding method substantially outperforms GroundingDINO on correctness (83.3% vs. 29.3% average score), demonstrating superior capability in localizing semantically complex and context-dependent entities such as geometric elements (e.g., “angle ABC”), chart components (e.g., “bar for Grace”), and referring expressions (e.g., “the second figure”). While GroundingDINO achieves higher inter-annotator agreement, this primarily reflects consistent failure modes rather than quality, as evidenced by its low absolute performance.

E.2 Evaluation of v1 Pointing Accuracy

We additionally evaluated the pointing accuracy of our trained v1 model to assess how effectively it grounds visual references during inference.

Table 8: Human evaluation of v1g dataset quality.

Method	Metric	Avg	Majority	Fleiss' κ	Agreement
Attention-based (Ours)	Correctness	83.3%	87.0%	0.352	Fair
	Comprehensive.	55.0%	56.0%	0.582	Moderate
	Tightness	46.0%	44.0%	0.436	Moderate
Grounding-DINO	Correctness	29.3%	30.0%	0.711	Substantial
	Comprehensive.	47.3%	49.0%	0.906	Almost Perfect
	Tightness	19.0%	18.0%	0.675	Substantial

Table 9: Human evaluation of v1 model pointing quality.

Metric	Avg	Majority	Fleiss' κ	Agreement
Correctness	82.7%	87.0%	0.558	Moderate
Comprehensiveness	55.7%	54.0%	0.689	Substantial
Tightness	49.3%	40.0%	0.280	Fair
Appropriateness	87.7%	90.0%	0.599	Moderate

Methodology. Using the same evaluation protocol, we sampled 100 outputs from v1 on the MathVision dataset. Annotators evaluated whether the model’s pointed regions (copied image tokens) correctly corresponded to the referenced objects in the reasoning trace. We added an **Appropriateness** criterion to assess whether the pointing action was contextually justified.

Results. Table 9 demonstrates that v1 maintains high grounding quality during inference, achieving 82.7% correctness; comparable to the training data quality. The high appropriateness score (87.7%) indicates that the model learns to selectively invoke the pointing mechanism when dynamic visual reference is genuinely beneficial for reasoning.

Discussion. The evaluation reveals that our attention-based grounding excels at capturing semantically rich visual references that are challenging for traditional object detectors. The moderate tightness scores across both methods reflect the inherent ambiguity in defining precise boundaries for abstract concepts (e.g., "angle 2" or "the second figure"), where multiple valid interpretations exist. The consistency between training data quality and model performance suggests that v1 successfully learns robust visual grounding capabilities from our automatically generated supervision.

F Bounding-Box Extraction from Cross-Attention

This section provides a high-level pseudocode description of our data annotation method for deriving bounding boxes from cross-attention in Qwen2.5-VL.

G Use of LLMs

This work uses LLMs for data generation (Gemini-2.0-Flash for reasoning trace decomposition) and evaluation (Gemini-2.0-Flash for answer matching), as described in Sections 4.3 and 6.1.

H Additional Qualitative Results

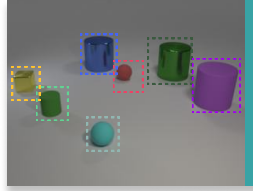
To further illustrate v1’s complex visual reasoning, this section provides additional qualitative examples, complementing Figure 4 from the main text. These examples highlight how v1 leverages the point-and-copy mechanism.

Algorithm 1 Bounding-Box Extraction from Cross-Attention (High-Level)**Input:** Image I , region description T **Output:** Bounding box b corresponding to T **1. Prepare multimodal input.**Concatenate I with a static visual-grounding instruction prompt and feed it to Qwen2.5-VL.**2. Extract attention with instruction.**From the final decoding position, obtain the cross-attention map A over image tokens. Use a predefined set of layers (selected empirically) and average across heads.**3. Extract baseline attention.**Remove the object name from the prompt, feed the modified prompt with I to the model, and extract the corresponding attention map A' using the same layers and averaging.**4. Compute attention contrast.**Compute the contrastive relevance for each image token: $R = A / A'$.**5. Derive bounding region.**Identify the peak region in R . Sweep over multiple candidate crop ratios; for each ratio, form a bounding region around the peak. Select the bounding box maximizing contrast sharpness between inside and outside regions. Convert the selected region to image-coordinate bounding box b .**6. Return.****return** b

Figure 8 demonstrates **v1** on a synthetic task (CLEVR-like) requiring object counting based on the query: “Subtract all red things, then subtract all tiny matte balls. How many objects are left?”. **v1** first localizes objects using its pointer mechanism. It then sequentially reasons, identifying “red” objects before revisiting relevant items, like the “cyan sphere,” to verify the combined “tiny” and “matte” attributes through targeted attention. This process demonstrates **v1**’s capacity for precise attribute grounding and multi-step compositional reasoning enabled by the point-and-copy mechanism.

In Figure 9, **v1** tackles a chart comprehension task: determining if the “Dark Violet” data series has the minimum area under the curve. **v1** initially grounds key chart elements, using its pointer to isolate data series such as “Dark Violet,” “Medium Mint,” and “Dark Cyan.” Later in its reasoning, it proactively revisits these series, performing a comparative analysis of their visual trajectories and relative y-axis values to infer their respective areas. Such selective re-focusing showcases its ability to perform nuanced comparisons within dense visual information.

These examples further affirm that **v1**’s architecture, by supporting active visual reference and precise grounding via its pointing mechanism, achieves robust, interpretable, and accurate multi-step visual reasoning.

**Question:**

Subtract all red things. Subtract all tiny matte balls. How many objects are left?

Answer: 5

Image-Question Pair

Let me see what objects are present. I need to identify all the objects first.

```
detect(query="yellow cube", objects=["<obj1><region>[0.00, 0.00, 0.21, 0.55]</copy_0></copy_1> ... </copy_45></copy_46></region>"])
```

```
detect(query="blue cylinder", objects=["<obj2><region>[0.29, 0.20, 0.46, 0.45]</copy_14></copy_15> ... </copy_48></copy_49></region>"])
```

```
detect(query="green cylinder", objects=["<obj3><region>[0.58, 0.20, 0.75, 0.50]</copy_17></copy_18> ... </copy_50></copy_51></copy_52></region>"])
```

```
detect(query="purple cylinder", objects=["<obj4><region>[0.75, 0.45, 0.96, 0.65]</copy_52></copy_53> ... </copy_64></copy_65></region>"])
```

```
detect(query="small green cylinder", objects=["<obj5><region>[0.12, 0.45, 0.29, 0.75]</copy_45></copy_46> ... </copy_68></copy_69></region>"])
```

```
detect(query="red sphere", objects=["<obj6><region>[0.46, 0.30, 0.54, 0.45]</copy_27></copy_38></copy_49></region>"])
```

```
detect(query="cyan sphere", objects=["<obj7><region>[0.33, 0.65, 0.46, 0.85]</copy_58></copy_59> ... </copy_81></copy_82></region>"])
```

The task is to subtract all red things and all tiny matte balls.

Let me consider the order: first, subtract all red things, then subtract all tiny matte balls.

Step 1: Subtract all red things. There's only one red sphere <obj6>. So, remove that.

Remaining objects: 1. Yellow cube <obj1> 2. Blue cylinder <obj2> 3. Green cylinder <obj3> 4. Purple cylinder <obj4> 5. Small green cylinder <obj5> 6. Cyan sphere <obj7>

Step 2: Subtract all tiny matte balls. Assuming the cyan sphere <obj7> is matte, it should be removed. So, remove the cyan sphere <obj7>.

Remaining objects: 1. Yellow cube <obj1> 2. Blue cylinder <obj2> 3. Green cylinder <obj3> 4. Purple cylinder <obj4> 5. Small green cylinder <obj5>

That's five objects.

Wait, but the problem might be considering only the red sphere <obj6> as matte, and the cyan sphere <obj7> is also matte. But since it's not specified, I'll assume both are matte.

Alternatively, perhaps "tiny matte balls" refer only to the cyan sphere <obj7>, and the red sphere <obj6> is not considered a "ball" or something. But the problem mentions "tiny matte balls," and both spheres are small and matte, so probably both are considered tiny matte balls. Therefore, after subtracting the red sphere <obj6> and the cyan sphere <obj7>, we're left with five objects: yellow cube <obj1>, blue cylinder <obj2>, green cylinder <obj3>, purple cylinder <obj4>, and small green cylinder <obj5>.

But wait, the user's message seems to indicate there are six objects initially, but lists seven, which is confusing. Maybe there's a mistake in counting. Alternatively, perhaps the small green cylinder <obj5> is not present, and there are only six objects: Yellow cube <obj1>, Blue cylinder <obj2>, Red sphere <obj6>, Green cylinder <obj3>, Purple cylinder <obj4>, Cyan sphere <obj7>

In this case, after subtracting the red sphere <obj6> and the cyan sphere <obj7><region>[0.33, 0.65, 0.46, 0.85]</copy_58></copy_59> ... </copy_81></copy_82></region>, we're left with four objects: yellow cube <obj1>, blue cylinder <obj2>, green cylinder <obj3>, purple cylinder <obj4>.

But the user mentioned six objects, including the small green cylinder <obj5>, so I'll stick with that.

Alternatively, maybe the small green cylinder <obj5> is separate from the green cylinder <obj3>, making it seven distinct objects. Regardless, the key is to subtract the red sphere <obj6> and the cyan sphere <obj7>, assuming both are matte.

So, final answer should be 5 or 4, depending on whether the small green cylinder is considered separate or part of the green cylinder. But since the user specified seven objects, including small green cylinder <obj5>, the answer should be 5. **Final Answer** { 5 }

Figure 8: Qualitative example of v1 tackling an attribute-based counting task in a synthetic domain.



Figure 9: Qualitative example of v1 performing comparative reasoning on a chart comprehension task.

Prompt for data generation

You are given text-only reasoning for visual question answering.

Your task is to convert this text-only reasoning into visually grounded reasoning.

STEP-BY-STEP INSTRUCTION

Please follow these instructions step-by-step, imitating human visual reasoning behavior by:

- 1) Start from the beginning of the reasoning and read EACH sentence.
- 2) When you think you'd better look at the object or region, use the detect() function.
- 3) Format: 'detect(query="visual item that you want to find", objects=["<obj#>"])'
- 4) After detection, reference the visual element with '<obj#>' tags every time you need to look at it again immediately after mentioning the item.
- 5) Use NEW object numbers ('<obj1>', '<obj2>', '<obj3>'...) for EACH new detection.

EXAMPLE:

Original text:

"Looking at the graph, I can see the function reaches its maximum at $x = 3$."

Corrected:

""

To answer the question, I need to look the graph.

detect(query="function graph", objects=["<obj1>"])

Looking at the graph <obj1>, I can see the function reaches its maximum at $x = 3$.

""

Later reference:

You can skip the <obj#> tag when you think you do not need to look it again.

""

The slope of the function becomes zero at this point on the graph.

""

KEY REQUIREMENTS:

- Every item in lists MUST have its own 'detect()' statement
- Put 'detect()' statements on their own lines
- NEVER skip any visual element mentioned in the reasoning
- Start object numbering at 'obj1' and increase by 1 for each new object

<OBJ#> REQUIREMENTS

- Visual element should be concrete, distinct, and explicit. Later you will localize the element based on the detect(). So make sure that the element not confusing.
- Use separate tags for each object (write "between the bus <obj1> and the car <obj2>" not "between <obj1 and obj2>").
- GOOD grounding: "I need to analyze this problem. detect(query="triangle", objects=["<obj1>"]) The triangle <obj1> has a right angle at vertex S."
- BAD grounding: "detect(query="triangle and rectangular", objects=["<obj1>"]) in the diagram, there are the triangle and rectangular has a right angle." (referring to non-atomic element)
- BAD grounding: "detect(query="region", objects=["<obj1>"]) The triangle <obj1> has a right angle." (referring to ambiguous element)

After completing the reasoning, list all objects detected:

```
{
  "obj1": {"type": "function_graph",
    "description": "Graph of a function with maximum at  $x = 3$ "},
  "obj2": {"type": "next_item",
    "description": "Description of next item"}
}
```

- We will localize the element with the open-world detector based on the description, so make sure to include well-described full self-contained description enough to uniquely identify the object.

FINAL FORMAT:

```
{
  "reasoning": "Your fully visually-grounded reasoning text",
  "obj_list": "Your JSON object list"
}
```

Now, strictly following the instruction and the example, please provide the object list and visually grounded reasoning for the following prompt and reasoning:

Example

Original Conversation

HUMAN:

[few_shot_question]

GPT:

[few_shot_answer]

Visually Grounded Reasoning

GPT:

[few_shot_reasoning]

Object List:

[few_shot_objects]

Now, given the conversation, please convert GPT's text-only reasoning into visually grounded reasoning

Original Conversation:

HUMAN:

[question]

GPT:

[answer]

Visually Grounded Reasoning:

Table 10: Prompt used for converting textual reasoning to grounded reasoning annotations in **v1g** data generation process.