LEARNING TO LOOK: A FOUNDATION FOR VISUAL UNDERSTANDING

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language models (VLMs) suffer from visual hallucination, such as saying an object that is not present in an image. We ask a more fundamental question: whether a VLM is even aware of where it is looking before understanding the semantic meaning in the image. We formulate this question as a LOOK task: identifying which parts of its visual input were available to the model. We construct the task in a self-supervised way without manually labeling the available region to the model. Our experiments show that existing VLMs fail to localize the visible region. We further show that fine-tuning VLMs on the proposed LOOK task improves a model's awareness of visual input.

1 INTRODUCTION

Vision-language models (VLMs) can describe images fluently while referring to objects that are not actually present. This failure mode, commonly termed visual hallucination, has been widely evidenced (Rohrbach et al., 2018; Li et al., 2023; Chen et al., 2024b) and persists even in frontier models. For example, Asadi et al. (2026) show that Gemini-3-Pro, Gemini-2.5-Pro, Claude-Opus-4.5, and GPT-5.1 can readily produce detailed image descriptions and elaborate reasoning traces even when no image is provided. This observation suggests that visual hallucination can arise prior to visual misinterpretation: the model may fail to reliably distinguish between the presence and absence of visual input. In this sense, current VLMs lack a basic form of visual-input awareness, the ability to recognize whether they are receiving visual information at all. This stands in sharp contrast to human visual perception, where an observer is inherently aware of whether visual input is available, for example, whether their eyes are open or closed.

However, existing efforts to address visual hallucination are limited at the level of semantics. They curate instruction data with negative and counterfactual examples (Liu et al., 2024a), align outputs with factual or human preferences (Sun et al., 2024), or intervene during decoding to suppress content the image does not support (Leng et al., 2024; Huang et al., 2024). Despite their differences, these methods address essentially the same question: whether the model's claims about the visual



Figure 1: **Hierarchy of visual understanding: LOOK → Perceive → Understand.** We introduce a hierarchical visual grounding structure and argue that visual hallucination can arise from any level. This work focuses on the fundamental *Look* level.

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.

scene are semantically correct. We instead ask a more fundamental question that precedes semantic interpretation: *Is a VLM aware of where it is looking in the first place before reasoning about what the visual input means?*

To this end, we argue that general visual grounding can be organized into a three-level hierarchy as shown in Fig. 1: *Look* identifies where the model is looking, *Perceive* determines what is present there, and *Understand* uses this evidence to answer a question or draw an inference. We hypothesize that visual hallucination may arise as early as the first level, *Look*, a capability that remains largely untested by existing benchmarks. A model should ground its claims in available visual input when asserting that it sees something and, more fundamentally, should be aware of whether visual input is available at all.

We therefore propose LOOK, a task designed to probe a VLM's awareness of the visual input available to it. Specifically, the model is given an image crop and must localize the visible region as a bounding box in the coordinate frame of the original image. We encode only the crop pixels, while assigning each crop patch the same positional embedding it would have received in the full image. Thus, although the excluded scene content is unavailable, the model retains all geometric information required to localize the crop. Intuitively, this is analogous to covering one of a person's eyes and asking them to identify which eye is covered. LOOK probes a model's awareness of the visual input available to it, rather than the semantic content of what it sees.

Our experiments primarily use Qwen3-VL, which preserves absolute positional information for image patches, whereas many other VLMs rely only on relative positional embeddings and therefore lack the spatial cues required for the LOOK task. We first show that pretrained VLMs fail to reliably localize the visual input available to them. We then fine-tune Qwen3-VL on automatically constructed LOOK data and find that the model can learn the task and generalize well to unseen examples. To evaluate whether LOOK improves a VLM's awareness of visual input more broadly, we further introduce the Visual Awareness Benchmark (VisABench). In VisABench, visual tokens are replaced with null <image_pad> tokens, and the model is asked whether a target object is present in the image. Models fine-tuned on LOOK substantially outperform the corresponding baselines, suggesting that learning to LOOK enhances a model's awareness of available visual evidence and naturally helps mitigate visual hallucination.

Our contributions are threefold:

(i) We introduce a three-level hierarchy of visual grounding. The hierarchy decomposes visual grounding into *Look*, *Perceive*, and *Understand*, providing a framework for studying visual hallucination at a more fundamental level.

(ii) We propose the LOOK task for improving visual awareness through self-supervision. LOOK training data can be generated from arbitrary image datasets by cropping images, with the crop region itself providing bounding-box supervision and requiring no semantic annotation.

(iii) We introduce VisABench to evaluate a VLM's awareness of visual input. We show that pretrained VLMs can hallucinate object presence even when visual evidence is absent, while fine-tuning on LOOK substantially improves visual awareness and reduces such hallucinations.

2 HIERARCHY OF VISUAL UNDERSTANDING

Visual understanding is commonly evaluated by the success rate on visual question answering (VQA) tasks. However, when a model produces an incorrect answer, it is often difficult to determine where the failure originates. We argue that visual understanding involves multiple stages, and that visual hallucination can arise at any of them. Effectively mitigating visual hallucination therefore requires identifying the stage at which the failure occurs and addressing its underlying cause.

To this end, we propose a three-level hierarchy of visual understanding as shown in Fig. 1. Let I denote an image, w a region of it, I_w the pixels inside w, and P_w the positions those pixels occupy in I. The three levels answer three different questions about the given image input:

$$\underbrace{\hat{w} = L(I_w, P_w)}_{\textit{Look}\text{: did I see and where is it?}} \quad , \qquad \underbrace{\hat{z} = P(I_w)}_{\textit{Perceive}\text{: what is in it?}} \quad , \qquad \underbrace{\hat{a} = U(\hat{z}, q)}_{\textit{Understand}\text{: what follows?}} \quad , \tag{1}$$



Figure 2: **Where does a hallucination originate?** In this example, we ask what animal is grazing in a photograph of a dog on grassland, a model answers "a sheep". The same wrong answer can arise at two levels of the hierarchy. At the *Perceive* level, the model uses the image but misrecognizes the dog. At the *Look* level, the model does not rely on the image at all, and "sheep" follows from the textual context.

where $\hat{z}$ denotes observations and q a question. Particularly, the ordering of the hierarchy expresses the dependence of evidence to conclude the final visual understanding: a claim about what the image contains (P) or implies (U) should presuppose some account of what input was available (L). Consider the limiting case where an input that carries no visual content at all. In this case, the correct output of L is that nothing is visible, and every positive claim of P is a hallucination without grounding (Asadi et al., 2026).

Source of visual hallucination. While existing work largely focuses on semantic-level visual perception and reasoning, our hierarchy highlights the more fundamental role of *Look* in visual grounding. Before interpreting visual content, a model should first be aware of the visual input available to it and distinguish visual evidence from textual context. For example (Fig. 2), if a model hallucinates a *sheep* in an image of grassland containing a *dog*, the failure may arise because it misperceives the dog as a sheep, corresponding to a failure at the *Perceive* level. More fundamentally, however, the model may fail to rely on the image at all and instead generate *sheep* from textual associations with the surrounding context, corresponding to a failure at the *Look* level. Distinguishing these failure modes is essential for identifying where visual hallucination originates and how it should be addressed.

3 The Look Task

Motivated by the proposed hierarchy and the goal of improving visual grounding at the fundamental *Look* level, we design the Look task to quantify a VLM's awareness of its available visual input. We further use the task as a self-supervised training objective to enhance this awareness. A central challenge, however, is to disentangle visual-input awareness from semantic shortcuts that could allow the model to solve the task without actually identifying where it is looking.

3.1 Constructing a Look example

Just as a person knows which part of their surroundings falls within their view, a model should know which part of an image it receives. In the Look task, the full image plays the role of the environment and a crop plays the role of the available view (Fig. 3). We sample a rectangular window aligned with the model's visual-token grid, encode only the pixels inside it, and ask the model for the bounding box of the window in the coordinate frame of the full image. Because we sample the window ourselves, the label can be automatically obtained and any image can be used as Look examples.

However, a crop encoded on its own carries no trace of where it came from. The same pixels could occur anywhere in a larger image, and a model could only guess from semantic cues, such as sky suggesting the top of a scene. We therefore assign each crop token the positional information it would have received in the full image. We implement Look on Qwen3-VL (Qwen Team, 2025),

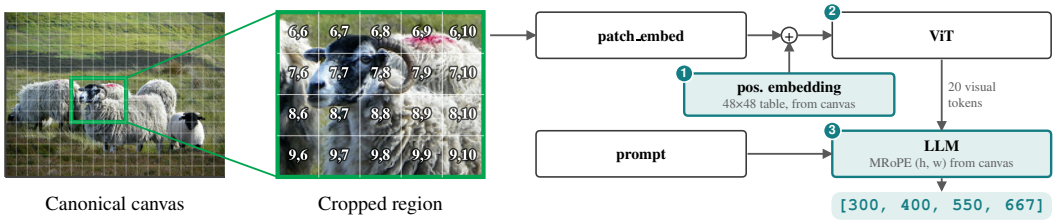


Figure 3: **Constructing a LOOK example.** We select a merge-block-aligned viewport and encode only its pixels for the VLM. Importantly, we restore positional metadata from the full canvas without encoding the excluded image content. The crop's coordinates provide the target automatically. Changed blocks indicate places where token positions are added.

which preserves absolute positional information for image patches and thus provides the spatial cues needed to identify the available region. Specifically, as illustrated in Fig. 3, we preserve the crop's original full-image positions at two points in the architecture: the vision encoder's absolute positional embeddings (1) and the language model's multimodal rotary positional embeddings (MRoPE) (3). In this way, the geometric information required for localization is explicitly available, allowing the model to identify its available visual region without relying on semantic cues to infer its location. The LOOK task therefore directly probes whether the model can recognize where its visual input comes from. Implementation details are provided in App. B.

3.2 TRAINING WITH LOOK DATA

Visual perception verbs as the interface between language and vision. To improve a model's visual awareness, we encourage it to be sensitive when generating expressions that explicitly invoke visual perception, such as `I see` and `I observe`. We view such expressions as a linguistic interface between the language and visual modalities: when a model claims to see something, that claim should be grounded in the visual input available to it. Accordingly, we equip each LOOK example with diverse localization prompts that invoke such visual-perception expressions, as listed in App. E.

Concretely, each LOOK example pairs a localization prompt q_{look} with the bounding box $y(w)$ of the sampled window, represented in the model's coordinate format as defined in Eq. (4). Given an existing instruction-tuning mixture dataset D_{inst} and the generated LOOK dataset D_{look}, we train on their union using the standard autoregressive cross-entropy objective:

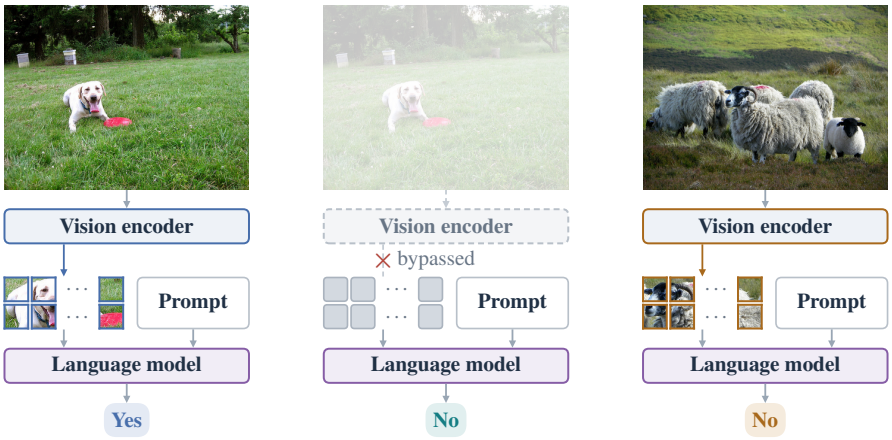
$$\mathcal{L}(\theta) = -\frac{1}{\sum_{(x,q,r)\in D} |r|} \sum_{(x,q,r)\in D} \sum_{t=1}^{|r|} \log p_\theta(r_t \mid x, q, r_{<t}), \qquad D = D_{\text{inst}} \cup D_{\text{look}}, \tag{2}$$

where r is the target response of an example and $|r|$ its length in tokens, so the loss is averaged over all response tokens. For LOOK examples, r is the serialized box $y(w)$ and $x = (I_w, P_w)$ denotes the cropped visual input and its positional information, while ordinary instruction examples retain their original image inputs and responses.

4 VISABENCH: EVALUATING VLMS' ILLUSION

The LOOK task examines where the available visual input lies within a scene. Its limiting case concerns whether any visual input is available at all. A model that is aware of its visual input should recognize when none is present, rather than describe content it has never received, the *illusion* failure shown in Asadi et al. (2026). To evaluate VLMs' illusion, we introduce the Visual Awareness Benchmark (VisABench).

Benchmark construction. VisABench uses all 10,752 referring expressions from the RefCOCO testA and testB splits (Yu et al., 2016), spanning 1,500 COCO images (Lin et al., 2014). For each expression, we ask the model: "*Does this image contain: {expression}? Answer with Yes or No.*" We evaluate three input conditions, illustrated in Fig. 4:



(i) Real image (ii) No image (iii) Wrong image

Figure 4: **Illustration of VisABench construction.** All conditions use the same prompt, "*Does this image contain: dog? Answer with Yes or No.*", with the only difference being the visual input. The grounded answer is Yes in (i) and No in (ii) and (iii).

(i) *Real image.* The expression is paired with its original image, which contains the referred object; the correct answer is therefore Yes.
(ii) *No image.* We bypass the vision encoder while retaining the <|image_pad|> placeholder embedding at each visual-token position. This preserves the sequence length, token positions, and attention mask of the corresponding real-image input while removing all visual content. The correct answer is therefore No.
(iii) *Wrong image.* Each image is paired with the expression of a different item, for which the expected answer is No.

We use the same 10,752 items, the same set of images and referring expressions throughout for the three conditions. The no-image condition serves as our primary test of visual awareness.

Metrics. We score the model from the logits ℓ at the first answer position instead of parsing generated text, since free-form responses can vary substantially. Answers such as "Yes.", "yes, it does", or a longer sentence that never clearly commits to either answer make string-based evaluation brittle and discard information about confidence. We therefore compute

$$p(\text{Yes}) = \sigma\Big(\log \sum_{t \in \mathcal{Y}} e^{\ell_t} - \log \sum_{t \in \mathcal{N}} e^{\ell_t}\Big), \tag{3}$$

where σ is the logistic function and $\mathcal{Y}$ and $\mathcal{N}$ contain every vocabulary token whose surface form, ignoring case and surrounding whitespace, is Yes and No, respectively. We classify an answer as Yes when $p(\text{Yes}) > 0.5$. This readout requires only a single forward pass, avoids parsing failures, and provides a continuous confidence score.

We report the following metrics for each of the three input conditions:

(1) *Yes rate*: the fraction of examples for which the model answers Yes. In the no-image condition it is our primary metric (↓), corresponding to false claims that the referred object is visually present. In the real-image condition (↑) it controls for whether the model still recognizes the referred object when the correct image is provided; a model that always answers No would score perfectly on the no-image Yes rate but fail here. In the wrong-image condition (↓) it gives a coarse measure of whether the model checks the referring expression against the actual image content.
(2) *Mean $p(Yes)$*: the average probability assigned to Yes, with the same direction as the Yes rate in each condition. It captures the model's confidence even when this remains below the decision threshold.
(3) *Gap* (↑): the difference between the real-image and no-image Yes rates, summarizing the model's sensitivity to the presence of visual evidence.

Table 1: **Compare different probe methods for visual awareness.** We report IoU of the pretrained Qwen3-VL when asked for the visible region over 1,000 views. IoU counts answers without a valid box as zero. Methods operating on pixels or patch embeddings fail to probe visual awareness because the model can exploit semantic shortcuts, often predicting the bounding box of the dominant object rather than the available visual region.

Probe methods	Full image	Pixel operation		Patch embedding		Crop	
		Black	Grey	Zero	Mean	Local pos.	Global pos. (LOOK)
IoU ↓	0.52	0.75	0.77	0.58	0.76	**0.18**	**0.18**

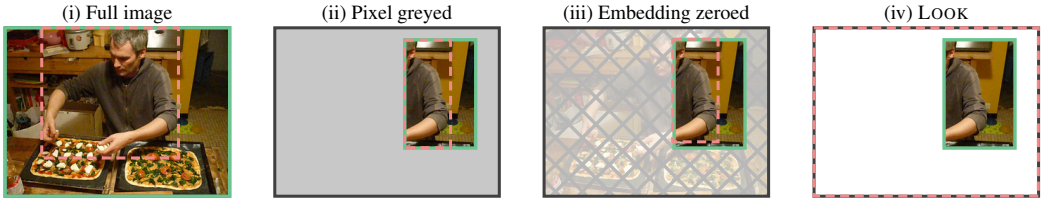


Figure 5: **Four ways to probe visual awareness.** The green box shows the available visual region while the red dashed box shows the prediction. (i) We input the full image, but the model boxes the dominant person rather than the whole canvas. (ii, iii) When the context pixels are greyed out or its embeddings are zeroed, the model still locates the dominant person, likely using semantic shortcuts. (iv) In LOOK, the model answers with the whole canvas.

5 EXPERIMENTS

We design experiments to answer four questions: (i) whether LOOK measures visual awareness without semantic shortcuts, (ii) whether the LOOK task is learnable, (iii) whether LOOK fine-tuning improves VisABench performance, and (iv) whether LOOK fine-tuning preserves the model's general visual perception capabilities.

5.1 SETUP

Training dataset. To improve visual awareness while preserving the VLM's general capabilities, we train on a mixture of replay data and the proposed LOOK data. (i) Replay mixture dataset. We use 46,000 instruction-tuning examples drawn from eight public datasets as detailed in App. A. (ii) LOOK dataset. We construct the LOOK data from the same replay images by sampling one window per image as described in Sec. 3 and pairing each crop with one of the 36 localization prompts listed in App. E. To study the effect of LOOK supervision, we combine the full replay mixture with 10%, 20%, and 100% of the LOOK dataset, denoted as LOOK-to-replay ratios of 1:10, 2:10, and 10:10.

Model training and baselines. We experiment with Qwen3-VL-8B-Instruct (Qwen Team, 2025) since it uses absolute positional embeddings for image patches. We fine-tune the model with LoRA (Hu et al., 2022) of rank 16 and scale 32 on all linear layers, keeping the vision encoder and the projector frozen, and train for two epochs with a learning rate of 10^{-4}, an effective batch size of 32, and a maximum sequence length of 4096, using the objective in Eq. (2). Training is implemented with *ms-swift* (Zhao et al., 2025) on four H200 GPUs. We compare three configurations. (i) The *SFT* baseline is trained on the replay instruction mixture alone. (ii) *Ours* is trained on the mixture together with LOOK data at the three ratios above. (iii) *LocPos* is an ablation that uses the same LOOK data as Ours (10:10) but disables the positional restoration of Sec. 3.1 during training, so that each crop carries crop-local positions. This ablation separates the contribution of global positional information from that of the crop data and the answer format.

5.2 HOW TO MEASURE VISUAL AWARENESS WITHOUT SEMANTIC SHORTCUTS?

We aim to measure a VLM's awareness of its available visual input while preventing it from exploiting semantic shortcuts. We compare four families of probes as illustrated in Fig. 5. (i) Full image.

Table 2: **Performance on LOOK task.** Results are tested on the same 1,000 views of Table 1. Success@0.5 is the fraction of views with an IoU of at least 0.5.

Method	Base	SFT	LocPos	Ours		
				1:10	2:10	10:10
IoU ↑	0.181	0.111	0.371	0.609	0.665	**0.742**
Success@0.5 ↑	0.025	0.012	0.322	0.734	0.808	**0.844**

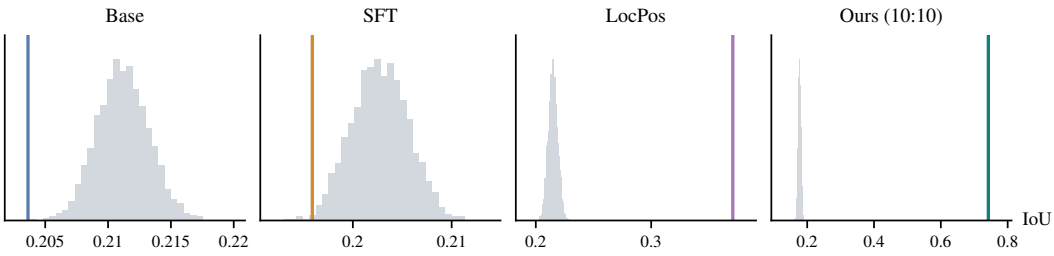


Figure 7: **Permutation test.** The grey bars mean IoU after re-pairing each prediction with the window of another image while the color lines show the IoU on the given image. The pretrained and SFT models show no positive dependence on the view.

We input the full image and ask the model about its visible region. (ii) Pixel operations. We paint the hidden region black or gray. (iii) Patch-embedding operations. We set the embeddings of the hidden patches to zero or to their mean. (iv) Crop based. We encode only the visible region, with either local or global positions used to compute the absolute positional embeddings in Qwen3-VL.

We evaluate each probe on the pretrained model using 1,000 views sampled from the RefCOCO validation images (Yu et al., 2016; Lin et al., 2014). As shown in Table 1, pixel- and patch-embedding-based probes achieve higher IoU, but this performance is largely driven by semantic shortcuts. The model often predicts the extent of dominant objects rather than the available visual region, a behavior also evident in the full-image setting. We therefore adopt cropping with global positional information for the LOOK task, as it removes the hidden content while preserving the geometric information required for localization.

5.3 LEARNING TO LOOK

In this section, we examine whether the LOOK task is learnable. Following the training setup in Sec. 5.1, we compare model performance on the LOOK task. As the results shown in Table 2 and the learning dynamics in Fig. 6, combining replay instruction data with LOOK data substantially improves localization performance. In contrast, LocPos fails to learn the task because it lacks the global positional information used by our method. Notably, even a 1:10 ratio of LOOK data to replay data yields a substantial improvement.

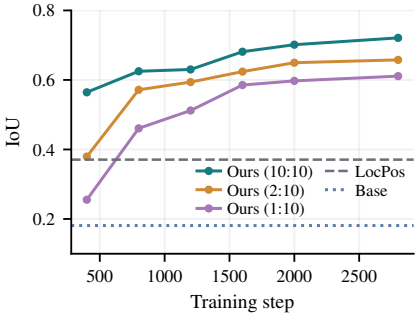


Figure 6: Learning to look dynamics.

Do predictions depend on the given view? To further assess if the model is truly aware of its visual input, we perform a permutation test by pairing each prediction with the target window from another image. As shown in Fig. 7, the Base and SFT models achieve even higher IoU under permutation (grey bars), indicating that their predictions are poorly grounded in the actual visual input.

Table 3: **VisABench.** All 10,752 items, identical for every model. No image: features withheld, correct answer No. Real image: the expression's own image, correct answer Yes. Wrong image: the image of a different item, approximately No. For each condition we report the Yes rate in percent and the mean $p(\text{Yes})$. Gap is the real-image minus the no-image Yes rate. The bottom rows are other pretrained VLMs (Bai et al., 2025; Liu et al., 2024b; Gemma Team, 2026). †The image placeholder embedding of this model is untrained, so it answers neither Yes nor No without an image and the no-image results are not reported.

	No image		Real image		Wrong image		
	Yes (%) ↓	p(Yes) ↓	Yes (%) ↑	p(Yes) ↑	Yes (%) ↓	p(Yes) ↓	Gap ↑
Base	2.5	**0.041**	93.3	**0.927**	15.6	0.158	90.7
SFT	31.2	0.409	94.9	0.915	17.0	0.196	63.7
LocPos	5.7	0.272	92.6	0.890	15.0	0.170	86.9
Ours (1:10)	2.3	0.202	94.1	0.910	15.7	0.183	91.8
Ours (2:10)	4.9	0.241	93.9	0.911	16.0	0.188	89.0
Ours (10:10)	**0.8**	0.101	94.1	0.909	15.1	0.175	**93.2**
Qwen2.5-VL-7B†	–	–	90.0	0.868	12.5	0.129	–
LLaVA-1.5-7B†	–	–	**95.3**	0.775	30.2	0.355	–
Gemma-4-12B	42.9	0.434	88.2	0.880	**12.2**	**0.123**	45.4

Table 4: **LOOK maintains general capabilities.** We report accuracy (%) on MMStar, MMMU and MMMU-Pro and F1 score on POPE. We evaluate 500 items per benchmark.

	MMStar	POPE	MMMU	MMMU-Pro
Base	66.4	88.0	55.1	39.4
SFT	63.6	88.5	56.5	40.8
LocPos	64.4	87.2	54.4	41.0
Ours (1:10)	65.4	88.6	56.0	41.0
Ours (2:10)	65.0	89.3	54.1	38.6
Ours (10:10)	63.8	88.5	56.2	39.2

5.4 DOES LOOK FINE-TUNING IMPROVE VISABENCH?

We next evaluate whether training on LOOK improves visual awareness as measured by VisABench. As shown in Table 3, LOOK fine-tuning consistently improves performance. In particular, Ours (10:10) reduces the no-image Yes rate to 0.8%, compared with 2.5% for the pretrained model and 31.2% for standard SFT, while maintaining a comparable real-image Yes rate. It also achieves the largest real-to-no-image gap of 93.2. These results indicate that LOOK improves the model's awareness of whether visual input is actually available without compromising its ability to recognize objects when an image is present. We note that this failure is also not specific to Qwen3-VL. For example, Gemma-4-12B (Gemma Team, 2026) answers Yes on 42.9% of no-image items.

5.5 DOES LOOK FINE-TUNING PRESERVE GENERAL CAPABILITIES?

Finally, we examine whether LOOK training affects the model's general capabilities. As shown in Table 4, on MMStar (Chen et al., 2024a), POPE (Li et al., 2023), MMMU (Yue et al., 2024), and MMMU-Pro (Yue et al., 2025), all of our models remain within 2.5 points of the SFT baseline. In particular, Ours (1:10), which uses the smallest amount of LOOK data while already substantially reducing hallucination in the absence of visual input, performs on par with or better than SFT across all four benchmarks. We hypothesize that the transfer of visual awareness learned by LOOK to the general VLM benchmarks needs more careful datasets and training design. Also note that the visual hallucination can arise from the semantic perception and understanding levels which need extra techniques to address in addition to the LOOK. We leave this to the future work.

6 Related work

Visual hallucination. Existing benchmarks such as CHAIR (Rohrbach et al., 2018), POPE (Li et al., 2023), ROPE (Chen et al., 2024b), and HallusionBench (Guan et al., 2024) measure different forms of mismatch between visual evidence and generated claims, and methods address such errors through instruction data (Liu et al., 2024a), preference alignment (Sun et al., 2024), or decoding (Leng et al., 2024; Huang et al., 2024). Other studies expose limits of visual representations (Tong et al., 2024), elementary perception (Rahmanzadehgervi et al., 2024), and spatial relations (Kamath et al., 2023). Recently, Asadi et al. (2026) show the illusion even in frontier models where models describe images they were never given. VisABench turns this observation into a quantitative measure, while LOOK targets the underlying capability: knowing what visual input is available before reasoning about what it contains.

Self-supervised methods to mitigate hallucination. Context prediction (Doersch et al., 2015), jigsaw puzzles (Noroozi & Favaro, 2016), rotation prediction (Gidaris et al., 2018), and masked reconstruction (He et al., 2022) derive supervision from transformations of the input image. Recent multimodal post-training methods, such as Visual Jigsaw (Wu et al., 2026) and SSL-R1 (Xie et al., 2026), extend this paradigm to language-model outputs and verifiable rewards. LOOK shares their annotation-free construction, but differs in asking the model to localize its available visual input within the coordinate frame of the original image. As a result, the task is solved through positional information rather than semantic cues from the image content.

7 Limitations

Our work has several limitations. First, our study focuses primarily on a single model family. The current implementation of LOOK requires a variable visual-token grid and access to absolute positional information, so architectures that encode geometry differently may require alternative task designs. Second, VisABench replaces visual features with a constant embedding, which is far from the distribution of natural images. As a result, awareness of missing visual input does not necessarily imply faithful grounding on challenging real images. Finally, visual hallucination may arise at the *Look*, *Perceive*, and *Understand* levels, and developing methods that jointly address failures across all three levels remains an open problem.

8 Conclusion

We study visual hallucination from a more fundamental perspective by asking whether a VLM is aware of the visual input available to it before reasoning about what that input contains. To structure this question, we introduce a three-level hierarchy of visual grounding—*Look*, *Perceive*, and *Understand*—and focus on the first level, which remains largely untested by existing benchmarks. We propose the self-supervised LOOK task, where a model must localize its available visual region using positional information while the excluded image content is never encoded. Our experiments show that pretrained VLMs struggle with this task, whereas fine-tuning on automatically generated LOOK data substantially improves visual awareness with little effect on general VLM performance. We further introduce VisABench to test whether a model recognizes when visual evidence is absent. On VisABench, standard instruction tuning increases the no-image false-Yes rate to 31.2%, while LOOK fine-tuning reduces it to 0.8% without degrading real-image performance. These results suggest that visual hallucination can arise as early as the *Look* level and that explicitly training visual-input awareness provides a simple, annotation-free way to improve visual grounding before semantic perception and reasoning begin.

Reproducibility statement

Sec. 3 specifies the LOOK construction and the positional restoration, Sec. 4 the VisABench construction and scoring, and Sec. 5.1 the training and evaluation protocol. App. A gives the instruction mixture and training recipe, App. E the prompts, and App. F the implementation checks. Accompanying scripts rebuild every table and figure from saved predictions without model inference and record hashes of their inputs.

Ethics statement

The study uses existing research datasets and collects no new human-subject data. Images may depict identifiable people; any release must respect the source datasets' access and licensing terms. Improved awareness of visual input does not guarantee factual or safe outputs, and the models should not be described as having human-like awareness.

AI assistance disclosure

AI tools assisted with manuscript editing, analysis-code preparation, and figure layout.

References

Mohammad Asadi, Jack W. O'Sullivan, Fang Cao, Tahoura Nedaee, Kamyar Rajabalifardi, Fei-Fei Li, Ehsan Adeli, and Euan Ashley. MIRAGE: The illusion of visual understanding. *arXiv preprint arXiv:2603.21687*, 2026.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. Are we on the right way for evaluating large vision-language models? In *Advances in Neural Information Processing Systems*, volume 37, 2024a.

Xuweiyi Chen, Ziqiao Ma, Xuejun Zhang, Sihan Xu, Shengyi Qian, Jianing Yang, David F. Fouhey, and Joyce Y. Chai. Multi-object hallucination in vision language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024b.

Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1422–1430, 2015.

Haodong Duan, Junming Yang, Yuxuan Qiao, Xinyu Fang, Lin Chen, Yuan Liu, Xiaoyi Dong, Yuhang Zang, Pan Zhang, Jiaqi Wang, Dahua Lin, and Kai Chen. 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.

Gemma Team. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026.

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations*, 2018.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6904–6913, 2017.

Tianrui Guan, Fuxiao Liu, Xiyang Wu, Ruiqi Xian, Zongxia Li, Xiaoyu Liu, Xijun Wang, Lichang Chen, Furong Huang, Yaser Yacoob, Dinesh Manocha, and Tianyi Zhou. HallusionBench: An advanced diagnostic suite for entangled language hallucination and visual illusion in large vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14375–14385, 2024.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009, 2022.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13418–13427, 2024.

Amita Kamath, Jack Hessel, and Kai-Wei Chang. What's "up" with vision-language models? investigating their struggle with spatial reasoning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 9161–9175, 2023.

Mehran Kazemi, Hamidreza Alvari, Ankit Anand, Jialin Wu, Xi Chen, and Radu Soricut. GeomVerse: A systematic evaluation of large models for geometric reasoning. *arXiv preprint arXiv:2312.12241*, 2023.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13872–13882, 2024.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne 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, 2014.

Fuxiao Liu, Kevin Lin, Linjie Li, Jianfeng Wang, Yaser Yacoob, and Lijuan Wang. Mitigating hallucination in large multi-modal models via robust instruction tuning. In *International Conference on Learning Representations*, 2024a.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024b.

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

Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. 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.

Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pp. 69–84, 2016.

Qwen Team. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Pooyan Rahmanzadehgervi, Logan Bolton, Mohammad Reza Taesiri, and Anh Totti Nguyen. Vision language models are blind. In *Proceedings of the Asian Conference on Computer Vision*, pp. 18–34, 2024.

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, 2018.

Dustin Schwenk, Apoorv Khandelwal, Christopher Clark, Kenneth Marino, and Roozbeh Mottaghi. A-OKVQA: A benchmark for visual question answering using world knowledge. In *European Conference on Computer Vision*, pp. 146–162, 2022.

Oleksii Sidorov, Ronghang Hu, Marcus Rohrbach, and Amanpreet Singh. TextCaps: A dataset for image captioning with reading comprehension. In *European Conference on Computer Vision*, pp. 742–758, 2020.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards VQA models that can read. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8317–8326, 2019.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Zhiqing Sun, Sheng Shen, Shengcao Cao, Haotian Liu, Chunyuan Li, Yikang Shen, Chuang Gan, Liang-Yan Gui, Yu-Xiong Wang, Yiming Yang, Kurt Keutzer, and Trevor Darrell. Aligning large multimodal models with factually augmented RLHF. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 13088–13110, 2024.

Shengbang Tong, Zhuang Liu, Yuexiang Zhai, Yi Ma, Yann LeCun, and Saining Xie. Eyes wide shut? exploring the visual shortcomings of multimodal LLMs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9568–9578, 2024.

Penghao Wu, Yushan Zhang, Haiwen Diao, Bo Li, Lewei Lu, and Ziwei Liu. Visual jigsaw post-training improves mllms. In *International Conference on Learning Representations*, 2026.

Jiahao Xie, Alessio Tonioni, Nathalie Rauschmayr, Federico Tombari, and Bernt Schiele. SSL-R1: Self-supervised visual reinforcement post-training for multimodal large language models. *arXiv preprint arXiv:2604.20705*, 2026.

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

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. MMMU: A massive multi-discipline multimodal understanding and reasoning benchmark for expert AGI. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9556–9567, 2024.

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

Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, Wenmeng Zhou, and Yingda Chen. SWIFT: A scalable lightweight infrastructure for fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 29733–29735, 2025.

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

A TRAINING DATA AND IMPLEMENTATION DETAILS

Table A1: The instruction mixture replay datasets. LOOK examples reuse its image pool.

Dataset	Rows	Task
VQAv2 (Goyal et al., 2017)	12,000	Visual question answering
TextVQA (Singh et al., 2019)	6,000	Text reading and question answering
Visual7W (Zhu et al., 2016)	6,000	Grounded question answering
A-OKVQA (Schwenk et al., 2022)	5,000	Knowledge-based question answering
ChartQA (Masry et al., 2022)	5,000	Chart reasoning
TextCaps (Sidorov et al., 2020)	4,000	Captioning with text
ScienceQA (Lu et al., 2022)	4,000	Science question answering
GeomVerse (Kazemi et al., 2023)	4,000	Geometric reasoning
Total	46,000	

We train in bfloat16 using scaled dot-product attention and apply LoRA to all linear layers with rank 16 and $\alpha = 32$, while keeping the vision encoder and projector frozen. We use a per-device batch size of 1 with 8 gradient-accumulation steps across four GPUs, giving an effective batch size of 32. Training runs for two epochs with a learning rate of 10^{-4}, a warmup ratio of 0.05, a maximum sequence length of 4096, and a maximum image area of $1{,}003{,}520$ pixels. We reserve 1% of the combined training examples for validation.

B IMPLEMENTATION OF THE LOOK TASK

Sampling the window. The processor resizes image I to a canonical canvas of $R_w \times R_h$ pixels. For patch size p and spatial merge factor m, each visual token corresponds to a merge block of side $c = pm$, so the canvas contains $B_w = R_w/c$ by $B_h = R_h/c$ blocks. We sample a rectangle $w = (b_{x_0}, b_{y_0}, b_{x_1}, b_{y_1})$ on this block grid, with sides between 15% and 80% of the canvas, and serialize the target in the integer coordinate convention of Qwen3-VL:

$$y(w) = 1000 \left(\frac{b_{x_0}}{B_w}, \frac{b_{y_0}}{B_h}, \frac{b_{x_1}}{B_w}, \frac{b_{y_1}}{B_h} \right). \tag{4}$$

Because each window is aligned with the visual-token grid, every token lies entirely inside or outside the selected region, making the target box exact up to coordinate rounding. We resize the full image before cropping and disable any subsequent resizing, so each crop patch is pixel-identical to the corresponding patch in the full image. The resulting crop features nevertheless differ from their full-image counterparts because the excluded patches do not participate in vision-encoder attention. We discard crops smaller than four merge blocks, which would otherwise be upsampled by the image loader. This removes 264 of the $46{,}000$ generated training examples.

Restoring global positions. We override the three positional pathways through input arguments and forward hooks, without introducing new parameters or modifying the attention computation.

(1) **Vision absolute positions.** Qwen3-VL interpolates a learned positional table to the current patch grid. We compute the interpolation indices and weights on the full-image grid and then select the entries corresponding to the crop patches.

(2) **Vision rotary positions.** We offset the crop's two-dimensional patch indices by (mb_{y_0}, mb_{x_0}). Because a shared translation cancels in the rotary query–key inner product (Su et al., 2024), this pathway does not by itself encode absolute location. We nevertheless restore it for consistency.

(3) **Language-model positions.** We shift the height and width components of the image tokens' multimodal rotary positions by (b_{y_0}, b_{x_0}), thereby preserving their spatial placement relative to the text tokens.

In a controlled comparison, overriding the vision absolute positions changes the vision-encoder output by a relative L_2 distance of 0.676, compared with only 0.033 for the rotary-position override. On Qwen2.5-VL (Bai et al., 2025), whose vision encoder uses only rotary positional encoding, restoring the crop positions leaves the output unchanged. This further motivates our requirement for an absolute positional signal in the LOOK task.

C VISABENCH DETAILS

We use the prompt "Does this image contain: {expression}? Answer with Yes or No.", inserting the raw RefCOCO expression. Images are resized to the model's canonical geometry, consistent with our other evaluations. In the no-image condition, the image is still loaded so that the number of visual tokens matches the real-image input, but the vision encoder is bypassed. The feature function instead returns the <|image_pad|> embedding at every visual-token position together with all-zero DeepStack features. For the wrong-image condition, we sort items by image and pair each item with the expression from the item halfway around the list, skipping pairs that originate from the same image. The sets $\mathcal{Y}$ and $\mathcal{N}$ include all vocabulary tokens whose decoded surface forms, after removing surrounding whitespace and ignoring case, are "yes" and "no", respectively. The two sets are disjoint by construction. Each model is evaluated with one forward pass per item and condition, for a total of $32{,}256$ forward passes per model.

D EVALUATION DETAILS

Protocol. MMStar (Chen et al., 2024a), POPE (Li et al., 2023), and MMMU-Pro (Yue et al., 2025) are capped at 500 items; MMMU (Yue et al., 2024) reports the 425 validation items within its cap. The harness is VLMEvalKit (Duan et al., 2024), and POPE reports F1.

Answer extraction. Without an LLM judge, the evaluation harness marks a multiple-choice response as incorrect whenever its answer-extraction rule fails, which disproportionately penalizes verbose outputs. Under the default extraction, the pretrained model obtains an MMMU score of only 0.089, largely because its chain-of-thought responses are not successfully parsed. We therefore re-score all saved outputs using the same fixed sequence of extraction rules for every model: the last explicit answer phrase, a bolded option letter, the harness's option rule applied to the final non-empty line, and a unique match to an option string within the final 300 characters. Responses that remain unresolved are counted as incorrect. For open-ended questions, we match the normalized final answer exactly or as a substring, using a 1% relative tolerance for numerical answers. With this corrected scoring procedure, the pretrained model's MMMU score increases to 0.551. This change reflects improved answer extraction rather than any modification to the model or its outputs.

E PROMPT INVENTORY

The evaluation cycles through 11 groups and samples one prompt within each group. Every prompt contains a format cue and asks for the visible region, not a description of its content. The inventory below is copied from the evaluation code.

Group	Prompt
see	*Which region of the full image can you see? Answer with a bounding box.*
	What part of the image is visible to you? Give a bounding box.
	Identify the area you are able to see; respond with a bounding box.
	Report the visible region as a bounding box.
look	*Where in the full image are you looking? Answer with a bounding box.*
	Which region are you looking at? Reply with box coordinates.
	Point out the region you are looking at, as a bounding box.
	Give the bounding box of the area you are currently looking at.
observe	*Which part of the image are you observing? Answer with a bounding box.*
	Mark the region you can observe; return a bounding box.
	What area is under your observation? Provide a bounding box.
perceive	*Which region of the scene do you perceive? Answer with a bounding box.*
	Delimit the portion of the image you perceive, in bounding-box form.

Group	Prompt
	State, as a bounding box, the region you can perceive.
view	*What is your field of view within the full image? Answer with a bounding box.*
	Which region falls inside your view? Express it as a bounding box.
	Give the bounding box of your viewport.
	Which rectangle of the original image are you viewing? Answer with its bounding box.
examine	*Which region are you able to examine? Answer with a bounding box.*
	Give the bounding box of the area you can inspect.
discern	*Which part of the image can you discern? Answer with a bounding box.*
	What region can you make out? Give its bounding box.
	Which area do you notice? Answer with a bounding box.
	Which portion of the image can you glimpse? Provide a bounding box.
access	*Which region of the image is accessible to you? Answer with a bounding box.*
	What portion of the image is available to you? Give a bounding box.
origin	*This is part of a larger image. Where does it come from? Answer with a bounding box.*
	Locate this view within the full image; return a bounding box.
	Where is this region situated in the original image? Answer with box coordinates.
	Return the bounding box of this view in the full image.
occlusion	*Everything outside one rectangle is hidden from you. Which rectangle is not? Answer with a bounding box.*
	Only one region of the image reaches you. Which one? Give its bounding box.
terse	*Visible region (bounding box):*
	Your viewport, as a bounding box.
	Looking at (bounding box):
	Seen region, as a box:

F VERIFYING THE POSITIONAL RESTORATION

We validate positional restoration independently of downstream performance. Translating the crop patch indices by the grid offset must exactly reproduce the corresponding rows of the full-image position indices. The same row mapping is used to select the absolute interpolation entries, so any indexing error affects both pathways and is detected by this check. We additionally use a painted-patch test to verify the processor's merge-block-major flattening order. After canonical resizing, cropping must leave the retained patch inputs unchanged, and the loader must not upsample the crop. During training, crop offsets must also survive the full dataset-loading pipeline. We therefore log the first active positional-override calls to detect any silent loss of these offsets.

The output parser supports both plain-text bounding boxes and the model's special-token box format. For all model comparisons, we verify that images, prompts, and target windows are identical, recompute IoU directly from the predicted coordinates, and record SHA-256 hashes for every input file. All permutation tests use fixed random seeds.