PINPOINT: TRAINING-FREE REFERRING IMAGE SEGMENTATION VIA PROMPT DISAMBIGUATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Modern referring image segmentation pipelines couple a vision-language model (VLM) for grounding with a promptable segmenter such as the Segment Anything Model (SAM) for mask generation. Prior training-free instances of this recipe often trail fine-tuned and reinforcement-learning (RL)-tuned specialists, and it has been unclear whether the gap comes from the VLM's grounding, SAM's capacity, or the prompt. We show that *prompt ambiguity* is an important bottleneck in this architecture; a VLM-proposed bounding box (bbox) leaves SAM to guess which pixels inside the bbox belong to the object the expression denotes. Interior points are the natural disambiguator, but where they are placed matters; naively sampled points can land on boundaries, distractors, and background clutter, and even *hurt* performance compared to the bbox alone. Supervised and RL-tuned methods improve spatial prompts through task-specific training; we show that informative point prompts can instead be constructed without such training. At a matched budget of up to five interior points and on the same frozen stack, our selector improves cumulative Intersection-over-Union (cIoU) by 12–18 points over uniform sampling across RefCOCO(+/g) splits. We introduce **PinPoint**, a deterministic, training-free point selector that fuses four visual cues into a consensus map, selects compact, spatially diverse points away from boundaries, and uses the frozen VLM to label each point and filter low-confidence labels. Without any task-specific training, PinPoint is competitive with supervised and RL-tuned specialists while issuing only two VLM calls per query.

1 INTRODUCTION

Referring image segmentation generates a pixel-level binary mask of the object that a free-form natural-language expression denotes (Kazemzadeh et al., 2014; Yu et al., 2016; Mao et al., 2016). Unlike traditional semantic or instance segmentation, which depend on a predefined label set (He et al., 2017; Cheng et al., 2022; Ding et al., 2025), referring segmentation must handle compositional, attribute-rich expressions such as *"the partially hidden zebra"* or *"the white pot in the upper right corner"*. This setting better reflects how humans describe objects in real-world use cases such as instruction-following agents (Hong et al., 2024) and image-editing tools (Brooks et al., 2023; Yu et al., 2023), while imposing higher demands on the model: nuanced grounding of language to image regions, fine-grained visual perception, and precise pixel-level mask generation. Modern systems for this task have converged on a modular recipe that pairs two pretrained foundation models: a vision-language model (VLM) (Liu et al., 2023b; Bai et al., 2025b;a) for semantic grounding, and a promptable segmenter such as the Segment Anything Model (SAM) (Kirillov et al., 2023; Ravi et al., 2024) for pixel-level mask generation (Lai et al., 2024; Chen et al., 2024; Xia et al., 2024; Liu et al., 2025b; Huang et al., 2026; Du et al., 2026).

This recipe is appealing because it reuses two strong off-the-shelf foundation models. However, the interface between them is fragile; prior training-free instances often remain behind their fine-tuned and RL-tuned counterparts, and it has not been clear whether the gap lies in the VLM's grounding, in SAM's capacity, or in the prompt that connects them. We argue that *prompt ambiguity* is an important bottleneck that can be addressed training free; a VLM-predicted bounding box is an under-specified region that contains the target alongside partial occluders, distractors, boundaries, and background clutter (Liu et al., 2023a). This leaves SAM to guess which pixels inside it belong to the referent. SAM, in turn, is the least stable near object boundaries (Zhang et al., 2025; Fan et al., 2025;

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

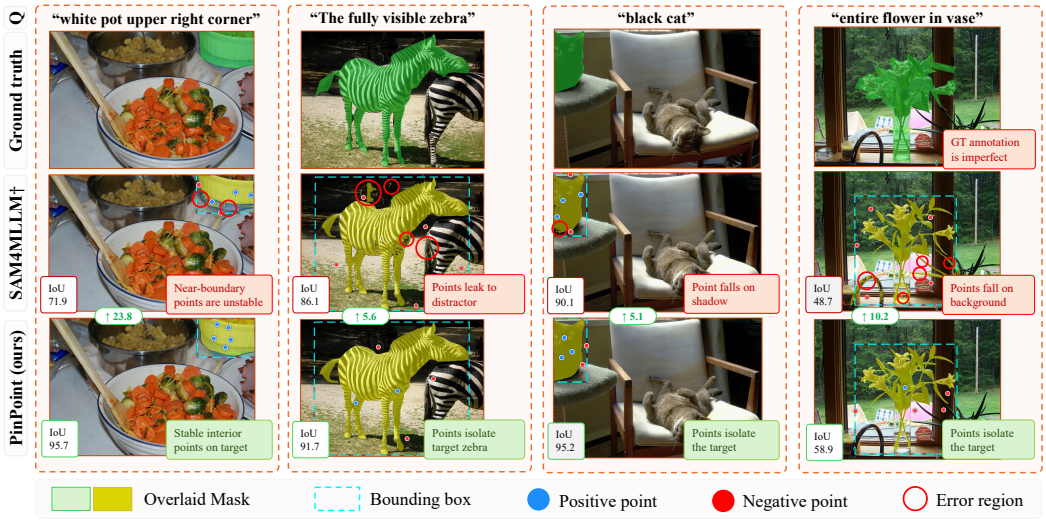


Figure 1: **Informative point prompts improve training-free referring segmentation.** For each query (columns), we show the ground truth (top), SAM4MLLM† (middle), and **PinPoint** (bottom). All methods use the same frozen stack and budget. IoU is per-example mask overlap (higher is better). † denotes our training-free reproduction of SAM4MLLM's PQPP Chen et al. (2024) setting.

Wang et al., 2025a), and its mask predictions degrade when prompted with redundant or off-target points (Fan et al., 2025; Dai et al., 2023); this makes *the choice of interior points* a critical question. Figure 1 illustrate this sensitivity: at a matched point budget, naïve interior sampling places points on boundaries and nearby distractors, while PinPoint selects stable points that isolate the target.

Existing work closes this gap through additional training, falling into three categories. The first adds segmentation-aware tokens to the VLM so that it can emit mask-shaped outputs (Lai et al., 2024; Ren et al., 2024; Yan et al., 2024; Xia et al., 2024; Pi et al., 2024; Yuan et al., 2025; Liu et al., 2025a); auxiliary threads layer chain-of-thought, decomposition, or mask-feedback loops on top (Kao et al., 2025b; Wang & Zhang, 2025; Wang et al., 2024; Kao et al., 2025a). The second fine-tunes the VLM to produce SAM-compatible point prompts directly (Chen et al., 2024; Dai & Yang, 2024). The third replaces supervision with reinforcement learning, post-training a VLM with GRPO on a few thousand curated samples to emit better single-shot points or refine them iteratively (Liu et al., 2025b; Huang et al., 2026; Du et al., 2026; Zhu et al., 2026). Across all three families, the objective is to produce a better spatial prompt for SAM. A simpler question has not been answered: do these informative points actually require learning, or can they be proposed without any training?

We argue that the points do not need to be *generated* by a trained policy; they can be *selected* from image-side evidence and *verified* by the frozen VLM. Classical visual cues already encode where reliable evidence sits inside a coarse box, and a frozen VLM is a competent per-point verifier. In this view, informative point prompts can be constructed without task-specific training.

We turn this idea into **PinPoint** (*Prompting with Informative Interior Points*), a training-free select–verify–filter pipeline that converts a coarse VLM bounding box into an informative composite prompt comprising the bounding box and a compact set of semantically labeled interior points. Inside each predicted bbox, PinPoint fuses visual cues into a consensus map, extracts a compact and spatially diverse interior point set, and reuses the frozen VLM to label each point as positive or negative; SAM is then run once with the prompt; the entire pipeline costs only two VLM calls per query.

At a matched point budget across RefCOCO(+/g), PinPoint improves cIoU by 12–18 points over uniform sampling, using identical VLM-predicted boxes and frozen backbones (Table 1). System-level comparisons show competitive performance against recent supervised and RL-tuned specialists without task-specific parameter updates (Section 4). Together, these results identify *prompt ambiguity* as an important bottleneck after localization and show that informative point prompts can be constructed through deterministic image-side selection and frozen-VLM verification, without task-specific point-policy training.

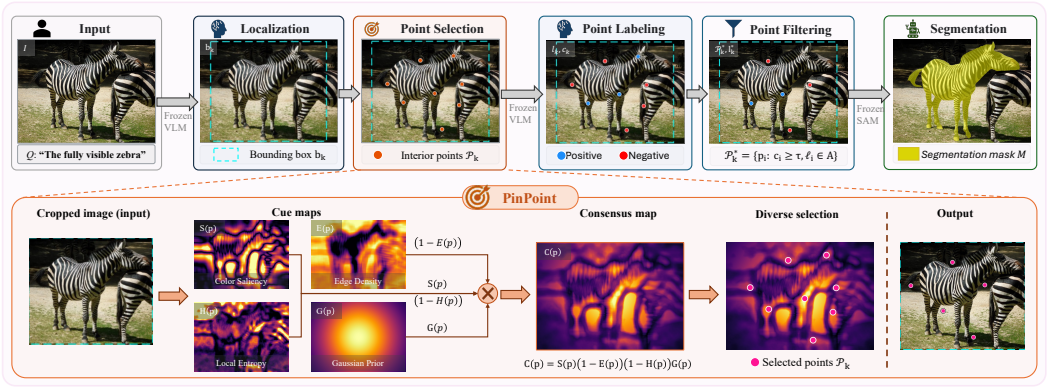


Figure 2: **Training-free referring segmentation with PinPoint.** A frozen VLM localizes the referent with candidate bboxes; PinPoint selects informative interior points inside each bbox; the same frozen VLM labels each point as positive or negative; confidence filtering removes unreliable labels; SAM produces the final mask in a single forward pass. The lower inset shows the PinPoint selector: cue maps computed on a crop, fused into a consensus map, and converted into a diverse interior point set.

2 Related Work

Supervised fine-tuning. A large body of work equips VLMs with segmentation-specific output interfaces, typically by adding mask-producing heads or segmentation tokens that decode into pixel masks (Lai et al., 2024; Ren et al., 2024; Yan et al., 2024; Xia et al., 2024; Pi et al., 2024; Yuan et al., 2025; Liu et al., 2025a). A closely related approach is SAM4MLLM's PQPP variant (Chen et al., 2024), which uniformly samples a grid of points inside the predicted box and then uses a fine-tuned VLM to label them. Its reported RefCOCO-family comparisons favor this *select-then-label* strategy over direct point generation. Related weakly supervised work, PPT (Dai & Yang, 2024), trains a point generator through a curriculum, including positive and negative prompts, while keeping CLIP and SAM frozen. PinPoint reuses select-then-label *without any task-specific tuning*.

Reinforcement learning for prompt policies. A more recent line uses reinforcement learning to improve VLM-based grounding and segmentation. Seg-Zero trains the VLM to emit bounding boxes and points for a frozen segmenter (Liu et al., 2025b), while SAM-R1 derives rewards directly from the resulting SAM masks (Huang et al., 2026). SAM-Veteran extends this design to an iterative agent that proposes refinement points and learns when to terminate the refinement loop (Du et al., 2026). LENS instead combines a learned context module with joint optimization of the VLM and segmentation decoder (Zhu et al., 2026). PinPoint *avoids task-specific point-policy training* by deterministically selecting informative points from image-side evidence and reusing the frozen VLM to label them.

Training-free prompt disambiguation. SAMAug (Dai et al., 2023) selects additional point prompts using saliency, entropy, or distance criteria conditioned on an initial SAM mask. LoTeR (Zhang et al., 2026) uses a language-conditioned patch-score map to guide SAM proposals. IteRPrimE refines Grad-CAM heatmaps for image–text matching (Wang et al., 2025b). Complementary methods explore chain-of-thought and attribute reasoning for images (Kao et al., 2025b; Wang & Zhang, 2025). PinPoint separates deterministic image-side candidate selection within VLM-predicted boxes from query-dependent frozen-VLM verification and confidence filtering, constructing the point prompt before SAM mask generation.

3 Method

PinPoint disambiguates the prompt passed to SAM *before* mask generation. Each VLM-predicted bounding box is converted into a composite prompt of *bounding box (bbox) + a small set of semantically labeled interior points*; with all backbones frozen and the point budget matched, the choice of these interior points has a large effect on cIoU (Section 4).

3.1 PROBLEM SETUP

We are given an image $I \in \mathbb{R}^{H\times W\times 3}$ and a natural-language query Q that may specify zero, one, or multiple visible target instances. The goal is a binary mask $M \in \{0,1\}^{H\times W}$ whose positive pixels correspond to the object(s) denoted by Q (or the all-zero mask when there is no visible referent). We construct M through a five-stage pipeline (Figure 2):

$$(I,Q) \xrightarrow{\text{Localize}} \mathcal{B} \xrightarrow{\text{PinPoint}} \{(b_k,\mathcal{P}_k)\} \xrightarrow{\text{Label}} \{(b_k,\mathcal{P}_k,\boldsymbol{\ell}_k,\mathbf{c}_k)\} \xrightarrow{\text{Filter}} \{(b_k,\mathcal{P}_k^\star,\boldsymbol{\ell}_k^\star)\} \xrightarrow{\text{SAM}} M. \quad (1)$$

Localization turns (I,Q) into candidate bboxes $\mathcal{B} = \{b_1,\dots,b_K\}$. PinPoint extracts a small set of interior points $\mathcal{P}_k$ inside each b_k. *Point labeling* attaches a label $\ell_{k,i} \in \{\texttt{positive},\texttt{negative}\}$ and confidence $c_{k,i} \in [0,1]$ to every $p_{k,i} \in \mathcal{P}_k$ via the same VLM; we collect these into the per-bbox tuples $\boldsymbol{\ell}_k = (\ell_{k,i})_{p_{k,i}\in\mathcal{P}_k}$ and $\mathbf{c}_k = (c_{k,i})_{p_{k,i}\in\mathcal{P}_k}$. *Filtering* keeps the subset $\mathcal{P}_k^\star \subseteq \mathcal{P}_k$ that passes a confidence threshold and a label allowlist, with $\boldsymbol{\ell}_k^\star$ the corresponding labels restricted to $\mathcal{P}_k^\star$. *Segmentation* hands $(b_k,\mathcal{P}_k^\star,\boldsymbol{\ell}_k^\star)$ to SAM in a single forward pass, and the per-bbox masks are combined into M.

3.2 LOCALIZATION

The first stage uses the frozen VLM as a grounded proposer that turns (I,Q) into a set of bboxes $\mathcal{B} = \{b_1,\dots,b_K\}$ in pixel coordinates, where K may be 0 (no detected referent), 1, or larger. We use grammar-guided decoding (Kwon et al., 2023) to constrain the VLM's output to valid JSON to avoid fragile text post-processing; the prompt and EBNF grammar are in Appendix C. We only require this stage to provide *coarse* spatial support; a predicted bbox may include background clutter or overlap nearby instances, and therefore remain an under-specified prompt for SAM.

3.3 POINT SELECTION VIA PINPOINT

PinPoint is applied inside each predicted bbox to convert this coarse signal into an informative prompt. A useful point prompt should identify stable visual evidence inside the localized region while avoiding regions where SAM and the VLM verifier are unreliable; SAM is least stable near object boundaries (Zhang et al., 2025) and its mask predictions degrade when prompted with redundant or off-target points (Fan et al., 2025; Dai et al., 2023). Given a bbox b and a budget N, PinPoint computes four pixel-wise cues over an extended crop, fuses them into a consensus score, and greedily selects up to N spatially diverse points inside b.

Cues. We score each pixel using four complementary cues (full formulas in Appendix A): *color-contrast saliency* S, the ℓ_2 distance between mean CIELAB colors in a local disk and an annulus around p (Itti et al., 1998); *edge density* E, a box-averaged Sobel gradient magnitude flagging boundary-adjacent pixels; *local Shannon entropy* H over quantized gradient magnitudes, identifying textural clutter; and a *Gaussian spatial prior* G centered at the bbox centroid with bandwidth proportional to the bbox diagonal. Cues are computed over an extended crop b_{ext} obtained by extending b by a factor α. Each raw map is rescaled to $[0,1]$ via a percentile-clipped min–max normalizer $\mathcal{N}(\cdot)$ (Eq. 12 in the appendix); we reuse S,E,H,G for the normalized cues. The cues are classical signals from the saliency literature (Harel et al., 2006; Hou & Zhang, 2007; Judd et al., 2009); PinPoint's contribution is how they are composed: inside each VLM-proposed bbox, coupled to a downstream verifier, with a budget tied to cue agreement through Soft-NMS (Bodla et al., 2017) so the selector can return fewer than N points when reliable evidence is sparse.

Fusion. We combine the normalized cues into a consensus score map C using one of two fusion modes. The *multiplicative consensus* variant requires all cues to agree:

$$C_{\text{mult}} \;=\; S \;\times\; (1-E) \;\times\; (1-H) \;\times\; G, \quad (2)$$

so that a pixel scores high only if it is visually distinctive, away from strong boundaries, locally homogeneous, and spatially plausible; a weak value in any cue suppresses the score, making this variant conservative in cluttered scenes. The *max-saliency-smooth* variant keeps saliency as the dominant signal and uses edge density and entropy as smoothness guards:

$$C_{\text{msm}} \;=\; S \;\times\; \max\big(1-\lambda_E E-\lambda_H H,\; 0\big) \;\times\; G, \quad (3)$$

Algorithm 1 PinPoint point selection inside a single bounding box. Defaults for all hyperparameters are listed in Appendix B.

Input: image I, bounding box b, budget N, fusion mode $m \in \{\text{mult}, \text{msm}\}$
Output: selected points $\mathcal{P} \subset \text{interior}(b)$ with $|\mathcal{P}| \leq N$
$b_{\text{ext}} \leftarrow \text{Extend}(b, \alpha)$
for each cue $X \in \{S, E, H, G\}$ **do**
    compute raw cue map X_{raw} over b_{ext} ▷ Eqs. 8–11
    $X \leftarrow \mathcal{N}(X_{\text{raw}})$ ▷ Eq. 12
end for
$C \leftarrow \text{Fuse}(S, E, H, G, m)$ ▷ Eq. 2 or Eq. 3
$\widetilde{\mathcal{P}} \leftarrow \text{LocalMaxima}(C) \cap \text{interior}(b)$
$\mathcal{P} \leftarrow \emptyset$
while $|\mathcal{P}| < N$ **and** $\widetilde{\mathcal{P}} \setminus \mathcal{P} \neq \emptyset$ **do**
    compute $\widetilde{C}_{\mathcal{P}}(p)$ for all $p \in \widetilde{\mathcal{P}} \setminus \mathcal{P}$ ▷ Eq. 13; $\widetilde{C}_{\emptyset}(p) = C(p)$
    if $\max_{p \in \widetilde{\mathcal{P}} \setminus \mathcal{P}} \widetilde{C}_{\mathcal{P}}(p) \leq \varepsilon$ **then**
        break
    end if
    $p^{\star} \leftarrow \arg\max_{p \in \widetilde{\mathcal{P}} \setminus \mathcal{P}} \widetilde{C}_{\mathcal{P}}(p)$
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{p^{\star}\}$
end while
return $\mathcal{P}$

where $\lambda_E, \lambda_H \geq 0$ control the strength of the edge and entropy penalties; textured but salient interior regions can still receive high scores while pixels with strong boundary evidence or heavy clutter are zeroed out by the $\max(\cdot, 0)$ rectifier. Both variants share the factored form $C = (\text{visual evidence}) \times G$ but differ in how strongly the smoothness cues gate the saliency signal: C_{mult} enforces strict conjunction, while C_{msm} shows that the strict-consensus constraint can be loosened without loss.

Candidate extraction and diversity selection. We extract local maxima of C as the *candidate pool* $\widetilde{\mathcal{P}} \subset \text{interior}(b)$ with initial scores $C(p)$ (cue maps live on the extended crop, but candidates are restricted to the original bbox), and grow a *selected set* $\mathcal{P}$ greedily from $\widetilde{\mathcal{P}}$. To avoid redundant prompts, the next point is chosen by a Soft-NMS-style (Bodla et al., 2017) virtual score $\widetilde{C}_{\mathcal{P}}(p)$ that multiplies $C(p)$ by a Gaussian penalty in distance (with bandwidth σ_{nms}) to the nearest point already in $\mathcal{P}$ (Eq. 13 in Appendix A). The first selection reduces to $\arg\max_p C(p)$; after each selection, virtual scores are updated, so nearby candidates are suppressed while distant high-scoring candidates remain competitive. The procedure stops in three cases: N points have been selected, no candidates remain, or every remaining virtual score has fallen below the floor ε. PinPoint can therefore return fewer than N points rather than padding with weak or redundant ones. PinPoint adds no VLM calls beyond localization and labeling, and its CPU cost is dominated by the surrounding VLM and SAM forward passes (Section 4.3).

3.4 Point labeling

PinPoint's points are geometrically interior but carry no semantic understanding of the target. We use the same frozen VLM as a per-point *semantic verifier*: for each candidate $p_{k,i} \in \mathcal{P}_k$, the VLM receives $(I, Q, b_k, p_{k,i})$ and returns a label and a verbalized confidence,

$$(\ell_{k,i},\, c_{k,i}) \;=\; g_{\text{ver}}\big(I,\, Q,\, b_k,\, p_{k,i}\big), \qquad \ell_{k,i} \in \{\texttt{positive}, \texttt{negative}\},\; c_{k,i} \in [0,1], \tag{4}$$

where $\ell_{k,i} = \texttt{positive}$ iff $p_{k,i}$ lies on the target described by Q; a point on a nearby distractor is therefore `negative` even when it lies on a foreground object. In implementation we batch all selected points from an image into a single grammar-guided generation call, reducing labeling to one VLM call per image regardless of N; for $K{>}1$ predicted bboxes we issue one prompt per bbox and dispatch them as a single vLLM batch, so per-image latency stays bounded by one VLM round-trip independent of K (full prompt and EBNF grammar in Appendix D). Verification helps when a point falls on a boundary, background, or distractor; it can also misfire on target-interior

points, and the reported confidence is a noisy self-report rather than a calibrated probability (Groot & Valdenegro-Toro, 2024; Zhang et al., 2024), so labeling complements rather than replaces the proactive boundary avoidance built into PinPoint (quantified in Section 4.3).

3.5 Filtering and segmentation

Point filtering. Before prompting SAM we filter the labeled points using a confidence threshold τ and a label allowlist $\mathcal{A} \subseteq \{\texttt{positive}, \texttt{negative}\}$:

$$\mathcal{P}_k^\star \;=\; \{\, p_{k,i} \in \mathcal{P}_k : c_{k,i} \geq \tau \;\wedge\; \ell_{k,i} \in \mathcal{A} \,\}, \qquad \boldsymbol{\ell}_k^\star \;=\; (\ell_{k,i})_{p_{k,i} \in \mathcal{P}_k^\star}. \tag{5}$$

$\mathcal{A} = \{\texttt{positive}\}$ gives a positive-only prompt; $\mathcal{A} = \{\texttt{positive}, \texttt{negative}\}$ passes both target-supporting and target-suppressing points; τ=0 disables confidence filtering. Unless otherwise stated we use τ=0.7 and $\mathcal{A}$=$\{\texttt{positive}, \texttt{negative}\}$.

SAM prompting and mask merging. For each retained bbox we run SAM once,

$$M_k \;=\; \mathrm{SAM}\big(I,\, b_k,\, \mathcal{P}_k^\star,\, \boldsymbol{\ell}_k^\star\big), \qquad M_k \in \{0,1\}^{H \times W}, \tag{6}$$

and combine per-bbox masks pixelwise:

$$M \;=\; \bigvee_{k=1}^{K} M_k. \tag{7}$$

When the VLM returns no bboxes (K=0), we set $M = \mathbf{0}_{H \times W}$, which contributes zero intersection and full ground-truth area to the cIoU denominator.

4 Experiments

4.1 Experimental setup

Implementation details. Unless otherwise stated, experiments use Qwen2.5-VL-7B (Bai et al., 2025b) (served via vLLM (Kwon et al., 2023)) as the frozen VLM and SAM 2-L (Ravi et al., 2024) as the frozen segmenter[1], and report both PinPoint fusion modes: PinPoint$_{\text{mult}}$ (multiplicative consensus, Eq. 2) and PinPoint$_{\text{msm}}$ (max-saliency-smooth, Eq. 3). Both VLM stages use grammar-guided structured decoding with $Temperature$=0.1 and a fixed seed, so PinPoint's end-to-end output is stable in practice; the matched-budget naive baseline (Table 1) involves stochastic sampling and is averaged over three seeds. All inference uses a single NVIDIA A6000 GPU at engine batch size 1. Verbatim prompts and EBNF grammars for the localization and labeling stages are reproduced in Appendices C–D; full hyperparameters are in Appendix B.

Datasets and Evaluation metrics. We evaluate on both referring and reasoning segmentation benchmarks. For referring segmentation, we use three standard benchmarks: **RefCOCO**, **RefCOCO+** (Kazemzadeh et al., 2014; Yu et al., 2016), and **RefCOCOg** (Mao et al., 2016). RefCOCO/+/g are built from MSCOCO images with natural-language referring expressions; RefCOCO+ excludes spatial-relation descriptions, while RefCOCOg contains longer and more complex expressions. For reasoning segmentation, we evaluate on **ReasonSeg** (Lai et al., 2024), where the model must segment the target specified by an implicit or reasoning-intensive instruction rather than an explicit referring expression. Following common practice, we report cumulative Intersection-over-Union (cIoU) on all benchmarks, and additionally report the per-sample averaged IoU (gIoU) on ReasonSeg[2]:

$$\mathrm{cIoU} = \frac{\sum_{i=1}^{N} |P_i \cap G_i|}{\sum_{i=1}^{N} |P_i \cup G_i|}, \qquad \mathrm{gIoU} = \frac{1}{N} \sum_{i=1}^{N} \frac{|P_i \cap G_i|}{|P_i \cup G_i|}.$$

Here, P_i is the predicted binary mask, G_i is the ground-truth mask, and N is the number of samples.

4.2 Comparisons

We separate controlled prompt comparisons in Table 1, which holds localization and the frozen backbones fixed to assess prompt construction, from system-level comparisons across training

[1]We refer to SAM 2-L simply as "SAM" in prose throughout for readability.

[2]We follow the LISA/ReasonSeg convention in calling the per-sample averaged IoU "gIoU"; the same quantity is sometimes denoted mIoU elsewhere in the segmentation literature.

Table 1: **Informative points close the prompt-ambiguity gap.** cIoU (↑) on RefCOCO(+/g) using identical stack and fixed budget. SAM4MLLM† applies PQPP-style uniform sampling within our frozen pipeline, rather than using the trained SAM4MLLM system reported in Table 2. Δ reports gains over this control. **Best** and second-best results are marked per column.

Method	RefCOCO			RefCOCO+			RefCOCOg	
	val	testA	testB	val	testA	testB	val(U)	test(U)
Bounding box only	69.2	72.0	63.9	60.7	68.7	56.9	65.1	67.4
SAM4MLLM†	61.9	65.5	57.6	53.2	59.6	46.9	55.9	57.1
PINPOINT$_{mult}$	**75.9**	81.7	71.1	**71.4**	76.9	62.5	72.3	69.0
PINPOINT$_{msm}$	75.4	**81.9**	**72.4**	71.3	**77.1**	**62.8**	**73.6**	**71.2**
Δ_{mult} *vs. SAM4MLLM*†	+14.0	+16.2	+13.5	+18.2	+17.3	+15.6	+16.4	+11.9
Δ_{msm} *vs. SAM4MLLM*†	+13.5	+16.4	+14.8	+18.1	+17.5	+15.9	+17.7	+14.1

Table 2: **Comparison across training regimes.** cIoU (↑) on RefCOCO(+/g) and gIoU/cIoU (↑) on ReasonSeg val split. **Best** and second-best results among the listed training-free methods are marked per metric and split. The RL block shares our Qwen2.5-VL-7B + SAM 2-L stack. *Empirical Upper Bound* uses ground-truth boxes and point labels. Baseline scores are from the cited papers; backbones and inference procedures vary. Absent cells (—) were not reported in the original paper.

Method (↑)	RefCOCO			RefCOCO+			RefCOCOg		ReasonSeg val	
	val	testA	testB	val	testA	testB	val(U)	test(U)	gIoU	cIoU
Oracle reference										
Empirical Upper Bound	*91.2*	*91.2*	*92.0*	*91.2*	*91.2*	*92.1*	*90.9*	*90.1*	*90.3*	*91.0*
Supervised Fine-tuning										
LISA-7B (Lai et al., 2024)	74.9	79.1	72.3	65.1	70.8	58.1	67.9	70.6	53.6	52.3
PixelLM-7B (Ren et al., 2024)	73.0	76.5	68.2	66.3	71.7	58.3	69.3	70.5	—	—
VISA-7B (Yan et al., 2024)	72.4	75.5	68.1	59.8	64.8	53.1	65.5	66.4	—	—
PerceptionGPT-7B (Pi et al., 2024)	—	78.6	—	—	73.9	—	—	71.7	—	—
SAM4MLLM (Chen et al., 2024)	77.1	80.9	72.5	71.5	76.8	64.7	74.5	75.2	46.7	48.1
Reinforcement-Learning Tuning										
Seg-Zero-7B (Liu et al., 2025b)	—	80.3	—	—	76.2	—	—	72.6	62.6	62.0
SAM-R1-7B (Huang et al., 2026)	—	79.2	—	—	74.7	—	—	73.1	64.0	55.8
SAM-Veteran-7B (Du et al., 2026)	—	80.8	—	—	76.6	—	—	73.4	68.2	67.3
Training-Free										
IteRPrimE (Wang et al., 2025b)	40.2	46.5	33.9	44.2	51.6	35.3	46.0	45.8	-	-
LoTeR (Zhang et al., 2026)	38.8	48.4	30.3	40.7	50.2	32.0	40.3	40.5	-	-
SAM3 agent (Carion et al., 2026)	59.4	64.3	55.0	51.4	57.0	44.9	57.2	58.8	62.2	49.1
PinPoint$_{mult}$ (ours)	**75.9**	81.7	71.1	**71.4**	76.9	62.5	72.3	69.0	63.1	**58.9**
PinPoint$_{msm}$ (ours)	75.4	**81.9**	**72.4**	71.3	**77.1**	**62.8**	**73.6**	**71.2**	**63.3**	58.8

regimes in Table 2, which compares PinPoint with published training-free, supervised, and RL-tuned systems. Among training-free methods, both PinPoint variants substantially outperform naive point sampling at the same budget, with the dedicated Δ rows showing improvements of $+12$ to $+18$ cIoU across all eight RefCOCO(+/g) splits. The only thing that differs between the SAM4MLLM† row and the PinPoint rows is *which* interior points are fed to SAM; the backbones, the bounding boxes, the budget, and the rest of the pipeline are identical, so the Δ rows cleanly isolate the contribution of prompt quality. Compared with training-required methods, PinPoint is on par with fine-tuned and RL-tuned baselines without any task-specific training; it matches or exceeds fine-tuned baselines like LISA-7B and PixelLM-7B across most splits, and on RefCOCO `testA` reaches 81.9 cIoU, edging past all listed baselines including the SFT entry SAM4MLLM and the reinforcement-learning entries Seg-Zero and SAM-R1. RL-tuned methods use the same Qwen2.5-VL-7B + frozen SAM2 stack as PinPoint plus a few thousand GRPO training samples; PinPoint matches or beats Seg-Zero and SAM-R1 on both `testA` splits with no task-specific updates whatsoever, *which indicates that*

competitive performance is achievable on this stack without task-specific post-training. On the ReasonSeg val, PinPoint surpasses training-free and supervised methods.

4.3 Ablation study

All ablations run on RefCOCO val with Qwen2.5-VL-7B + SAM 2. The detailed end-to-end ablation table is reported in Appendix E; rows differ only in the point selector and the verifier configuration, with every other stage held fixed. Below, we summarize our key findings.

Hyperparameter sensitivity. We sweep the three scalar hyperparameter classes whose effect on cIoU is non-trivial: the Gaussian-prior bandwidth σ_G, the edge/entropy penalty weights (λ_E, λ_H) in C_{msm}, and the percentile-clip endpoints (q_5, q_{95}) (Appendix B). PinPoint is robust to moderate perturbations of any single knob; across every grid point, the cIoU on RefCOCO val and testB stays close to the default configuration, with (λ_E, λ_H) the only axis where the choice meaningfully matters. The default $(\lambda_E, \lambda_H)=(0.5, 1.5)$ achieves the highest mean cIoU across val and testB (73.9, vs. 73.2 for the runner-up $(1.5, 0.5)$); it wins val by 1.2 cIoU and trails the testB-best by only 0.5, where the top three testB settings cluster within 0.5 cIoU of each other. Holding $\lambda_E{=}0.5$ and increasing λ_H from 0.5 to 1.5 improves val by $+2.4$ and testB by $+1.6$, suggesting λ_H has a slightly stronger marginal effect than λ_E inside C_{msm}, though the global landscape is non-monotonic and the two splits do not pick the same operating point.

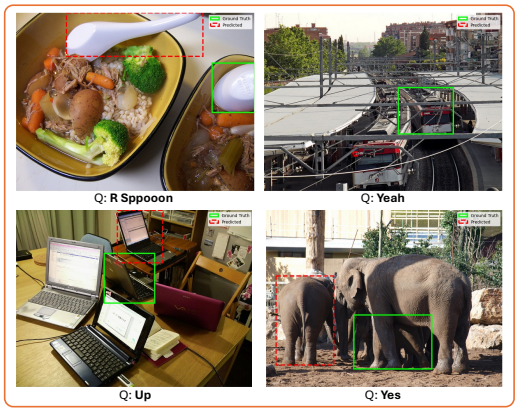
Effect of filtering. Restricting $\mathcal{A}$ to a single label class is brittle: positive-only at $\tau{=}0.7$ gains $+0.4$ cIoU on val but loses 1.5 on testB, and at $\tau{=}0.8$ the optimum flips again (positive-only wins on val, negative-only on testB), so the best label class is split-dependent. The threshold τ is more robust but bounded: verbalized confidences below ~ 0.7 are systematically mis-labeled and benefit from removal, while above that cutoff confidence and correctness become uncorrelated, so pushing τ higher discards correct labels alongside wrong ones, and once the kept-prompt count drops the box-only fallback rate dominates (-6.7 cIoU; Table 5). At $\tau{=}0.9$ the both-labels cIoU sits -0.8 below default on both splits despite a $\sim 45\%$ discard rate. We therefore keep $\tau{=}0.7$, $\mathcal{A}{=}\{\texttt{positive}, \texttt{negative}\}$ as the most stable operating point; full grids in Appendix E.

Inference efficiency. Under our default configuration, a single query costs *two VLM calls*, one for localization and one batched for point-labeling, independent of the point budget N, the number of proposed boxes K, and the choice of fusion mode. PinPoint avoids the attribute-specific calls of decomposition (Wang & Zhang, 2025) and the repeated calls of iterative refinement pipelines (Kao et al., 2025a; Du et al., 2026) whose call counts scale with expression length or refinement rounds. On RefCOCO val (single A6000) the two VLM calls together account for $\sim 95\%$ of per-query wall-clock and PinPoint's geometry-aware selector consumes only $\sim 1\%$; the per-stage breakdown and a detailed call-count comparison are in Appendix F.

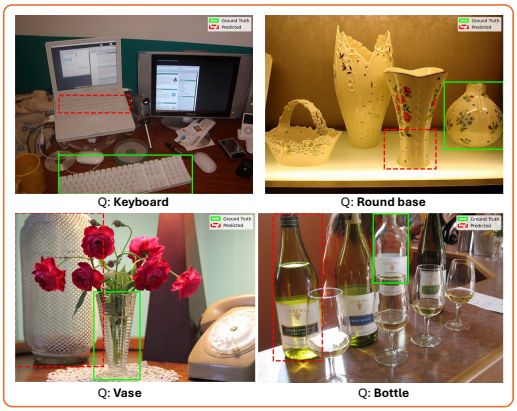
Benchmark noise and the oracle gap. The empirical oracle reference in Table 2 replaces VLM localization and point labeling with ground-truth-derived inputs while keeping PinPoint and SAM unchanged. The remaining gap to 100 cIoU therefore should not be interpreted as a VLM-grounding error; rather, it reflects the limits of SAM mask generation and the benchmark annotation protocol. PinPoint's gap to this oracle reference primarily reflects the portion still attributable to imperfect localization and point verification. Figure 3 illustrates one residual source: some queries are corrupted, unnatural, or under-specified, making the intended referent ambiguous (Fig. 3a), while others are valid but admit multiple plausible targets even though the benchmark provides only one (Fig. 3b).

5 Conclusion

We argued that, in training-free VLM+SAM referring segmentation, *prompt ambiguity* is an important bottleneck after localization. Augmenting the VLM-predicted bounding box with semantically labeled points can reduce this ambiguity. Still, point placement is decisive and poorly placed points can drift onto boundaries and distractors, and can degrade performance. The question that motivated this paper was whether informative point prompts require task-specific training.



(a) **Noisy or corrupted queries.**



(b) **Multiple valid targets.**

Figure 3: **Dataset noise and ambiguity limit the performance.** Even when a system selects a visually plausible region, evaluation can penalize it because the benchmark provides only one annotated target. **(a)** Some queries are corrupted, unnatural, or under-specified, so the intended referent is not recoverable from language alone. **(b)** Other queries are valid but ambiguous: multiple objects satisfy the expression, while only one is labeled as ground truth. Green boxes denote the benchmark target; red dashed boxes denote the predicted or plausible alternative target.

Our answer is PinPoint, a training-free *select–verify–filter* pipeline that fuses four classical cues into a consensus map, deterministically selects compact and spatially diverse candidates, and pairs frozen-VLM semantic verification with confidence filtering. The method leaves the predicted boxes unchanged and remains dependent on upstream grounding and reasoning. Its cue priors can disadvantage off-center, camouflaged, or heavily textured referents, and the verifier can assign high confidence to incorrect labels. Improving verification and confidence calibration is a promising direction within the same pipeline.

Despite these limitations, PinPoint closes 12–18 cIoU over uniform sampling at a matched budget on RefCOCO(+/g), and on the same stack as recent SFT and RL methods, it is competitive with them across RefCOCO(+/g) and ReasonSeg at only two VLM calls per query. Together, these results establish image-side selection paired with semantic verification and confidence filtering as an effective approach to post-localization prompt construction without task-specific point-policy training.

AI USE STATEMENT

In this work, we used generative AI tools to assist with manuscript drafting and editing, identify relevant literature, and obtain feedback on experimental design and the interpretation and presentation of results. We have not used generative AI tools to develop theoretical models or conceptual frameworks, formulate mathematical claims, propose or refine research hypotheses, implement the methods, translate text, clean or reformat datasets, or conduct qualitative or thematic data analyses. Synthetic dataset generation and the development or writing of mathematical proofs are not applicable to this work. We reviewed all AI-assisted content for factual accuracy and consistency with our methods and experimental results. The authors take responsibility for the final manuscript, including all claims, results, and AI-assisted content.

REPRODUCIBILITY STATEMENT

Section 3 and Algorithm 1 describe the pipeline and point-selection procedure. Section 4.1 describes the backbones, datasets, evaluation metrics, and hardware. Appendix A provides the cue definitions, normalization, and Soft-NMS recurrence. Appendix B lists the default hyperparameters and decoding settings. Appendices C and D provide the localization and point-labeling prompts, output grammars, and error handling. Appendices E–F document the ablation configurations and per-stage runtime measurements.

REFERENCES

Shuai Bai, Yuxuan Cai, Keqin Chen, Junyang Lin, Bowen Yu, Jingren Zhou, et al. Qwen3-VL technical report, 2025a. URL https://arxiv.org/abs/2511.21631.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo 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, 2025b. URL https://arxiv.org/abs/2502.13923.

Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S Davis. Soft-NMS–improving object detection with one line of code. In *Proceedings of the IEEE international conference on computer vision*, pp. 5561–5569, 2017.

Tim Brooks, Aleksander Holynski, and Alexei A. Efros. InstructPix2Pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. URL https://arxiv.org/abs/2211.09800.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, Jie Lei, Tengyu Ma, Baishan Guo, Arpit Kalla, Markus Marks, Joseph Greer, Meng Wang, Peize Sun, Roman Rädle, Triantafyllos Afouras, Effrosyni Mavroudi, Katherine Xu, Tsung-Han Wu, Yu Zhou, Liliane Momeni, Rishi Hazra, Shuangrui Ding, Sagar Vaze, Francois Porcher, Feng Li, Siyuan Li, Aishwarya Kamath, Ho Kei Cheng, Piotr Dollár, Nikhila Ravi, Kate Saenko, Pengchuan Zhang, and Christoph Feichtenhofer. Sam 3: Segment anything with concepts, 2026. URL https://arxiv.org/abs/2511.16719.

Yi-Chia Chen, Wei-Hua Li, Cheng Sun, Yu-Chiang Frank Wang, and Chu-Song Chen. SAM4MLLM: Enhance multi-modal large language model for referring expression segmentation. In *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part LXXXI*, pp. 323–340, Berlin, Heidelberg, 2024. Springer-Verlag. ISBN 978-3-031-73003-0. doi: 10.1007/978-3-031-73004-7_19. URL https://doi.org/10.1007/978-3-031-73004-7_19.

Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. URL https://arxiv.org/abs/2112.01527.

Haixing Dai, Chong Ma, Zhiling Yan, Zhengliang Liu, Enze Shi, Yiwei Li, Peng Shu, Xiaozheng Wei, Lin Zhao, Zihao Wu, Fang Zeng, Dajiang Zhu, Wei Liu, Quanzheng Li, Lichao Sun, Shu Zhang, Tianming Liu, and Xiang Li. SAMAug: Point prompt augmentation for segment anything model. *ArXiv*, abs/2307.01187, 2023. URL https://arxiv.org/abs/2307.01187.

Qiyuan Dai and Sibei Yang. Curriculum point prompting for weakly-supervised referring image segmentation, 2024. URL https://arxiv.org/abs/2404.11998.

Henghui Ding, Song Tang, Shuting He, Chang Liu, Zuxuan Wu, and Yu-Gang Jiang. Multimodal referring segmentation: A survey, 2025. URL https://arxiv.org/abs/2508.00265.

Tianyuan Du, Haopeng Li, Zhen Fan, Jiarui Zhang, Panwang Pan, and Yang Zhang. SAM-Veteran: An MLLM-based human-like SAM agent for reasoning segmentation. In *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026. URL https://openreview.net/forum?id=oN55r8iJJW.

Qi Fan, Xin Tao, Lei Ke, Mingqiao Ye, Di Zhang, Pengfei Wan, Yu-Wing Tai, and Chi-Keung Tang. Stable segment anything model. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=ooxj2Audlq.

Tobias Groot and Matias Valdenegro-Toro. Overconfidence is key: Verbalized uncertainty evaluation in large language and vision-language models. In *Proceedings of the 4th Workshop on Trustworthy Natural Language Processing (TrustNLP), NAACL*, 2024. URL https://arxiv.org/abs/2405.02917.

Jonathan Harel, Christof Koch, and Pietro Perona. Graph-based visual saliency. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 19, pp. 545–552, 2006.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. URL https://arxiv.org/abs/1703.06870.

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxuan Zhang, Juanzi Li, Bin Xu, Yuxiao Dong, Ming Ding, and Jie Tang. CogAgent: A visual language model for GUI agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. URL https://arxiv.org/abs/2312.08914.

Xiaodi Hou and Liqing Zhang. Saliency detection: A spectral residual approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8, 2007. doi: 10.1109/CVPR.2007.383267.

Jiaqi Huang, Zunnan Xu, Jun Zhou, Ting Liu, Yicheng Xiao, Mingwen Ou, Bowen Ji, Xiu Li, and Kehong Yuan. SAM-r1: Leveraging SAM for reward feedback in multimodal segmentation via reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=dHOSTp8MBl.

Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11): 1254–1259, 1998. doi: 10.1109/34.730558.

Tilke Judd, Krista Ehinger, Frédo Durand, and Antonio Torralba. Learning to predict where humans look. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2106–2113, 2009. doi: 10.1109/ICCV.2009.5459462.

Shiu-hong Kao, Yu-Wing Tai, and Chi-Keung Tang. CoT-RVS: Zero-shot chain-of-thought reasoning segmentation for videos, 2025a. URL https://arxiv.org/abs/2505.18561.

Shiu-hong Kao, Yu-Wing Tai, and Chi-Keung Tang. Think before you segment: High-quality reasoning segmentation with gpt chain of thoughts, 2025b. URL https://arxiv.org/abs/2503.07503.

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. ReferItGame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 787–798, 2014.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollar, and Ross Girshick. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4015–4026, October 2023.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. LISA: Reasoning segmentation via large language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9579–9589, June 2024.

Chang Liu, Henghui Ding, and Xudong Jiang. GRES: Generalized referring expression segmentation. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23592–23601. IEEE, June 2023a. doi: 10.1109/cvpr52729.2023.02259. URL http://dx.doi.org/10.1109/cvpr52729.2023.02259.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023b. URL https://arxiv.org/abs/2304.08485.

Ye Liu, Zongyang Ma, Junfu Pu, Zhongang Qi, Yang Wu, Ying Shan, and Chang Wen Chen. UniPixel: Unified object referring and segmentation for pixel-level visual reasoning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025a. URL https://arxiv.org/abs/2509.18094.

Yuqi Liu, Bohao Peng, Zhisheng Zhong, Zihao Yue, Fanbin Lu, Bei Yu, and Jiaya Jia. Seg-Zero: Reasoning-chain guided segmentation via cognitive reinforcement, 2025b. URL https://arxiv.org/abs/2503.06520.

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11–20, 2016.

Renjie Pi, Lewei Yao, Jiahui Gao, Jipeng Zhang, and Tong Zhang. PerceptionGPT: Effectively fusing visual perception into LLM. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16–22, 2024*, pp. 27114–27123. IEEE, 2024. doi: 10.1109/CVPR52733.2024.02561.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Radle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollar, and Christoph Feichtenhofer. SAM 2: Segment anything in images and videos, 2024. URL https://arxiv.org/abs/2408.00714.

Zhongwei Ren, Zhicheng Huang, Yunchao Wei, Yao Zhao, Dongmei Fu, Jiashi Feng, and Xiaojie Jin. PixelLM: Pixel reasoning with large multimodal model. *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26364–26373, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Ren_PixelLM_Pixel_Reasoning_with_Large_Multimodal_Model_CVPR_2024_paper.html.

Ruiqi Wang and Hao Zhang. RESAnything: Attribute prompting for arbitrary referring segmentation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

Xiaoqi Wang, Clint Sebastian, Wenbin He, and Liu Ren. ProSAM: Enhancing the robustness of SAM-based visual reference segmentation with probabilistic prompts. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025a. URL https://arxiv.org/abs/2506.21835.

XuDong Wang, Shaolun Zhang, Shufan Li, Konstantinos Kallidromitis, Kehan Li, Yusuke Kato, Kazuki Kozuka, and Trevor Darrell. SegLLM: Multi-round reasoning segmentation, 2024. URL https://arxiv.org/abs/2410.18923.

Yuji Wang, Jingchen Ni, Yong Liu, Chun Yuan, and Yansong Tang. IteRPrimE: Zero-shot referring image segmentation with iterative Grad-CAM refinement and primary word emphasis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025b. URL https://ojs.aaai.org/index.php/AAAI/article/view/32880.

Zhuofan Xia, Dongchen Han, Yizeng Han, Xuran Pan, Shiji Song, and Gao Huang. GSVA: Generalized segmentation via multimodal large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3858–3869, June 2024.

Cilin Yan, Haochen Wang, Shilin Yan, Xiaolong Jiang, Yao Hu, Guoliang Kang, Weidi Xie, and Efstratios Gavves. VISA: reasoning video object segmentation via large language models. In Ales Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024 – 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part XV*, Lecture Notes in Computer Science, pp. 98–115. Springer, 2024. doi: 10.1007/978-3-031-72633-0_6.

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

Tao Yu, Runseng Feng, Ruoyu Feng, Jinming Liu, Xin Jin, Wenjun Zeng, and Zhibo Chen. Inpaint anything: Segment anything meets image inpainting, 2023. URL https://arxiv.org/abs/2304.06790.

Haobo Yuan, Xiangtai Li, Tao Zhang, Yueyi Sun, Zilong Huang, Shilin Xu, Shunping Ji, Yunhai Tong, Lu Qi, Jiashi Feng, and Ming-Hsuan Yang. Sa2VA: Marrying SAM2 with LLaVA for dense grounded understanding of images and videos, 2025. URL https://arxiv.org/abs/2501.04001.

Lei Zhang, Yongqiu Huang, Yingjun Du, Fang Lei, Zhiying Yang, Cees G.M. Snoek, and Yehui Wang. Loter: Localized text prompt refinement for zero-shot referring image segmentation, 2026. ISSN 1077-3142. URL https://www.sciencedirect.com/science/article/pii/S1077314225003194.

Mozhi Zhang, Mianqiu Huang, Rundong Shi, Linsen Guo, Chong Peng, Peng Yan, Yaqian Zhou, and Xipeng Qiu. Calibrating the confidence of large language models by eliciting fidelity. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024. URL https://aclanthology.org/2024.emnlp-main.173/.

Yixin Zhang, Nicholas Konz, Kevin Kramer, and Maciej A. Mazurowski. Quantifying the limits of segmentation foundation models: Modeling challenges in segmenting tree-like and low-contrast objects, 2025. URL https://arxiv.org/abs/2412.04243.

Lianghui Zhu, Bin Ouyang, Yuxuan Zhang, Tianheng Cheng, Rui Hu, Haocheng Shen, Longjin Ran, Xiaoxin Chen, Li Yu, Wenyu Liu, and Xinggang Wang. LENS: Learning to segment anything with unified reinforced reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026. URL https://arxiv.org/abs/2508.14153.

A PinPoint cue definitions, normalization, and Soft-NMS

This appendix gives the per-pixel formulas for the four cues used by PinPoint, the percentile-clipped normalization that puts them on a common scale, and the Soft-NMS recurrence used for diversity-aware point selection. All defaults are listed in Appendix B.

Color-contrast saliency. S rewards pixels whose local color distribution departs from their surroundings:

$$S(p) \;=\; \| \, \mu_{\text{in}}^{\text{lab}}(p) \;-\; \mu_{\text{ann}}^{\text{lab}}(p) \, \|_2, \tag{8}$$

where $\mu_{\text{in}}^{\text{lab}}(p)$ and $\mu_{\text{ann}}^{\text{lab}}(p)$ are mean CIELAB colors in a local disk around p and in the surrounding annulus, respectively. We work in CIELAB because it is approximately perceptually uniform: the ℓ_2 distance between mean colors is only a meaningful contrast signal in a space where geometric distance tracks perceived color difference, which is not the case for RGB or HSV. CIELAB's perceptual uniformity is itself defined relative to a fixed reference illuminant (D65 in our implementation), and we do not perform white-balance or colour-constancy correction prior to computing S. Under strong shadows or specular highlights the ℓ_2 contrast can therefore over- or under-estimate true perceptual difference, so we treat S as a robust-but-approximate cue rather than a calibrated photometric quantity.

Edge density. E flags boundary-adjacent pixels, precisely the regions where SAM's decoder is most uncertain:

$$E(p) \;=\; \big(K_{\text{box}} * \|\nabla I_{\text{gray}}\|\big)(p), \tag{9}$$

where "$*$" denotes 2D convolution and K_{box} is a box-average kernel applied to the Sobel gradient magnitude $\|\nabla I_{\text{gray}}\|$. Prior to gradient computation, I_{gray} is convolved with a Gaussian of $\sigma_{\text{pre}} = 1$ pixel to suppress high-frequency noise that would otherwise inflate E (and H) on flat regions.

Local Shannon entropy. H measures textural clutter: a homogeneous neighborhood scores low, a busy one scores high:

$$H(p) \;=\; -\sum_{v \in \mathcal{V}} \widehat{q}_v(p) \, \log \widehat{q}_v(p), \tag{10}$$

where $\widehat{q}_v(p)$ is the empirical frequency of quantized gradient-magnitude bin v over a disk-shaped neighborhood of p, and $\mathcal{V}$ is a set of $|\mathcal{V}|=256$ equal-width bins. Concretely, we compute the per-pixel gradient magnitude $\|\nabla I_{\text{gray}}\|$, clip it to its 5th and 95th percentiles over the extended crop, rescale the clipped values linearly to the 8-bit integer range $[0, 255]$, and compute a 256-level histogram over each disk-shaped neighborhood. The fixed $(5, 95)$ percentile clip used here is for entropy quantization and is independent of the percentile-clip endpoints (q_5, q_{95}) used in the cue normalizer (Eq. 12); both happen to default to the same percentiles. Like E, H identifies regions where both SAM's decoder and the VLM labeler are least reliable.

Gaussian spatial prior. G softly favors central positions inside the box:

$$G(p) \;=\; \exp\!\Big(-\tfrac{\|p-c_b\|^2}{2\,\sigma_G^2}\Big), \qquad \sigma_G = 0.25\, d_{\text{diag}}(b), \tag{11}$$

where c_b and $d_{\text{diag}}(b)$ are the center and the diagonal length of the bounding box b, and σ_G is the Gaussian-prior bandwidth. G acts as a weak geometric prior: it biases the consensus toward the interior without forbidding peripheral points.

Normalization. The four raw cue maps have different scales, so we normalize each map to $[0, 1]$ using percentile-clipped min–max scaling:

$$\mathcal{N}(X)(p) \;=\; \text{clip}\!\left(\frac{X(p) \;-\; q_5(X)}{q_{95}(X) \;-\; q_5(X) \;+\; \delta},\; 0,\; 1\right), \tag{12}$$

where $q_\alpha(X)$ is the α-th percentile of X over the extended bounding box and $\delta = 10^{-8}$ is a small numerical constant that prevents division by zero on degenerate (constant-valued) maps. We reuse S, E, H, G to denote the normalized cues. Since high E and high H indicate unstable regions, the fusion rules in Section 3.3 use $(1 - E)$ and $(1 - H)$ as smoothness/homogeneity evidence to be rewarded.

Soft-NMS diversity selection. Given a partially-selected set $\mathcal{P} \subseteq \widetilde{\mathcal{P}}$, each remaining candidate $p \in \widetilde{\mathcal{P}} \setminus \mathcal{P}$ receives the virtual score

$$\widetilde{C}_{\mathcal{P}}(p) \;=\; C(p)\,\left(1 - \exp\!\left(-\frac{\min_{p'\in\mathcal{P}}\|p-p'\|^2}{2\sigma_{\text{nms}}^2}\right)\right), \tag{13}$$

with the convention $\min_{p'\in\emptyset}\|p-p'\|^2 = +\infty$ so that $\widetilde{C}_\emptyset(p) = C(p)$ and the first selection reduces to $\arg\max_{p\in\widetilde{\mathcal{P}}} C(p)$. Subsequent points are chosen by $p^\star = \arg\max_p \widetilde{C}_{\mathcal{P}}(p)$. The selection terminates when $|\mathcal{P}| = N$, when $\widetilde{\mathcal{P}} \setminus \mathcal{P} = \emptyset$, or when every remaining virtual score falls below the floor ε (early termination prevents PinPoint from padding the prompt with weak or redundant points). Algorithm 1 in the main text is the operational form of this recurrence.

Complexity. Each cue map is computed over the extended crop, so cue extraction is $\mathcal{O}(|b_{\text{ext}}|)$ with small constants from box filtering, Sobel gradients, and local histograms. Local-maxima extraction is also $\mathcal{O}(|b_{\text{ext}}|)$, and Soft-NMS (Bodla et al., 2017) selection over the candidate pool costs $\mathcal{O}(N \cdot |\widetilde{\mathcal{P}}|)$. PinPoint is deterministic image processing and adds no VLM calls beyond the localization and point-labeling stages; in practice, its cost is dominated by the surrounding VLM and SAM forward passes.

B Reproducibility details

This appendix lists the default values for every symbol introduced in Section 3. All PinPoint radii are expressed relative to the diagonal length $d_{\text{diag}}(b)$ of the VLM-proposed bounding box so that the selector is scale-invariant across images and objects. Unless stated otherwise, these are the values used throughout every experiment in Section 4.

Deterministic behavior. Every stage of PinPoint is deterministic for a fixed image, box, and budget. The only stochastic component is VLM sampling in the localization and point-labeling stages; we use a low temperature (0.1) and a fixed nucleus (0.9) so that the structured JSON outputs are stable in practice, and every benchmark number reported in Section 4 is a single pass with a fixed seed.

Table 3: Default hyperparameters for PinPoint, point labeling, filtering, and SAM. Radii marked with $d_{\text{diag}}(b)$ scale with the per-box diagonal; fixed values are constants across all boxes and images.

Symbol	Meaning	Default	Reference
α	Box-extension factor for cue-map ROI	0.20	§3.3
r_{in}	Saliency inner (local) window radius	$0.03 \cdot d_{\text{diag}}(b)$	Eq. 8
r_{ann}	Saliency annular window radius	$0.09 \cdot d_{\text{diag}}(b)$	Eq. 8
$\lvert K_{\text{box}}\rvert$	Edge-density box-average kernel side	$0.05 \cdot d_{\text{diag}}(b)$	Eq. 9
σ_{pre}	Pre-Sobel Gaussian smoothing	1.0 pixel	Eq. 9
r_H	Entropy disk-neighborhood radius	$0.03 \cdot d_{\text{diag}}(b)$	Eq. 10
$\lvert\mathcal{V}\rvert$	Gradient-magnitude quantization bins	256 (8-bit)	Eq. 10
σ_G	Gaussian-prior bandwidth	$0.25 \cdot d_{\text{diag}}(b)$	Eq. 11
(q_5, q_{95})	Cue-normalizer percentile-clip endpoints	(5th, 95th)	Eq. 12
δ	Numerical-stability constant in $\mathcal{N}(\cdot)$	10^{-8}	Eq. 12
$(p^H_{\text{lo}}, p^H_{\text{hi}})$	Gradient-mag. clip for H quantization	(5th, 95th)	Eq. 10
λ_E	Edge penalty in C_{msm}	0.5	Eq. 3
λ_H	Entropy penalty in C_{msm}	1.5	Eq. 3
W_{peak}	Local-maxima peak-finding window	3×3, 8-connected	§3.3
σ_{nms}	Soft-NMS Gaussian suppression radius	$0.10 \cdot d_{\text{diag}}(b)$	Eq. 13
ε	Soft-NMS termination floor	10^{-3}	§3.3
N	Requested point budget per box	5	§3.3
τ	Confidence threshold for filtering	0.7	Eq. 5
$\mathcal{A}$	Label allowlist	{`positive`, `negative`}	Eq. 5
VLM decoding			
Temperature	Sampling temperature	0.1	§3.4
top-p	Nucleus sampling	0.9	§3.4
max-new-tokens	Generation budget per call	512	§3.4
Labeling batching	All $\mathcal{P}_k$ batched; $\sim$2 calls/image	on	§3.4
SAM			
Checkpoint	SAM 2 model identifier (HF Hub)	`facebook/sam2-hiera-large`	§3.5
`multimask_output`	Mask proposals per prompt	`False` (single-mask)	Eq. 6

Hardware. All inference runs use a single NVIDIA A6000 GPU with batch size 1 at the engine level; vLLM internally batches the per-image point-labeling calls. The two-VLM-call cost structure is hardware-independent; see Section 4.3 for the call-count analysis.

Hyperparameter sensitivity. Table 4 sweeps the three scalar hyperparameter classes whose effect on cIoU is non-trivial: the Gaussian-prior bandwidth σ_G (Eq. 11), the edge / entropy penalty weights (λ_E, λ_H) in C_{msm} (Eq. 3), and the percentile-clip endpoints (q_5, q_{95}) in the cue normalizer (Eq. 12). Each block sweeps one parameter while holding the remaining defaults from Table 3 fixed; the reported cIoU values are based on runs on RefCOCO val and the more spatially-challenging testB split. Three observations follow. First, σ_G is essentially flat across the reasonable range $[0.10, 0.33]$ (val cIoU spans only 1.1 points), with a meaningful drop only at the over-smoothed endpoint σ_G=0.45; the default 0.25 is robust. Second, the (λ_E, λ_H) grid is the most informative axis but the two splits do not agree on a single winner: $(0.5, 1.5)$ wins val with 75.4 cIoU and reaches 72.4 on testB, while $(1.5, 0.5)$ tops testB with 72.9 but loses 1.9 cIoU on val. We adopt $(0.5, 1.5)$ as the default reflected in Table 3 because (i) it wins val by a clear margin and is within 0.5 cIoU of the best on testB (where the top settings cluster within a narrow band of 0.5 cIoU), and (ii) holding λ_E=0.5 fixed and sweeping $\lambda_H \in \{0.5, 1.0, 1.5\}$ yields a monotonic gain on both splits (+2.4 val, +1.6 testB), indicating that within the smallest-edge-penalty slice the entropy penalty does have a slightly stronger marginal effect; the global 3×3 landscape is, however, non-monotonic, so this should be read as a local rather than universal trend. Third, the percentile-clip endpoints (q_5, q_{95}) are stable: the wider $(1, 99)$ clip edges $(5, 95)$ on val by 0.4 cIoU but loses 1.6 cIoU on testB, so we keep the more stable $(5, 95)$ default. Across every grid point the cIoU stays close to the default configuration (highlighted in bold), confirming that PinPoint does not depend on a brittle hyperparameter setting.

Table 4: **Hyperparameter sensitivity.** cIoU on RefCOCO val and testB while sweeping σ_G (as a fraction of $d_{\text{diag}}(b)$), (λ_E, λ_H), and (q_5, q_{95}); Δ columns report the gap to the block's default row. **Best** and second-best per metric per block; the default row labels are bolded. The (λ_E, λ_H) block is reported under C_{msm} since these weights only appear in Eq. 3; the σ_G and (q_5, q_{95}) blocks are reported under the paper-default C_{mult}, which is why the "default" rows do not all share the same cIoU value.

Parameter	Setting	val cIoU ↑	Δ val	testB cIoU ↑	Δ testB
Gaussian-prior bandwidth σ_G (frac. of $d_{diag}(b)$)					
σ_G	0.10	75.9	0.0	69.9	−1.2
σ_G	0.18	76.6	+0.7	70.8	−0.3
$\boldsymbol{\sigma_G}$	**0.25** *(default)*	**75.9**	**0.0**	**71.1**	**0.0**
σ_G	0.33	75.5	−0.4	70.7	−0.4
σ_G	0.45	73.6	−2.3	70.5	−0.6
Edge / entropy penalty weights (λ_E, λ_H) in C_{msm}					
(λ_E, λ_H)	$(0.5, 0.5)$	73.0	−2.4	70.8	−1.6
(λ_E, λ_H)	$(0.5, 1.0)$	72.6	−2.8	71.0	−1.4
$\boldsymbol{(\lambda_E, \lambda_H)}$	$\mathbf{(0.5, 1.5)}$ *(default)*	**75.4**	**0.0**	**72.4**	**0.0**
(λ_E, λ_H)	$(1.0, 0.5)$	74.2	−1.2	70.0	−2.4
(λ_E, λ_H)	$(1.0, 1.0)$	73.0	−2.4	70.5	−1.9
(λ_E, λ_H)	$(1.0, 1.5)$	73.0	−2.4	72.7	+0.3
(λ_E, λ_H)	$(1.5, 0.5)$	73.5	−1.9	72.9	+0.5
(λ_E, λ_H)	$(1.5, 1.0)$	71.9	−3.5	71.4	−1.0
(λ_E, λ_H)	$(1.5, 1.5)$	73.1	−2.3	69.6	−2.8
Percentile-clip endpoints (q_5, q_{95}) in the cue normalizer					
(q_5, q_{95})	$(1, 99)$	76.3	+0.4	69.5	−1.6
$\boldsymbol{(q_5, q_{95})}$	$\mathbf{(5, 95)}$ *(default)*	**75.9**	**0.0**	**71.1**	**0.0**
(q_5, q_{95})	$(10, 90)$	75.6	−0.3	69.6	−1.5

C LOCALIZATION PROMPT AND GRAMMAR

The first stage of the pipeline uses the VLM as a grounded proposer: given an image I and a natural-language query Q, it returns a (possibly empty) set of candidate bounding boxes. To make the output consumable by downstream stages we constrain the generation with a formal EBNF grammar via vLLM's structured outputs. The prompt template below is what the VLM sees; the grammar enforces that its output is a single valid JSON object with chain-of-thought and box coordinates.

Localization Prompt

```
You are a visual grounding assistant.
Given an image and a natural-language query, return every object in
the image
that the query refers to, as a bounding box in pixel coordinates.
Image size: {H} × {W} (height × width).
Query: "{Q}"
Respond with a single JSON object that conforms to the grammar
below.
Include a short reasoning string explaining your choice, followed by
a
boxes list in which each element has integer fields
x_min, y_min, x_max, y_max, all in pixel coordinates of the image.
If the query does not refer to any visible object, return an empty
list for
boxes and briefly explain why in reasoning.
```

EBNF Grammar (Localization)

```
root        ::= "{" ws "\"reasoning\"" ws ":" ws string
                ws "," ws "\"boxes\"" ws ":" ws box_list
                ws "}"

box_list    ::= "[" ws "]" | "[" ws box (ws "," ws box)* ws "]"

box         ::= "{" ws "\"x_min\"" ws ":" ws integer
                ws "," ws "\"y_min\"" ws ":" ws integer
                ws "," ws "\"x_max\"" ws ":" ws integer
                ws "," ws "\"y_max\"" ws ":" ws integer
                ws "}"

integer     ::= "0" | [1-9] [0-9]*
string      ::= "\"" ( [^"\\] | "\\" . )* "\""
ws          ::= ( " " | "\t" | "\n" | "\r" )*
```

Implementation note. The grammar is enforced at decoding time via vLLM's `StructuredOutputsParams`. If the grammar-constrained decoder truncates early (e.g., the generation budget is exhausted mid-JSON), we fall back to a single unconstrained retry and flag the sample only if the retry also fails to produce a parseable object; empirically, this happens on fewer than 0.1% of the RefCOCO(+/g) validation samples with the higher max-token budget, which can be mitigated by adjusting generation hyperparameters.

D POINT-LABELING PROMPT AND GRAMMAR

The third stage of the pipeline uses the VLM as a semantic verifier: for each PinPoint-selected point $p_{k,i} \in \mathcal{P}_k$ inside the proposed box b_k, the VLM returns a label $\ell_{k,i} \in \{\texttt{positive}, \texttt{negative}\}$ and a verbalized confidence $c_{k,i} \in [0, 1]$. As discussed in Section 3.4, the VLM receives the image, the query, the box, and the point coordinates via text; no pixel-level annotations (dots, crosses, rectangles) are rendered onto the image before it is shown to the VLM. All points inside an image are labeled in a single batched generation call.

Point-Labeling Prompt

```
You are a visual grounding assistant.
Given an image, a natural-language query, a bounding box, and a list
of points
inside that bounding box, decide for each point whether it lies on
the object
described by the query (positive) or on the background (negative),
and report your confidence.
Image size: {H} × {W} (height × width).
Query: "{Q}"
Bounding box (pixel coordinates): {x_min, y_min, x_max, y_max}.
Points: [{x_1, y_1}, {x_2, y_2}, ..., {x_n, y_n}].
Respond with a single JSON object that conforms to the grammar
below.
The labels field must contain exactly one entry per input point, in
the
same order. Each entry has a label string (positive or
negative) and a confidence number in [0,1].
Do not hedge: pick the label that better describes the point, and
report your
own honest confidence.
```

EBNF Grammar (Point Labeling)

```
root        ::= "{" ws "\"labels\"" ws ":" ws label_list ws "}"

label_list  ::= "[" ws label (ws "," ws label)* ws "]"

label       ::= "{" ws "\"label\"" ws ":" ws label_enum
                    ws "," ws "\"confidence\"" ws ":" ws confidence
                    ws "}"

label_enum  ::= "\"positive\"" | "\"negative\""

confidence  ::= "0" | "1" | "0." [0-9]+ | "1.0" | "1.00"

ws          ::= ( " " | "\t" | "\n" | "\r" )*
```

Batching. In practice, all points for a single image (and, when the benchmark is run in batched mode, all points across a minibatch of images) are passed to the VLM in one structured-output call. This collapses point labeling from $|\mathcal{P}_k|$ sequential forward passes to a single forward pass per image, and the whole pipeline consumes approximately two VLM calls per image in total: one for localization and one for the entire labeling batch. The template above handles one box per call; multi-box images dispatch K such prompts concurrently as a single vLLM batch, so per-image latency stays bounded by one VLM round-trip independent of K.

Length-mismatch fallback. The EBNF grammar enforces the JSON structure of the response but does not constrain the `labels` array length to match the input point count. We handle the two failure modes deterministically: if the VLM returns a `labels` array *longer* than the input point list, we truncate to the input length; if it returns a *shorter* array, the missing trailing entries are treated as low-confidence labels and are dropped by the confidence filter (Eq. 5). No retry is issued, and length mismatches do not propagate to the SAM call.

E END-TO-END ABLATION

This appendix reports the detailed end-to-end ablation summarized in Section 4.3. Each row holds every stage of the PinPoint pipeline fixed except the named factor; $N{=}5$ throughout.

Table 5: **End-to-end ablation on RefCOCO val.** Each row holds every stage of the pipeline fixed except the named factor; $N{=}5$ throughout. "Random" samples points uniformly inside the VLM box (averaged over three seeds); "All-positive" skips the VLM verifier and labels every PinPoint point as `positive`; "$\tau{=}0$" disables confidence filtering. Bold marks the default PinPoint configuration (also the best result) used throughout the rest of the paper; underline marks the second-best.

Selector	Verifier	Filter	cIoU ↑	Δ *vs. default*
Bounding box only	—	—	69.2	−6.7
Naive sampling	VLM labels	$\tau{=}0.7$	61.9	−14.0
PinPoint$_{\text{mult}}$	all-positive	—	68.7	−7.2
PinPoint$_{\text{mult}}$	VLM labels	$\tau{=}0$	73.6	−2.3
PinPoint$_{\text{mult}}$	**VLM labels**	$\boldsymbol{\tau{=}0.7}$ *(default)*	**75.9**	**0.0**
PinPoint$_{\text{msm}}$	VLM labels	$\tau{=}0.7$	<u>75.4</u>	−0.5

Roles of the verifier and the filter. The verifier and the confidence filter contribute distinct, complementary roles even though they both operate on the labeled point set. PinPoint's point selection is semantically blind: the four cues observe the image and the bounding box, but never the referring expression. Treating every selected point as positive (the all-positive row) therefore makes the prompt more ambiguous, since any candidate that lands on co-occurring background or a distractor object pollutes the support; the VLM verifier converts these geometrically-chosen candidates into positive/negative SAM prompts that align with the referring expression. Numerically,

dropping the verifier costs 7.2 cIoU while disabling the threshold (τ=0) costs only 2.3 cIoU, so the verifier carries the bulk of the work and the filter acts as a stabilizer that discards low-confidence labels so occasional VLM misjudgments do not propagate into the SAM mask. The two fusion modes (last two rows) sit within 0.5 cIoU of each other on this split, supporting our framing of C_{msm} as a softer relaxation of C_{mult} rather than a separate method.

Threshold sensitivity and VLM portability. Two questions follow from the -2.3 cIoU gap between τ=0 and τ=0.7 in the ablation above. First, is τ=0.7 brittle on the VLM we use, or is the entire $[0, 0.7]$ range close to the default? The two endpoints give a 2.3 cIoU envelope, and verbalized confidences from grammar-guided Qwen2.5-VL are concentrated above 0.5 in our runs, so most of that envelope is contributed by a small slice of low-confidence labels rather than by a sharp peak around 0.7; PinPoint is therefore not relying on a finely tuned cutoff for this VLM. Second, does the same value of τ port to a different VLM? Verbalized confidences are systematically overconfident and miscalibrated, with the degree of miscalibration differing meaningfully across language and vision–language models tested side by side (Groot & Valdenegro-Toro, 2024), so the absolute scale of $c_{k,i}$ is VLM-specific and the optimal τ is unlikely to transfer numerically. The role of the threshold (suppress the lowest-confidence labels), however, is a structural property of the filter and not of the value itself: any VLM that reports a usable confidence score will benefit from removing its bottom slice, even if the cut-off needs to be re-located. We therefore recommend a coarse one-pass val sweep over $\tau \in \{0.0, 0.3, 0.5, 0.7, 0.9\}$ when porting PinPoint to a new VLM, rather than treating 0.7 as a universal default; the verifier itself, the only stage with a large effect, is unchanged across this sweep.

Stage decomposition and per-cue robustness. Table 5 holds the pipeline fixed at the default configuration and ablates one factor at a time; we complement it here with two analyses that probe different axes. Table 6 walks a candidate prompt set through four progressive pipeline configurations, from the box-only floor to the full PinPoint pipeline, isolating what each stage contributes on top of the previous one. Table 7 zeros each of the four PinPoint cues (colour-contrast saliency S, edge density E, local entropy H, and the Gaussian spatial prior G) one at a time, asking whether the gain is robust to dropping any single cue. Together the two answer two questions Table 5 does not address by construction: *where* along the pipeline the gain enters, and *whether* the four-cue selector relies on any single cue. The cue ablation runs under a diagnostic linear-penalty fusion mode, $C(p) = S(p) \cdot \max(1 - \lambda_E E(p) - \lambda_H H(p),\, 0) \cdot G(p)$, which is the only fusion that admits a clean off-switch for E and H via the existing λ_E and λ_H knobs (the headline C_{mult} multiplies S, $(1 - E)$, $(1 - H)$, G with no scalar weights, so an off-switch for any single cue would change the formula rather than zero a parameter); we therefore report ΔcIoU vs. the all-cues-on baseline rather than absolute cIoU. S enters every fusion as a multiplicative factor and admits no zero-weight off-switch in any mode, so a no_S row is intentionally omitted.

Table 6: **Per-stage contribution on RefCOCO val.** Four progressive configurations from the box-only floor to the full PinPoint pipeline; N=5 throughout. Each row adds one stage on top of the previous: + PinPoint points introduces geometry-aware points as all-positive prompts without the verifier; + VLM labels enables the verifier; + filter enables confidence filtering at τ=0.7. Bold marks the full pipeline (the headline run from Table 1); underline marks the second-best.

Stage	Description	cIoU ↑
s1	Bounding box only	69.2
s2	+ PinPoint points (no labels)	68.7
s3	+ VLM labels (no filter)	<u>73.6</u>
s4	**+ Confidence filter τ=0.7 (full pipeline)**	**75.9**

Where each stage helps. Adding all-positive PinPoint points to the box-only baseline (s1→s2) *costs* 0.5 cIoU: SAM treats every passed point as a positive prompt, so candidates landing on co-occurring background or distractors pollute the support and shift the predicted mask away from the referent. The largest gain comes from the VLM verifier (s2→s3, +4.9 cIoU): converting a geometric set into per-point positive/negative labels gives SAM both anchors *and* explicit exclusions, which is what disambiguates the referent from co-occurring objects inside the same box. Confidence filtering on top of labelling (s3→s4, +2.3 cIoU) is the final stabiliser, suppressing the small fraction of

low-confidence labels the verifier mislabels under grammar-guided decoding; Tables 8 and 9 examine this stage further.

Table 7: **Per-cue robustness on RefCOCO val.** Each row zeros one cue of the four-cue PinPoint selector under the diagnostic linear-penalty fusion that admits clean off-switches for E and H via λ_E and λ_H, and for G via $\sigma_G \to \infty$. We report ΔcIoU relative to the all-cues-on baseline of this fusion; the absolute level under the diagnostic fusion is not directly comparable to the headline C_{mult} runs in Tables 1 and 5.

Setting	**Cue removed**	**ΔcIoU** vs. all-on
all-on	—	0.0
no_E	edge density (λ_E=0)	-0.7
no_G	Gaussian prior ($\sigma_G \to \infty$)	-1.5
no_H	local entropy (λ_H=0)	$+1.7$

Robustness to single-cue dropout. No single cue moves cIoU by more than 1.7 points on RefCOCO val, supporting the robustness claim already made in §4.3 (Hyperparameter sensitivity): PinPoint does not rely on any single cue. Removing the Gaussian prior costs the most (-1.5 cIoU), consistent with G's role as a soft centroid prior that suppresses peripheral candidates close to the box border. Removing the edge-density penalty costs only -0.7 cIoU at unit weight. The small *positive* Δ for no_H ($+1.7$) is a property of the unit-weighted linear-penalty diagnostic, not of the headline fusion: the entropy penalty at λ_H=1 is somewhat over-aggressive, which is consistent with the C_{msm} sweep in §4.3 preferring a different operating point rather than H being unhelpful in general. The headline C_{mult} uses $(1-E)$ and $(1-H)$ as smoothness/homogeneity factors with no scalar weights, so E and H contribute as fixed multiplicative penalties there and cannot be probed by zeroing a parameter; the diagnostic mode used in Table 7 exists precisely to make these per-cue interventions well-defined.

Filter sweep: confidence threshold. The verifier ablation in Table 5 reports two threshold endpoints, τ=0 and τ=0.7. Table 8 sweeps the interior, $\tau \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$, on the multiplicative variant with the default allowlist $\mathcal{A}$={`positive`, `negative`}; we report cIoU on the full RefCOCO val and testB splits along with the percentage of the five-point budget filtered out and the mean number of points passed to SAM (N=5 throughout). cIoU is non-monotone in τ on both splits and peaks at τ=0.8 on val and τ=0.6 on testB; the τ=0.5, τ=0.6, τ=0.7 rows sit within a 1.0 cIoU band on val. This pattern is consistent with the calibration of grammar-guided Qwen2.5-VL: verbalized confidences below $\sim$0.7 are systematically mis-labeled and benefit from removal, but above $\sim$0.7 confidence and label correctness are essentially uncorrelated, so further filtering discards correct labels along with the wrong ones. Pushing past τ=0.8 then begins to cost: at τ=0.9, the mean number of kept prompts on val falls to 2.60 from 4.16 at default, and a non-trivial sample fraction retains no prompts at all and falls back to SAM's box-only output (69.2 cIoU; Table 5). The both-labels cIoU at τ=0.9 ends -0.8 below default on both splits, giving a clean illustration of the box-only-fallback tradeoff.

Table 8: **Confidence-threshold sweep on RefCOCO val and testB.** cIoU at each τ on the default PinPoint$_{\text{mult}}$ configuration with $\mathcal{A}$={`positive`, `negative`}, N=5. "% filt." reports the number of filtered points as a percentage of the five-point budget; PinPoint may return fewer than five candidates. "Mean kept" is the mean number of points passed to SAM. The **bold** row marks the paper default; underlined cells are the best within each split.

τ	val cIoU	testB cIoU	val % filt.	testB % filt.	val mean kept	testB mean kept
0.5	76.0	69.0	0.9	1.0	4.71	4.75
0.6	75.2	<u>73.0</u>	4.3	3.1	4.56	4.57
0.7	**75.9**	**71.1**	**12.8**	**8.8**	**4.16**	**4.29**
0.8	<u>78.1</u>	71.9	29.0	20.6	3.58	3.84
0.9	75.1	70.3	47.8	42.4	2.60	2.69

Filter sweep: label allowlist. Table 9 extends the sweep to the joint grid of $\tau \in \{0.7, 0.8, 0.9\}$ and $\mathcal{A} \in \{\{+,-\},\{+\},\{-\}\}$. The optimal allowlist is split-dependent. Val prefers positive-only at τ=0.8 (80.3 cIoU, +4.4 over default), while testB prefers negative-only at the same τ (72.1 cIoU, +1.0); the two splits flip the sign of the allowlist effect, so committing to a single label class is brittle. SAM uses positives to *anchor* and negatives to *exclude*, and the relative importance of the two roles depends on the cluttered-box population of the split: testB has more distractors, so removing negatives hurts more there. Beyond τ=0.8 all three allowlists move toward or past the box-only-fallback regime, as positive-only retains only ∼1.1 prompts on val and ∼1.2 on testB at τ=0.9, and the both-labels cell drops below default on both splits.

Table 9: **Threshold × label-allowlist sweep on RefCOCO val and testB.** cIoU under three allowlists $\mathcal{A}$: both labels, positive-only ($\{+\}$), negative-only ($\{-\}$). N=5, multiplicative fusion, oracle off. The bold row is the paper default; underlined cells are the best within each split.

τ	$\mathcal{A}$	val cIoU	testB cIoU	val mean kept	testB mean kept
0.7	**{+, −}**	**75.9**	**71.1**	**4.16**	**4.29**
0.7	{+}	76.3	69.6	1.46	1.88
0.7	{−}	73.1	69.4	2.70	2.41
0.8	{+, −}	78.1	71.9	3.58	3.84
0.8	{+}	80.3	71.4	1.33	1.79
0.8	{−}	77.2	72.1	2.25	2.05
0.9	{+, −}	75.1	70.3	2.60	2.69
0.9	{+}	79.0	68.8	1.09	1.18
0.9	{−}	77.4	69.9	1.51	1.51

F PER-STAGE RUNTIME BREAKDOWN

This appendix expands the brief efficiency discussion in Section 4.3. We characterise PinPoint's inference cost in terms of VLM calls per query, the dominant cost contributor in any training-free VLM + SAM pipeline and a hardware-independent measure of architectural efficiency, before reporting the per-stage wall-clock breakdown.

Why the per-query budget is exactly two VLM calls. Under PinPoint's default configuration (grammar-guided structured decoding in both the localization and point-labeling stages, no separate chain-of-thought reasoning pass, and batched per-image labeling), a single query results in (i) one grammar-guided localization call producing the candidate set $\mathcal{B}$, and (ii) one batched grammar-guided point-labeling call across all (b_k, p) pairs. The two-call budget is therefore independent of the point budget N, the number of proposed boxes K, and the choice of fusion mode.

Call-count comparison with related training-free pipelines. Training-free referring-segmentation methods divide cleanly along their inference-time call budget. Decomposition pipelines such as RESAnything (Wang & Zhang, 2025) parse the referring expression into multiple semantic attributes and issue one VLM call per attribute. Loop-based pipelines such as CoT-RVS (Kao et al., 2025a) iteratively feed segmentation outputs back into the VLM for refinement, spending three or more VLM calls per refinement round. RL-trained single-shot methods such as Seg-Zero (Liu et al., 2025b) and SAM-R1 (Huang et al., 2026) issue a single VLM call at inference but require GRPO post-training on thousands of curated samples.

Per-stage wall-clock. Table 10 reports the per-query wall-clock decomposition of PinPoint on RefCOCO val with Qwen2.5-VL-7B + SAM, measured on a single A6000. Means and standard deviations are computed across the val split. Total cost is dominated by the two VLM calls (localization and point labeling, together ∼95% of wall-clock); PinPoint's geometry-aware point selection contributes only 1.3% and confidence filtering is below 0.1 ms.

The ratio of PinPoint point-selection cost to total VLM cost is $62.2 / (3290.1 + 1413.7) \approx 1.3\%$, confirming that the per-query budget is dominated by the two structured-decoding VLM calls and

Table 10: **Per-query wall-clock breakdown on RefCOCO val (single A6000).** Confidence filtering was disabled in this measurement; its arithmetic cost is below 0.1 ms.

Stage	Mean (ms)	Std (ms)	% of total
Localization (VLM)	3290.1	952.9	66.5%
Point selection (PinPoint)	**62.2**	**14.3**	**1.3%**
Point labeling (VLM)	1413.7	196.9	28.6%
Point filtering	< 0.1	< 0.1	0.0%
SAM segmentation	182.5	23.1	3.7%
Total	**4948.5**		**100.0%**

that PinPoint's overhead is negligible despite being more expensive than uniform random sampling inside the box.

G GENERALIZED REFERRING SEGMENTATION ON GREFCOCO

We evaluate PinPoint on gRefCOCO under the GRES protocol (Liu et al., 2023a), using the same frozen Qwen2.5-VL-7B + SAM 2-L stack and a point budget of $N=5$ per predicted box. Table 11 compares PinPoint with the bounding-box-only control; both rows reuse identical localization outputs.

Table 11: **Generalized referring segmentation on gRefCOCO.** Each cell reports gIoU / cIoU / no-target accuracy (N-acc), all in percent (↑). The localization outputs and frozen backbones are identical across rows.

Method	val	testA	testB
Bounding box only	40.2/20.3/45.7	44.1/34.4/39.1	39.2/30.1/34.8
$\text{PinPoint}_{\text{msm}}$	49.7/48.1/45.7	47.6/43.2/39.1	41.8/42.3/34.8

PinPoint improves gIoU by $+9.5/+3.5/+2.6$ and cIoU by $+27.8/+8.8/+12.2$ on val/testA/testB, while N-acc remains unchanged. When localization returns $K=0$, the pipeline emits an empty mask; for $K>1$, it refines each returned box and unions the per-box masks. PinPoint never creates, removes, or revises the predicted box set, so no-target recognition and which target instances can be covered remain governed by localization. These results show that post-localization mask refinement transfers to generalized RIS. Informative points improve the segmentation of returned instances but cannot recover targets excluded by the frozen localization stage.

H NON-UNIFORM POINT-SELECTOR COMPARISONS

Table 12 extends the controlled comparison to non-uniform point selectors. Across all point-based rows, the predicted boxes, frozen Qwen2.5-VL-7B + SAM 2-L stack, VLM verifier, confidence filter, and budget $N{=}5$ are identical; only the selected coordinates change. The uniform row applies PQPP's point-placement rule within our frozen pipeline.

Table 12: **Non-uniform selector comparison on RefCOCO.** cIoU (↑) under the matched single-pass protocol. SAMAug selection rules are adapted to the box interior. FPS denotes farthest-point sampling. Bold marks the full PinPoint$_{\text{mult}}$ configuration.

Selector	val	testA	testB
Bounding box only	69.2	72.0	63.9
Naive uniform sampling	61.9	65.5	57.6
Centroid + 4 quadrant centres	67.1	66.7	65.9
Saliency peaks	67.2	69.0	65.2
SAMAug max-difference-entropy	67.1	68.4	64.8
SAMAug max-distance	69.5	71.3	65.9
Distance transform peak + FPS	69.0	71.6	64.9
$(S \times G)$ + Soft-NMS only	73.5	77.3	70.9
PinPoint, four cues, no Soft-NMS	73.4	76.9	70.6
PinPoint$_{\text{mult}}$ (full)	**75.9**	**81.7**	**71.1**

PinPoint exceeds the strongest competing non-uniform selector on each split by $+6.4/+10.1/+5.2$ cIoU on val/testA/testB. The $S \times G$ simplification recovers much of the gain, but full cue fusion adds $+2.4/+4.4/+0.2$ over it, and Soft-NMS adds $+2.5/+4.8/+0.5$ over the four-cue selector without diversity suppression. The gains also hold against non-uniform selectors, although full cue fusion adds little on testB.

SAMAug (Dai et al., 2023) conditions its augmented points on an initial SAM mask. Table 13 reports these mask-conditioned controls using SAM(box), which requires a second SAM pass. The box-interior adaptations in Table 12 are stronger on every split. The saliency-peaks row uses classical saliency; it does not reproduce SAMAug's learned-saliency variant, which introduces an additional pretrained Visual Saliency Transformer.

Table 13: **Mask-conditioned selector controls on RefCOCO.** cIoU (↑) with points selected inside the initial SAM(box) mask, requiring two SAM passes.

Selector inside SAM(box)	val	testA	testB
SAMAug max-distance	68.4	70.2	65.4
Distance transform peak + FPS	65.9	67.2	64.1
SAMAug max-difference-entropy	63.3	64.9	64.3