Appendix

A. Table of Contents

1. Sec. B provides detailed implementation information on identifying localization heads and obtaining the language-guided localization prior using these heads.
2. Sec. C describes detailed experimental setup, including datasets and preprocessing, baseline methods, and prompt templates.
3. Sec. D provides additional experimental results and analyses, including latency analysis and an ablation study on the effects of different visual encoders.
4. Sec. E presents qualitative results of our method.
5. Sec. F discusses the limitations of our approach and directions for future work.

B. Implementation Details

In this section, we provide implementation details. We first describe how specific attention heads with strong visual grounding capabilities are selected in Sec. B.1. We then explain how text-to-image attention maps are obtained with these selected heads from different MLLMs in Sec. B.2. Finally, we provide the details on how we fuse the textual and visual cues in Sec. B.3.

B.1. Finding Localization Heads

To identify localization heads (Sec. 3), we follow the approach of Kang et al. [13], using two criteria: overall attention strength and spatial concentration.

First, for each attention head, we compute the total attention from the final text token to all image tokens and average this value over 1,000 randomly sampled queries from the RefCOCO training set [14]. Heads with average attention above the elbow point of the resulting distribution are retained. Next, to assess spatial concentration, we binarize each retained head's attention map at the mean value, extract connected components using 8-neighbor connectivity, and compute the spatial entropy of the component size distribution [3, 23] . Lower entropy indicates that attention is compactly focused on object regions, while higher entropy reflects dispersed patterns. For each query, we rank retained heads by entropy and record the top-10 with the lowest values. Finally, heads that consistently appear in these top-10 lists across queries are selected as localization heads, ensuring both strong and spatially concentrated attention.

B.2. Language-guided Localization Priors

Here, we describe how the localization heads identified in the previous section is leveraged to obtain the language-guided localization prior introduced in Sec. 4.4.

By default, we adopt the top three most frequently selected heads from the previous stage, following the setting in prior work [13]. Specifically, given an image and a text query, we extract the attention maps from the last text token to all image tokens of the selected localization heads. Each attention map has the shape $\mathbb{R}^{H_l W_l}$, where H_l and W_l denote the number of patches along the height and width, respectively. These maps are then reshaped to $\mathbb{R}^{H_l \times W_l}$ and further smoothed with a Gaussian filter, using a kernel size of $k = 7$ and a standard deviation of $\sigma = 1.0$. Finally, the smoothed maps are aggregated via element-wise summation to produce the final language-guided localization score map.

Among the models used in this work, Molmo-7B-D [9] and InternVL-3-8B [40] divide the input image into multiple square crops that tile the image, along with a global thumbnail. For Molmo-7B-D, we leverage both the attention maps to the square crops and to the thumbnail, where the thumbnail attention is interpolated to match the resolution of the square crops. For InternVL-3-8B, for simplicity, we only use the attention to the thumbnail image. In contrast, Qwen2.5-VL-7B [2], Ovis2.5-9B [21], and Kimi-VL-A3B [30] do not divide the image into square crops, so we can directly utilize the attention from text to image.

B.3. Combining Textual and Visual Cues

Here, we provide details on how we fuse the textual and visual cues. After obtaining the language-guided localization score map $S_{\text{Text}} \in \mathbb{R}^{H_l \times W_l}$ and the visual semantic correspondence score map $S_{\text{Visual}} \in \mathbb{R}^{H_v \times W_v}$, we combine them to enable more robust localization. Since these two maps may have different spatial resolutions (with S_{Visual} typically being higher, e.g., 64×64 for DINOv3 at a 1024-pixel input resolution in our setup), we first normalize both maps to the range $[0, 1]$, and then upsample S_{Text} to match the resolution of S_{Visual} using bilinear interpolation. Finally, we obtain the fused score map $S_{\text{Final}} \in \mathbb{R}^{H_v \times W_v}$ by performing element-wise multiplication.

C. Experimental Details

In this section, we present the details of our experimental setup. We begin by introducing the datasets and preprocessing steps in Sec. C.1. Next, we describe the baselines used in Sec. C.2. Finally, we provide the prompt templates employed for evaluating different models in Sec. C.3.

C.1. Datasets

In this section, we describe the datasets used in our experiments. We adapt three part segmentation datasets—PACO-LVIS [24], InstructPart [33], and PartImageNet++ [17]—to the few-shot part-level pointing task. For all datasets, the training split is used as the support set candidates, and the test split is used for evaluation.

PACO-LVIS contains 75 object categories and 456 part categories, covering common everyday objects. Its offi-

cial test split provides 5,729 instance-level referring expressions that specify both objects and parts. For our task, we pair each referred object instance with its annotated parts to construct the evaluation set. Each sample thus consists of a target part and a referring expression for an object instance, with the goal of predicting the part of the instance. To ensure reliable evaluation, we discard samples with ground truth masks smaller than 196 pixels, resulting in 18,154 evaluation samples. Applying the same filter to the training split yields 120,777 support samples. This setup enables part-level pointing evaluation in real-world images with instance-level references. Hereafter, we refer to PACO-LVIS as PACO for brevity.

InstructPart is designed for task-oriented part-level understanding, featuring instructions about object affordances and functionalities. It contains 2,400 images across 48 object categories and 44 part categories in household tasks, with 1,800 training (supporting) and 600 test samples. The dataset supports two tasks proposed in the original work: Task Reasoning Part Localization (e.g., "find the part in the image that can be gripped") and Oracle Referring Part Localization (e.g., "locate the handle of the mug"). We focus on the latter, where each sample specifies an object's part to be localized. We directly use the dataset without additional processing to evaluate models' ability to localize parts in household scenes.

PartImageNet++ provides high-quality part segmentation annotations for all ImageNet-1K categories [25], covering a diverse range of objects. The dataset defines 3,308 object–part categories, with 100 images for each of the 1,000 object categories. Each sample involves locating a specific part of an object category. We use the first 90 images per category for training (supporting) and the remaining 10 for testing, whose part annotations yield 26,747 part-level test samples. We evaluate on this dataset primarily to broaden the diversity of tested part–object categories.

C.2. Baselines

To contextualize our few-shot results, we include several baselines. For zero-shot reference, we report the performance of pointing-capable MLLMs, including Qwen2.5-VL-7B [2], Ovis2.5-9B [21], Molmo-7B-D [9], and the closed-source model GPT-4.1 (GPT-4.1-mini-2025-04-14).

For additional reference, we compare against several widely used open-vocabulary and reasoning segmentation models: X-Decoder [41], SEEM [42], VL-Part [28], and LISA [16]. X-Decoder and SEEM are strong baselines for open-vocabulary and referring segmentation; we evaluate them using checkpoints with the Focal-L [35] backbone. VL-Part, in contrast, is a specialist model for open-vocabulary part segmentation, trained jointly on multiple object-level and part-level datasets (LVIS [10], PACO, PartImageNet, and Pascal-Part) with a SwinBase Cascade Mask R-CNN architecture [6, 20]. We include it as a reference for part-focused specialists. Finally, we include LISA [16], which integrates the language generation capabilities of MLLMs with mask prediction, serving as a representative grounding-specialist MLLM. LISA is also trained on part-level datasets such as PACO and Pascal-Part, and we use the LISA-7B-v1-explanatory checkpoint from the official repository.

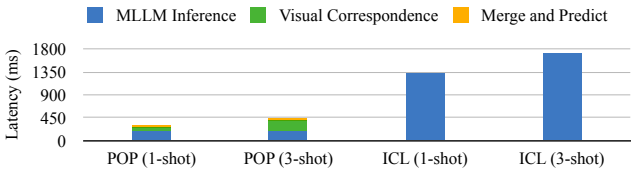

Figure 5. Latency Analysis using Qwen2.5-VL-7B on PACO. POP achieves a lower end-to-end latency compared to ICL.

For the few-shot baseline, we first evaluate pointing-capable MLLMs under an in-context learning setup [5], where support examples are provided directly in the context. In addition, we consider two strong training-free few-shot segmentation models, Matcher [19] and GF-SAM [37]. Both leverage visual foundation models (DINOv2-ViT-L/14 [22] and SAM [15]) to perform training-free few-shot segmentation, making them suitable baselines for our setting. For fairness, we replace DINOv2-ViT-L/14 with DINOv3-ViT-L/16 [27] and set the input resolution for DINOv3 to 1024 in our experiments. During evaluation, these models are provided with full support masks, whereas our method only receives a single support point, following our problem formulation.

For segmentation models that output masks, we consider two strategies for extracting a representative point: selecting the pixel with the maximum logit and selecting the innermost point determined by the distance transform [4]. We observe that the maximum-logit strategy generally yields better results, so we adopt it as the default. For models that only produce binary masks, such as VL-Part, Matcher, and GF-SAM, we instead rely on the distance-transform method.

C.3. Prompt Template

In this section, we first describe the problem format for each dataset and then present the prompt templates used in this work. We follow the official prompt formats from the corresponding repositories or papers as closely as possible.

For evaluation, we design problem prompts tailored to each dataset. For PACO, where referring expressions are often short sentences, we use the template: *"For {referring expression}, point to its {part}."* For InstructPart and PartImageNet++, we adopt the template: *"Point to the {object}'s {part}."* Only minimal modifications are made to align with the official recommendations. Across all ex-

Table 8. Ablation study on the effect of different image encoder.

Method	MLLM 0-shot	POP (DINOv2)	POP (DINOv3)
Molmo-7B-D	51.2	53.4	55.8
Qwen2.5-VL	47.7	49.3	51.6
Ovis-2.5-9B	47.5	51.1	53.2

periments, we maintain a consistent problem format for fair comparison across models. For non-pointing-capable MLLMs, we replace "point to" with "locate" to align with their native prompt format for visual grounding tasks.

For each model, we follow its official prompt specification as closely as possible. Using the PACO dataset and the query *"For the laptop, point to its keyboard."* as an example, Fig. 6, Fig. 7, and Fig. 8 show the exact prompt formats used to evaluate Qwen2.5-VL-7B, Ovis2.5-9B, and Molmo-7B-D respectively. For non-pointing-capable MLLMs, Fig. 9 and Fig. 10 present the corresponding formats used for InternVL-3-8B and Kimi-VL-A3B.

D. Additional Results and Analysis

In this section, we provides additional experimental results and analyses, including latency analysis and an ablation study on the effects of different visual encoders.

D.1. Latency Analysis

We measure the average latency on the PACO dataset using Qwen2.5-VL-7B with CUDA events and GPU synchronization on an RTX A6000 GPU. As shown in Fig. 5, in-context learning (ICL) is slower due to autoregressive decoding and the longer context introduced by support examples, while POP requires only a single MLLM forward pass. Although POP incorporates DINOv3, we observe that it does not introduce significant overhead. Based on these factors, POP achieves a lower end-to-end latency compared to ICL.

D.2. Effect of Different Visual Encoder

We evaluate POP with DINOv2 on PACO dataset. As shown in Tab. 8, POP with DINOv2 consistently outperforms the MLLM baselines. Moreover, we observe that DINOv3 consistently performs better than DINOv2, indicating that the performance of POP can improve further as more powerful visual encoders become available.

E. More Qualitative Results

We present several qualitative results to illustrate the effectiveness of our approach. Fig. 11, Fig. 12, and Fig. 13 show representative examples from Qwen2.5-VL-7B, Ovis2.5-9B, and Molmo-7B-D. Ground-truth regions are highlighted with red masks, and for clearer visualization, the prediction is shown on a cropped image.

F. Limitations and Future Work

In this section, we discuss the limitations of our approach and potential directions for future work.

1. Our current evaluation is limited to tasks requiring a single-point prediction. Extending the method to multi-target scenarios is an important avenue for future work.
2. The effectiveness of our approach depends on the quality of the underlying attention maps. When they are not well aligned with semantic parts, incorporating few-shot exemplars yields only modest gains.
3. Predicted points correspond to patch centers, so localization accuracy is constrained by encoder resolution: even when the correct patch is identified, the predicted center may deviate from the true location.

```
System: You are a helpful assistant
User: For laptop with blue case, point to its keyboard by returning its 2D point
      coordinates in JSON format.
Assistant:
```

Figure 6. Prompt format used for Qwen2.5-VL-7B evaluation on the PACO dataset.

```
User: For laptop with blue case, point to <ref>its keyboard</ref>. Please provide the
      point coordinates.
Assistant:
```

Figure 7. Prompt format used for Ovis2.5-9B evaluation on the PACO dataset.

```
User: For laptop with blue case, point to its keyboard.
Assistant:
```

Figure 8. Prompt format used for Molmo-7B-D evaluation on the PACO dataset.

```
User: For laptop with blue case, locate the region this sentence describes: <ref>
      its keyboard </ref>.
Assistant:
```

Figure 9. Prompt format used for InternVL-3-8B evaluation on the PACO dataset.

```
User: For laptop with blue case, locate its keyboard.
Assistant:
```

Figure 10. Prompt format used for Kimi-VL-A3B evaluation on the PACO dataset.

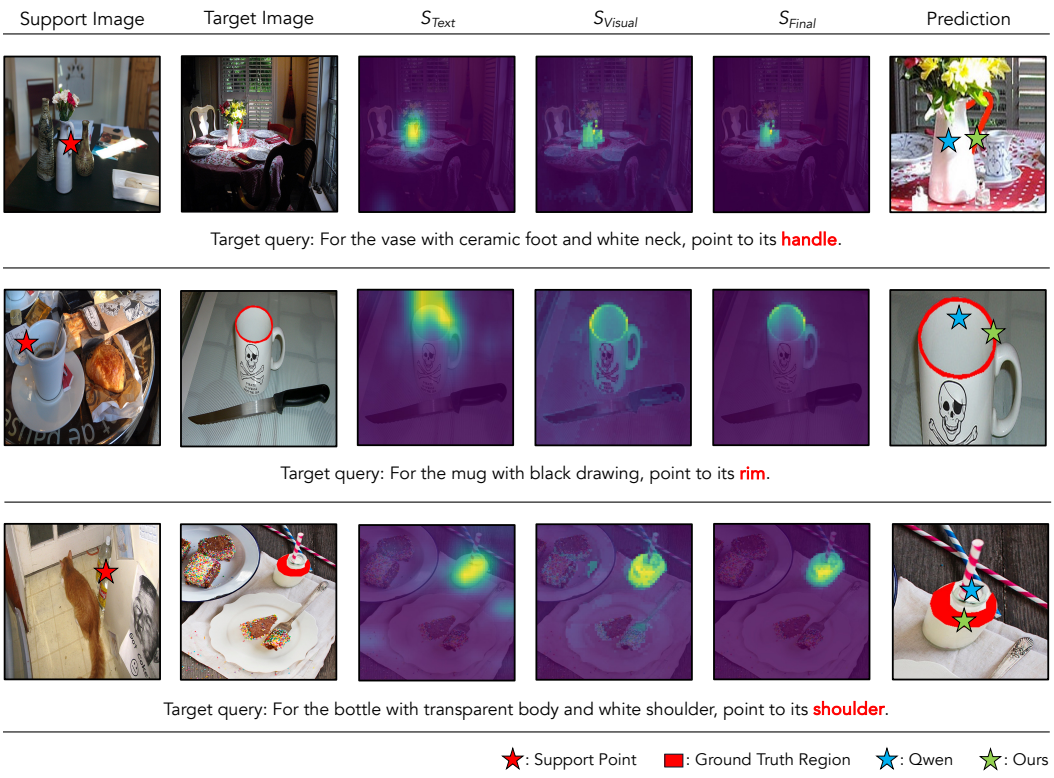

Figure 11. Qualitative results of our method with Qwen2.5-VL-7B. For clearer visualization, the prediction is shown on a cropped image.

Figure 12. Qualitative results of our method with Ovis2.5-9B. For clearer visualization, the prediction is shown on a cropped image.

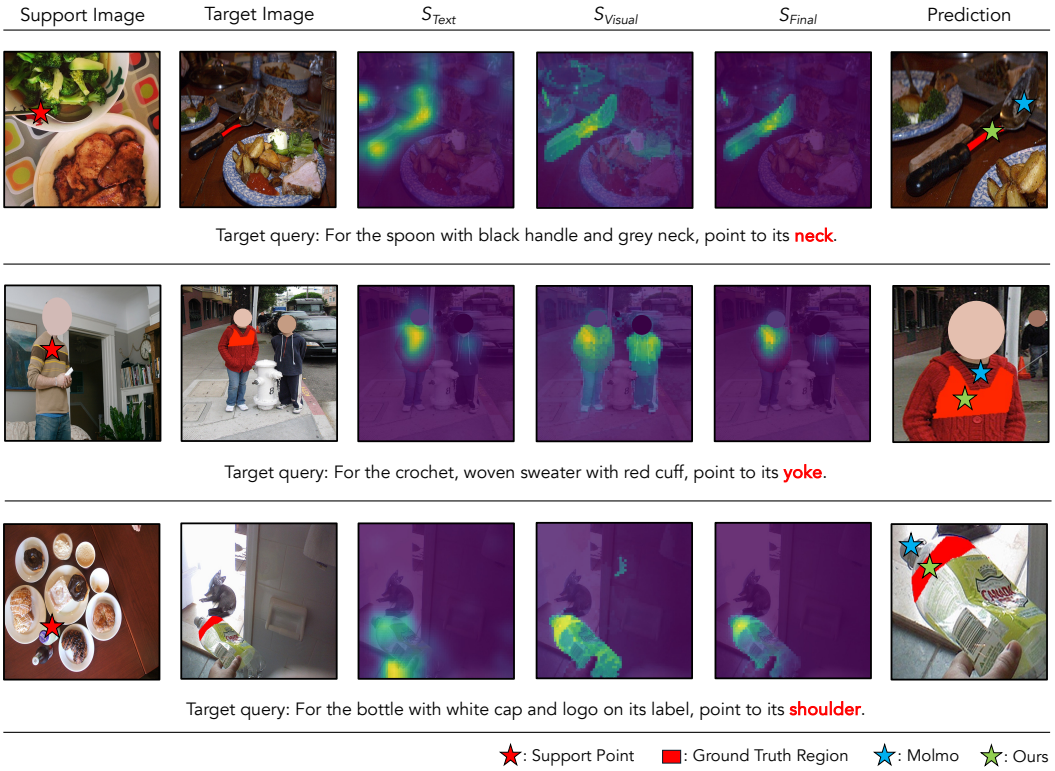

Figure 13. Qualitative results of our method with Molmo-7B-D. For clearer visualization, the prediction is shown on a cropped image.