PARALLEL INSTANCE DECODING FOR LANGUAGE-GUIDED LOCALIZATION IN AERIAL IMAGES

Anonymous authors
Paper under double-blind review

ABSTRACT

Earth observation (EO) localization is challenging due to large scene sizes, complex backgrounds, large variations in object scale and varying numbers of target objects. Existing remote-sensing vision-language models (RSVLMs) typically perform localization tasks, such as grounding, detection, and pointing by autoregressively generating bounding-box coordinates. As the object count increases, this step-by-step decoding becomes slow because boxes are produced one at a time and each prediction relies on earlier ones. To address this limitation, we introduce Parallel Instance Decoding (PID), a unified generative localization framework that separates instance identification from spatial localization. Given a textual reference (open vocabulary query), PID first generates a variable-length set of point addresses, with one address representing each matching object instance. It then predicts the bounding boxes for all identified instances in a single parallel decoding pass. To enable independent box prediction, we use an instance-isolated attention mask that allows each box to attend to the full image and its corresponding address while preventing attention to other boxes. Under the same training data and experimental setup, PID improves F1@0.50 over the direct-box baseline by 8.3 points on DIOR and 7.3 points on the DOTAv2 dataset. It also reduces complete-response latency by 22.4% and 43.6%, respectively. Moreover, PID performs favorably well against general-purpose VLMs and specialized RSVLMs on grounding and pointing tasks. We will publicly release our code and models.

1 INTRODUCTION

Localization in Earth observation (EO) often requires machine learning models to identify targets in remote sensing imagery, determine their locations, and estimate their spatial extent. This is particularly challenging as objects in satellite imagery are often small and densely distributed, vary greatly in scale, and are surrounded by repetitive structures that can cause visual ambiguity (Xia et al., 2018; Yuan et al., 2023). For vision-language models (VLMs), the model must also understand object attributes and spatial relationships to identify the object instances that match the given description (Zhan et al., 2023; Li et al., 2024; Xia et al., 2018; Yuan et al., 2023). Depending on the input text, the correct output may be a single object or multiple object instances distributed across the image.

Recent remote-sensing VLMs have improved category-conditioned detection and expression-conditioned grounding through large-scale multimodal supervision and stronger spatial representations (Pan et al., 2025; Zhou et al., 2024; Zhang et al., 2026). Many of these methods formulate localization as a sequence-generation task, where the model outputs semantic and coordinate tokens to represent detected objects. Some approaches can predict multiple coordinate tokens of a single bounding box in parallel. However, object instances are typically generated autoregressively, where the model determines which object to output, completes its bounding box, and then proceeds to the next object. Thus, even when the coordinates of each box can be predicted in parallel, different objects are still localized one after another. This sequential prediction creates dependencies between objects, since later predictions are conditioned on the previously generated outputs. In dense aerial scenes that contain many objects, this increases the decoding time, and propagates errors across sequential generations of boxes which may exhibit large semantic and spatial variations.

We introduce *Parallel Instance Decoding* (PID), a unified framework for text-conditioned localization (Fig. 1 (B)). Given an image and a textual reference (open vocabulary query), PID first identifies the

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.

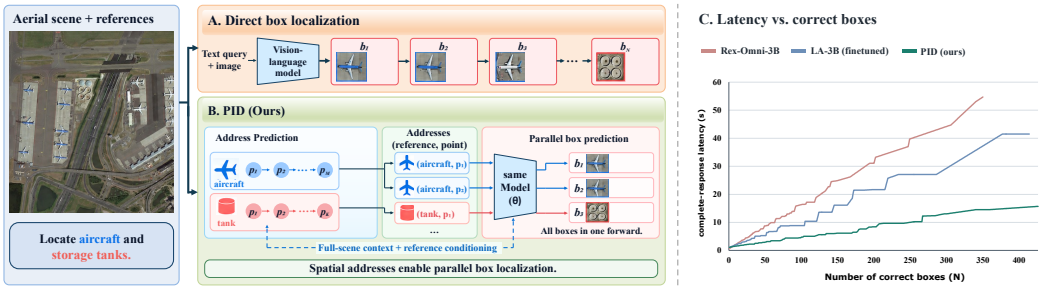

Figure 1: Parallel Instance Decoding (PID). **(A)** Direct box localization decodes boxes sequentially, even when coordinates within a box are predicted in parallel. **(B)** PID first predicts one point address per object instance for each textual reference (e.g., class name), and then predicts the boxes of all addresses in a single parallel pass using instance-isolated attention. **(C)** Complete-response latency on DOTAv2 dataset versus the number of correctly predicted boxes per image. The latency of direct-box models grows steeply as images contain more objects, whereas PID's latency grows much more slowly, because all boxes are predicted in one parallel pass.

matching object instances by predicting a variable-length set of point *addresses*, with one address assigned to each instance. These addresses then serve as instance-specific cues for localization, where each address is combined with the textual reference and shared image context to predict the bounding box of its corresponding instance. Since each box is determined only by its own address, all boxes can be decoded in parallel. We enforce this with instance-isolated attention, an attention mask that lets each box attend to the full image and its own address but not to other boxes. This removes the bottleneck of sequential box decoding, and a wrong box cannot influence any other box. Address prediction, however, remains autoregressive within each reference.

We represent addresses as predicted spatial anchor points, following point-based object representations and language-conditioned pointing (Zhou et al., 2019; Deitke et al., 2025). In contrast to settings where candidates follow a predefined external ordering, such as temporal indices in video (Rasheed et al., 2026), aerial images do not provide a fixed set or ordering of object instances. Therefore, Parallel Instance Decoding predicts addresses directly from the image and textual reference, while a null option allows it to reject unsupported candidates. These predicted addresses then guide box prediction. Each address-conditioned box is predicted using the full image and textual reference, while being masked from the geometry outputs of other object instances. This design avoids dependencies between boxes and enables parallel instance decoding in a single pass.

We evaluate Parallel Instance Decoding on remote-sensing detection, expression-conditioned grounding, and pointing benchmarks, considering both localization quality and end-to-end response efficiency. The proposed approach provides a unified interface for localizing targets specified by either category references or free-form expressions. In dense detection settings, it further improves detection throughput via parallel extent decoding of object instances.

Our main contributions are:

- **Point-addressed localization.** We introduce a localization approach that separates *which* object instances should be detected from *where* their bounding boxes should be placed. The model first predicts a point address for each referred object and then predicts one bounding box conditioned on each address.
- **Parallel address-conditioned box prediction.** We train a single model to predict both point addresses and their bounding boxes. With instance-isolated attention, each box depends only on the image, its reference, and its own address, so all boxes can be predicted in one parallel pass.
- **Improved accuracy and decoding efficiency.** Under the same data, training curriculum, and trainable components, PID outperforms the direct-box baseline by 8.3 F1@0.50 points on DIOR and 7.3 points on DOTAv2, while doubling small-object recall on DIOR. PID also reduces complete-response latency by 22.4% on DIOR and 43.6% on DOTAv2 relative to the direct-box baseline, with a larger advantage in the denser DOTAv2 scenes.

2 Related Work

Remote-sensing detection and grounding. Remote-sensing localization has expanded from category-level detection to language-conditioned tasks, including open-vocabulary aerial detection (Pan et al., 2025) and referring-expression grounding benchmarks such as RSVG (Zhan et al., 2023) and VRSBench (Li et al., 2024). In parallel, general vision-language models have been adapted to Earth observation (EO): GeoChat supports region-level queries and grounded dialogue (Kuckreja et al., 2024), GeoGround unifies horizontal boxes, oriented boxes, and masks (Zhou et al., 2024), and EarthGPT, SkyEyeGPT, GeoPixel, and EarthDial extend instruction tuning to EO data (Zhang et al., 2024; Zhan et al., 2025; Shabbir et al., 2025; Soni et al., 2025). More recent work emphasizes geospatial reasoning (Liu et al., 2026) and reward-guided search for small targets in large scenes (Zhang et al., 2026). Our focus is complementary: we study how to structure decoding when a single textual reference may match an unknown (and potentially large) number of object instances.

Generative localization and point-based anchors. Many modern systems either (i) use detector-style architectures with language conditioning, such as Grounding DINO (Liu et al., 2024), or (ii) generate boxes as sequences of tokens, as in Pix2Seq and SeqTR (Chen et al., 2021; Zhu et al., 2022) and multimodal LLM localizers such as Shikra, Kosmos-2, Ferret, Florence-2, and Rex-Omni (Chen et al., 2023; Peng et al., 2023; You et al., 2024; Xiao et al., 2024; Jiang et al., 2026). In sequence-generation interfaces, the model must decide which objects to output while it is also generating box geometry, which typically leads to sequential box-by-box decoding. Points provide a compact alternative representation: they are used as object centers (Zhou et al., 2019), as point-level supervision for detection (Chen et al., 2022), and as outputs for language-conditioned pointing in Molmo, Molmo2, and MolmoPoint (Deitke et al., 2025; Clark et al., 2026b;a); Groma and PaDT further explore region- and patch-level tokens for grounding (Ma et al., 2024; Su et al., 2026). PID uses points as *addresses*: it first predicts which object instances to localize as textual reference-conditioned points, and then predicts one box per point in parallel.

Reducing sequential decoding cost. Several works attempt to reduce the sequential cost of generating geometry. LocateAnything introduces Parallel Box Decoding, which predicts each box as an atomic unit in a single decoding step rather than coordinate-by-coordinate (Wang et al., 2026); however, boxes are still produced one after another. For spatio-temporal video grounding, Parallel Tube Decoding uses time indices as predefined output slots: it predicts the temporal interval and then decodes all per-frame boxes in parallel (Rasheed et al., 2026). Aerial detection and grounding do not provide a natural index over candidate objects, so PID instead learns the output slots by first predicting point addresses, and then predicting their boxes in a single parallel pass.

3 Method

We introduce Parallel Instance Decoding, a unified framework for remote-sensing localization built on LocateAnything-3B (Wang et al., 2026), which connects a MoonViT vision encoder (Team et al., 2025) to a Qwen2.5 language model (Qwen et al., 2025) through an MLP projector. Given an image and a textual reference, Parallel Instance Decoding first predicts a variable-length set of point *addresses*, each identifying a relevant object instance. For each address, the model then predicts a bounding box conditioned on the address, shared image context, and textual reference. This separates instance selection from box prediction and allows the boxes to be predicted in parallel. Both components are implemented within the same model and trained jointly (Fig. 2).

3.1 From Sequential Boxes to Point-Addressed Boxes

Task formulation. The input is an aerial image I and a set of C textual references $\mathcal{Q} = (q_1, \ldots, q_C)$. A reference q_c may be a category label, as in object detection, or a free-form referring expression, as in visual grounding. For each reference q_c, the goal is to output the bounding boxes of all m_c object instances it refers to, where the count m_c is unknown in advance and may be zero. The model must therefore infer both *how many* object instances each reference matches and *where* they are. This is challenging in large aerial scenes with dense, repetitive, and visually similar objects.

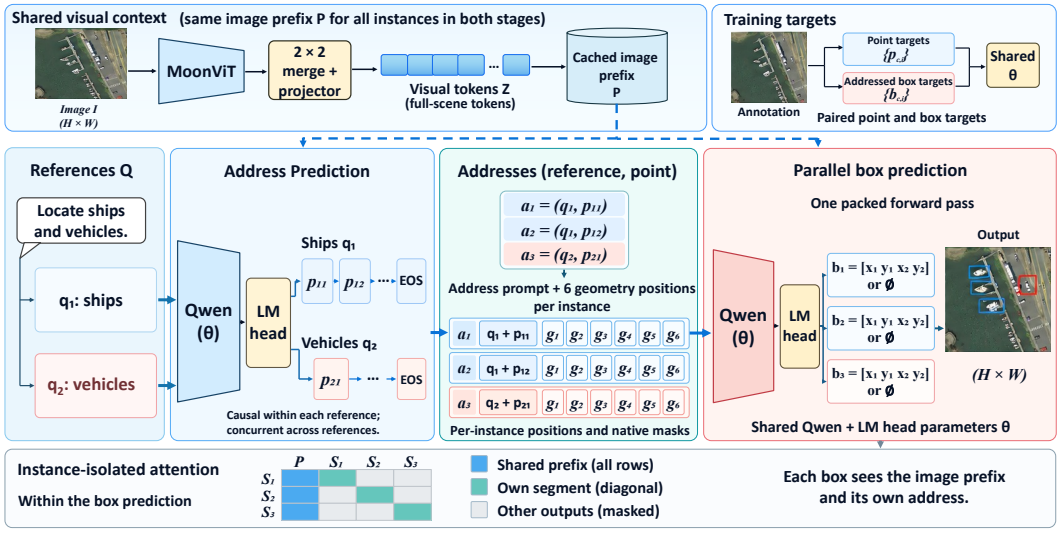

Figure 2: Parallel Instance Decoding (PID) overview. The model first proposes textual reference-conditioned address points and then predicts one horizontal bounding box (HBB) per address. A shared image prefix provides global context, and masking prevents each address-conditioned box prediction from depending on other predicted boxes, enabling parallel decoding.

Conventional box localization. A conventional generative localizer first encodes the image into visual tokens $Z = V_\theta(I)$, where V_θ is the visual encoder and θ denotes all model parameters. It then generates boxes as a single output sequence $B = (b_1, \ldots, b_M)$, where M is the total number of generated boxes, with each box conditioned on previously generated boxes:

$$p_\theta(B \mid Z, \mathcal{Q}) = \prod_{j=1}^{M} p_\theta(b_j \mid Z, \mathcal{Q}, b_{<j}), \tag{1}$$

where $b_{<j} = (b_1, \ldots, b_{j-1})$. Parallel Box Decoding (PBD) (Wang et al., 2026) predicts the coordinates of an individual box jointly, but different boxes are still produced sequentially. As a result, selecting the next object instance to output remains entangled with estimating its geometry, and the number of decoding steps grows with the number of object instances.

Point-addressed box Localization. We represent each candidate object instance by an *address* $a_{c,i} = (q_c, p_{c,i})$, which pairs the reference q_c with a point $p_{c,i} = (u_{c,i}, v_{c,i})$ inside the i-th candidate. For each reference, the model outputs a variable-length list of points $A_c = (p_{c,1}, \ldots, p_{c,n_c})$, where n_c is determined at inference time and need not match the ground-truth count m_c.

Given these addresses, the model predicts one output $b_{c,i}$ per address. Each output is either a horizontal bounding box in normalized image coordinates or a null response $\varnothing$ indicating that the address does not correspond to a valid instance. Denoting all addresses by $A = (A_1, \ldots, A_C)$ and all box outputs by B, PID factorizes the prediction as

$$p_\theta(A, B \mid Z, \mathcal{Q}) = \prod_{c=1}^{C} \Big[\underbrace{p_\theta(A_c \mid Z, q_c)}_{\text{address prediction}} \prod_{i=1}^{n_c} \underbrace{p_\theta(b_{c,i} \mid Z, q_c, p_{c,i})}_{\text{box prediction}} \Big]. \tag{2}$$

The address term is autoregressive within each reference: each new point depends on the points already produced for that reference. The box term predicts each box from the image, its reference, and its own address point, without conditioning on other boxes. This removal of inter-box dependence enables parallel instance decoding.

3.2 Our Parallel Instance Decoding

Step 1: Address prediction. For each reference q_c, the model generates address points A_c autoregressively, with each point conditioned on the previous points of the same reference; different

references are decoded concurrently. Coordinates are integers in $[0, 1000]$, and a point is written as `<box><u><v></box>`. Each point sequence ends with EOS, and a reference with no matching instance emits EOS immediately.

Step 2: box prediction from addresses. For each address $(q_c, p_{c,i})$, the model predicts a box `<box><x1><y1><x2><y2></box>`, or `<box>None</box>` if the addressed object is not supported by the image. Coordinates are predicted jointly with the base model's native PBD mechanism, which assigns six geometry positions per box (Wang et al., 2026). Since the address identifies the instance, this step only estimates box geometry.

Shared visual context. Both steps use the same backbone and output head on the full image, without cropping. The image-only prefix of the Qwen backbone is computed once and reused in both steps; reference text and generated outputs are excluded from it. For detection and grounding, predicted points serve as addresses for box prediction; for pointing, only Step 1 is run and its points are returned directly.

Instance-isolated attention. To implement Eq. 2 in a single transformer, we isolate instances with an attention mask. The input is a shared image prefix $\mathcal{P}$ (image tokens only) followed by segments $\mathcal{S}_1, \ldots, \mathcal{S}_K$: an *address segment* holds a reference and its points, and a *box segment* holds a reference-address pair and its box. Attention follows three rules (Fig. 2, bottom): (i) prefix tokens attend causally within the prefix; (ii) tokens in $\mathcal{S}_k$ attend to the prefix and within $\mathcal{S}_k$, using the native pattern (causal for addresses, PBD for boxes); and (iii) attention across segments is blocked. Each segment thus sees the full image and its own reference and address, but not other segments' outputs. The same mask is used in training, so box tokens never attend to other instances' geometry.

Parallel box decoding. Because each box depends only on the prefix and its own address, the $N = \sum_c n_c$ box segments are packed into one sequence and decoded in a single forward pass. Each segment is an address prompt followed by six geometry positions, filled with the native anchor and mask tokens rather than target coordinates. The mask and per-address positional indices preserve the conditioning above, so the model is not run separately per address.

Sequential decoding cost. We count sequential forward passes after the shared prefix. The base model, LocateAnything-3B, decodes each box or point as one atomic PBD step (Wang et al., 2026). Direct decoding writes all boxes of all references into one sequence, with $s_{\text{box}} \geq 1$ steps per box (one PBD step plus any label or separator tokens). PID decodes the address lists of all references concurrently, using one step per point and one EOS step per reference, and then predicts all N boxes in one packed pass:

$$D_{\text{direct}} \approx \sum_c n_c \, s_{\text{box}} = N \, s_{\text{box}}, \qquad D_{\text{PID}} \approx \max_c (n_c + 1) + \mathbb{1}[N > 0], \tag{3}$$

where D is the number of sequential passes and $\mathbb{1}[N > 0]$ is 1 if any address is produced and 0 otherwise. PID thus replaces the sum over references with a maximum and removes geometry and label tokens from the sequential path. The gain is largest when many instances span several references; for a single reference with few instances, as in single-target grounding, the extra address step can make PID slightly slower. The computation and memory of the packed pass still grow with N. Positional indexing and attention visibility are detailed in Appendix B.2, and packed decoding in Appendix D.3.

3.3 Joint Localization Learning

Paired supervision. Each training episode contains an image, its references, and annotated instances. Each positive instance yields two aligned targets: an *address target* (where the address should be placed) and a *box target* (the box predicted from it). In detection, a category reference is supervised with all its annotated instances; in grounding, each expression keeps its original target. We also use two negative targets: references with no matching instance must emit EOS immediately, and unsupported reference-address pairs must return the null response, teaching the model to reject invalid addresses. Episode construction and address collisions are described in Appendix A.1.

Joint objective. Both steps use token-level cross-entropy $\ell_\theta(t)$, computed under the attention mask above. Since the number of supervised tokens varies strongly with scene density, we average the loss separately over address and box tokens in each episode and weight them equally:

$$\mathcal{L}_e = \tfrac{1}{2}\mathcal{L}_{e,\mathrm{addr}} + \tfrac{1}{2}\mathcal{L}_{e,\mathrm{box}}, \qquad \mathcal{L}_{e,r} = \frac{1}{|\mathcal{T}_{e,r}|}\sum_{t\in\mathcal{T}_{e,r}} \ell_\theta(t), \tag{4}$$

where e indexes the episode, $r \in \{\mathrm{addr}, \mathrm{box}\}$ denotes the component, and $\mathcal{T}_{e,r}$ is its set of supervised tokens. This keeps the relative weight of the two components independent of the instance count. Episode losses are averaged over the batch, with no additional direct-box objective. Both components are trained jointly throughout the coarse-to-dense curriculum; stages differ only in data, not in objective. They are coupled through shared parameters and paired targets, and gradients do not pass through discrete point generation. Box prediction is conditioned on annotation-derived addresses during training and on predicted addresses at inference.

4 Experiments

We evaluate our Parallel Instance Decoding (PID) on three remote-sensing localization tasks, namely category-conditioned detection, expression-conditioned grounding, and pointing. PID follows a two-stage process. First, the model predicts a point address for each target instance, providing a spatial reference for localization. It then uses these point addresses to jointly predict the boxes of all target instances in a single forward pass. Our experiments focus on two key aspects of this design, specifically whether point-based instance addressing improves localization accuracy compared with direct box generation and whether joint box prediction reduces the computational cost of producing complete multi-object outputs.

4.1 Experimental Setup

Training stages. We train the model in two stages using a coarse-to-dense curriculum. Stage 1 contains 116,386 episodes, and Stage 2 contains 39,851 episodes, including 10,943 replay episodes (Appendix Table 6). PID is trained using paired address (point) and box targets, without additional pointing-only records. Each stage is trained for two epochs with an effective batch size of 32. After each stage, we select the checkpoint with the lowest validation loss averaged across detection and grounding. The selected Stage 2 checkpoint is then used for all downstream tasks. For a controlled comparison, the direct-box baselines use the same training data, curriculum, trainable components, and checkpoint-selection procedure as PID. They differ only in the output format and corresponding supervision targets. Additional details on the training data and optimization settings are provided in Appendix A and Appendix Table 8.

Benchmarks and query scope. We evaluate detection, grounding, and pointing across four remote-sensing benchmarks. For detection, we use 586 DIOR and 874 DOTAv2 images (Pan et al., 2025), resulting in 861 and 1,618 image-category pairs, respectively. Each query contains a category annotated in the corresponding image but does not provide object counts, points, or bounding boxes. This setup focuses the evaluation on localizing queried categories rather than identifying categories that are absent from the image. For grounding, we use 7,500 DIOR-RSVG and 16,159 VRSBench-VG image-expression pairs with their original referring expressions (Zhan et al., 2023; Li et al., 2024). Finally, pointing is evaluated on both settings: category queries are used for the detection benchmarks, while referring expressions are used for the grounding benchmarks.

Metrics. We follow the LocateAnything evaluation protocol (Wang et al., 2026), with one modification: for detection, a prediction is considered a match only when both the localization and category are correct (Appendix C). We evaluate detection using F1@0.50, F1@0.75, and mF1, where mF1 is the average F1 score over IoU thresholds from 0.50 to 0.95 in steps of 0.05. For grounding, we use the first valid predicted box; for PID, this corresponds to the box obtained from the first predicted address. We report Acc@0.5, Acc@0.7, and mIoU, with an IoU of zero assigned to missing or invalid predictions. For pointing, we report F1@Point with one-to-one matching. A predicted point is considered correct if it falls within the corresponding ground-truth HBB. All accuracy values are reported as percentages. Additional evaluation details are provided in Appendix C.

4.2 RESULTS

Horizontal-box detection. Table 1 presents category-conditioned detection results on aerial imagery. Compared with our finetuned direct-box baseline LA-3B (FT)†, PID improves performance on all metrics. In particular, PID increases mF1 by $+5.97$ on DIOR and $+7.40$ on DOTAv2. On DOTAv2, it also improves F1@0.75 by $+7.82$, so the gains hold under a stricter localization threshold. Among all methods, PID achieves the best overall performance on DIOR. It matches LAE-DINO at F1@0.50 (82.24 vs. 82.20) and outperforms it by more than 10 points at F1@0.75 and in mF1. On the denser DOTAv2 dataset, Rex-Omni achieves the highest F1@0.50 among VLMs (70.12 vs. 66.90 for PID), while PID is slightly ahead at F1@0.75 (44.15 vs. 44.02) and in mF1 (42.68 vs. 42.56). LAE-DINO, which is pretrained on the large-scale LAE-1M corpus, achieves the strongest overall results on DOTAv2. Given this additional pretraining, we use LA-3B (FT)†, trained on the same data as PID, as the primary controlled comparison for evaluating our formulation.

Table 1: Category-conditioned horizontal bounding-box detection under annotated-category queries. mF1 denotes mean F1 over ten IoU thresholds (0.50-0.95, step 0.05). **Bold**: best; underline: second best.

	DIOR			DOTAv2		
Model	F1@.5	F1@.75	mF1	F1@.5	F1@.75	mF1
Specialized detectors						
Grounding DINO-T	27.48	18.44	18.03	17.17	8.58	9.12
LAE-DINO	**82.20**	**58.98**	**54.75**	**72.79**	**50.64**	**46.94**
Autoregressive VLMs						
Qwen3-VL-4B	65.85	45.80	44.48	42.50	18.42	21.29
Qwen3-VL-8B	64.90	45.05	43.72	40.64	17.73	20.46
Rex-Omni-3B	54.53	44.00	40.87	**70.12**	44.02	42.56
LA-3B	54.16	40.92	39.40	36.93	18.17	19.53
LA-3B (FT)†	73.94	63.75	59.94	59.62	36.33	35.28
PID (ours)	**82.24**	**69.83**	**65.91**	66.90	**44.15**	**42.68**

Expression-conditioned grounding. Table 2 evaluates single-target localization using referring expressions. PID outperforms LA-3B (FT)† across all metrics on both benchmarks. Although the improvements are modest (up to $+0.77$), they indicate that the instance-addressed formulation preserves strong single-instance localization performance. Compared with general-purpose VLMs, PID achieves more than 10 points higher Acc@0.5. On VRSBench-VG, PID also surpasses the strongest remote-sensing-specific method, GeoViS*, by $+7.75$ in Acc@0.5 and $+11.51$ in Acc@0.7. On DIOR-RSVG, GeoViS* performs slightly better in Acc@0.5 and mIoU, while PID achieves the highest Acc@0.7. Results marked with * are taken from prior work under their original evaluation protocols and are included for reference.

Pointing. Table 3 evaluates point prediction directly, rather than deriving points from predicted boxes. We consider two query types: category queries, which require a point on every instance of a class, and expression queries, which require a point on one referred instance. PID improves over LA-3B (FT)† on all four benchmarks. The gains are largest in the denser category-conditioned settings: $+3.52$ F1 on DIOR and $+7.68$ F1 on DOTAv2. Because PID is explicitly trained to predict one point per instance as its address, this supervision carries over to pointing, even though no pointing-only data are used. On DOTAv2, Rex-Omni achieves a slightly higher F1 than PID (70.12 vs. 68.28).

4.3 INFERENCE EFFICIENCY

Table 2: Single-target referring-expression grounding on DIOR-RSVG and VRSBench-VG. Metrics are Acc@0.5, Acc@0.7, and mean IoU (mIoU). **Bold**: best; underline: second best. † indicates our finetuned variant and * indicates results reported in prior work.

	DIOR-RSVG			VRSBench-VG		
Model	Acc@.5	Acc@.7	mIoU	Acc@.5	Acc@.7	mIoU
Qwen3-VL-4B	47.08	34.61	43.80	51.98	31.98	45.86
Qwen3-VL-8B	51.24	37.93	47.53	55.70	34.20	49.13
Rex-Omni-3B	53.25	41.87	47.78	65.90	48.02	56.75
LA-3B	41.87	30.84	41.13	50.94	33.37	45.86
GeoGround*	77.70	-	-	49.80	20.00	-
GeoViS*	**79.80**	70.10	**72.60**	68.50	45.70	59.20
LA-3B (FT)†	78.77	70.08	69.82	75.63	56.80	64.91
PID (ours)	78.99	**70.67**	70.59	**76.25**	**57.21**	**65.11**

Table 3: Pointing performance (F1) for category-conditioned (DIOR, DOTAv2) & expression-conditioned (RSVG, VRSBench-VG) queries. † indicates our finetuned variant.

	Category		Expression	
	DIOR	DOTAv2	RSVG	VRSB.
Model	F1@Point			
Qwen3-VL-4B	73.34	47.89	65.09	71.53
Qwen3-VL-8B	77.71	49.85	67.25	72.79
Rex-Omni-3B	73.12	**70.12**	70.05	78.32
Molmo-7B-D	73.20	45.02	64.85	68.78
LA-3B	69.22	53.84	59.66	67.43
LA-3B (FT)†	81.79	60.60	82.68	86.17
PID (ours)	**85.31**	68.28	**83.64**	**86.53**

We measure the efficiency of PID under parallel extent localization on NVIDIA A100 PCIe 40 GB GPUs. CUDA-synchronized timings include preprocessing, visual encoding, generation, and output parsing, excluding loading, initialization, warm-up, queueing, and scoring. Table 4 reports latency (TPI) and throughput (BPS and CBPS); these measurements are separate from the final rotary-stabilized detection evaluation.

Table 4: Complete-response detection efficiency under annotated-category queries. TPI: seconds per image; BPS/CBPS: valid/correctly matched boxes per second (IoU 0.50 for correctness). Rates are reported per worker; full results for all methods are in Appendix Table 15. † indicates our finetuned variant. **Bold**: best; underline: second best.

	DIOR			DOTAv2		
Method	TPI↓	BPS↑	CBPS↑	TPI↓	BPS↑	CBPS↑
Rex-Omni-3B	1.614	4.18	1.98	19.704	6.32	0.73
LA-3B (FT)†	0.990	3.99	2.90	3.930	10.01	3.27
PID (ours)	**0.768**	**6.99**	**5.50**	**2.218**	**13.30**	**8.33**

PID achieves the lowest latency and highest throughput on both datasets. Relative to LA-3B (FT)†, PID reduces latency by 22.4% on DIOR and 43.6% on DOTAv2, while increasing CBPS by $1.90\times$ and $2.55\times$, respectively. Against Rex-Omni-3B, it is $2.10\times$ and $8.88\times$ faster, respectively, with $11.4\times$ higher CBPS on DOTAv2.

The gain increases with scene density: from DIOR to DOTAv2, latency grows by $12.2\times$ for Rex-Omni, $4.0\times$ for LA-3B (FT)†, and only $2.9\times$ for PID. This follows Eq. 3: direct-box models scale sequentially with the total number of boxes N, whereas PID depends only on the largest reference and predicts all boxes in one packed pass.

4.4 Ablation Studies

Direct vs. instance-addressed localization. Table 5A compares joint category prediction (LA-3B (FT)†), class-isolated prediction (LA-3B (FT, class-iso.)§), and instance-addressed prediction (PID) under the same training setup. Class isolation yields modest F1@0.50 gains ($+1.51$ DIOR; $+2.04$ DOTAv2), whereas PID adds $+5.84$ and $+5.64$ mF1 over class isolation. This indicates a larger contribution from instance addressing, consistent with the inference-only results in Appendix C.6 (Table 14). However, PID also uses box-center point supervision at no additional annotation cost, so its effect cannot be fully separated from instance addressing.

Table 5: Ablation of instance-addressed localization. (A) Output formulations under matched training: LA-3B (FT)† jointly predicts boxes for all requested categories, while LA-3B (FT, class-iso.)§ predicts boxes for one category at a time. (B) PID with predicted versus annotation-derived addresses using the same weights. Δ denotes the absolute gain from annotation-derived addresses. **Bold**: best; underline: second best within each block.

	DIOR			DOTAv2		
Model	F1@.5	F1@.75	mF1	F1@.5	F1@.75	mF1
(A) Output formulation (matched training)						
LA-3B (FT)†	73.94	63.75	59.94	59.62	36.33	35.28
LA-3B (FT, class-iso.)§	75.45	63.69	60.07	61.66	38.57	37.04
PID (ours)	**82.24**	**69.83**	**65.91**	**66.90**	**44.15**	**42.68**
(B) Address source (fixed PID weights)						
PID, predicted addr.	82.24	69.83	65.91	66.90	44.15	42.68
PID, annotation addr.	91.32	76.58	72.25	89.22	55.71	54.26
Δ	+9.08	+6.75	+6.34	+22.31	+11.55	+11.58

Predicted vs. annotation-derived addresses. Table 5B evaluates address quality. Replacing predicted with annotation-derived addresses improves mF1 by $+6.34$ on DIOR and $+11.58$ on DOTAv2, suggesting that address prediction is a larger error source in denser scenes. Because these addresses provide counts and approximate locations, but not box boundaries, the comparison does not fully separate address-prediction from box-localization errors.

4.5 Qualitative Results

Figure 3 compares PID with Rex-Omni-3B and LA-3B (FT)† on the three tasks. In the urban detection scene with 66 annotated objects, PID returns 59 correct boxes, compared with 53 for Rex-Omni and 50 for LA-3B (FT)†, and it is the only model that localizes the roundabout. In grounding, Rex-Omni selects the wrong ship (IoU 0.09), whereas PID and LA-3B (FT)† select the correct one, with PID fitting it more tightly (IoU 0.86 vs. 0.77). In the harbor pointing scene with 78 annotated ships, PID returns 74 points, compared with 70 and 62. Overall, PID covers dense scenes more completely and localizes targets more precisely. Additional qualitative examples are provided in Appendix E.



Figure 3: Qualitative comparison on detection (top; DOTAv2), grounding (middle; VRSBench-VG), and pointing (bottom; DOTAv2) across ground truth, Rex-Omni-3B, LA-3B (FT)†, and PID. Colors denote detection categories; boxes denote grounding targets/predictions; and dashed boxes and dots denote pointing targets/predictions. Labels show counts or grounding IoU.

5 Limitations

PID relies on accurate point-address prediction; missing, repeated, or ambiguous addresses can lead to missed, duplicate, or incorrect box predictions. This sensitivity is also affected by the quality of the object inventory and point locations, while overlapping HBBs can make object identification ambiguous in dense scenes. Our evaluation is limited to HBBs and points and does not consider oriented geometry. In addition, autoregressive address prediction and the growing memory cost of parallel box prediction may limit efficiency, particularly as the number of objects increases.

6 Conclusion

We presented Parallel Instance Decoding, a unified framework for text-conditioned localization in remote-sensing imagery. By identifying instances with point addresses before box prediction, it separates instance identification from localization and enables parallel decoding. Across benchmarks, Parallel Instance Decoding improves multi-instance detection, response efficiency, and pointing while remaining competitive in single-expression grounding, demonstrating the effectiveness of point-based addressing for variable-cardinality localization.

AI USE STATEMENT

We did not use generative AI tools for any activities subject to mandatory disclosure under the ICLR AI policy. We used ChatGPT to improve the clarity and readability of manuscript drafts and Codex to support code debugging. All AI-assisted outputs were reviewed by the authors: AI-assisted text was carefully checked, and AI-assisted code was validated and tested for correctness. The authors take full responsibility for the final content of this work, including all text, claims, and artifacts produced with the assistance of generative AI.

REPRODUCIBILITY STATEMENT

We provided the necessary details to support the reproducibility of our work. The main paper describes the proposed method and experimental setup. Appendices A, B, and C provide additional details on the training, inference, and evaluation protocols, respectively. Appendix D.2 describes the timing protocol used for the efficiency measurements. Together, these details specify the procedures used to obtain the results reported in this work.

REFERENCES

Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. Shikra: Unleashing multimodal llm's referential dialogue magic. *arXiv preprint arXiv:2306.15195*, 2023.

Pengfei Chen, Xuehui Yu, Xumeng Han, Najmul Hassan, Kai Wang, Jiachen Li, Jian Zhao, Humphrey Shi, Zhenjun Han, and Qixiang Ye. Point-to-box network for accurate object detection via single point supervision. In *European Conference on Computer Vision*, pp. 51–67. Springer, 2022.

Ting Chen, Saurabh Saxena, Lala Li, David J Fleet, and Geoffrey Hinton. Pix2seq: A language modeling framework for object detection. *arXiv preprint arXiv:2109.10852*, 2021.

Christopher Clark, Yue Yang, Jae Sung Park, Zixian Ma, Jieyu Zhang, Rohun Tripathi, Mohammadreza Salehi, Sangho Lee, Taira Anderson, Winson Han, et al. Molmopoint: Better pointing for vlms with grounding tokens. *arXiv preprint arXiv:2603.28069*, 2026a.

Christopher Clark, Jieyu Zhang, Zixian Ma, Jae Sung Park, Rohun Tripathi, Sangho Lee, Mohammadreza Salehi, Jason Ren, Chris Dongjoo Kim, Yinuo Yang, et al. Molmo2: Open weights and data for vision-language models with video understanding and grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 28652–28668, 2026b.

Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, et al. Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 91–104. IEEE, 2025.

Qing Jiang, Junan Huo, Xingyu Chen, Yuda Xiong, Zhaoyang Zeng, Yihao Chen, Tianhe Ren, Junzhi Yu, and Lei Zhang. Detect anything via next point prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 25472–25483, 2026.

Kartik Kuckreja, Muhammad Sohail Danish, Muzammal Naseer, Abhijit Das, Salman Khan, and Fahad Shahbaz Khan. Geochat: Grounded large vision-language model for remote sensing. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 27831–27840. IEEE, 2024.

K Li, G Wan, G Cheng, L Meng, and J Han. Dior: a large-scale dataset for object detection in optical remote sensing images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2020.

Xiang Li, Jian Ding, and Mohamed Elhoseiny. Vrsbench: A versatile vision-language benchmark dataset for remote sensing image understanding. *Advances in Neural Information Processing Systems*, 37:3229–3242, 2024.

Jiaqi Liu, Lang Sun, Ronghao Fu, and Bo Yang. Towards faithful reasoning in remote sensing: A perceptually-grounded geospatial chain-of-thought for vision-language models. In *International Conference on Learning Representations*, volume 2026, pp. 6535–6563, 2026.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European conference on computer vision*, pp. 38–55. Springer, 2024.

Chuofan Ma, Yi Jiang, Jiannan Wu, Zehuan Yuan, and Xiaojuan Qi. Groma: Localized visual tokenization for grounding multimodal large language models. In *European Conference on Computer Vision*, pp. 417–435. Springer, 2024.

Jiancheng Pan, Yanxing Liu, Yuqian Fu, Muyuan Ma, Jiahao Li, Danda Pani Paudel, Luc Van Gool, and Xiaomeng Huang. Locate anything on earth: Advancing open-vocabulary object detection for remote sensing community. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 6281–6289, 2025.

Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, and Furu Wei. Kosmos-2: Grounding multimodal large language models to the world. *arXiv preprint arXiv:2306.14824*, 2023.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Hanoona Rasheed, Haania Siddiqui, Ming-Hsuan Yang, Fahad Shahbaz Khan, and Salman Khan. Locate anything in videos: Rethinking efficient generative spatio-temporal video grounding. *arXiv preprint arXiv:2608.28192*, 2026.

Akashah Shabbir, Mohammed Zumri, Mohammed Bennamoun, Fahad S Khan, and Salman Khan. Geopixel: Pixel grounding large multimodal model in remote sensing. *arXiv preprint arXiv:2501.13925*, 2025.

Sagar Soni, Akshay Dudhane, Hiyam Debary, Mustansar Fiaz, Muhammad Akhtar Munir, Muhammad Sohail Danish, Paolo Fraccaro, Campbell D Watson, Levente J Klein, Fahad Shahbaz Khan, et al. Earthdial: Turning multi-sensory earth observations to interactive dialogues. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14303–14313. IEEE, 2025.

Yongyi Su, Haojie Zhang, Shijie Li, Nanqing Liu, Jingyi Liao, Junyi Pan, Yuan Liu, Xiaofen Xing, Chong Sun, Chen Li, et al. Patch-as-decodable-token: Towards unified multi-modal vision tasks in mllms. In *International Conference on Learning Representations*, volume 2026, pp. 147893–147919, 2026.

Kimi Team, Angang Du, Bohong Yin, Bowei Xing, Bowen Qu, Bowen Wang, Cheng Chen, Chenlin Zhang, Chenzhuang Du, Chu Wei, et al. Kimi-vl technical report. *arXiv preprint arXiv:2504.07491*, 2025.

Shihao Wang, Shilong Liu, Yuanguo Kuang, Xinyu Wei, Yangzhou Liu, Zhiqi Li, Yunze Man, Guo Chen, Andrew Tao, Guilin Liu, et al. Locateanything: Fast and high-quality vision-language grounding with parallel box decoding. In *European Conference on Computer Vision*, pp. 336–357. Springer, 2026.

Gui-Song Xia, Xiang Bai, Jian Ding, Zhen Zhu, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, and Liangpei Zhang. Dota: A large-scale dataset for object detection in aerial images. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3974–3983. IEEE, 2018.

Bin Xiao, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yumao Lu, Michael Zeng, Ce Liu, and Lu Yuan. Florence-2: Advancing a unified representation for a variety of vision tasks. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4818–4829. IEEE, 2024.

Haoxuan You, Haotian Zhang, Zhe Gan, Xianzhi Du, Bowen Zhang, Zirui Wang, Liangliang Cao, Shih-Fu Chang, and Yinfei Yang. Ferret: Refer and ground anything anywhere at any granularity. In *International Conference on Learning Representations*, volume 2024, pp. 57153–57180, 2024.

Yiqin Yuan, Lang Li, Xiwen Yao, Lingjun Li, Xiaoxu Feng, Gong Cheng, and Junwei Han. A comprehensive review of optical remote-sensing image object detection datasets. *Journal of Remote Sensing*, 27(12):2671–2687, 2023.

Yang Zhan, Zhitong Xiong, and Yuan Yuan. Rsvg: Exploring data and models for visual grounding on remote sensing data. *IEEE transactions on geoscience and remote sensing*, 61:1–13, 2023.

Yang Zhan, Zhitong Xiong, and Yuan Yuan. Skyeyegpt: Unifying remote sensing vision-language tasks via instruction tuning with large language model. *ISPRS Journal of Photogrammetry and Remote Sensing*, 221:64–77, 2025.

Peirong Zhang, Yidan Zhang, Luxiao Xu, Jinliang Lin, Zonghao Guo, Fengxiang Wang, Xue Yang, Kaiwen Wei, and Lei Wang. Geovis: Geospatially rewarded visual search for remote sensing visual grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14335–14345, 2026.

Wei Zhang, Miaoxin Cai, Tong Zhang, Yin Zhuang, and Xuerui Mao. Earthgpt: A universal multimodal large language model for multisensor image comprehension in remote sensing domain. *IEEE Transactions on Geoscience and Remote Sensing*, 62:1–20, 2024.

Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019.

Yue Zhou, Mengcheng Lan, Xiang Li, Litong Feng, Yiping Ke, Xue Jiang, Qingyun Li, Xue Yang, and Wayne Zhang. Geoground: A unified large vision-language model for remote sensing visual grounding. *arXiv preprint arXiv:2411.11904*, 2024.

Chaoyang Zhu, Yiyi Zhou, Yunhang Shen, Gen Luo, Xingjia Pan, Mingbao Lin, Chao Chen, Liujuan Cao, Xiaoshuai Sun, and Rongrong Ji. Seqtr: A simple yet universal network for visual grounding. In *European Conference on Computer Vision*, pp. 598–615. Springer, 2022.

A Data and Training

A.1 Source Episodes and Spatial Supervision

Each training episode consists of an aerial image, a request, and the corresponding annotated target set. The supervision depends on the type of request. For category-conditioned requests, all annotated instances belonging to the requested categories are used as targets. For expression-conditioned requests, the original referring expression and its associated spatial region are preserved. This distinction allows the model to learn two different spatial tasks: identifying all instances of a requested category and localizing a specific referent when multiple similar objects appear in the aerial scene.

Detection data are drawn from DIOR, DOTA variants, FAIR1M, xView, HRSC2016, NWPU-VHR-10, RSOD, SODA-A, KFGOD, and Power-Plant, while grounding supervision mainly comes from DIOR-RSVG and VRSBench. During source conversion, we correct the xView category identifiers and regenerate the corresponding positive and negative requests. Ambiguous records are excluded rather than assigned artificial replacement expressions. The corrected source records are then used consistently for PID and the two finetuned direct-box baselines. Finally, checks based on both image files and decoded image content are used to remove exact overlaps with benchmark images. These checks, however, do not guarantee geographic independence between nearby aerial tiles.

Table 6 summarizes the training episodes used in the two curriculum stages. Stage 1 contains 116,386 episodes, including 62,618 detection and 53,768 grounding episodes, drawn from 88,767 distinct images. The scenes in this stage are relatively sparse, with 238,475 positive target occurrences in total, corresponding to about two targets per episode and a maximum of 19 targets in a single episode.

Stage 2 contains fewer episodes but substantially denser scenes. It consists of 39,851 episodes (23,435 detection and 16,416 grounding) from 36,460 images, with 1,619,629 positive target occurrences. This corresponds to about 41 targets per episode, with as many as 1,942 targets in a single episode. Thus, despite having only about one third as many episodes as Stage 1, Stage 2 provides approximately $6.8\times$ more positive target supervision.

Both stages also include negative box requests, for which the null targets teach the model to reject unsupported addresses. Stage 1 contains 26,493 such requests, while Stage 2 contains 179,966, accounting for approximately 10% of box requests in each stage. To help retain performance on sparse scenes, Stage 2 additionally replays 10,943 episodes from Stage 1, representing 27.46% of its records.

The validation split contains 740 episodes, including 397 detection and 343 grounding episodes, with 84,377 positive targets and up to 682 targets in a single episode. By including both sparse and dense scenes, this split provides a common basis for checkpoint selection across both curriculum stages.

No additional pointing-only episodes are used. Each positive HBB provides a quantized center address. If multiple boxes within the same reference map to the same address during training, deterministic interior offsets are used to resolve these collisions. If a collision cannot be resolved, the entire episode is excluded to maintain complete supervision. Null box targets are generated from annotation-verified background regions, wrong-category requests, and absent-reference requests. The validity of these null targets is determined based on the available source annotations and does not require every visible object in the image to be annotated.

A.2 Optimization and Checkpoint Selection

PID and the direct-box baselines are fine-tuned using the same training procedure. We update all four model components: the MoonViT vision encoder, multimodal projector, Qwen language model, and output head. Training is performed on eight GPUs, with one episode per GPU and four gradient-accumulation steps, resulting in an effective batch size of 32 episodes. Each curriculum stage is trained for two epochs. Stage 2 is initialized from the best Stage 1 checkpoint for the corresponding formulation, selected based on validation loss. The optimizer and learning-rate schedule are reset at the start of Stage 2. To select checkpoints, we compute the mean validation loss for detection and grounding episodes separately and then average the two task-level losses. The Stage 2 checkpoint

Table 6: Composition of training episodes before epoch repetition. Image counts are computed independently per split; target occurrences may include repeated supervision of the same physical object across episodes.

Statistic	Stage 1	Stage 2	Validation
Episodes	116,386	39,851	740
Detection episodes	62,618	23,435	397
Grounding episodes	53,768	16,416	343
Distinct images within split	88,767	36,460	740
Positive target occurrences	238,475	1,619,629	84,377
Negative box requests	26,493	179,966	9,375
Maximum targets per episode	19	1,942	682
Replay episodes	0	10,943	0

selected using this criterion is used for all evaluation tasks, without any additional task-specific checkpoint selection.

Optimizer and numerical configuration. We store trainable parameters in FP32 and use BF16 autocast for model computation. To reduce memory usage, optimizer states are sharded across GPUs using ZeroRedundancyOptimizer, and activation checkpointing is enabled. We optimize the model using fused AdamW with a weight decay of 0.01 and clip the gradient norm to 1.0. The learning rate is linearly warmed up over the first 3% of training updates, corresponding to 218 of 7,276 updates in Stage 1 and 75 of 2,492 updates in Stage 2. After warmup, it follows a cosine schedule and decays to 10% of the peak learning rate for each parameter group. The learning rates used for individual model components are reported in Table 8. All experiments use a fixed random seed of 20260908.

Dataset and manifest identifiers. PID and LA-3B (FT)† are trained using the same corrected data release, with identical hashes for the Stage 1, Stage 2, and validation files recorded in their saved configurations. Table 7 reports the first 12 hexadecimal characters of the SHA-256 hashes for these training files, as well as for the frozen evaluation manifests used in our experiments. These hashes identify the exact file versions used in each run. Note that changing only the image-path prefix changes the file hash, while leaving the image membership and annotations unchanged.

Table 8 summarizes the training configuration for both curriculum stages. The number of optimizer updates is determined by the dataset size, batch size, and number of training epochs. In Stage 1, the 116,386 training samples with a batch size of 32 result in $\lceil 116{,}386/32 \rceil = 3{,}638$ updates per epoch, giving a total of 7,276 updates over two epochs. Similarly, Stage 2 contains 39,851 samples, resulting in $\lceil 39{,}851/32 \rceil = 1{,}246$ updates per epoch and 2,492 updates over two epochs.

Normalization of the joint objective. Let m_{rb} be the number of supervised token positions in answer b of component $r \in \{\mathrm{addr}, \mathrm{box}\}$ (address prediction and box prediction), and let $\bar{\ell}_{rb}$ be the mean token-level cross-entropy of that answer. PID minimizes

$$\mathcal{L}_e = \frac{1}{2} \sum_{r \in \{\mathrm{addr},\mathrm{box}\}} \sum_{b} \frac{m_{rb}}{\sum_{b'} m_{rb'}} \bar{\ell}_{rb}, \tag{5}$$

which is the token-weighted form of Eq. 4: each component's loss is the mean over all of its supervised tokens in the episode, and the two components receive equal weight. The denominator always sums over the entire episode, including when a dense episode is processed in several memory-bounded groups. As a result, splitting an episode into groups does not change its contribution to the loss. The normalization is applied per token within each component, not per object or per reference. PID uses no additional direct-box loss, while the direct-box baselines are trained with their standard answer-level cross-entropy on the same data.

B Localization Interfaces

B.1 Reference and Address Conditioning

PID performs address prediction separately for each textual reference. For category requests, the input is divided into one reference for each category, whereas for expression requests, the complete

Table 7: Training-file and frozen evaluation-manifest identifiers. Training counts are episodes; detection evaluation counts are images; grounding evaluation counts are image-expression queries. Hashes are abbreviated prefixes that identify the files, not full integrity checksums.

Record file / evaluation set	Records	SHA-256 prefix
Training: `stage1.jsonl`	116,386	`2fd6c5e8df29`
Training: `stage2.jsonl`	39,851	`c5d0e3156f2c`
Training: `validation.jsonl`	740	`00e14c17087b`
DIOR	586	`44b493d6e497`
DOTAv2	874	`ada8bbdc3ff1`
DIOR-RSVG	7,500	`11443dbd68a1`
VRSBench-VG	16,159	`c96aeb3d8aad`

Table 8: Full-parameter training configuration. Both stages use seed 20260908, and each formulation selects its own checkpoint by validation loss.

Setting	Stage 1	Stage 2
Epochs	2	2
Optimizer updates	7,276	2,492
Warmup updates (3%)	218	75
Effective global batch	32	32
Qwen learning rate	2×10^{-6}	10^{-6}
Output-head learning rate	10^{-6}	5×10^{-7}
MoonViT learning rate	10^{-6}	10^{-6}
Projector learning rate	10^{-5}	10^{-5}
Pre-merge image-patch limit	6,000	6,000
Token-packing budget	16,384	16,384
Maximum sequence length	32,768	32,768
Selected PID checkpoint (update)	6,900	2,400
Selected LA-3B (FT)† checkpoint (update)	7,200	1,200

referring expression is treated as a single reference. Given each reference, the model predicts a variable-length sequence of point addresses over the full image, terminated by an EOS token. If no matching instance exists for a reference, the model outputs EOS directly. Each predicted point is then paired with its corresponding reference to form an address-conditioned box query. From this query, the model predicts either an HBB or a null response. The semantic identity of the predicted box is inherited directly from the input reference, so no separate category label is generated. Both address and box prediction operate on the full image, without using magnified crops or local re-encoding around the predicted points.

Pointing uses the same address-prediction step with the pointing prompt in Table 9, and the predicted points are returned directly without box prediction. For expression requests, the pointing prompt is identical to the expression address prompt. For category requests, the pointing prompt names a single category, whereas the category address prompt used in detection follows the detection query format. The pointing results in Table 3 therefore come from separate runs with the pointing prompt, not from the addresses produced during detection. Table 9 summarizes all prompts and output formats.

B.2 Shared Context and Masking Across Addresses

Both address prediction and box prediction reuse the same image-only prefix computed by the Qwen backbone. This prefix ends immediately after the image-closing token and contains neither reference text nor previously generated outputs. Thus, both stages have access to the full image context.

During box prediction, each box segment attends to the shared image prefix, its own reference-address prompt, and its own geometry block. Attention to the prompts and geometry blocks of other addresses is masked. Therefore, each box is predicted using the global image context and its corresponding address, without depending on boxes predicted for other addresses.

All box segments use the same position indices relative to the shared prefix. Within each segment, the prompt tokens follow causal attention, while the six geometry positions use the native PBD attention pattern (Wang et al., 2026). Following the base model, the final prompt token is repeated as the PBD

Table 9: PID query and response formats. Prompts are reproduced verbatim; {category}, {expression}, and {reference} are replaced by the request text. Each category address prompt names a single category, since PID predicts addresses separately for each reference (Appendix B.1). Coordinates are integers from the native vocabulary (0-1,000). Points reuse the box delimiters but contain only two coordinates (u, v), whereas boxes contain four corner coordinates.

Interface	Prompt or output format
Queries	
Category addresses	Point to all object instances that match the following categories: {category}.
Expression addresses	Point to: {expression}.
Pointing	Point to: {reference}.
Address-conditioned box	Locate the single {reference} containing point `<box><u><v></box>` in horizontal-box format. Return None if absent.
Responses	
Points	`<box><u><v></box>`, one per proposed instance, followed by EOS
Box	`<box><x1><y1><x2><y2></box>`
Null	`<box>None</box>`

anchor at the same position index. Its original occurrence in the prompt is masked from the geometry queries.

For N predicted addresses, predicting the boxes one at a time would require N forward passes, whereas packed prediction requires a single forward pass when $N > 0$ and none when $N = 0$.

B.3 Rotary-Position Consistency

Box segments use prefix-relative positions, while the length of the packed sequence increases with the number of addresses. If the packed sequence exceeds the precomputed rotary-embedding length, the Qwen implementation extends its rotary cache. In BF16, this extension can introduce small numerical differences in the cached values. To ensure consistent rotary embeddings, we compute the inverse frequencies, position indices, and sine and cosine values in FP32 outside autocast, and cast the completed cache to the inference precision only afterward. The frequencies are recomputed directly from the configured rotary base and dimension rather than from an already rounded buffer. This procedure is used for the final detection experiments (Table 1) and the annotation-address intervention (Table 5B). It changes only the numerical precision of the rotary-cache computation; the model weights and attention masks remain unchanged.

We verify the stability of this procedure with an acceptance test. A single DOTAv2 image produces 1,074 addresses, which are processed together in one packed box pass. This increases the cache length from 32,768 to 37,904 positions in each of the 36 rotary modules, while the first 2,048 cached positions remain byte-identical after the extension. We then process a smaller image before and after this cache growth and obtain identical points, box outputs, and parsed predictions for both predicted and annotation-derived addresses.

We also compare packed and sequential execution using 16 addresses. In this test, 15 of the 16 raw box outputs are identical. Thus, packed execution remains stable when the rotary cache is extended, although it is not expected to be bitwise identical to sequential execution because batched attention can alter the order of floating-point accumulation. Finally, the efficiency measurements reported in Section 4.3 and Appendix D.2 were collected before this numerical-precision change and are therefore reported separately from the final detection results.

C Evaluation Protocols

C.1 Benchmark Scope and Output Provenance

Detection and category-conditioned pointing use 586 DIOR images (Li et al., 2020) and 874 DOTAv2 images (Xia et al., 2018), containing 3,379 and 29,329 annotated HBBs, respectively. Each request names the categories annotated in the image but provides no instance counts, points, or boxes. All annotations are kept in the scorer, including 875 DOTAv2 boxes that extend beyond the image boundary. Predictions are clipped to the image, but ground-truth boxes are never modified. Expression-

conditioned evaluation uses all 7,500 DIOR-RSVG and 16,159 VRSBench-VG requests without modification.

The category scope follows the annotated-category detection setting of Rex-Omni (Jiang et al., 2026), and matching and aggregation follow the dense-localization evaluation released with LocateAnything (Wang et al., 2026). This setting measures how well a model localizes the requested categories, not whether it can recognize and reject absent categories from the full benchmark vocabulary. We make one change to the released LocateAnything matcher: it ignores category labels, whereas we require the predicted and ground-truth categories to agree. This change is applied identically to every model.

The PID detection results in Table 1 come from fresh inference with predicted addresses, using the rotary-stabilized implementation (Appendix B.3). Results computed from saved per-category outputs rather than new inference, such as the per-category requests in Table 14, keep only the outputs for the categories annotated in each image. This selection depends only on which category was requested, never on prediction correctness, overlap, or target count. Duplicate, invalid, and unmatched outputs in the retained streams are kept and scored as errors. Because these results are rescored rather than re-run, we do not report timings for them.

C.2 Matching and Aggregation

For each image and IoU threshold τ, matching is greedy. Ground-truth boxes are processed in annotation order, and each one is matched to the unused prediction with the highest IoU among those that are eligible, that is, predictions with the same category and IoU $\geq \tau$. Each prediction and each ground-truth box can be matched at most once. Unmatched predictions count as false positives and unmatched ground-truth boxes as false negatives. With M images, the detection metrics are

$$\begin{aligned} \bar{P}_\tau &= \frac{1}{M}\sum_i \frac{\mathrm{TP}_{i,\tau}}{\mathrm{TP}_{i,\tau}+\mathrm{FP}_{i,\tau}}, \qquad \bar{R}_\tau = \frac{1}{M}\sum_i \frac{\mathrm{TP}_{i,\tau}}{\mathrm{TP}_{i,\tau}+\mathrm{FN}_{i,\tau}}, \\ \mathrm{F1}_\tau &= \frac{2\bar{P}_\tau\bar{R}_\tau}{\bar{P}_\tau+\bar{R}_\tau}, \qquad\qquad \mathrm{mF1} = \frac{1}{10}\sum_{k=0}^{9}\mathrm{F1}_{0.50+0.05k}. \end{aligned} \tag{6}$$

A ratio with a zero denominator is set to zero, so an image with no predictions contributes zero precision. F1 is computed from image-averaged precision and recall. It is therefore neither micro-F1 over pooled counts nor the mean of per-image F1 scores.

Pointing uses the same aggregation. Each ground-truth HBB is matched to the first unused eligible point that lies inside or on the box boundary. For category requests, an eligible point must also carry the matching category. A point does not need to lie near the object center to count as correct, and the criterion differs from mask-based pointing evaluation.

For grounding, let u_j be the IoU between the first parsed box for query j and the annotated referent, with $u_j = 0$ if the box is missing or has zero area. Over Q queries, we report

$$\mathrm{Acc@}\tau = \frac{1}{Q}\sum_{j=1}^{Q}\mathbf{1}[u_j \geq \tau], \qquad \mathrm{mIoU} = \frac{1}{Q}\sum_{j=1}^{Q} u_j. \tag{7}$$

The target is defined by the complete expression, so no category list is given and any generated labels are ignored. Only the first box is scored; later boxes cannot replace an incorrect first prediction. The frozen VRSBench-VG manifest contains 13 zero-area ground-truth boxes. We keep them for all models, and they always score zero IoU.

C.3 Prompts, Parsing, and Operating Points

LA-3B, LA-3B (FT)[†], and Rex-Omni receive all annotated categories of an image in a single joint request. LA-3B (FT, class-iso.)[§] and PID both handle each category as an isolated reference. The class-isolated variant generates boxes for each reference as a separate request, whereas PID decodes the addresses of all references concurrently and then predicts all boxes in a single packed forward pass (Section 3). Qwen3-VL receives one request per category, and the outputs are merged per image.

The specialized detectors receive all categories as a single period-separated text prompt. Pointing uses a separate point request for each category or expression. No prompt reveals the number of targets.

Outputs from LA-family models, including PID, are parsed with the released LocateAnything parser, which swaps reversed corners and clips predictions to the image. Rex-Omni and Qwen3-VL use their own native coordinate parsers. A prediction with an explicit wrong category label is scored as an error. In address-conditioned box responses, where the category is already specified by the request, a missing label takes the requested category. Malformed candidates are scored as errors, and text that fails to parse is never turned into a prediction. We apply no additional non-maximum suppression, deduplication, or limit on the number of detections (such as maxDets = 100).

LA-family models and Qwen3-VL decode with sampling at temperature 0.7, top-p 0.9, and repetition penalty 1.1; Qwen3-VL additionally uses top-$k = 0$. Rex-Omni uses the deterministic settings of its released wrapper. Each generation call has a budget of 8,192 tokens, which applies per call rather than per image. We record separately how often outputs reach this limit. Image preprocessing follows each model's native pipeline: LA-family models use at most 6,000 pre-merge patches, and Qwen3-VL uses at most 1,176,000 pixels. Grounding DINO-T and LAE-DINO use a confidence threshold of 0.25, Grounding DINO also uses a text threshold of 0.25, and LAE-DINO keeps its native top-300 output selection.

Detector sensitivity. Table 10 varies the LAE-DINO operating point using the same saved candidate pool. Removing the top-300 selection has almost no effect, whereas raising the confidence threshold to 0.30 lowers F1@0.50 by about 3 points on both datasets. The main comparison uses the fixed setting in the first row.

Parser sensitivity. Table 11 re-parses the same raw outputs of the LA-family models with a strict parser. The strict parser accepts only well-formed outputs and applies no corner repair and no label fallback. PID's F1@0.50 is unchanged after rounding on both datasets, and the other models change by at most 0.34 points. The comparison is therefore not sensitive to the choice of parser.

C.4 Complementary Set-Based Grounding

The released referring-object scorers of LocateAnything and Rex-Omni match the complete set of predicted boxes against the ground-truth set, without category agreement. Table 12 applies these rules to our saved grounding outputs, with prompts, weights, parsing, and predictions unchanged. Precision and recall are averaged over expressions before F1 is computed, and mF1 averages the ten IoU thresholds. Both released implementations and an independent count-based check give identical results on these outputs. This confirms that our matching is compatible with theirs; it does not imply that our end-to-end inference reproduces those papers.

Set-based scores can differ from first-box accuracy when a model returns more than one box. Rex-Omni returns multiple boxes for 420 DIOR-RSVG and 1,622 VRSBench-VG expressions, whereas PID does so for only four and two. Extra boxes can recover a target missed by the first prediction, but they also lower set precision. Under set-based matching, LA-3B (FT)† scores slightly above PID in mF1 on both datasets (62.15 vs. 61.93 and 50.22 vs. 49.09), even though PID is higher in first-box Acc@0.5 (Table 2). The two models are therefore comparable on single-target grounding, which is consistent with the small margins in the main results. When every expression has exactly one ground-truth box and one returned box, set F1 at a threshold equals first-box accuracy at that threshold; otherwise the two can differ. mIoU and threshold-averaged mF1 are different quantities and should not be compared directly.

C.5 Pointing Precision and Recall

Table 13 reports the precision and recall behind the category-conditioned pointing results. These points are produced by the address-prediction step with the pointing prompt (Appendix B.1); they are not box centers. PID improves both precision and recall over LA-3B (FT)† on both datasets. The largest gain is in DOTAv2 recall ($+11.13$ points, from 54.40 to 65.53), which is consistent with better coverage of dense scenes.

Table 10: Sensitivity of LAE-DINO to its output operating point under annotated-category detection. "All native pairs" uses the full candidate pool produced by the model before top-300 selection.

	DIOR			DOTAv2		
LAE-DINO output policy	F1@0.50	F1@0.75	mF1	F1@0.50	F1@0.75	mF1
Top-300, > 0.25 (main)	82.20	58.98	54.75	72.79	50.64	46.94
All native pairs, > 0.25	82.20	58.98	54.75	72.90	50.70	47.00
Top-300, > 0.30	79.46	57.48	53.22	69.67	49.77	45.69

Table 11: Sensitivity to the parsing policy with fixed raw model outputs. Equal values after rounding do not imply identical parsed predictions. † indicates our finetuned variant.

	DIOR F1@0.50		DOTAv2 F1@0.50	
Model	LA-native	Strict	LA-native	Strict
LA-3B	54.16	54.16	36.93	36.59
LA-3B (FT)†	73.94	73.81	59.62	59.51
PID (ours)	82.24	82.24	66.90	66.90

Table 12: Set-based grounding evaluation on unchanged saved outputs. Values are set F1 scores over the complete predicted box set, not conversions of first-box Acc or mIoU, and are therefore not directly comparable to published first-box results. † indicates our finetuned variant.

	DIOR-RSVG				VRSBench-VG			
Model	F1@0.50	F1@0.75	F1@0.95	mF1	F1@0.50	F1@0.75	F1@0.95	mF1
Qwen3-VL-4B	47.08	30.37	4.60	30.04	51.98	26.17	1.99	28.27
Qwen3-VL-8B	51.24	33.39	4.92	32.64	55.70	28.23	1.76	30.33
Rex-Omni-3B	54.48	38.79	6.84	36.98	65.04	40.75	5.13	40.43
LA-3B	41.87	26.91	5.87	27.06	50.94	27.93	2.77	29.18
LA-3B (FT)†	80.77	65.99	22.29	62.15	76.63	50.76	10.59	50.22
PID (ours)	78.99	66.06	23.12	61.93	76.25	49.78	7.57	49.09

C.6 Address and Reference Diagnostics

The annotation-address intervention in Table 5B keeps the PID weights, requested categories, image preprocessing, box prompts, parsing, and scoring fixed. Only the addresses change. Each annotated object supplies one quantized address, ordered first by category and then by annotation order. For objects that cross the image boundary, the address is the center of the visible part of the box, so that it lies inside the image, while scoring still uses the original HBB. All 3,379 DIOR and 29,329 DOTAv2 annotations contribute an address, including three quantized addresses that coincide in DOTAv2. We verify that every address lies inside its source box. The box decoder never receives the annotated box boundaries.

With predicted addresses, PID proposes 3,497 addresses on DIOR (103% of the annotated count) and 27,274 on DOTAv2 (93%). These totals do not measure completeness by themselves, because missed, repeated, and incorrect addresses can occur together. Even with annotation-derived addresses, the box decoder returns a null response for 183 DIOR and 691 DOTAv2 addresses. The intervention therefore measures localization when the correct instance count and approximate positions are given. It is not an additive split of the total error between address prediction and box prediction, and it is not necessarily an upper bound for a learned address model. Its timing excludes address prediction and is not used as a speed result.

Inference-only category separation. Table 14 changes only the request interface of the trained LA-3B (FT)† checkpoint, without retraining. Each category is requested separately, and the outputs are merged per image; predictions with a label other than the requested category are kept and scored as errors. Separating the categories at inference improves DIOR F1@0.50 by $+1.80$ but lowers DOTAv2 F1@0.50 by -0.58. Separating categories without instance addressing therefore yields at most small and inconsistent gains, which agrees with the trained class-isolated variant LA-3B (FT, class-iso.)§ in Table 5A.

Table 13: Category-conditioned pointing under annotated-category requests. Precision (P) and recall (R) are image-averaged percentages; F1 is computed before rounding. † indicates our finetuned variant.

	DIOR			DOTAv2		
Model	P	R	F1@Point	P	R	F1@Point
Qwen3-VL-4B	80.73	67.18	73.34	67.30	37.17	47.89
Qwen3-VL-8B	82.25	73.65	77.71	66.00	40.05	49.85
Rex-Omni-3B	77.26	69.41	73.12	67.98	72.40	70.12
LA-3B	67.48	71.04	69.22	58.08	50.17	53.84
LA-3B (FT)†	86.44	77.62	81.79	68.38	54.40	60.60
PID (ours)	90.23	80.90	85.31	71.27	65.53	68.28

Table 14: Inference-only category separation with fixed LA-3B (FT)† weights. Per-category results are rescored from saved per-category requests. Token budgets are per request, and sampling seeds are not paired with the joint run, so this is not a matched-compute comparison.

	DIOR			DOTAv2		
Request interface	F1@0.50	F1@0.75	mF1	F1@0.50	F1@0.75	mF1
Joint request	73.94	63.75	59.94	59.62	36.33	35.28
Per-category requests	75.74	63.72	60.39	59.04	35.79	34.72

D Additional Analyses

D.1 Object Scale and Scene Density

We stratify the detection outputs of Table 1 by object scale and scene density, without changing the requests or re-running inference. Object scale is the area of the annotated HBB divided by the image area: small targets cover less than 0.1% of the image, medium targets 0.1% to 1%, and large targets 1% or more. These relative-area bins differ from the absolute pixel-area bins used in COCO. Matching is performed on the whole image first, and the matched ground-truth targets are then assigned to their area bins. Scale-wise recall is the fraction of annotated targets in each bin that are recovered; predictions are not filtered by their own area. The three bins together contain all 3,379 DIOR and 29,329 DOTAv2 targets.

Figure 4 (left) shows that PID achieves higher recall than LA-3B (FT)† in all three scale groups on both datasets. The largest gains are for small targets: recall roughly doubles on DIOR, from 28.64% to 59.09%, and rises from 27.71% to 44.81% on DOTAv2. Because the two models share the same data, training curriculum, and trainable components, this improvement can be attributed to the instance-addressed formulation as a whole. The stratification alone, however, cannot separate the contributions of address prediction and box prediction. The advantage also depends on scale: for large DOTAv2 targets, Rex-Omni achieves higher recall than PID.

For scene density, images are grouped by the number of annotated targets: 0-5, 6-20, and more than 20. Every evaluated image contains at least one annotated target. Within each group, F1@0.50 is computed from image-averaged precision and recall, as in Eq. 6. In the densest group, PID reaches 74.95% F1 on DIOR and 67.51% on DOTAv2, compared with 59.22% and 56.19% for LA-3B (FT)† and 64.53% and 56.27% for Rex-Omni. This group contains only 25 DIOR images but 258 DOTAv2 images, so the DOTAv2 result is the more reliable estimate. The advantage is not uniform across density levels. On DOTAv2, Rex-Omni achieves higher F1 than PID in the two lower-density groups, and PID performs on par with LA-3B (FT)† in the 1-5 group. The gains of PID are thus concentrated in dense scenes, where the number of targets is largest. This pattern is consistent with the removal of dependencies between boxes, which matters most when many boxes are predicted, but the stratification alone does not isolate this effect from other differences between the formulations.

D.2 Complete-Response Efficiency

We measure the time each model needs to produce a complete response for an image, covering all requested categories. Timing is CUDA-synchronized. It includes preprocessing, visual encoding,

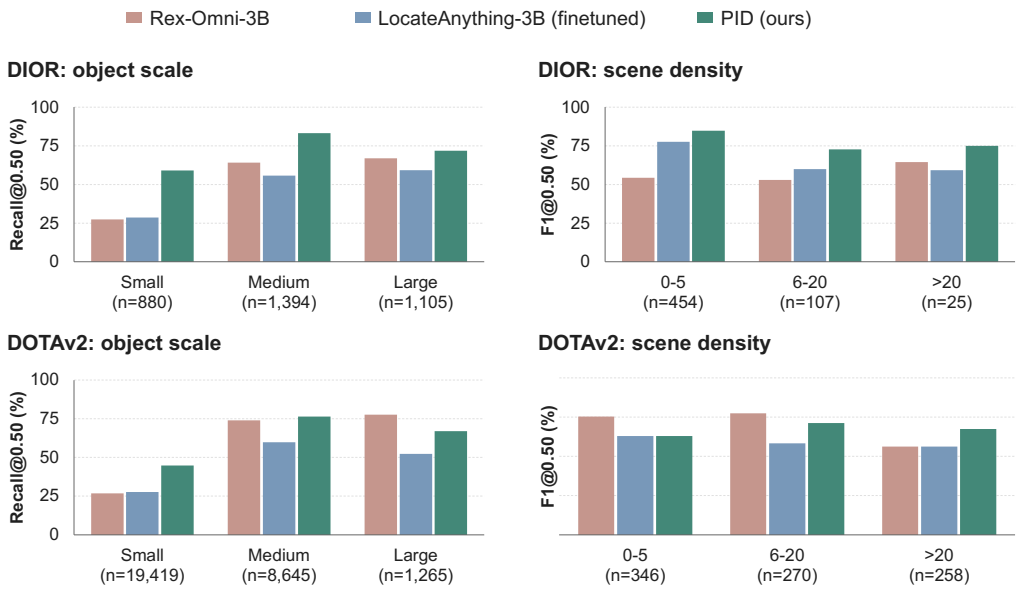

Figure 4: Detection performance of Rex-Omni-3B, LA-3B (FT)†, and PID stratified by object scale and scene density. Left: recall at IoU 0.50 after whole-image matching, binned by the area of the ground-truth (GT) horizontal bounding box (HBB) relative to the image. Right: F1@0.50 from image-averaged precision and recall within each density bin. Numbers in parentheses give the number of GT targets (scale) or images (density).

generation, and output parsing, and excludes disk image loading, model initialization, warm-up, queue waits, and scoring. For models that issue one request per category (Qwen3-VL), the time of all requests for an image is summed. For image i with response time t_i, B_i valid returned boxes, and C_i boxes matched to a ground-truth box of the same category at IoU 0.50, we report over M images

$$\mathrm{TPI} = \frac{\sum_i t_i}{M}, \qquad \mathrm{BPS} = \frac{\sum_i B_i}{\sum_i t_i}, \qquad \mathrm{CBPS} = \frac{\sum_i C_i}{\sum_i t_i}. \tag{8}$$

BPS and CBPS divide total counts by total time; they are not averages of per-image rates. CBPS uses the actual number of matched boxes rather than image-averaged precision multiplied by BPS. Malformed boxes count as false positives in the accuracy metrics but are excluded from the valid-box count B_i. All measurements use NVIDIA A100 PCIe 40 GB GPUs on a shared server. Rates are reported per worker and are not multiplied by the number of concurrent workers.

Table 15 reports latency and throughput for all methods. Specialized detectors are the fastest, since they predict all boxes in a single forward pass without autoregressive generation. Among the generative VLMs, PID achieves the highest BPS and CBPS on both datasets and the lowest latency on DIOR. On DOTAv2, the pretrained LA-3B is about 10% faster than PID (1.992 s vs. 2.218 s), but PID produces $2.8\times$ more correct boxes per second (8.33 vs. 3.00 CBPS), which reflects its much higher accuracy (Table 1). Relative to LA-3B (FT)†, PID is faster and more productive on both datasets, as discussed in Section 4.3.

D.3 Packed Box Decoding Details

In packed box decoding, PID first predicts a (possibly empty) set of addresses for every reference in the image, decoding the references concurrently. It then gathers the N addresses from all references and predicts their boxes together in a single forward pass. Each box segment attends only to the shared image prefix and to its own address (Appendix B.2), so packing changes the number of sequential passes but not what each box can attend to. Compared with predicting the boxes one at a time, this reduces the box stage from N sequential passes to one, which is the $\mathbb{1}[N > 0]$ term in Eq. 3. The total computation and the memory needed for the packed sequence, however, still grow with N. All latency and throughput results for PID in Table 4 and Table 15 use packed box decoding and follow the timing protocol of Appendix D.2.

Table 15: Complete-response detection efficiency under annotated-category requests. TPI is seconds per image; BPS and CBPS are valid and correctly matched boxes per second (correct at IoU 0.50). All models use their native preprocessing and query interfaces, and measurements may come from different runs. † indicates our finetuned variant. **Bold**: best among generative VLMs.

	DIOR			DOTAv2		
Method	TPI↓	BPS↑	CBPS↑	TPI↓	BPS↑	CBPS↑
Specialized detectors						
Grounding DINO-Swin-T	0.189	15.77	4.12	0.200	20.55	6.02
LAE-DINO	0.164	23.02	21.06	0.183	154.19	130.74
Autoregressive VLMs						
Qwen3-VL-4B	4.777	0.73	0.50	10.010	0.68	0.37
Qwen3-VL-8B	6.295	0.65	0.38	10.088	0.65	0.35
Rex-Omni-3B	1.614	4.18	1.98	19.704	6.32	0.73
LA-3B	0.858	6.43	3.32	**1.992**	7.78	3.00
LA-3B (FT)†	0.990	3.99	2.90	3.930	10.01	3.27
PID (ours)	**0.768**	**6.99**	**5.50**	2.218	**13.30**	**8.33**

E Qualitative Comparisons

This section gives additional qualitative comparisons of Rex-Omni-3B, LA-3B (FT)†, and PID. Within each row, all models receive the same image and request. The examples were selected to show typical differences in target coverage, referent selection, and localization precision; they are not representative of average performance, which is reported in Section 4.2.

E.1 Multi-Category Detection

Figure 5 shows three detection scenes: a DIOR sports complex with tennis courts and parked vehicles, a DIOR harbor with ships and vehicles, and a DOTAv2 scene with vehicles and tennis courts. Each scene combines many small, repeated instances with a few large, spatially extended objects such as harbors and courts. Unlike the main detection evaluation, which requests only the categories annotated in each image, these examples request the full benchmark vocabulary (20 categories for DIOR and 18 for DOTAv2). Models must therefore also avoid predicting absent categories. In the DOTAv2 scene, for example, LA-3B (FT)† labels part of the small-vehicle cluster as large vehicles, whereas PID assigns the correct category.

E.2 Expression-Conditioned Grounding

Figure 6 shows one DIOR-RSVG and two VRSBench-VG examples. Each expression requires distinguishing the target from similar neighboring objects: a chimney identified by its central position, a tennis court identified by its order from the top, and a ship identified by its proximity to the right edge. PID selects the correct referent in all three examples. Rex-Omni returns no valid box for the chimney and selects the wrong court and the wrong ship. LA-3B (FT)† selects the wrong chimney and the wrong court but finds the correct ship. Each panel shows the first parsed box for the complete expression, which is the box that is scored; later boxes are never used to replace it.

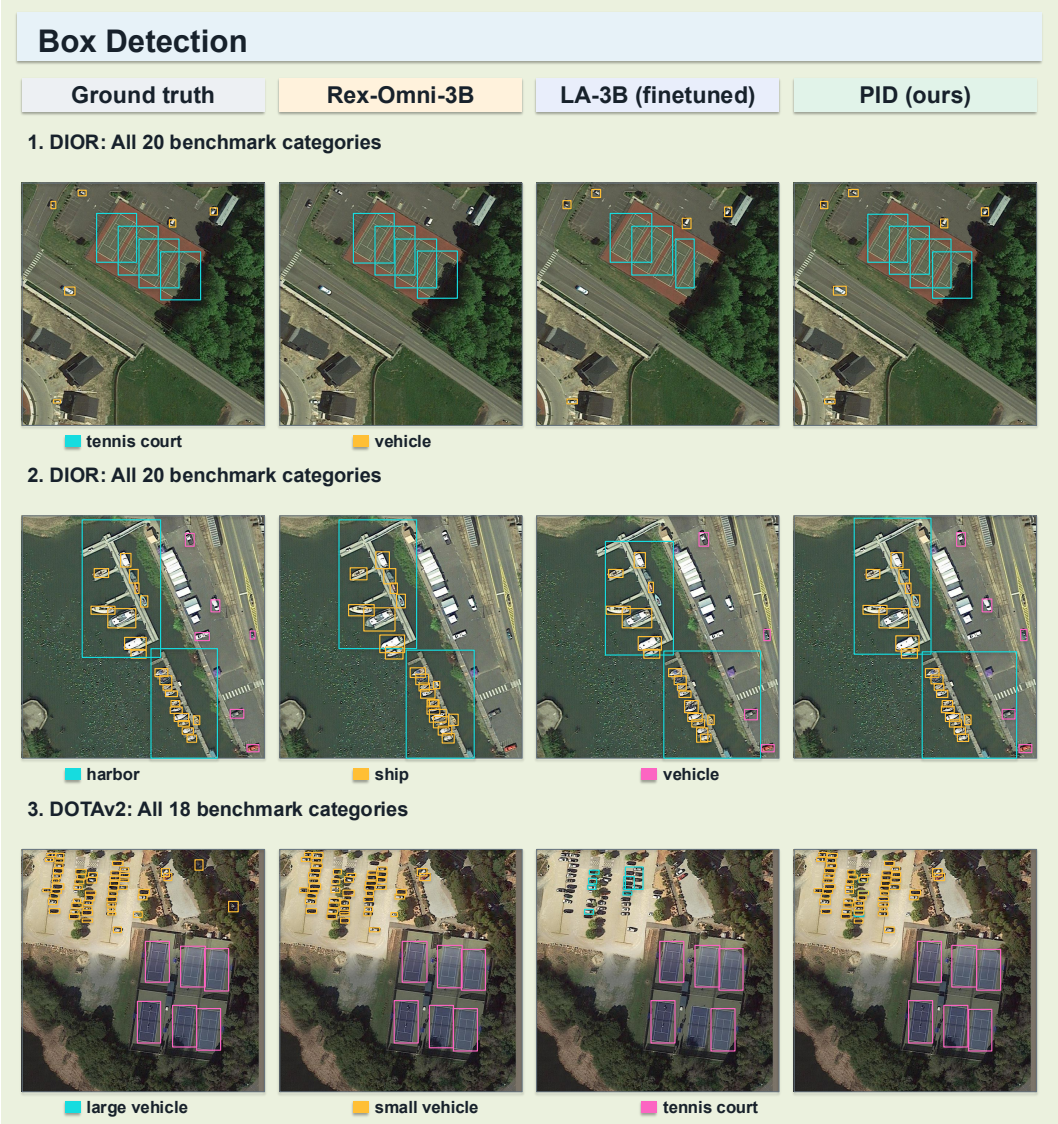

Figure 5: Qualitative full-vocabulary detection on DIOR (top and middle) and DOTAv2 (bottom). All benchmark categories are requested (20 for DIOR; 18 for DOTAv2), unlike the annotated-category setting of the main detection tables. Columns show ground truth, Rex-Omni-3B, LA-3B (FT)†, and PID. Colors indicate categories and are consistent across columns within each row.

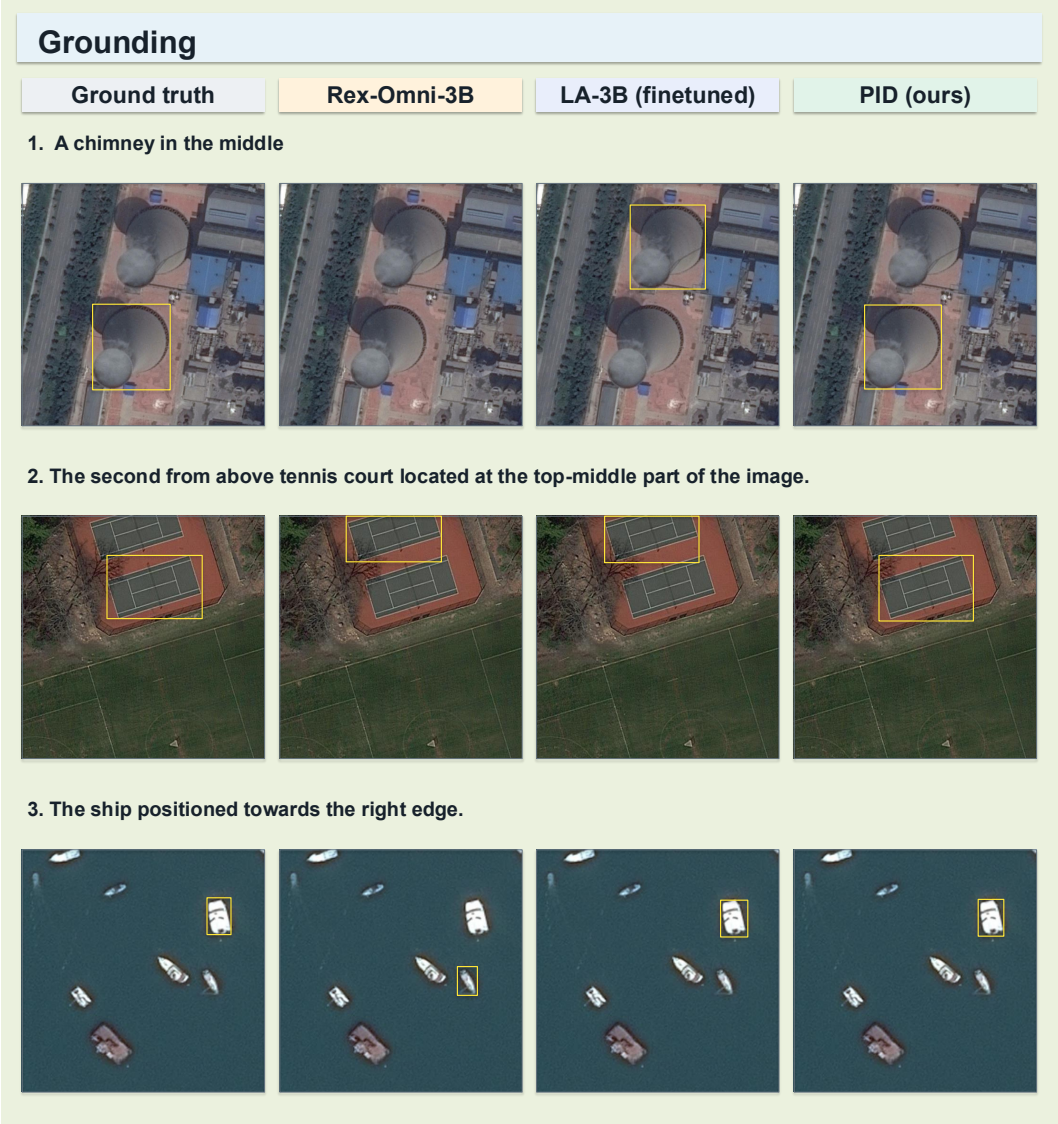

Figure 6: Qualitative referring-expression grounding on DIOR-RSVG (top) and VRSBench-VG (middle and bottom). The expressions refer to a central chimney, a tennis court specified by its order, and a ship near the right image edge. Columns show ground truth, Rex-Omni-3B, LA-3B (FT)†, and PID. Each prediction panel shows the first parsed box; a panel without a box means that no valid box was parsed.

E.3 POINTING

Figure 7 shows DIOR pointing examples for tennis courts, aircraft, and storage tanks. These requests test instance enumeration across a regular grid of identical objects, objects of very different sizes, and closely spaced objects. In the tennis-court scene, PID and Rex-Omni place one point on each of the twelve courts, whereas LA-3B (FT)† covers only half of them. In the aircraft scene, LA-3B (FT)† also places a point on empty ground. The ground-truth column shows the annotated target regions as dashed boxes, and the prediction columns show the predicted points. A point is correct if it lies inside an unmatched ground-truth box (Appendix C.2); it does not need to lie at the object center.

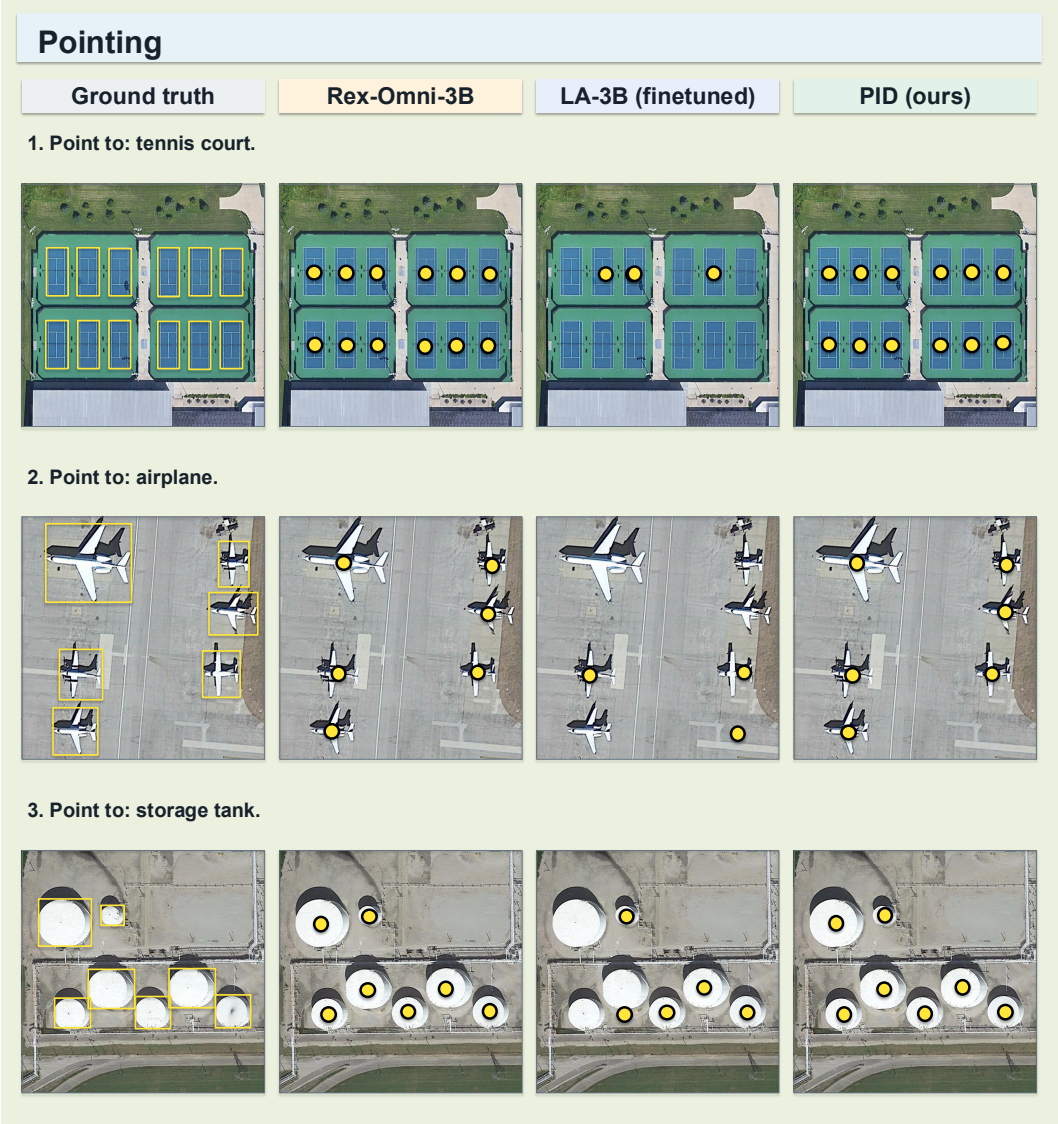

Figure 7: Qualitative category-conditioned pointing on DIOR: tennis courts (top), aircraft (middle), and storage tanks (bottom). Columns show ground truth, Rex-Omni-3B, LA-3B (FT)†, and PID. Dashed rectangles mark annotated targets and appear only in the ground-truth column. Model columns show the predicted points, which come from dedicated point requests rather than box centers.

Attribution of remaining errors. The remaining errors of PID include missed instances, duplicate predictions, incorrect referents, and imprecise boxes. The final outputs alone do not reveal which step caused a given error. For example, a missed object may have received no address, or it may have received an address for which the box decoder returned a null response. Separating these cases

requires the intermediate addresses. The annotation-address intervention (Table 5B) isolates box prediction by supplying correct addresses, while the pointing results (Table 3) reflect the quality of address prediction alone. Qualitative overlays by themselves cannot identify the internal cause of an error.