MVP: Multiple View Prediction Improves GUI Grounding

Supplementary Material

1. Details About Attention Heuristic Cropping

This section details our exploration of leveraging attention scores to better locate regions containing target UI elements in screenshots. Large Vision-Language Models (LVLMs) inherently possess strong text-visual alignment capabilities. Prior work indicates that text-to-vision attention scores from specific decoder layers can effectively locate instruction-relevant visual patches [2]. Furthermore, models adaptively adjust the attention assigned to visual tokens during text generation. Denoting the visual tokens as $V \in \mathbb{R}^{L_v \times d}$, we experiment with different text tokens as queries to compute the attention scores:

- Using all instruction tokens T_{instruct} as queries, averaging the final scores over the text length dimension.
- Using the first generated token "<im_start>" as the query.
- Using the comma token from the generated coordinate format "(x, y)" as the query, an insight inspired by GUI-Actor [1].
- Using the final generated token "<im_end>" as the query.

We conduct experiments with GTA1-7B on the ScreenSpot-Pro benchmark. Following the cropping procedure described in Section 3.1, we derive attention scores from the 20th decoder layer, set $k = 100$ and $m = 4$, and then evaluate two metrics: the ratio of top-m regions containing the target bounding box, and the final grounding accuracy after clustering.

Query Tokens	Target BBox Containing Ratio	SS-Pro Avg.
T_{instruct}	79.5%	60.5
$T_{<\text{im\_start}>}$	73.1%	52.2
$T_{<\text{im\_end}>}$	50.9%	33.3
T_{comma}	**83.4%**	**61.7**

Table 1. Comparison of cross-attention scores computed using different query tokens. The comma token yields the best performance and is therefore chosen as our default setting.

Our results(Table 1) show that using the comma token as the query yields the best localization performance, with 83.4% of the 4 selected views containing the target bounding box, which also translates to the highest final grounding accuracy. Consequently, we adopt this as our default configuration.

2. Additional Analysis and Ablations

Comparison with Attention-Driven Methods Following [2], we compare MVP with attention-driven grounding methods on ScreenSpot-Pro. The baseline "Top-1 att." directly outputs the center coordinate of the visual token with the highest attention score as the final prediction, without any generation step.

As shown in Table 2, this pure attention approach outperforms text-coordinate generation on weaker models (14.8 vs. 11.3 for Qwen2.5VL-3B), suggesting that attention scores provide useful spatial signal when generation capability is limited. However, on stronger models, text-based coordinate generation substantially surpasses the attention baseline (49.8 vs. 15.1 for GTA1-7B; 55.0 vs. 14.8 for Qwen3VL-8B), indicating that attention scores alone are too coarse for precise localization. MVP bridges these two paradigms: it uses attention scores as a lightweight signal to propose informative views, then relies on the model's generation capability to produce precise coordinates within each view. This hybrid design allows MVP to benefit from the spatial guidance of attention while remaining robust to its imprecision.

Method	**Qwen2.5VL-3B**	**GTA1-7B**	**Qwen3VL-8B**
Direct Generation	11.3	49.8	55.0
Top-1 Attention	14.8	15.1	14.8
MVP (Ours)	**19.8**	**61.7**	**65.3**

Table 2. Comparison with attention-driven grounding methods on ScreenSpot-Pro. MVP outperforms both pure attention and pure generation baselines by using attention for view proposal and generation for precise coordinate prediction.

Comparison with Repeated Sampling A natural question is whether MVP's gains simply arise from increased computation rather than from its design choices. To investigate this, we compare MVP against repeated sampling (temperature=1.0) with the same total inference budget of 5 forward passes on ScreenSpot-Pro, as shown in Table 3. Repeated sampling yields a Pass@N of only 52.4% and maintains the same accuracy as the single-view baseline (49.8%), since its diverse but undirected samples do not reliably cover the target region. In contrast, MVP achieves a substantially higher Pass@N of 70.2% through attention-guided view proposal, which concentrates views around instruction-relevant regions, and further distills these into an accurate final prediction via multi-coordinate clustering (61.7%). These results confirm that MVP's improvements stem from its principled view selection and aggregation strategy, not merely from increased computation.

Instability Analysis across Models and Perturbations To verify that prediction instability is a general phenomenon

Method	Number of Inferences	Pass@N	Acc
Repeated Sampling	5	52.4%	49.8%
MVP ($m = 4$)	5	**70.2%**	**61.7%**

Table 3. Comparison with repeated sampling under the same inference budget on ScreenSpot-Pro using GTA1-7B.

rather than an artifact of a specific model or perturbation type, we evaluate both GTA1-7B and Qwen3VL-8B under four distinct perturbation types: resolution scaling ($\times 1.5$), pixel shifting (28 pixels), addition of specific visual element(e.g. triangle with random colors), and Gaussian noise (std=25). For each setting, we report the rates of correct-to-wrong (C→W) and wrong-to-correct (W→C) prediction flips in Table 4. Across all perturbation types and both models, we observe substantial bidirectional flip rates, confirming that instability is a common phenomenon across models rather than a model-specific or perturbation-specific artifact. These results collectively motivate the need for a general inference-time solution such as MVP.

Perturbation	GTA1-7B (C→W / W→C)	Qwen3VL-8B (C→W / W→C)
Scaling ($\times 1.5$)	9.8% / 7.5%	3.7% / 6.0%
Shifting (28 pixels)	7.3% / 7.8%	4.6% / 3.6%
Adding visual element	6.6% / 4.6%	3.5% / 3.8%
Gaussian noise (std=25)	7.2% / 5.8%	5.4% / 3.2%

Table 4. Prediction flip rates under different perturbation types. Substantial flip rates across all conditions confirm that instability is a general phenomenon in current grounding models, affecting both model architectures and diverse perturbation types.

Efficiency Analysis We measure the inference time, peak GPU memory, and total FLOPs per sample for GTA1-7B on ScreenSpot-Pro using an RTX A6000, as reported in Table 5. MVP processes all sub-region views in a single batched forward pass rather than sequentially, which significantly reduces the per-view overhead compared to naive sequential inference. With $m = 2$, MVP achieves an 11.2-point accuracy gain at only $1.65\times$ the inference time and a moderate memory increase of 0.5 GB, representing a highly favorable efficiency–accuracy trade-off. Scaling to $m = 4$ yields a further 0.7-point gain at $2.02\times$ overhead. These measurements are based on a standard PyTorch implementation; integration with optimized inference frameworks such as vLLM or SGLang could further reduce latency, which we leave as future work.

Necessity of Clustering A key design choice in MVP is the use of Multi-Coordinate Clustering rather than simply selecting the single best view for final prediction. To validate this choice, we compare our clustering strategy against a *single-best-view* baseline, which directly performs inference on the view containing the highest concentration of top-k attended visual tokens (i.e., the top-ranked view from the Region Ranking step) without any clustering. As shown in Table 6, the single-best-view strategy achieves 58.0% on ScreenSpot-Pro, while our clustering approach with $m = 4$ views reaches 61.7%, a gain of 3.7 points. This demonstrates that even when the best view is selected by a strong attention-guided criterion, aggregating predictions across multiple views through spatial clustering provides additional robustness. The improvement stems from clustering's ability to identify spatial consensus: while any single view may produce an imprecise prediction due to residual instability, the correct coordinates from multiple independently cropped views consistently cluster near the true target, allowing the centroid of the largest cluster to be more accurate than any individual view prediction.

Method	Time	Mem.	FLOPs	Overhead	Acc.
GTA1-7B	1.12s	21.4GB	3.91e+13	1.0×	49.8
+MVP (m=2)	1.85s	21.9GB	4.94e+13	1.65×	61.0
+MVP (m=4)	2.26s	22.2GB	5.96e+13	2.02×	61.7

Table 5. Inference efficiency of MVP on ScreenSpot-Pro using GTA1-7B on an RTX A6000. MVP uses batched inference over sub-regions to minimize overhead. At $m = 2$, an 11.2-point accuracy gain is achieved with only $1.65\times$ inference time, demonstrating a favorable efficiency–accuracy trade-off.

Method	Number of Views	SS-Pro Avg.
Baseline (Single Full Image)	1	49.8
Single-Best-View Prediction	1	58.0
Multi-Coordinate Clustering (Ours)	4	**61.7**

Table 6. Comparison between single-best-view prediction and our Multi-Coordinate Clustering on ScreenSpot-Pro using GTA1-7B. Single-best-view prediction selects the top-ranked view by attention score and directly outputs its coordinate without clustering. The 3.7-point gap confirms that aggregating predictions across multiple views through spatial clustering provides additional robustness beyond selecting the single best view.

Benchmark	$m = 2$	$m = 4$	$m = 6$	$m = 10$
ScreenSpot-Pro	65.3	65.3	**65.8**	64.8
UI-Vision	31.9	32.4	**32.5**	31.6

Table 7. Hyperparameter study on the number of views m using Qwen3VL-8B-Instruct on ScreenSpot-Pro and UI-Vision.

Hyperparameter Study We investigate the sensitivity of MVP to the number of views m across different benchmarks and model scales. Table 7 reports results for Qwen3VL-8B-Instruct on ScreenSpot-Pro and UI-Vision under $m \in$

$\{2, 4, 6, 10\}$. Performance remains stable across a wide range of m values on both benchmarks, with peak accuracy achieved at $m = 6$ on ScreenSpot-Pro (65.8%) and $m = 6$ on UI-Vision (32.5%), but the differences relative to $m = 2$ or $m = 4$ are marginal (within 0.5 points). Beyond $m = 6$, performance slightly decreases, consistent with the saturation effect discussed in the main paper: additional views do not introduce new spatial clusters but increase inference cost. Based on this analysis, we adopt $m = 2$ for stronger models such as Qwen3VL to balance accuracy and efficiency, and $m = 4$ for other models where the additional views provide a more meaningful diversity gain.

Ablation on Qwen3VL-8B To verify that the ablation findings reported in the main paper generalize beyond GTA1-7B, we reproduce the key ablation studies using Qwen3VL-8B-Instruct. Table 9 compares view proposal strategies, and Table 8 compares coordinate aggregation strategies, both on ScreenSpot-Pro.

For view proposal, our Attention-Guided View Proposal achieves 65.3%, substantially outperforming both the single full-image baseline (55.0%) and the Border Padding strategy (60.2%), mirroring the pattern observed with GTA1-7B. For aggregation, our Multi-Coordinate Clustering achieves 65.3%, outperforming coordinate averaging (61.0%) and random selection (60.1%). Notably, coordinate averaging again underperforms random selection, reinforcing the finding that naive averaging is sensitive to scattered incorrect predictions. These consistent results across two architectures confirm that both the attention-guided view proposal and the clustering-based aggregation are general components that transfer effectively across different grounding models.

Aggregation Method	**Number of Views**	**SS-Pro Avg.**
Baseline (Single Full Image)	–	55.0
Average of Coordinates	2	61.0
Random Selection	2	60.1
Multi-Coordinate Clustering	2	**65.3**

Table 8. Ablation on aggregation strategies using Qwen3VL-8B-Instruct on ScreenSpot-Pro.

View Proposal Method	**Number of Views**	**SS-Pro Avg.**
Baseline (Single Full Image)	–	55.0
Border Padding	2	60.2
Attention-Guided View Proposal	2	**65.3**

Table 9. Ablation on view proposal methods using Qwen3VL-8B-Instruct on ScreenSpot-Pro.

3. Coordinate Selection via Trained Model

In this section, we explore an alternative to clustering: training a dedicated model to select the correct coordinate from multiple candidate predictions. The motivation stems from Figure 1(b), which shows that the probability of having at least one correct prediction among the views (Pass@N) increases with the number of views. However, as shown in Table 10, while our clustering method significantly surpasses the single-view baseline, its accuracy remains lower than the Pass@N upper bound. This indicates a potential performance gap that could be bridged by a perfect selection model.

View Number	Clustering Acc	Pass@N Acc
2	61.0	69.0
4	61.7	70.2
10	60.6	73.0

Table 10. Comparison between clustering accuracy and Pass@N accuracy. The gap indicates the potential room for improvement with an ideal selection model.

Prompt For Coordinate Selection Model

```
System Prompt:
You are an expert UI element verifier.
    Given a original GUI screenshot, the
    GUI screenshot annotated with some
    numbered candidate points (each marked
     with a red dot and a corresponding
    number label under the dot) and a user
    's instruction, you are expected to
    choose single most appropriate point
    that user most likely to click based
    on the instruction step by step.
    Return the annotated number under the
    optimal point in bracket: [Number].

User Prompt:
Instruction + Annotated Image

Output Format:
[Number Label]
```

Data Preparation We utilize the open-source GUI grounding dataset from GTA1 [3]. The data is firstly filtered with the following rules: (1) image resolution larger than 2560×1440; (2) bounding box area smaller than 500 pixels^2. This process yields approximately 20k samples. For each sample, we annotate 2-4 distinct red points on the image, each with a numerical label, as shown in Figure 1. One point is placed within the target bounding box, while the others are randomly distributed outside it. The annotation metadata, including the instruction, target bounding box, point coordinates, and the image, is saved for training.

Model Training We employ GRPO (Guided Reinforcement Policy Optimization) to train a model to directly output the numerical label of the correct point. The model takes

the annotated image and user instruction as input. The rule-based reward is defined as follows: if the model outputs the correct point label, the reward is 1; otherwise, it is 0. We use Qwen3VL-4B-Instruct as the base model and train it on 8 A6000 GPUs, with 8 rollouts per group and a gradient accumulation step of 32, for a total of 170 optimization steps. The average reward converged, rising from 0.47 to 0.68.

Evaluation and Analysis We evaluate the trained model by having it determine the final coordinate from multiple view predictions, with the expectation that it could achieve performance close to the Pass@N upper bound. Specifically, after obtaining coordinate predictions from diverse views, we annotate them as red dots with number labels on the screenshot and prompt the trained model to generate the label of the point a user is most likely to click based on the instruction.

Base Model	Aggregation Method	SS-Pro Avg.
GTA1-7B	Qwen3VL-4B-Instruct	60.5
GTA1-7B	Qwen3VL-4B-Instruct (Trained)	62.8
GTA1-7B	Clustering (Ours)	**61.7**
Qwen3VL-8B-Instruct	Qwen3VL-4B-Instruct	62.7
Qwen3VL-8B-Instruct	Qwen3VL-4B-Instruct (Trained)	65.3
Qwen3VL-8B-Instruct	Clustering (Ours)	**65.5**

Table 11. Performance comparison when using another LVLM versus our clustering method for coordinate aggregation. The training improves performance of selector model over it's baseline, but still fails to consistently outperform the simple clustering.

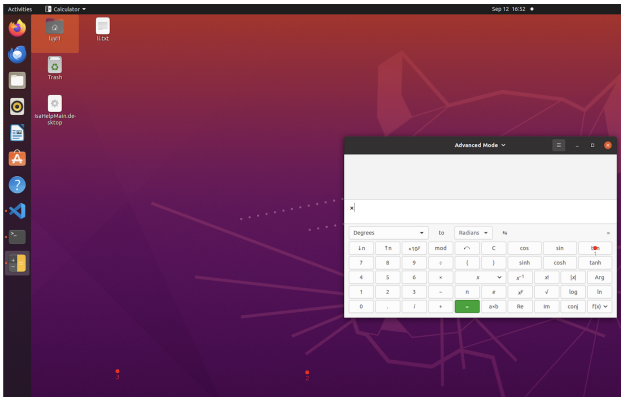

Figure 1. Example of annotated image. We annotate 2-4 visible red dots with corresponding numerical label for each sample. The model is trained to directly output the correct label.

As shown in Table 11, the trained selector model fails to consistently surpass our clustering method. While it shows a minor improvement, it is outperformed by clustering in the critical comparison with Qwen3VL-8B-Instruct. This result suggests that training a separate model for this selection task is not an effective strategy, as the performance gain is marginal and inconsistent, failing to justify the additional complexity and training cost. The clustering method remains a more robust and reliable aggregation strategy.

4. Case Study

We found 46 cases (2.9%) where MVP fails while origin model succeeds on ScreenSpot-Pro for Qwen3VL-8B. Although MVP introduces some failure cases, it corrects 209 samples (13.2%) originally predicted incorrectly, far outweighing the errors it produces.

Among the 46 failure cases, we identify three failure types. The first type accounts for 16 cases, in which the attention-guided cropping fails to include the target bounding box in any of the generated views. This occurs when the user instruction is semantically ambiguous or overly generic (e.g., "forward", "zoom in"), causing the attention scores to diffuse across irrelevant regions rather than localizing the target UI element. The second type accounts for 23 cases, in which the coordinate predictions across views are too spatially dispersed to form any dominant cluster. In such cases, MVP falls back to outputting the prediction from the highest-ranked single view via Equation 8, which may itself be incorrect. This failure mode typically arises when the target element is visually ambiguous across different cropped perspectives, causing the model to produce inconsistent predictions without a clear spatial consensus. The third type accounts for the remaining 7 cases, in which a majority of views independently converge on the same incorrect coordinate, causing an erroneous cluster to dominate over the correct one. This occurs when a salient but incorrect UI element consistently attracts the model's attention across multiple views, leading the clustering to select the wrong target.

To provide intuitive insight into these failure modes and the cases where MVP successfully corrects the original model's predictions, we visualize some representative samples.

References

[1] Qianhui Wu, Kanzhi Cheng, Rui Yang, Chaoyun Zhang, Jianwei Yang, Huiqiang Jiang, Jian Mu, Baolin Peng, Bo Qiao, Reuben Tan, Si Qin, Lars Liden, Qingwei Lin, Huan Zhang, Tong Zhang, Jianbing Zhang, Dongmei Zhang, and Jianfeng Gao. Gui-actor: Coordinate-free visual grounding for gui agents, 2025. 1

[2] Hai-Ming Xu, Qi Chen, Lei Wang, and Lingqiao Liu. Attention-driven gui grounding: Leveraging pretrained multimodal large language models without fine-tuning. In *The 39th Annual AAAI Conference on Artificial Intelligence*, 2025. 1

[3] Yan Yang, Dongxu Li, Yutong Dai, Yuhao Yang, Ziyang Luo, Zirui Zhao, Zhiyuan Hu, Junzhe Huang, Amrita Saha, Zeyuan Chen, Ran Xu, Liyuan Pan, Silvio Savarese, Caiming Xiong, and Junnan Li. Gta1: Gui test-time scaling agent, 2025. 3

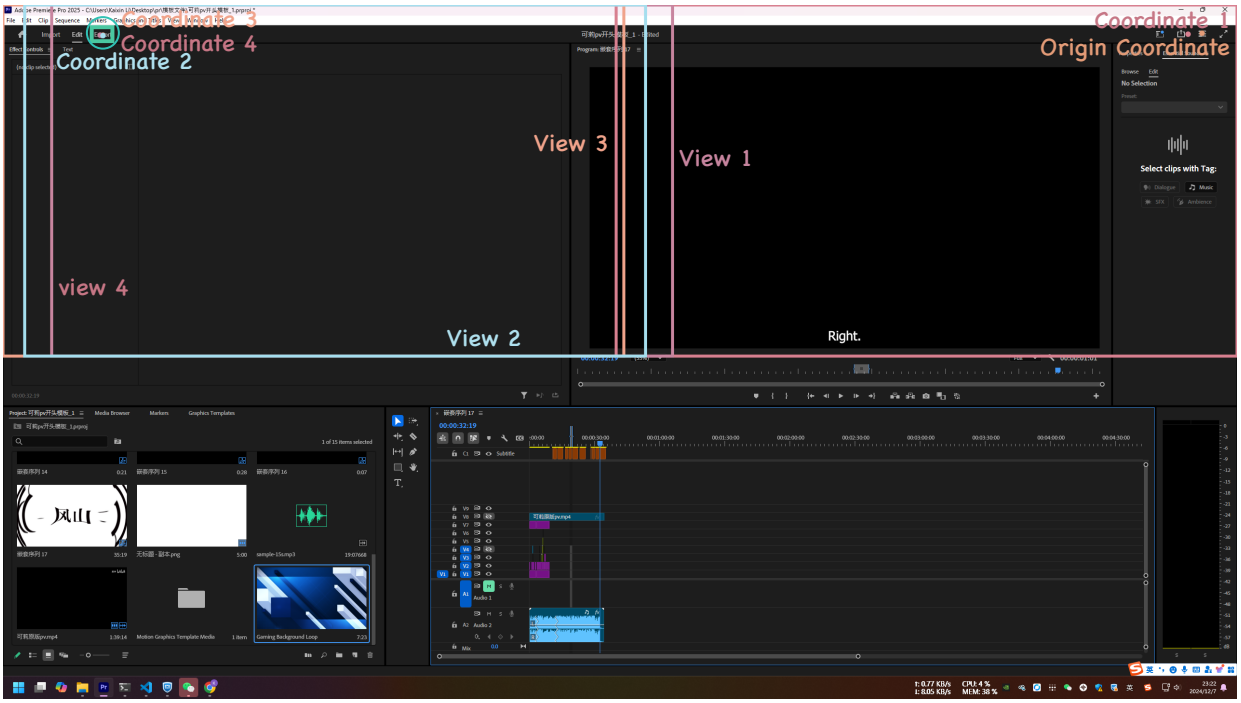

Figure 2. Multi-view example from SS-Pro evaluated by GTA1-7B. Instruction is “change to export workspace”.

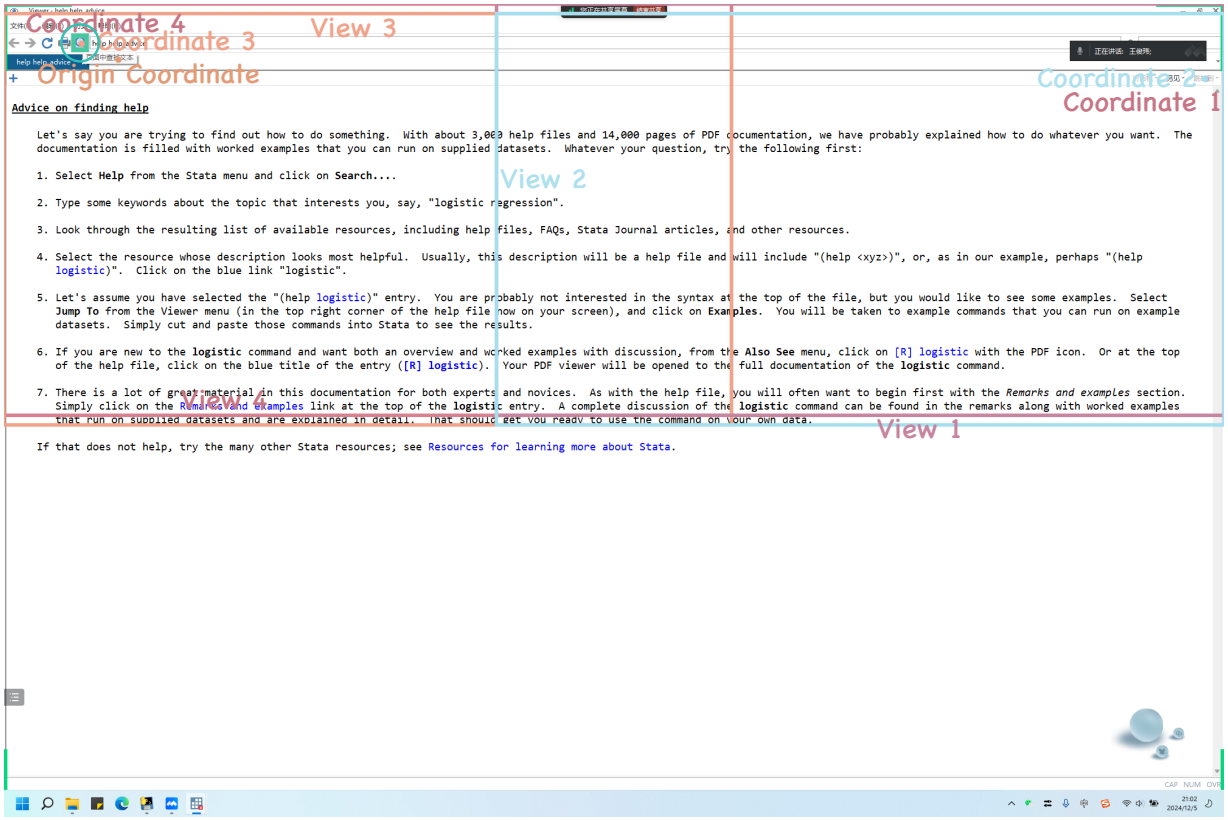

Figure 3. Multi-view example from SS-Pro evaluated by GTA1-7B. Instruction is “find text on the page”.

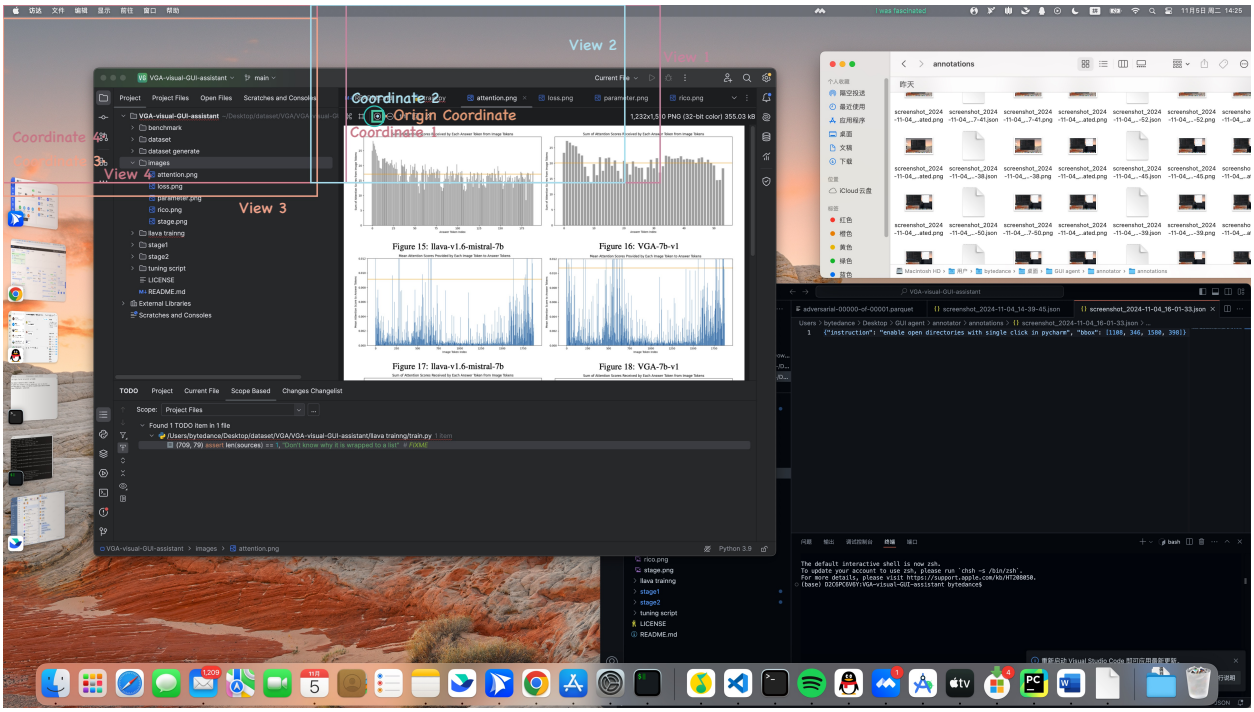

Figure 4. Multi-view example from SS-Pro evaluated by GTA1-7B. Instruction is “zoom in the image in pycharm”.

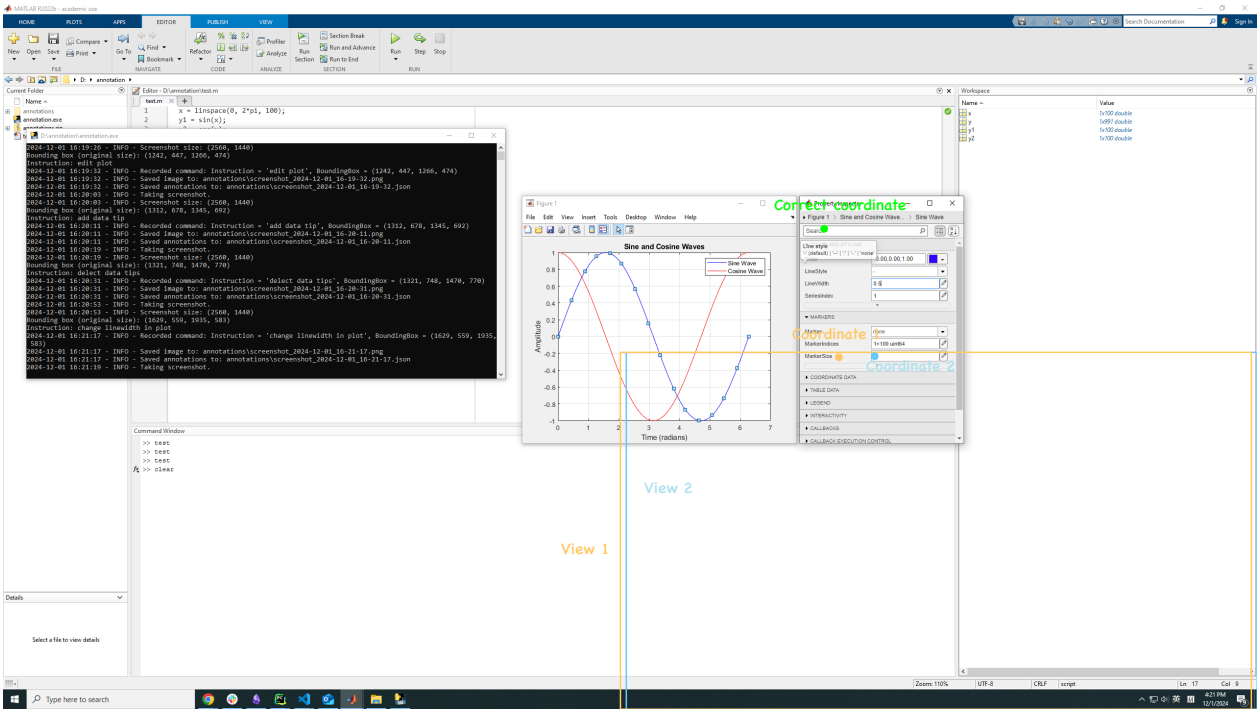

Figure 5. MVP failure case. Instruction is “change simulator language in vivado”. Target Bbox is outside the cropped views.

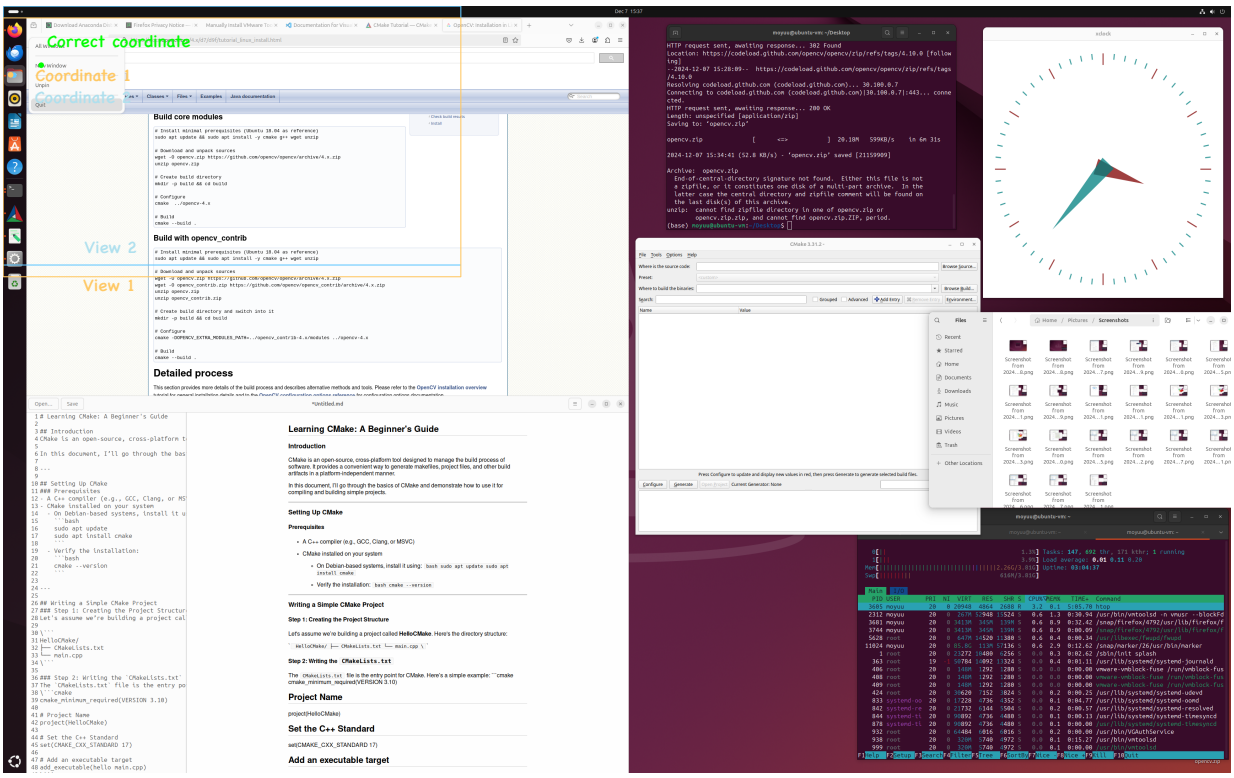

Figure 6. MVP failure case. Instruction is “launch a new file explorer”. Incorrect predictions accidentally clustered together.

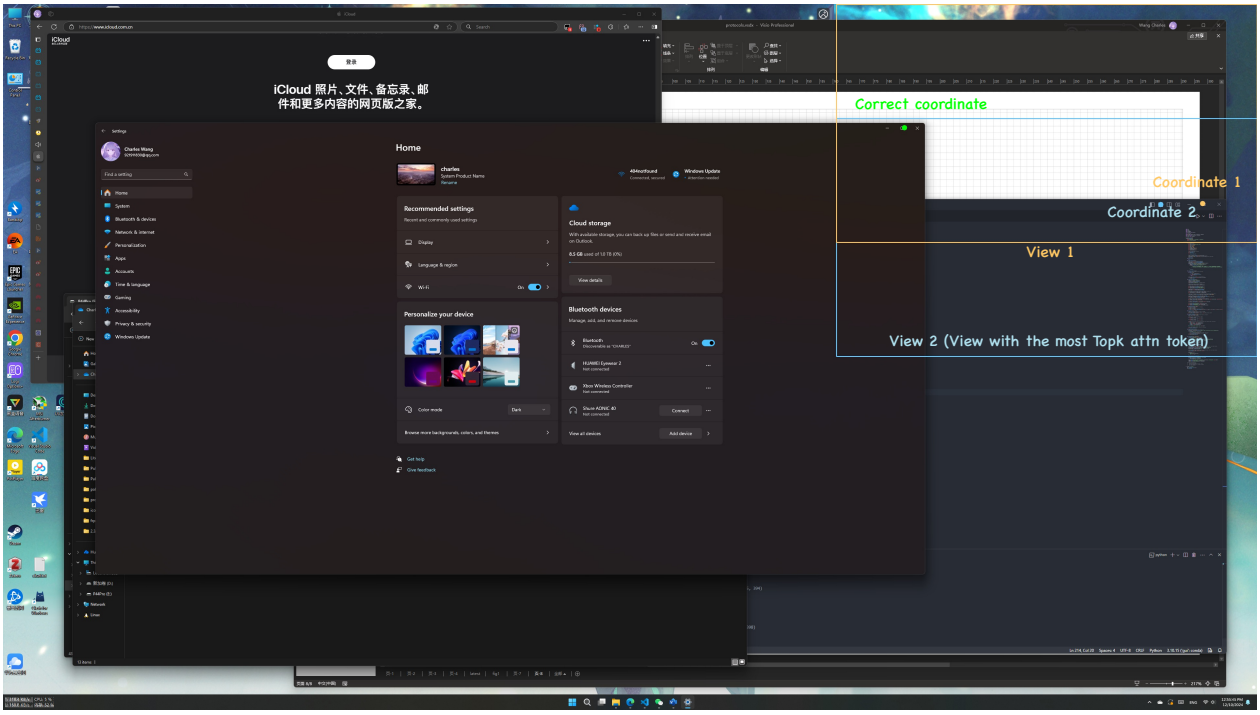

Figure 7. MVP failure case. Instruction is “maximize window or change split view for settings”. Predicted coordinates are too spatially dispersed to form any dominant cluster.