BAMI: Training-Free Bias Mitigation in GUI Grounding

Supplementary Material

Table of Contents for Supplementary Material

A. Usage of Large Models in Paper Writing

During the conduct of this research, we utilized the GPT-5 for auxiliary support, primarily encompassing the following two aspects:

- **Manuscript Polishing:** Leveraging the text generation capability of GPT-5, we polished the draft of this manuscript, focusing on correcting grammatical errors, addressing expression inconsistencies, and other related issues. It should be emphasized that all content of the manuscript was still manually composed; the LLM was not involved in formulating the research logic of the paper. Additionally, all text generated by the LLM underwent manual review and revision to ensure its quality and accuracy.
- **Literature Survey:** We employed the knowledge retrieval capability (Retrieval-Augmented Generation, RAG) of GPT-5 to search for relevant literature. To guarantee retrieval accuracy, all retrieved literature was subject to manual review and verification. Subsequently, we screened out literature relevant to the research topic, followed by thorough reading and systematic organization of the selected materials.

B. Details of the Proposed Methods

B.1. Detailed Algorithm of MPD Attribution

To investigate the root causes of errors in grounding models, we propose a method for rapidly computing the decision attribution of models, namely **Masked Prediction Distribution (MPD) Attribution**. The detailed steps of this algorithm are presented as follows:

1. Masked Prediction Distribution (MPD) Attribution Algorithm

Require: GUI image I, query q, grid size (M, N), number of samples K

Ensure: Set of predicted points $\mathcal{P} = \{(x_c^{(k)}, y_c^{(k)})\}_{k=1}^{K}$

Partition the image I into $M \times N$ grid blocks $\{B_{i,j}\}_{i=1,j=1}^{M,N}$
for $k = 1$ to K **do**
  Randomly select a masking ratio α and sample $\lfloor \alpha \cdot M \cdot N \rfloor$ grid blocks to mask
  Generate the masked image $I^{(k)}$, where masked regions are filled with zero vectors
  Compute the model prediction: $t^{(k)} = f(q, I^{(k)})$
  Extract the center coordinates: $(x_c^{(k)}, y_c^{(k)})$
end for
Visualize all predicted points $\{(x_c^{(k)}, y_c^{(k)})\}_{k=1}^{K}$ as a scatter plot

C. Experimental Details

C.1. Prompt Design

The design of prompts is crucial for injecting prior information of coordinate space into the candidate box selection process. In the experiments presented in Table 4 (main paper), we compare prompts with different content. Among these, the vanilla prompt is as follows:

Prompt

```
You are comparing two images to determine which one better fulfills the user's intent.

User Command: "{user_query}"

Image 1: Shows a GUI element marked with a green box labeled "1"
Image 2: Shows a GUI element marked with a red box labeled "2"

Your task: Determine which image shows the element that will best fulfill the user's command.

**OUTPUT FORMAT**:
<answer>1 or 2</answer>"""
```

This simplistic prompt design fails to rectify the model's ambiguity bias. Therefore, in our BAMI method, we incorporate two critical structures—chain of thought and key principle—to enhance the model's understanding of prior information regarding the coordinate space. The final prompt we employed is presented as follows:

Prompt

```
You are comparing two images to determine which one better fulfills the user's intent.

User Command: "{user_query}"

Image 1: Shows a GUI element marked with a green box labeled "1"
Image 2: Shows a GUI element marked with a red box labeled "2"

Your task: Determine which image shows the element that will best fulfill the user's command.

ANALYSIS APPROACH:
1. Examine what GUI element is highlighted in each image
2. Consider which element better matches the user's intent
3. Think about standard GUI patterns and user expectations
4. Choose the image that shows the more appropriate interaction target

KEY PRINCIPLES:
- Focus on the functional purpose of the highlighted elements
- Consider standard UI patterns (buttons for actions, text fields for input, etc.)
- Choose interactive elements over static text/labels
- If one shows a selected state and the other shows normal state, prefer the normal state
- ELEMENT QUALITY HIERARCHY (best to worst):
   - Icon + Text together (most informative and complete)
   - Complete icon alone (clear visual indicator)
   - Complete text alone (readable label)
   - Multiple elements in one box OR incomplete elements (ambiguous target)

COMMON PITFALLS TO AVOID:
    - Don't choose based on keyword matching alone
    - Don't overlook the user's actual goal in favor of literal interpretation

Remember: Provide SPECIFIC analysis based on what you actually observe, not generic descriptions.

**OUTPUT FORMAT**:
<analysis>
Image 1: [Describe what element is highlighted and its purpose]
Image 2: [Describe what element is highlighted and its purpose]
Comparison: [Explain which better serves the user's intent and why]
</analysis>

<answer>1 or 2</answer>
<reason>Brief explanation of why this image shows the better choice</reason>
```

C.2. Model Inference Details

The models employed in this study can be broadly categorized into two types:

- **Bounding box-output models**: Such as OS-Atlas-7B [9] and TianXi-Action-7B [7]
- **Click point-output models**: Such as UGround [3] and UI-TARS-1.5-7B [6]

For **bounding box-output models**, the implementation of masked prediction is straightforward—only the pixels within the output bounding boxes need to be masked. In contrast, for **click point-output models**, we first expand the region around each click point by a fixed number of pixels (e.g., 25 pixels) in the up, down, left, and right directions, and then mask the expanded region.

C.3. Local Correction Model Training

To enable offline deployment of BAMI, we trained a specialized correction model based on Qwen3-VL-8B using LoRA fine-tuning. The training dataset contains 128,487 dual-box samples automatically generated via our five-step pipeline, sourced from GUIAct (70K samples) and Desktop domain (423K samples, then downsampled). Labels are determined by comparing against ground truth using dual criteria: IoU $\geq$ 0.5 or center point within GT bbox. When both boxes satisfy the criteria, we prioritize bbox1 (baseline) to reflect regrounding's role as a fallback mechanism, resulting in a 92:8 label distribution (bbox1:bbox2).

We fine-tune only the language model component via LoRA (rank r=128, alpha α=256, dropout=0.05) while freezing the vision encoder and projection layers, yielding approximately 200M trainable parameters (2.5% of total). This design leverages the pre-trained visual understanding while adapting the decision-making capability for dual-box selection. Training employs 8$\times$ A100 80GB GPUs with DeepSpeed ZeRO-2 optimization, effective batch size 128, learning rate 1e-4 with cosine annealing, for 3 epochs (approximately 12 hours). The model uses the same 24-line instruction prompt as GPT-5 (detailed in Section C.1) to ensure consistent task understanding and incorporate GUI-specific priors.

On ScreenSpot-Pro evaluation, the local model selects bbox2 in 9.7% of cases, closely matching the training distribution (8%), indicating proper learning of the selection strategy without overfitting to always choose baseline. The 56.2% accuracy demonstrates that comparable-scale models (8B parameters) can effectively perform correction tasks without requiring significantly larger architectures.

D. More Experiments

D.1. Comparison on ScreenSpot-V2

In addition to validating the BAMI method on the ScreenSpot-Pro [5] dataset, we also conducted validation on the simpler ScreenSpot-V2 [9] dataset. Unlike ScreenSpot-Pro, most grounding models already achieve satisfactory accuracy on ScreenSpot-V2; this is attributed to the lower resolution of samples and the simpler elements contained in individual samples within the latter dataset. When we applied the BAMI method to the OS-Atlas-7B and UI-TARS-1.5-7B models, further performance improvements were observed. However, the magnitude of these improvements is smaller than that achieved on the ScreenSpot-Pro dataset.

Table 1. Comparison with various models on ScreenSpot-V2.

Grounding Model	Mobile		Desktop		Web		Avg.
	Text	Icon	Text	Icon	Text	Icon	
InternVL-2-4B [1]	9.2	4.8	4.6	4.3	0.9	0.1	4.3
Qwen2-VL-7B [8]	61.3	39.3	52.0	45.0	33.0	21.8	42.9
CogAgent [4]	67.0	24.0	74.2	20.0	70.4	28.6	47.4
SeeClick [2]	78.0	52.0	72.2	30.0	55.7	32.5	53.4
OS-Atlas-4B [9]	85.7	58.5	72.2	45.7	82.6	63.1	70.1
UGround-7B [3]	82.8	82.8	82.8	63.6	80.4	70.4	73.3
OS-Atlas 7B [9]	92.1	68.7	88.7	60.7	89.7	75.9	81.2
+ BAMI	**92.4**	**67.3**	**88.7**	**66.4**	**89.3**	**79.8**	**82.2**
UI-TARS-1.5-7B [6]	94.1	80.6	88.7	76.4	88	84.2	86.4
+ BAMI	**94.1**	**80.6**	**88.7**	**76.4**	**88**	**84.7**	**86.5**

D.2. Why Masking Is Adopted Instead of Random Sampling?

In conventional approaches for generating candidate detection boxes, random sampling is typically employed. Specifically, when predicting the next token, instead of using the `torch.argmax` function to greedily select the token corresponding to the highest score, top-k/top-p sampling methods are utilized to obtain candidate tokens. However, our experiments reveal that in GUI grounding models during candidate box generation, the score difference between the top-1 token and top-2 token is substantial. This directly leads to a significant issue: candidate boxes generated via random sampling tend to cluster in a single region. As illustrated in Figure 1a, the red boxes represent candidate boxes obtained through random sampling. It is evident that these boxes exhibit almost complete overlap and lack diversity, which renders the subsequent selection process largely meaningless.

To address this limitation, we propose a masking strategy: pixels within the already predicted candidate boxes are masked first. This ensures that subsequently predicted candidate boxes are mutually exclusive with the already predicted ones. As shown in Figure 1b, the green boxes are candidate boxes generated using the masked prediction method. These boxes demonstrate significantly greater distribution diversity, thereby enhancing the upper performance limit of selection manipulation.

D.3. More Visualizations of BAMI

To better demonstrate the process by which BAMI corrects the baseline model, 8 samples were randomly selected from cases where the baseline model made incorrect predictions but BAMI achieved accurate corrections, as shown in Figure 2. In the figure, green boxes represent ground truth, red boxes denote the baseline model's prediction results (incorrect), and blue boxes indicate the corrected results by BAMI (correct). Specifically, BAMI utilized 2 candidate boxes in each prediction round of this experiment, with GPT-5 as the correction model. In these samples, it can be observed that accurately predicting bounding boxes in accordance with user instructions is considerably challenging, as the figures contain substantial interfering information. By alleviating precision bias and ambiguity bias, BAMI successfully achieves correct predictions in these samples.

D.4. More Attribution Results

We present additional attribution results herein to comprehensively demonstrate the attribution capability of the Masked Prediction Distribution (MPD) method. Specifically, we randomly selected samples from four categories (`Correct / Knowledge Gap / Precision Bias / Ambiguity Bias`) as illustrated in Figure 3.

References

[1] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *CVPR*, pages 24185–24198, 2024.

[2] Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, Jianbing Zhang, and Zhiyong Wu. Seeclick: Harnessing gui grounding for advanced visual gui agents. *arXiv*, abs/2401.10935, 2024.

[3] Boyu Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. Navigating the digital world as humans do: Universal visual grounding for gui agents. In *ICLR*, 2025.

[4] Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, et al. Cogagent: A visual language model for gui agents. In *CVPR*, 2024.

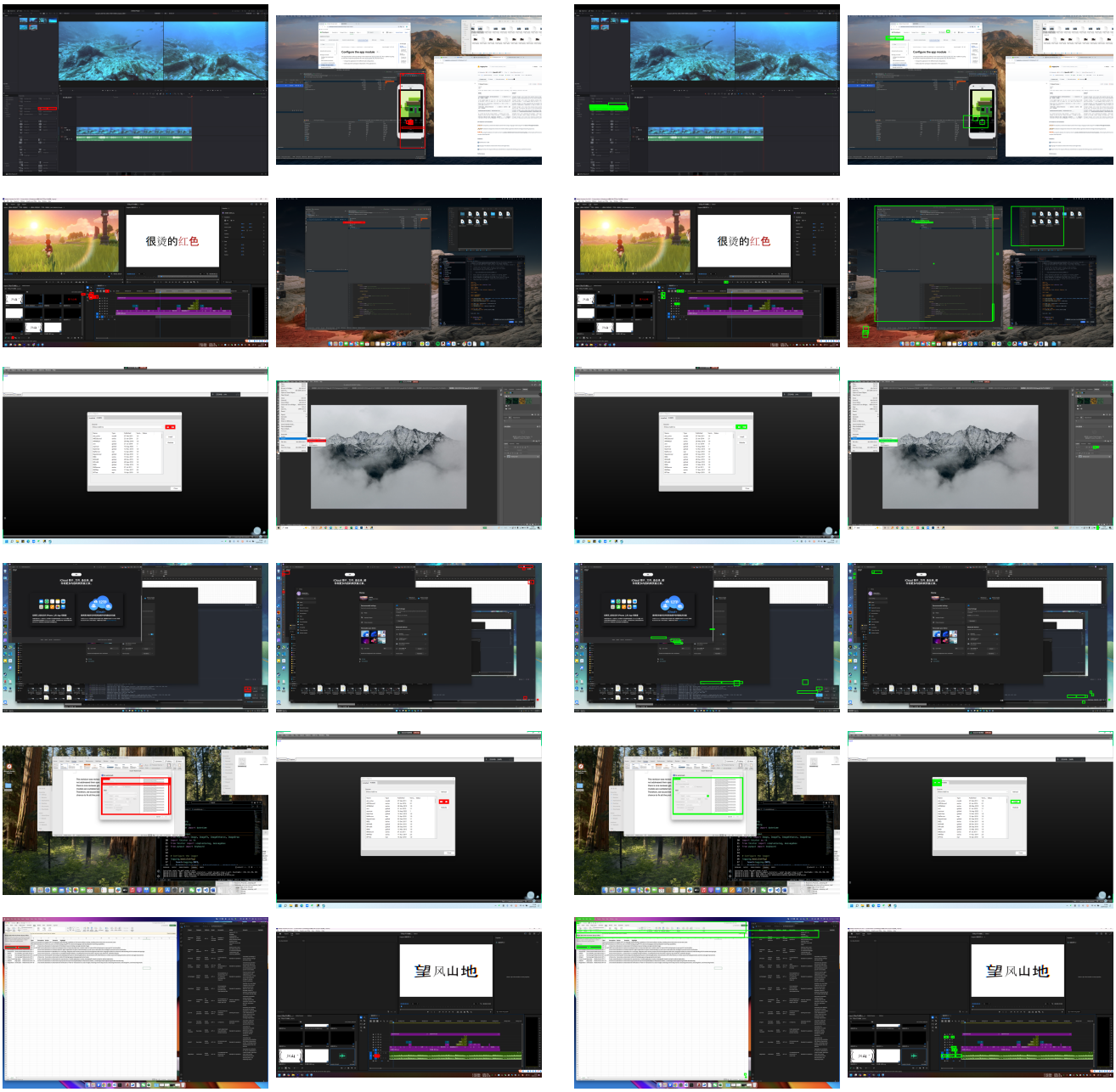
[5] Kaixin Li, Ziyang Meng, Hongzhan Lin, Ziyang Luo, Yuchen Tian, Jing Ma, Zhiyong Huang, and Tat-Seng Chua. Screenspot-pro: Gui grounding for professional high-resolution computer use. *arXiv*, abs/2504.07981, 2025.

[6] Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, et al. Ui-tars: Pioneering automated gui interaction with native agents. *arXiv*, abs/2501.12326, 2025.

[7] Liang Tang, Shuxian Li, Yuhao Cheng, Yukang Huo, Zhepeng Wang, Yiqiang Yan, Kaer Huang, Yanzhe Jing, and Tiaonan Duan. Sea: Self-evolution agent with step-wise reward for computer use. *arXiv*, abs/2508.04037, 2025.

[8] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv*, abs/2409.12191, 2024.

[9] Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, et al. Os-atlas: A foundation action model for generalist gui agents. *arXiv*, abs/2410.23218, 2024.



(a) Candidate boxes with random sampling.

(b) Candidate boxes with masked prediction.

Figure 1. Comparison of candidate box generation strategies.

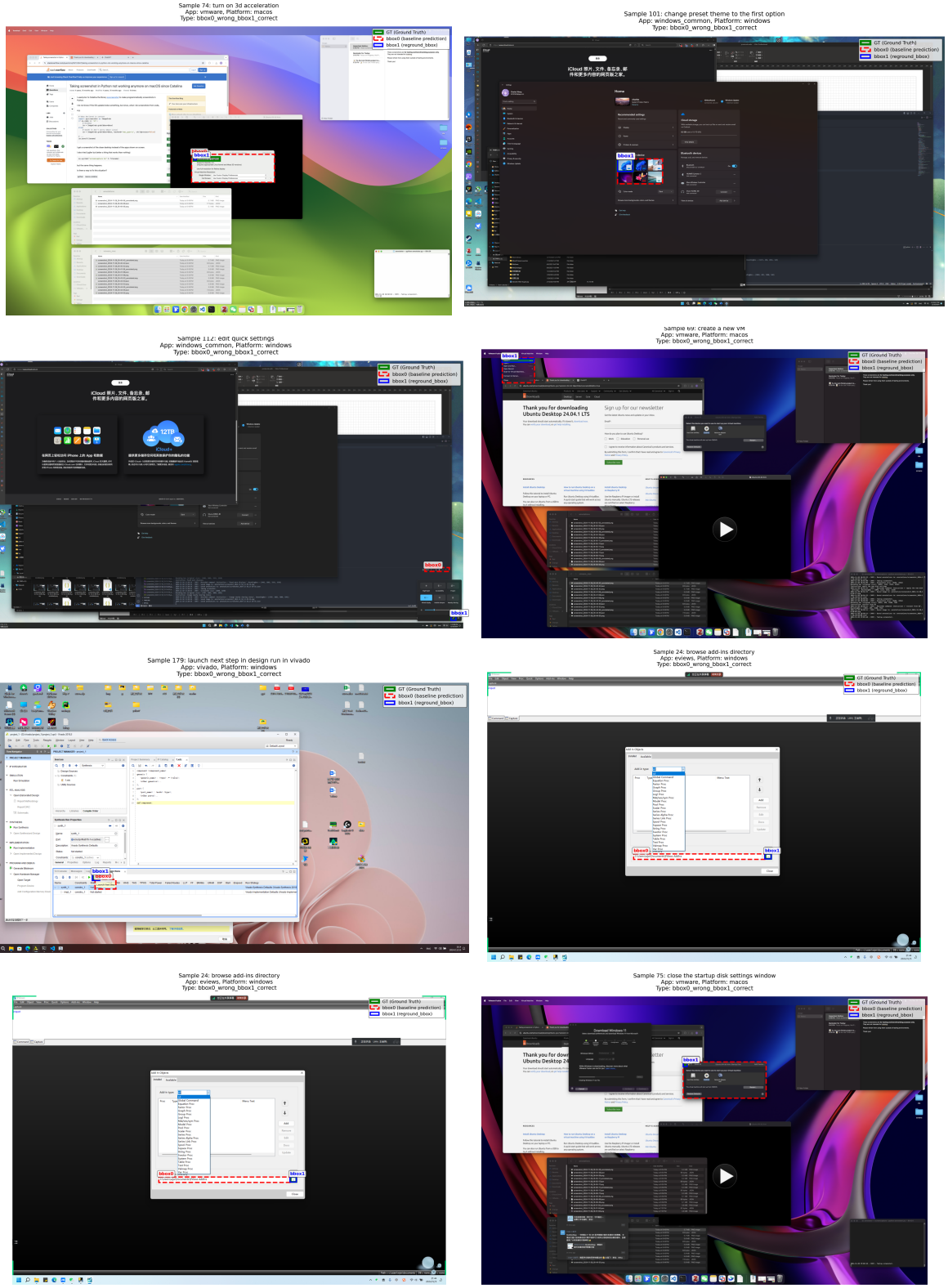


Figure 2. Visualizations of BAMI corrections.

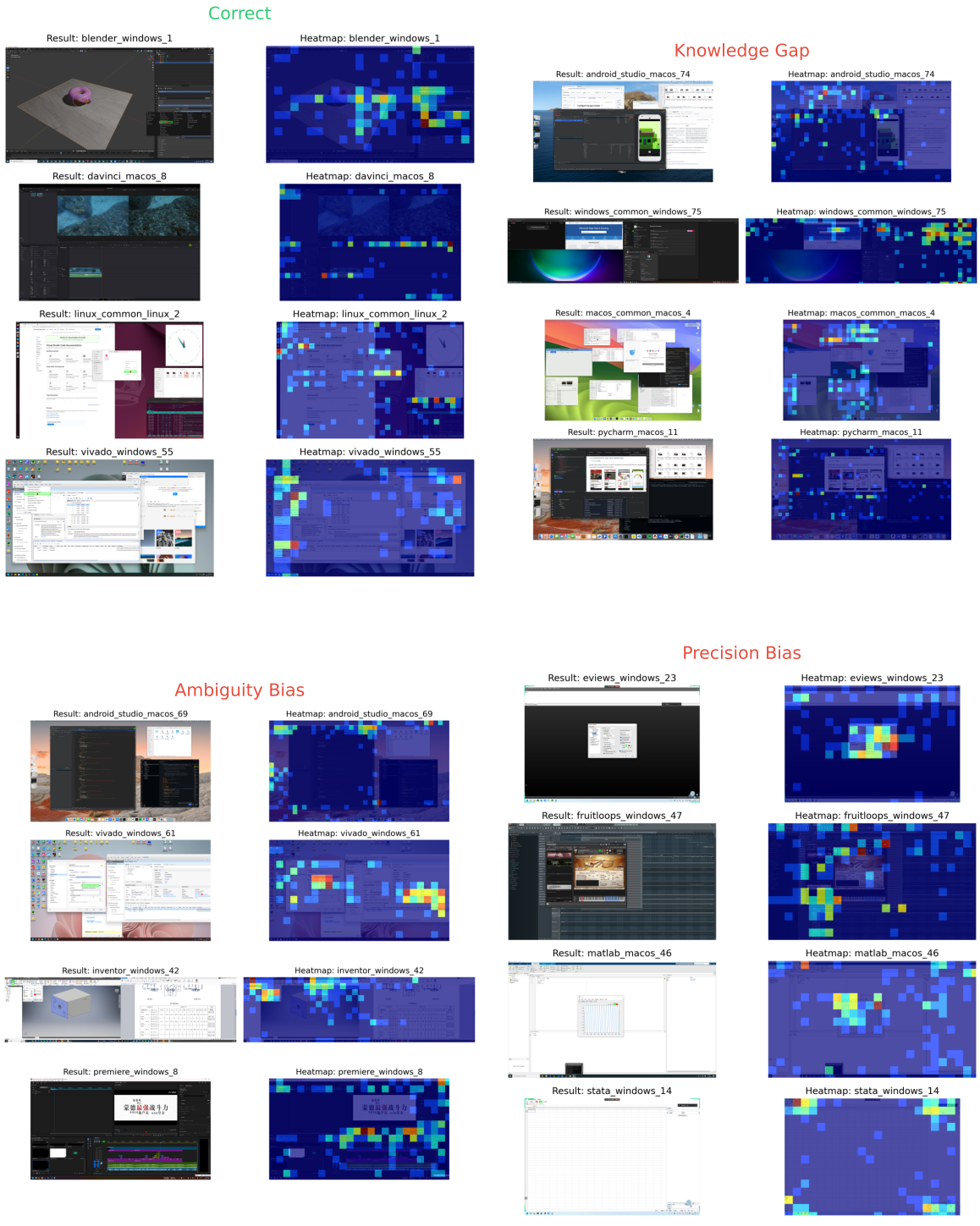


Figure 3. More Attributions Visualizations.