COMPUTER-USING WORLD MODEL

Anonymous authors
Paper under double-blind review

ABSTRACT

Agents operating in complex software environments benefit from reasoning about the consequences of their actions, as even a single incorrect user interface (UI) operation can derail long, artifact-preserving workflows. This challenge is particularly acute for computer-using scenarios, where real execution does not support counterfactual exploration, making large-scale trial-and-error learning and planning impractical despite the environment being fully digital and deterministic. We introduce the Computer-Using World Model (CUWM), a world model for desktop software that predicts the next UI state given the current state and a candidate action. CUWM adopts a two-stage factorization of UI dynamics: it first predicts a textual description of agent-relevant state changes, and then realizes these changes visually to synthesize the next screenshot. CUWM is trained on offline UI transitions collected from agents interacting with real Microsoft Office applications, and further refined with a lightweight reinforcement learning stage that aligns textual transition predictions with the structural requirements of computer-using environments. We evaluate CUWM via test-time action search, where a frozen agent uses the world model to simulate and compare candidate actions before execution. Across a range of Office tasks, world-model-guided test-time scaling improves decision quality and execution robustness.

1 INTRODUCTION

The performance of Large language models (LLMs) has improved consistently through scaling; natural language has been effectively modeled by training large models on massive static corpora (Brown et al., 2020; Hoffmann et al., 2022; OpenAI et al., 2024). In contrast, agents operate in non-stationary environments: their actions shape future observations and thus the data they learn from changes as learning progresses (Sutton et al., 1999; 1998; Yao et al., 2022; 2023). Since an agent's actions change the world's state, reliable decision-making requires counterfactual reasoning: anticipating the consequences of alternative actions before choosing one.

This requirement is particularly acute for *computer-using* agents in desktop applications (Zhang et al., 2025b;a;c; Bonatti et al., 2024). Although software is fully digital and largely deterministic, interaction is neither cheap nor safely reversible: UI actions incur substantial latency (Endo et al., 1996), undo is limited and context-dependent (Prakash & Knister, 1994), and a single mistake can corrupt artifacts or derail long workflows. Deter-

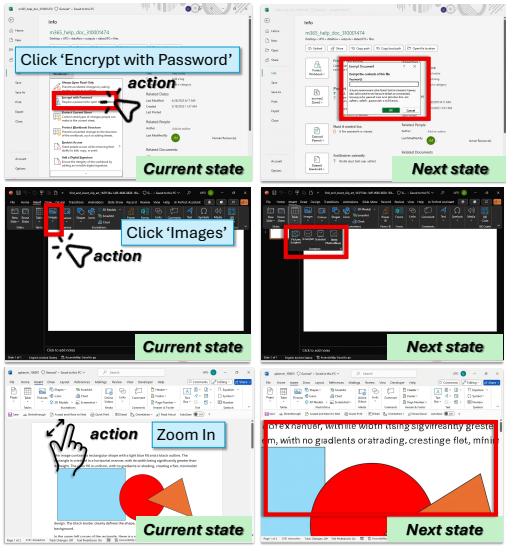

Figure 1: UI state transitions generated by CUWM. Each row is one example transition.

minism therefore does not imply cheap rollouts; without simulation, desktop agents cannot effectively and safely perform counterfactual explorations, making trial-and-error learning and real-execution tree search impractical (Xie et al., 2024; Zhou et al., 2023; Chen et al., 2025).

While model-based reinforcement learning has demonstrated the value of learned dynamics in robotics and games (Ha & Schmidhuber, 2018; Levine, 2022; Schrittwieser et al., 2020; Hafner et al., 2019), world models remain underexplored for *GUI-based desktop software* in the era of LLMs (Zhang et al., 2024). Recent LLM-based world modeling efforts have primarily focused on implicit latent dynamics, textual or semantic state transitions for web and mobile agents, or visual observation prediction in mobile UI settings (Hafner et al., 2019; 2023; Chae et al.; Li et al., 2025a;b; Luo et al., 2025; Cao et al., 2026; Xiang et al., 2025), rather than interactive desktop GUIs. Desktop software for computer use poses unique challenges, combining high-dimensional visual observations, rich compositional GUI actions, and long-horizon, artifact-preserving workflows where early mistakes persist and compound Wang et al. (2024).

In this paper, we take a first step toward world modeling for computer use by introducing the *Computer-Using World Model* (CUWM) for real-world desktop software. We instantiate CUWM in the Microsoft Office suite, including *Word*, *Excel*, and *PowerPoint*, which are widely used productivity applications. World modeling in this domain is especially important because actions such as editing, formatting, or deleting content can have irreversible consequences; a faithful world model enables agents to simulate action outcomes for safer planning, faster evaluation, and more reliable automation without interacting with live user data.

CUWM predicts the next UI state from the current state and a candidate action by factorizing UI dynamics into two stages. A textual transition model first predicts the action-induced, decision-relevant UI changes, and a visual realization model then renders these changes as the next screenshot. This separation of *what changes* from *how it appears* focuses model capacity on structurally salient transitions while retaining pixel-level state generation required by desktop agents. Figure 1 shows example UI state transitions predicted by CUWM, where candidate actions induce localized but consequential interface changes. CUWM is trained on offline UI transitions collected from agents interacting with real Microsoft Office applications. Supervised fine-tuning provides a faithful initialization of UI dynamics, which is further refined with a lightweight reinforcement learning stage that uses an LLM-based judge and a length penalty to encourage concise textual transitions that are aligned with the structural organization of software UIs.

We evaluate CUWM using *test-time action search* with a frozen LLM agent, where candidate actions are simulated by CUWM and a single action is executed. This enables improved decision quality through additional test-time computation without further training or risky exploration, and we evaluate both agent performance and the fidelity of predicted textual transitions and next-state screenshots. In summary, this work makes the following:

- To our best knowledge, **we present the first Computer-Using World Model (CUWM) that explicitly models UI state transitions**, enabling test-time planning for productivity use-cases by learning to reconstruct Microsoft Office (Word, Excel, and PowerPoint) applications.
- **We propose a two-stage world model learning framework** that factorizes UI state transitions into a *textual abstraction* of action-induced changes followed by a *visual state realization*. CUWM is initialized via supervised learning on offline UI transitions and further refined with a reinforcement learning stage that aligns textual transitions with the structural organization of software UIs, promoting concise descriptions.
- Taken together, our world model evaluations show that test-time simulation of UI consequences can substantially improve reliable decision making in software systems.

2 Related Work

Implicit World Models. Model-based reinforcement learning has long studied *implicit* world models (Ha & Schmidhuber, 2018; Levine, 2022; Schrittwieser et al., 2020; Hafner et al., 2019) that encode environment

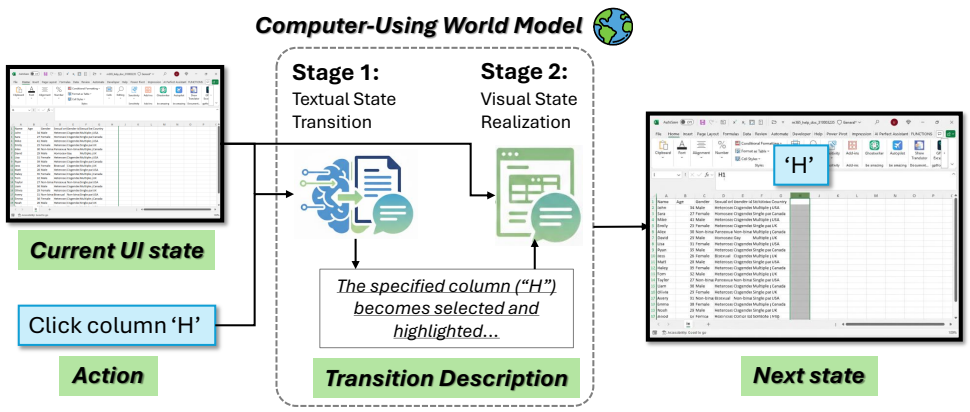

Figure 2: **Overview of the CUWM.** The world model state transitions proceed in two stages, in the first stage, given the current UI state and an action, the world model predicts a textual state-transition description of the next state. In the second stage, the world model conditions on the current UI state and the transition description to render the next UI state.

dynamics in latent representations for planning and value prediction, rather than explicit state reconstruction. Representative methods include World Models (Ha & Schmidhuber, 2018), PlaNet (Hafner et al., 2019), Dreamer (Hafner et al., 2023), and MuZero (Schrittwieser et al., 2020). While effective in games and robotics, these latent models are not designed to be interpretable or aligned with explicit UI semantics, limiting their applicability to computer-using agents. **Textual and Semantic World Models.** Recent work explores explicit textual or semantic world models, primarily for web and mobile agents. Web Agents with World Models (Chae et al.) predicts semantic state transitions for test-time action search in web navigation, and MobileWorldBench (Li et al., 2025a) argues for the effectiveness of semantic prediction in mobile environments. Related studies such as "From Word to World" (Li et al., 2025b) examine whether large language models can serve as implicit world models in purely text-based settings. However, these approaches do not model the visual realization of UI changes, which is often critical for desktop software. **Visual GUI World Models.** Another line of work focuses on predicting future GUI observations. ViMo (Luo et al., 2025) and MobileDreamer (Cao et al., 2026) synthesize future app screens using visual world models, while UI-SIM (Xiang et al., 2025) explores image-based UI simulation. In contrast, we present a factorized, multimodal world model for desktop productivity software that, to the best of our knowledge, is the first world model explicitly tailored for *GUI-based computer use* in software environments, where small UI changes can have outsized effects in long, artifact-preserving workflows.

3 METHOD

Our goal is to learn a world model that captures software UI dynamics and supports agent decision-making through imagined trajectories. Given a current UI state and a candidate action, the model predicts the resulting next state using annotated UI transitions collected from real applications. We instantiate this objective in desktop productivity software and propose a two-stage *computer-using world model*, as illustrated in Figure 2. The model factorizes UI dynamics into two stages. Stage 1 (*Textual State Transition*) predicts a structured natural-language description of the localized, decision-relevant change induced by the action. Stage 2 (*Visual State Realization*) conditions on the current UI state and the predicted transition to synthesize the next UI screenshot, preserving unchanged regions while applying the specified edits. This factorization separates *what changes* from *how it appears*, enabling interpretable and controllable modeling of UI dynam-

ics. The world model is trained primarily via supervised learning on offline UI transitions, and further refined with reinforcement learning to improve the decision relevance and conciseness of predicted transitions.

3.1 Two-Stage World Model Architecture

We model computer-using interaction as a sequential decision process and instantiate our study in desktop productivity software. At time step t, the environment is in a UI state s_t represented by a screenshot image, and executing a natural-language action a_t induces a transition to the next UI state s_{t+1}.

Directly predicting the next UI state s_{t+1} from (s_t, a_t) in pixel space is computationally inefficient due to the highly structured and sparse nature of software state changes. **We define this structure by three key characteristics: UI transitions are typically *localized* in space, *compositional* in nature, and *causally aligned* with the triggering action.** In desktop applications, most actions induce *localized* updates, such as changing a selection, spawning a dialog box, or moving a text cursor, while the majority of the interface remains unchanged. While this sparsity is rooted in a strong underlying structure, it complicates monolithic pixel-level prediction by forcing models to simultaneously track massive invariant backgrounds and tiny, decision-critical updates. Consequently, end-to-end pixel prediction often wastes modeling capacity on these static regions and fails to emphasize the action-relevant components of the transition.

To exploit this structure, we adopt a two-stage decomposition of software UI dynamics. Specifically, we separate *what changes* from *how it appears* by first predicting a textual abstraction of the action-induced state transition, followed by a visual realization of the next UI state. This factorization allows the model to explicitly represent semantically meaningful changes, such as which UI element is affected and in what manner, while delegating image synthesis to a specialized visual model. By aligning the model architecture with the compositional and localized nature of UI interactions, the world model can focus its capacity on decision-relevant dynamics rather than static visual details.

Stage 1: Textual State Transition Model. This stage predicts a textual description of the UI transition. We employ Qwen2.5-VL (Bai et al., 2025) as a vision-language model that takes the current UI state s_t (including the screenshot and associated textual context) and the action a_t as input, and outputs a textual transition description Δ_t: $\Delta_t = f_{\text{text}}(s_t, a_t)$. Rather than describing the entire UI, Δ_t focuses on decision-relevant changes, such as selection shifts, content edits, dialog appearances, or mode transitions. This abstraction significantly reduces the prediction space and provides an interpretable representation of software dynamics that is well aligned with agent decision-making.

Stage 2: Visual State Realization Model. The second stage translates the abstract transition description into a concrete visual outcome. We use Qwen-Image-Edit[1] (Wu et al., 2025), a diffusion-based conditional image editing model, to synthesize the next-state screenshot conditioned on the current UI and the predicted transition: $\hat{s}_{t+1} = f_{\text{image}}(s_t, \Delta_t)$. By conditioning on both the current screenshot and the textual transition, the visual model is responsible only for rendering localized changes, while preserving unchanged regions.

This two-stage design cleanly separates semantic state transitions from visual realization. By allocating model capacity to the most informative components of the prediction problem other than pixel-level synthesis, the architecture improves interpretability, modularity, and scalability, and forms the basis for effective training and downstream agent usage.

3.2 Supervised Training with GPT-Annotated Transitions

We first initialize the *Computer-Using World Model* (CUWM) using supervised learning. Supervised training provides a natural starting point for learning faithful software dynamics, as it allows the model to directly

[1] We use the Qwen-Image-Edit-2509 checkpoint throughout this work.

observe how UI states change in response to actions. However, manually annotating UI transitions at scale is prohibitively expensive, motivating the use of automated supervision.

To obtain training data, we use the GUI-360 (Mu et al., 2025) dataset, which consists of UI interaction trajectories generated by multiple computer-using agents interacting with Office applications. Each trajectory is represented as a sequence of screenshot-based UI states and text-described GUI/API actions, yielding transition tuples (s_t, a_t, s_{t+1}), as illustrated in Figure 1. Based on these transitions, we use GPT-5 as an automated annotator to generate concise natural-language description of UI state changes by conditioning on the triplet (s_t, a_t, s_{t+1}), explicitly identifying which elements change and which remain unchanged.

This process yields *ground-truth* transition descriptions of the form $(s_t, a_t) \rightarrow \Delta_t^{\mathrm{GT}}$, where Δ_t^{GT} summarizes the semantic differences between the consecutive UI states s_t and s_{t+1}. We apply supervised fine-tuning to both stages of CUWM. In Stage 1, the textual transition model (Qwen-VL 2.5) is trained to predict the GPT-annotated transition description Δ_t^{GT} from the current screenshot and action (s_t, a_t), producing a predicted transition Δ_t. In Stage 2, the visual realization model (Qwen-Image-Edit) is trained via diffusion-based image editing to reconstruct the next UI state s_{t+1} conditioned on the current screenshot and the predicted transition (s_t, Δ_t). This supervised training grounds both stages in real software behavior and provides a faithful initialization of UI dynamics and a strong foundation for subsequent refinement.

3.3 Structure-Aware Reinforcement Learning for Textual Transitions

Supervised fine-tuning provides a faithful initialization of textual UI transitions, but it does not ensure that the predicted descriptions consistently capture the UI structures most critical for downstream reasoning. In computer-using environments, effective planning depends on whether key interface components, such as selection state, active controls, and visible panes, are accurately and concisely represented.

We therefore apply a lightweight reinforcement learning refinement to the textual transition model. The model is treated as a policy conditioned on the current screenshot and action (s_t, a_t), generating a transition description Δ_t, and is optimized to maximize $\mathbb{E}_{\Delta_t \sim f_{\text{text}}(\cdot|s_t,a_t)}\left[R(s_t, a_t, \Delta_t)\right]$. The reward combines an LLM-as-a-Judge score and a length penalty: $R(s_t, a_t, \Delta_t) = R_{\text{judge}}(\Delta_t, \Delta_t^{\mathrm{GT}}) - \beta R_{\text{len}}(\Delta_t)$. The judge assigns a normalized score in $[0, 1]$ by evaluating correctness across predefined UI structural aspects (e.g., ribbon state, editing area, and side panes). The length penalty softly penalizes predictions that deviate from a target length range defined relative to the ground-truth transition, discouraging overly long or short descriptions that tend to introduce unsupported or noisy UI changes. We optimize the textual transition model using a relative preference objective based on Group Relative Policy Optimization (GRPO) (Shao et al., 2024). Further details regarding the GRPO implementation and reward formulation are in Appendix A.2.4.

3.4 World-Model-Guided Test-Time Action Search

We evaluate CUWM using world-model-guided test-time action search with a frozen agent policy, following (Luo et al., 2025). At inference time, the agent proposes a set of candidate actions from the current UI state s_t. CUWM simulates the resulting next UI state for each candidate, and the agent selects a single action based on the predicted outcomes. The agent policy remains unchanged throughout this process, and the world model is used solely as a simulator. This *think-then-act* procedure allows decision quality to improve with additional test-time computation, which is particularly important for long-horizon computer-using tasks where errors are costly to reverse.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Dataset. We train and evaluate CUWM using data from the GUI-360 dataset (Mu et al., 2025), focusing on real desktop software, including Microsoft Word, Excel, and PowerPoint. Each sample is constructed as a state–action–next-state tuple (s_t, a_t, s_{t+1}). More details about the dataset can be found in Appendix A.1.

Training. We implement a two-stage pipeline: LoRA-based SFT for both textual (Qwen2.5-VL) and visual (Qwen-Image-Edit) models, followed by GRPO refinement for the textual model (details in Appendix A.2).

Metrics. We evaluate CUWM from two perspectives: (i) *world model fidelity*, assessing the quality of predicted transitions independent of agents, and (ii) *agent-level evaluation*, measuring CUWM's impact on agent performance via test-time action search.

4.2 WORLD MODEL FIDELITY

In this section, we evaluate whether CUWM accurately captures UI dynamics independent of any downstream agent, focusing on both textual state transitions and visual state realization.

4.2.1 TEXTUAL STATE TRANSITION EVALUATION

We evaluate the textual state transition model independently, as it provides the explicit representation of UI dynamics used for visual realization and agent reasoning. We compare three variants: **Base**, an untrained Qwen2.5-VL model; **SFT**, trained with supervised fine-tuning; and **SFT+RL**, the full CUWM model with additional RL refinement. We evaluate textual transitions using two complementary metrics.

LLM-as-a-Judge Score. Predicted transitions are compared against GPT-5 generated ground-truth descriptions derived from (s_t, s_{t+1}) using an LLM-as-a-Judge. The judge evaluates consistency across multiple UI aspects (e.g., application state, executed actions, and major UI components), assigning scores of 0, 0.5, or 1 per aspect. We use GPT-5 as the judge and report the average score across aspects (See more details in Appendix A.3.1). As shown in Table 1, scores improve from Base to SFT and further to SFT+RL.

Table 1: Textual state transition model evaluation: LLM-as-a-Judge Score.

Model	Base	SFT	SFT+RL
Judge Score ↑	0.6027	0.6834	**0.6883**

Action Consistency Score. To evaluate whether the generated textual transitions preserve decision-relevant information, we introduce the **Action Consistency Score(ACS)**. Focusing on a *single-step prediction* setting, this metric measures the functional equivalence between the real UI and the generated text. Specifically, we compute the agreement rate between actions selected by a frozen agent policy when conditioned on two distinct inputs: (i) the ground-truth UI screenshot and (ii) the world model's predicted textual transition(see more details in Appendix A.3.2). We evaluate this metric using two representative agent backbones, GPT-4.1-mini and Gemini-2.0-Flash, to assess robustness across different multimodal reasoning models. As shown in Table 2, the SFT+RL variant achieves the highest action consistency across both backbones. Crucially, this improvement in ACS translates into tangible gains in downstream agent performance (as detailed in Table 13), confirming that RL-driven refinement effectively captures decision-critical UI structures necessary for accurate planning.

Table 2: Action consistency across agent backbones.

Model	**Agent**	**Score**
Base	GPT-4.1-mini	0.4990
Base	Gemini-2.0-Flash	0.3860
SFT	GPT-4.1-mini	0.5450
SFT	Gemini-2.0-Flash	0.4368
SFT+RL	GPT-4.1-mini	**0.5642**
SFT+RL	Gemini-2.0-Flash	**0.4732**

4.3 Visual State Realization Evaluation

The visual state realization model is a critical component of CUWM, as it translates predicted textual state transitions into concrete UI screenshots that can be directly perceived by agents. We evaluate this model using two complementary criteria: image-based quality metrics and text perception score, which together assess visual fidelity and the correctness of rendered UI text. We study how different sources of textual state transitions affect visual realization quality and compare our approach against the off-the-shelf Qwen-Image-Edit-2509 model. Specifically, we consider four evaluation settings: (1) **Action-Only + Qwen-Edit**: directly conditions Qwen-Image-Edit-2509 on the UI screenshot and action.

(2) **Base-Text + Qwen-Edit**: conditions Qwen-Image-Edit-2509 on textual state transitions generated by the Base textual state transition model. (3) **SFT-Text + Qwen-Edit**: conditions Qwen-Image-Edit-2509 on textual state transitions generated by the SFT textual state transition model. (4) **SFT-Text + Finetuned-Visual (CUWM)**: conditions a finetuned visual state realization model on SFT textual transitions and corresponds to the full CUWM setting. **Image-Based Metrics** To evaluate the similarity between the generated and ground-truth next-state UI screenshots, we adopt standard image quality metrics, including PSNR (Hore & Ziou, 2010), SSIM (Wang et al., 2004), LPIPS (Zhang et al., 2018), and FID (Heusel et al., 2018) (details in Appendix A.3.3). As shown in Table 3, incorporating textual state transitions significantly improves visual fidelity compared to directly conditioning on the previous screenshot and action. Performance is further enhanced by fine-tuning the textual transition model, while jointly fine-tuning both textual and visual components (CUWM) achieves the best results across all metrics, demonstrating superior accuracy and perceptual faithfulness in next-state UI generation.

Table 3: Visual state realization: image-based metrics.

Method	PSNR	SSIM	LPIPS	FID
Action-Only + Edit	11.09	0.49	0.48	136.14
Base-Text + Edit	12.45	0.53	0.39	32.21
SFT-Text + Edit	12.86	0.54	0.39	34.59
CUWM	**14.91**	**0.67**	**0.21**	**20.48**

Text Perception Score Text perception is critical in CUWM, as UI applications rely heavily on textual content to convey semantics. We therefore evaluate the readability and semantic consistency of rendered UI text across state transitions using an automated vision-based parser (Lu et al., 2024)(details in Appendix A.3.4).

Table 4 reports results across Word, Excel, and PowerPoint. CUWM achieves the best performance across all applications, indicating that jointly fine-tuning the textual state predictor and the visual renderer substantially improves text preservation during UI transitions. As shown in Figure 5, the Text Perception Score generally increases over training epochs.

Table 4: Visual state realization: text perception accuracy.

	Text Perception ↑			
Method	**Word**	**Excel**	**PPT**	**Overall**
Action-Only + Edit	0.314	0.269	0.339	0.307
Base-Text + Edit	0.621	0.614	0.542	0.597
SFT-Text + Edit	0.591	0.655	0.455	0.574
CUWM	**0.742**	**0.707**	**0.689**	**0.716**

4.4 World-Model-Guided Test-Time Action Search

In this section, we evaluate the impact of the proposed CUWM on agent performance. We follow the agent design and test-time planning procedure described in Section 3.4, and use it as our evaluation protocol.

We evaluate CUWM through controlled comparisons across different world model configurations and agent backbones. Specifically, we consider four agent backbones: Qwen3-VL-8B (Yang et al., 2025), GPT-4.1-mini (OpenAI et al., 2024), GPT-4o (OpenAI et al., 2024), and Gemini-2.0-Flash (Team et al., 2025).

For each backbone, we examine variants without a world model (None), with the Textual State Transition Model only, with the Visual State Realization Model only, as well as their combinations (CUWM) under different integration strategies. In addition, we compare CUWM with two representative image-generation-

Table 5: Agent task scores across different visual state realization models for different agents.

Agent	None	Text	Qwen-Edit		CUWM (ours)		GPT-Image	
			Image	Image+Text	Image	Image+Text	Image	Image+Text
Qwen3-VL-8B	0.3895	0.4102	0.4051	0.4120	**0.4189**	0.4137	0.4137	0.4080
GPT-4.1-mini	0.4361	0.4279	0.4196	0.4286	**0.4418**	0.4089	0.4189	0.4127
GPT-4o	0.4558	0.4625	0.4506	0.4668	**0.4720**	0.4624	0.4514	0.4587
Gemini-2.0-Flash	0.3923	0.4008	0.4053	0.3728	**0.4073**	0.3649	0.4004	0.3821

based world model baselines, Qwen-Image-Edit-2509 (Wu et al., 2025) and GPT-Image-1.5 (OpenAI, 2025). Table 5 reports agent task completion rates under these settings. CUWM with image-only input improves performance across all agent backbones, with gains of 4% for GPT-4o and 8% for Qwen3-VL-8B. These results demonstrate the effectiveness of our proposed world model in enhancing GUI agent decision-making. Compared with existing world models, our image-based methods consistently outperform both text-based world models and image-generation baselines across all agents. Further details on the agent evaluation methodology and calculations are provided in Appendix A.4.

Contrary to expectations, combining text and image predictions degraded agent performance across most configurations. We hypothesize two potential explanations: (1) cross-modal conflict, where textual descriptions contradict visually salient elements, forcing agents to decide between inconsistent signals without a learned resolution strategy, and (2) noise accumulation, where independent prediction errors in each modality compound rather than complement each other when provided together. These findings highlight limitations in current VLMs' capacity for integrated, multimodal reasoning.

4.5 Case Study

Case 1: Capturing Structurally Salient UI Changes. Figure 3 shows cases where the predicted UI state closely matches the ground-truth next state and faithfully reflects the underlying changes. CUWM correctly captures action-induced updates such as text entry, tab switches (e.g., *Pictures*), and opening the *File* view. By accurately modeling these structural transitions and rendering realistic next screens, CUWM enables the agent to anticipate the updated interaction context. **Case 2: World-Model-Guided Test-Time Action Search.** Figure 4 illustrates how CUWM supports action selection by *simulating* the outcomes of multiple candidates before execution. Given the task "Add password protection to the Excel workbook", the agent proposes several candidate actions (e.g., clicking "coordinate", "Title", or "Protect Workbook"). Then CUWM correctly simulates the subsequent states for each candidate respectively, providing accurate visual evidence of their distinct outcomes. Guided by these predictions, the agent identifies "Protect Workbook" as the optimal action consistent with the goal, effectively precluding live trial-and-error.

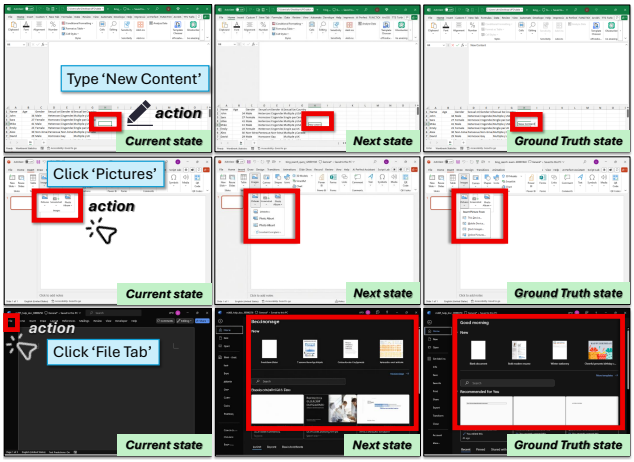

Figure 3: Qualitative comparison of CUWM predictions and ground truth under representative UI actions, showing close alignment in both layout and panel states.

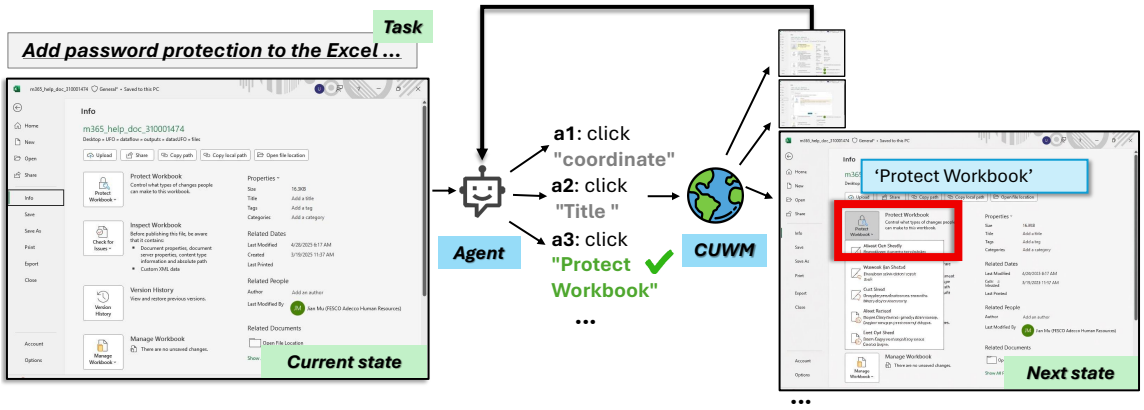

Figure 4: **World-model-guided action selection.** Given the current Excel UI state and candidate actions, CUWM correctly simulates the respective next states for each action, guiding the agent to select "Protect Workbook" based on goal alignment.

4.6 Insights: How World Models help GUI agents

A dominant failure mode of VLM-based agents is their inability to anticipate the consequences of actions (Shi et al., 2026), which often leads to ineffective or repetitive interactions. CUWM directly addresses this by enabling agents to reason over predicted post-action UI states before execution. We find that world models are especially valuable for structural UI transitions, opening modals, expanding dropdowns, or activating side panes, where changes are visually localized but fundamentally alter the interaction context and valid subsequent actions. By explicitly predicting such transitions, CUWM provides actionable signals about interface changes that VLMs frequently fail to infer from the current state alone. A notable insight is that agent performance correlates more strongly with access to high-level structural information (e.g., "a dropdown menu appeared") than with pixel-level fidelity (e.g., "a precise icon rendered in the dropdown"). This explains why CUWM-based agents outperform those using GPT-Image-1.5 (Table 5) despite lower visual fidelity metrics (Table 3). Finally, world-model-based simulation mitigates a common planning failure: action loops where agents repeatedly select actions that return the interface to its current state. By previewing outcomes, agents can distinguish actions that advance toward the goal from those resulting in stagnation.

5 Conclusion

We introduced the Computer-Using World Model (CUWM), a two-stage world model for desktop productivity software that factorizes UI dynamics into textual transition prediction and visual state realization. This design captures structurally salient UI changes while remaining compatible with pixel-level agent interaction, and can be trained from offline UI transitions with lightweight refinement. Across a range of Microsoft Office tasks, CUWM serves as an effective test-time simulator for computer-using agents, enabling world-model-guided action search that improves decision quality and execution robustness without modifying agent policies. Importantly, these gains hold even in deterministic software environments, highlighting the value of test-time simulation for reliable computer use. Several promising directions could be explored in future work. One potential avenue is to incorporate reinforcement learning on top of DiT fine-tuning to further align the world model with downstream decision-making objectives. Another direction is to design reward functions that more directly reflect the usefulness of the world model for agent performance. Finally, improving the joint training of textual and visual components may help better preserve decision-relevant information during state transitions.

REFERENCES

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025. URL https://arxiv.org/abs/2502.13923.

Rogerio Bonatti, Dan Zhao, Francesco Bonacci, Dillon Dupont, Sara Abdali, Yinheng Li, Yadong Lu, Justin Wagle, Kazuhito Koishida, Arthur Bucker, et al. Windows agent arena: Evaluating multi-modal os agents at scale. *arXiv preprint arXiv:2409.08264*, 2024.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Yilin Cao, Yufeng Zhong, Zhixiong Zeng, Liming Zheng, Jing Huang, Haibo Qiu, Peng Shi, Wenji Mao, and Wan Guanglu. Mobiledreamer: Generative sketch world model for gui agent. *arXiv preprint arXiv:2601.04035*, 2026.

Hyungjoo Chae, Namyoung Kim, Minju Gwak, Gwanwoo Song, Jihoon Kim, Kai Tzu-iunn Ong, Sunghwan Kim, Dongha Lee, and Jinyoung Yeo. World models for web agents. In *The First Workshop on System-2 Reasoning at Scale, NeurIPS'24*.

Zhaorun Chen, Zhuokai Zhao, Kai Zhang, Bo Liu, Qi Qi, Yifan Wu, Tarun Kalluri, Sara Cao, Yuanhao Xiong, Haibo Tong, et al. Scaling agent learning via experience synthesis. *arXiv preprint arXiv:2511.03773*, 2025.

Yasuhiro Endo, Zheng Wang, J Bradley Chen, and Margo I Seltzer. Using latency to evaluate interactive system performance. *ACM SIGOPS Operating Systems Review*, 30(si):185–199, 1996.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2(3), 2018.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pp. 2555–2565. PMLR, 2019.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium, 2018. URL https://arxiv.org/abs/1706.08500.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Alain Hore and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th international conference on pattern recognition*, pp. 2366–2369. IEEE, 2010.

Sergey Levine. Understanding the world through action. In *Conference on Robot Learning*, pp. 1752–1757. PMLR, 2022.

Shufan Li, Konstantinos Kallidromitis, Akash Gokul, Yusuke Kato, Kazuki Kozuka, and Aditya Grover. Mobileworldbench: Towards semantic world modeling for mobile agents. *arXiv preprint arXiv:2512.14014*, 2025a.

Yixia Li, Hongru Wang, Jiahao Qiu, Zhenfei Yin, Dongdong Zhang, Cheng Qian, Zeping Li, Pony Ma, Guanhua Chen, Heng Ji, et al. From word to world: Can large language models be implicit text-based world models? *arXiv preprint arXiv:2512.18832*, 2025b.

Yadong Lu, Jianwei Yang, Yelong Shen, and Ahmed Awadallah. Omniparser for pure vision based gui agent, 2024. URL https://arxiv.org/abs/2408.00203.

Dezhao Luo, Bohan Tang, Kang Li, Georgios Papoudakis, Jifei Song, Shaogang Gong, Jianye Hao, Jun Wang, and Kun Shao. Vimo: A generative visual gui world model for app agents. *arXiv preprint arXiv:2504.13936*, 2025.

Jian Mu, Chaoyun Zhang, Chiming Ni, Lu Wang, Bo Qiao, Kartik Mathur, Qianhui Wu, Yuhang Xie, Xiaojun Ma, Mengyu Zhou, et al. Gui-360: A comprehensive dataset and benchmark for computer-using agents. *arXiv preprint arXiv:2511.04307*, 2025.

OpenAI. Gpt image 1.5 model. Online; accessed 2026-02-03, 2025. URL https://platform.openai.com/docs/models/gpt-image-1.5. State-of-the-art image generation model from OpenAI API.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael,

Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

Atul Prakash and Michael J Knister. A framework for undoing actions in collaborative systems. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 1(4):295–330, 1994.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Chenrui Shi, Zedong YU, Zhi Gao, Ruining Feng, Enqi Liu, Yuwei Wu, Yunde Jia, Liuyu Xiang, Zhaofeng He, and Qing Li. GUI knowledge bench: Revealing the knowledge gap behind VLM failures in GUI tasks, 2026. URL https://openreview.net/forum?id=S3U0m4Z3ZT.

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy Lillicrap, Angeliki Lazaridou, Orhan Firat, James Molloy, Michael Isard, Paul R. Barham, Tom Hennigan, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, Ryan Doherty, Eli Collins, Clemens Meyer, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, Jack Krawczyk, Cosmo Du, Ed Chi, Heng-Tze Cheng, Eric Ni, Purvi Shah, Patrick Kane, Betty Chan, Manaal Faruqui, Aliaksei Severyn, Hanzhao Lin, YaGuang Li, Yong Cheng, Abe Ittycheriah, Mahdis Mahdieh, Mia Chen, Pei Sun, Dustin Tran, Sumit Bagri, Balaji Lakshminarayanan, Jeremiah Liu, Andras Orban, Fabian Güra, Hao Zhou, Xinying Song, Aurelien Boffy, Harish Ganapathy, Steven Zheng, HyunJeong Choe, Ágoston Weisz, Tao Zhu, Yifeng Lu, Siddharth Gopal, Jarrod Kahn, Maciej Kula, Jeff Pitman, Rushin Shah, Emanuel Taropa, Majd Al Merey, Martin Baeuml, Zhifeng Chen, Laurent El Shafey, Yujing Zhang, Olcan Sercinoglu, George Tucker, Enrique Piqueras, Maxim Krikun, Iain Barr, Nikolay Savinov, Ivo Danihelka, Becca Roelofs, Anaïs White, Anders Andreassen, Tamara von Glehn, Lakshman Yagati, Mehran Kazemi, Lucas Gonzalez, Misha

Khalman, Jakub Sygnowski, Alexandre Frechette, Charlotte Smith, Laura Culp, Lev Proleev, Yi Luan, Xi Chen, James Lottes, Nathan Schucher, Federico Lebron, Alban Rrustemi, Natalie Clay, Phil Crone, Tomas Kocisky, Jeffrey Zhao, Bartek Perz, Dian Yu, Heidi Howard, Adam Bloniarz, Jack W. Rae, Han Lu, Laurent Sifre, Marcello Maggioni, Fred Alcober, Dan Garrette, Megan Barnes, Shantanu Thakoor, Jacob Austin, Gabriel Barth-Maron, William Wong, Rishabh Joshi, Rahma Chaabouni, Deeni Fatiha, Arun Ahuja, Gaurav Singh Tomar, Evan Senter, Martin Chadwick, Ilya Kornakov, Nithya Attaluri, Iñaki Iturrate, Ruibo Liu, Yunxuan Li, Sarah Cogan, Jeremy Chen, Chao Jia, Chenjie Gu, Qiao Zhang, Jordan Grimstad, Ale Jakse Hartman, Xavier Garcia, Thanumalayan Sankaranarayana Pillai, Jacob Devlin, Michael Laskin, Diego de Las Casas, Dasha Valter, Connie Tao, Lorenzo Blanco, Adrià Puigdomènech Badia, David Reitter, Mianna Chen, Jenny Brennan, Clara Rivera, Sergey Brin, Shariq Iqbal, Gabriela Surita, Jane Labanowski, Abhi Rao, Stephanie Winkler, Emilio Parisotto, Yiming Gu, Kate Olszewska, Ravi Addanki, Antoine Miech, Annie Louis, Denis Teplyashin, Geoff Brown, Elliot Catt, Jan Balaguer, Jackie Xiang, Pidong Wang, Zoe Ashwood, Anton Briukhov, Albert Webson, Sanjay Ganapathy, Smit Sanghavi, Ajay Kannan, Ming-Wei Chang, Axel Stjerngren, Josip Djolonga, Yuting Sun, Ankur Bapna, Matthew Aitchison, Pedram Pejman, Henryk Michalewski, Tianhe Yu, Cindy Wang, Juliette Love, Junwhan Ahn, Dawn Bloxwich, Kehang Han, Peter Humphreys, Thibault Sellam, James Bradbury, Varun Godbole, Sina Samangooei, Bogdan Damoc, Alex Kaskasoli, Sébastien M. R. Arnold, Vijay Vasudevan, Shubham Agrawal, Jason Riesa, Dmitry Lepikhin, Richard Tanburn, Srivatsan Srinivasan, Hyeontaek Lim, Sarah Hodkinson, Pranav Shyam, Johan Ferret, Steven Hand, Ankush Garg, Tom Le Paine, Jian Li, Yujia Li, Minh Giang, Alexander Neitz, Zaheer Abbas, Sarah York, Machel Reid, Elizabeth Cole, Aakanksha Chowdhery, Dipanjan Das, Dominika Rogozińska, Vitaliy Nikolaev, Pablo Sprechmann, Zachary Nado, Lukas Zilka, Flavien Prost, Luheng He, Marianne Monteiro, Gaurav Mishra, Chris Welty, Josh Newlan, Dawei Jia, Miltiadis Allamanis, Clara Huiyi Hu, Raoul de Liedekerke, Justin Gilmer, Carl Saroufim, Shruti Rijhwani, Shaobo Hou, Disha Shrivastava, Anirudh Baddepudi, Alex Goldin, Adnan Ozturel, Albin Cassirer, Yunhan Xu, Daniel Sohn, Devendra Sachan, Reinald Kim Amplayo, Craig Swanson, Dessie Petrova, Shashi Narayan, Arthur Guez, Siddhartha Brahma, Jessica Landon, Miteyan Patel, Ruizhe Zhao, Kevin Villela, Luyu Wang, Wenhao Jia, Matthew Rahtz, Mai Giménez, Legg Yeung, James Keeling, Petko Georgiev, Diana Mincu, Boxi Wu, Salem Haykal, Rachel Saputro, Kiran Vodrahalli, James Qin, Zeynep Cankara, Abhanshu Sharma, Nick Fernando, Will Hawkins, Behnam Neyshabur, Solomon Kim, Adrian Hutter, Priyanka Agrawal, Alex Castro-Ros, George van den Driessche, Tao Wang, Fan Yang, Shuo yiin Chang, Paul Komarek, Ross McIlroy, Mario Lučić, Guodong Zhang, Wael Farhan, Michael Sharman, Paul Natsev, Paul Michel, Yamini Bansal, Siyuan Qiao, Kris Cao, Siamak Shakeri, Christina Butterfield, Justin Chung, Paul Kishan Rubenstein, Shivani Agrawal, Arthur Mensch, Kedar Soparkar, Karel Lenc, Timothy Chung, Aedan Pope, Loren Maggiore, Jackie Kay, Priya Jhakra, Shibo Wang, Joshua Maynez, Mary Phuong, Taylor Tobin, Andrea Tacchetti, Maja Trebacz, Kevin Robinson, Yash Katariya, Sebastian Riedel, Paige Bailey, Kefan Xiao, Nimesh Ghelani, Lora Aroyo, Ambrose Slone, Neil Houlsby, Xuehan Xiong, Zhen Yang, Elena Gribovskaya, Jonas Adler, Mateo Wirth, Lisa Lee, Music Li, Thais Kagohara, Jay Pavagadhi, Sophie Bridgers, Anna Bortsova, Sanjay Ghemawat, Zafarali Ahmed, Tianqi Liu, Richard Powell, Vijay Bolina, Mariko Iinuma, Polina Zablotskaia, James Besley, Da-Woon Chung, Timothy Dozat, Ramona Comanescu, Xiance Si, Jeremy Greer, Guolong Su, Martin Polacek, Raphaël Lopez Kaufman, Simon Tokumine, Hexiang Hu, Elena Buchatskaya, Yingjie Miao, Mohamed Elhawaty, Aditya Siddhant, Nenad Tomasev, Jinwei Xing, Christina Greer, Helen Miller, Shereen Ashraf, Aurko Roy, Zizhao Zhang, Ada Ma, Angelos Filos, Milos Besta, Rory Blevins, Ted Klimenko, Chih-Kuan Yeh, Soravit Changpinyo, Jiaqi Mu, Oscar Chang, Mantas Pajarskas, Carrie Muir, Vered Cohen, Charline Le Lan, Krishna Haridasan, Amit Marathe, Steven Hansen, Sholto Douglas, Rajkumar Samuel, Mingqiu Wang, Sophia Austin, Chang Lan, Jiepu Jiang, Justin Chiu, Jaime Alonso Lorenzo, Lars Lowe Sjösund, Sébastien Cevey, Zach Gleicher, Thi Avrahami, Anudhyan Boral, Hansa Srinivasan, Vittorio Selo, Rhys May, Konstantinos Aisopos, Léonard Hussenot, Livio Baldini Soares, Kate Baumli, Michael B. Chang, Adrià Recasens, Ben Caine, Alexander Pritzel, Filip Pavetic, Fabio Pardo, Anita Gergely, Justin Frye, Vinay Ramasesh, Dan Horgan, Kartikeya Badola, Nora Kassner, Subhrajit Roy, Ethan Dyer, Víctor Cam-

pos Campos, Alex Tomala, Yunhao Tang, Dalia El Badawy, Elspeth White, Basil Mustafa, Oran Lang, Abhishek Jindal, Sharad Vikram, Zhitao Gong, Sergi Caelles, Ross Hemsley, Gregory Thornton, Fangxiaoyu Feng, Wojciech Stokowiec, Ce Zheng, Phoebe Thacker, Çağlar Ünlü, Zhishuai Zhang, Mohammad Saleh, James Svensson, Max Bileschi, Piyush Patil, Ankesh Anand, Roman Ring, Katerina Tsihlas, Arpi Vezer, Marco Selvi, Toby Shevlane, Mikel Rodriguez, Tom Kwiatkowski, Samira Daruki, Keran Rong, Allan Dafoe, Nicholas FitzGerald, Keren Gu-Lemberg, Mina Khan, Lisa Anne Hendricks, Marie Pellat, Vladimir Feinberg, James Cobon-Kerr, Tara Sainath, Maribeth Rauh, Sayed Hadi Hashemi, Richard Ives, Yana Hasson, Eric Noland, Yuan Cao, Nathan Byrd, Le Hou, Qingze Wang, Thibault Sottiaux, Michela Paganini, Jean-Baptiste Lespiau, Alexandre Moufarek, Samer Hassan, Kaushik Shivakumar, Joost van Amersfoort, Amol Mandhane, Pratik Joshi, Anirudh Goyal, Matthew Tung, Andrew Brock, Hannah Sheahan, Vedant Misra, Cheng Li, Nemanja Rakićević, Mostafa Dehghani, Fangyu Liu, Sid Mittal, Junhyuk Oh, Seb Noury, Eren Sezener, Fantine Huot, Matthew Lamm, Nicola De Cao, Charlie Chen, Sidharth Mudgal, Romina Stella, Kevin Brooks, Gautam Vasudevan, Chenxi Liu, Mainak Chain, Nivedita Melinkeri, Aaron Cohen, Venus Wang, Kristie Seymore, Sergey Zubkov, Rahul Goel, Summer Yue, Sai Krishnakumaran, Brian Albert, Nate Hurley, Motoki Sano, Anhad Mohananey, Jonah Joughin, Egor Filonov, Tomasz Kepa, Yomna Eldawy, Jiawern Lim, Rahul Rishi, Shirin Badiezadegan, Taylor Bos, Jerry Chang, Sanil Jain, Sri Gayatri Sundara Padmanabhan, Subha Puttagunta, Kalpesh Krishna, Leslie Baker, Norbert Kalb, Vamsi Bedapudi, Adam Kurzrok, Shuntong Lei, Anthony Yu, Oren Litvin, Xiang Zhou, Zhichun Wu, Sam Sobell, Andrea Siciliano, Alan Papir, Robby Neale, Jonas Bragagnolo, Tej Toor, Tina Chen, Valentin Anklin, Feiran Wang, Richie Feng, Milad Gholami, Kevin Ling, Lijuan Liu, Jules Walter, Hamid Moghaddam, Arun Kishore, Jakub Adamek, Tyler Mercado, Jonathan Mallinson, Siddhinita Wandekar, Stephen Cagle, Eran Ofek, Guillermo Garrido, Clemens Lombriser, Maksim Mukha, Botu Sun, Hafeezul Rahman Mohammad, Josip Matak, Yadi Qian, Vikas Peswani, Pawel Janus, Quan Yuan, Leif Schelin, Oana David, Ankur Garg, Yifan He, Oleksii Duzhyi, Anton Älgmyr, Timothée Lottaz, Qi Li, Vikas Yadav, Luyao Xu, Alex Chinien, Rakesh Shivanna, Aleksandr Chuklin, Josie Li, Carrie Spadine, Travis Wolfe, Kareem Mohamed, Subhabrata Das, Zihang Dai, Kyle He, Daniel von Dincklage, Shyam Upadhyay, Akanksha Maurya, Luyan Chi, Sebastian Krause, Khalid Salama, Pam G Rabinovitch, Pavan Kumar Reddy M, Aarush Selvan, Mikhail Dektiarev, Golnaz Ghiasi, Erdem Guven, Himanshu Gupta, Boyi Liu, Deepak Sharma, Idan Heimlich Shtacher, Shachi Paul, Oscar Akerlund, François-Xavier Aubet, Terry Huang, Chen Zhu, Eric Zhu, Elico Teixeira, Matthew Fritze, Francesco Bertolini, Liana-Eleonora Marinescu, Martin Bölle, Dominik Paulus, Khyatti Gupta, Tejasi Latkar, Max Chang, Jason Sanders, Roopa Wilson, Xuewei Wu, Yi-Xuan Tan, Lam Nguyen Thiet, Tulsee Doshi, Sid Lall, Swaroop Mishra, Wanming Chen, Thang Luong, Seth Benjamin, Jasmine Lee, Ewa Andrejczuk, Dominik Rabiej, Vipul Ranjan, Krzysztof Styrc, Pengcheng Yin, Jon Simon, Malcolm Rose Harriott, Mudit Bansal, Alexei Robsky, Geoff Bacon, David Greene, Daniil Mirylenka, Chen Zhou, Obaid Sarvana, Abhimanyu Goyal, Samuel Andermatt, Patrick Siegler, Ben Horn, Assaf Israel, Francesco Pongetti, Chih-Wei "Louis" Chen, Marco Selvatici, Pedro Silva, Kathie Wang, Jackson Tolins, Kelvin Guu, Roey Yogev, Xiaochen Cai, Alessandro Agostini, Maulik Shah, Hung Nguyen, Noah Ó Donnaile, Sébastien Pereira, Linda Friso, Adam Stambler, Adam Kurzrok, Chenkai Kuang, Yan Romanikhin, Mark Geller, ZJ Yan, Kane Jang, Cheng-Chun Lee, Wojciech Fica, Eric Malmi, Qijun Tan, Dan Banica, Daniel Balle, Ryan Pham, Yanping Huang, Diana Avram, Hongzhi Shi, Jasjot Singh, Chris Hidey, Niharika Ahuja, Pranab Saxena, Dan Dooley, Srividya Pranavi Potharaju, Eileen O'Neill, Anand Gokulchandran, Ryan Foley, Kai Zhao, Mike Dusenberry, Yuan Liu, Pulkit Mehta, Ragha Kotikalapudi, Chalence Safranek-Shrader, Andrew Goodman, Joshua Kessinger, Eran Globen, Prateek Kolhar, Chris Gorgolewski, Ali Ibrahim, Yang Song, Ali Eichenbaum, Thomas Brovelli, Sahitya Potluri, Preethi Lahoti, Cip Baetu, Ali Ghorbani, Charles Chen, Andy Crawford, Shalini Pal, Mukund Sridhar, Petru Gurita, Asier Mujika, Igor Petrovski, Pierre-Louis Cedoz, Chenmei Li, Shiyuan Chen, Niccolò Dal Santo, Siddharth Goyal, Jitesh Punjabi, Karthik Kappaganthu, Chester Kwak, Pallavi LV, Sarmishta Velury, Himadri Choudhury, Jamie Hall, Premal Shah, Ricardo Figueira, Matt Thomas, Minjie Lu, Ting Zhou, Chintu Kumar, Thomas Jurdi, Sharat Chikkerur, Yenai Ma, Adams Yu, Soo Kwak, Victor Ähdel, Sujeevan Rajayogam, Travis Choma, Fei Liu, Aditya

Barua, Colin Ji, Ji Ho Park, Vincent Hellendoorn, Alex Bailey, Taylan Bilal, Huanjie Zhou, Mehrdad Khatir, Charles Sutton, Wojciech Rzadkowski, Fiona Macintosh, Roopali Vij, Konstantin Shagin, Paul Medina, Chen Liang, Jinjing Zhou, Pararth Shah, Yingying Bi, Attila Dankovics, Shipra Banga, Sabine Lehmann, Marissa Bredesen, Zifan Lin, John Eric Hoffmann, Jonathan Lai, Raynald Chung, Kai Yang, Nihal Balani, Arthur Bražinskas, Andrei Sozanschi, Matthew Hayes, Héctor Fernández Alcalde, Peter Makarov, Will Chen, Antonio Stella, Liselotte Snijders, Michael Mandl, Ante Kärrman, Paweł Nowak, Xinyi Wu, Alex Dyck, Krishnan Vaidyanathan, Raghavender R, Jessica Mallet, Mitch Rudominer, Eric Johnston, Sushil Mittal, Akhil Udathu, Janara Christensen, Vishal Verma, Zach Irving, Andreas Santucci, Gamaleldin Elsayed, Elnaz Davoodi, Marin Georgiev, Ian Tenney, Nan Hua, Geoffrey Cideron, Edouard Leurent, Mahmoud Alnahlawi, Ionut Georgescu, Nan Wei, Ivy Zheng, Dylan Scandinaro, Heinrich Jiang, Jasper Snoek, Mukund Sundararajan, Xuezhi Wang, Zack Ontiveros, Itay Karo, Jeremy Cole, Vinu Rajashekhar, Lara Tumeh, Eyal Ben-David, Rishub Jain, Jonathan Uesato, Romina Datta, Oskar Bunyan, Shimu Wu, John Zhang, Piotr Stanczyk, Ye Zhang, David Steiner, Subhajit Naskar, Michael Azzam, Matthew Johnson, Adam Paszke, Chung-Cheng Chiu, Jaume Sanchez Elias, Afroz Mohiuddin, Faizan Muhammad, Jin Miao, Andrew Lee, Nino Vieillard, Jane Park, Jiageng Zhang, Jeff Stanway, Drew Garmon, Abhijit Karmarkar, Zhe Dong, Jong Lee, Aviral Kumar, Luowei Zhou, Jonathan Evens, William Isaac, Geoffrey Irving, Edward Loper, Michael Fink, Isha Arkatkar, Nanxin Chen, Izhak Shafran, Ivan Petrychenko, Zhe Chen, Johnson Jia, Anselm Levskaya, Zhenkai Zhu, Peter Grabowski, Yu Mao, Alberto Magni, Kaisheng Yao, Javier Snaider, Norman Casagrande, Evan Palmer, Paul Suganthan, Alfonso Castaño, Irene Giannoumis, Wooyeol Kim, Mikołaj Rybiński, Ashwin Sreevatsa, Jennifer Prendki, David Soergel, Adrian Goedeckemeyer, Willi Gierke, Mohsen Jafari, Meenu Gaba, Jeremy Wiesner, Diana Gage Wright, Yawen Wei, Harsha Vashisht, Yana Kulizhskaya, Jay Hoover, Maigo Le, Lu Li, Chimezie Iwuanyanwu, Lu Liu, Kevin Ramirez, Andrey Khorlin, Albert Cui, Tian LIN, Marcus Wu, Ricardo Aguilar, Keith Pallo, Abhishek Chakladar, Ginger Perng, Elena Allica Abellan, Mingyang Zhang, Ishita Dasgupta, Nate Kushman, Ivo Penchev, Alena Repina, Xihui Wu, Tom van der Weide, Priya Ponnapalli, Caroline Kaplan, Jiri Simsa, Shuangfeng Li, Olivier Dousse, Fan Yang, Jeff Piper, Nathan Ie, Rama Pasumarthi, Nathan Lintz, Anitha Vijayakumar, Daniel Andor, Pedro Valenzuela, Minnie Lui, Cosmin Paduraru, Daiyi Peng, Katherine Lee, Shuyuan Zhang, Somer Greene, Duc Dung Nguyen, Paula Kurylowicz, Cassidy Hardin, Lucas Dixon, Lili Janzer, Kiam Choo, Ziqiang Feng, Biao Zhang, Achintya Singhal, Dayou Du, Dan McKinnon, Natasha Antropova, Tolga Bolukbasi, Orgad Keller, David Reid, Daniel Finchelstein, Maria Abi Raad, Remi Crocker, Peter Hawkins, Robert Dadashi, Colin Gaffney, Ken Franko, Anna Bulanova, Rémi Leblond, Shirley Chung, Harry Askham, Luis C. Cobo, Kelvin Xu, Felix Fischer, Jun Xu, Christina Sorokin, Chris Alberti, Chu-Cheng Lin, Colin Evans, Alek Dimitriev, Hannah Forbes, Dylan Banarse, Zora Tung, Mark Omernick, Colton Bishop, Rachel Sterneck, Rohan Jain, Jiawei Xia, Ehsan Amid, Francesco Piccinno, Xingyu Wang, Praseem Banzal, Daniel J. Mankowitz, Alex Polozov, Victoria Krakovna, Sasha Brown, MohammadHossein Bateni, Dennis Duan, Vlad Firoiu, Meghana Thotakuri, Tom Natan, Matthieu Geist, Ser tan Girgin, Hui Li, Jiayu Ye, Ofir Roval, Reiko Tojo, Michael Kwong, James Lee-Thorp, Christopher Yew, Danila Sinopalnikov, Sabela Ramos, John Mellor, Abhishek Sharma, Kathy Wu, David Miller, Nicolas Sonnerat, Denis Vnukov, Rory Greig, Jennifer Beattie, Emily Caveness, Libin Bai, Julian Eisenschlos, Alex Korchemniy, Tomy Tsai, Mimi Jasarevic, Weize Kong, Phuong Dao, Zeyu Zheng, Frederick Liu, Fan Yang, Rui Zhu, Tian Huey Teh, Jason Sanmiya, Evgeny Gladchenko, Nejc Trdin, Daniel Toyama, Evan Rosen, Sasan Tavakkol, Linting Xue, Chen Elkind, Oliver Woodman, John Carpenter, George Papamakarios, Rupert Kemp, Sushant Kafle, Tanya Grunina, Rishika Sinha, Alice Talbert, Diane Wu, Denese Owusu-Afriyie, Cosmo Du, Chloe Thornton, Jordi Pont-Tuset, Pradyumna Narayana, Jing Li, Saaber Fatehi, John Wieting, Omar Ajmeri, Benigno Uria, Yeongil Ko, Laura Knight, Amélie Héliou, Ning Niu, Shane Gu, Chenxi Pang, Yeqing Li, Nir Levine, Ariel Stolovich, Rebeca Santamaria-Fernandez, Sonam Goenka, Wenny Yustalim, Robin Strudel, Ali Elqursh, Charlie Deck, Hyo Lee, Zonglin Li, Kyle Levin, Raphael Hoffmann, Dan Holtmann-Rice, Olivier Bachem, Sho Arora, Christy Koh, Soheil Hassas Yeganeh, Siim Põder, Mukarram Tariq, Yanhua Sun, Lucian Ionita, Mojtaba Seyedhosseini, Pouya Tafti, Zhiyu Liu, Anmol Gulati, Jasmine Liu,

Xinyu Ye, Bart Chrzaszcz, Lily Wang, Nikhil Sethi, Tianrun Li, Ben Brown, Shreya Singh, Wei Fan, Aaron Parisi, Joe Stanton, Vinod Koverkathu, Christopher A. Choquette-Choo, Yunjie Li, TJ Lu, Abe Ittycheriah, Prakash Shroff, Mani Varadarajan, Sanaz Bahargam, Rob Willoughby, David Gaddy, Guillaume Desjardins, Marco Cornero, Brona Robenek, Bhavishya Mittal, Ben Albrecht, Ashish Shenoy, Fedor Moiseev, Henrik Jacobsson, Alireza Ghaffarkhah, Morgane Rivière, Alanna Walton, Clément Crepy, Alicia Parrish, Zongwei Zhou, Clement Farabet, Carey Radebaugh, Praveen Srinivasan, Claudia van der Salm, Andreas Fidjeland, Salvatore Scellato, Eri Latorre-Chimoto, Hanna Klimczak-Plucińska, David Bridson, Dario de Cesare, Tom Hudson, Piermaria Mendolicchio, Lexi Walker, Alex Morris, Matthew Mauger, Alexey Guseynov, Alison Reid, Seth Odoom, Lucia Loher, Victor Cotruta, Madhavi Yenugula, Dominik Grewe, Anastasia Petrushkina, Tom Duerig, Antonio Sanchez, Steve Yadlowsky, Amy Shen, Amir Globerson, Lynette Webb, Sahil Dua, Dong Li, Surya Bhupatiraju, Dan Hurt, Haroon Qureshi, Ananth Agarwal, Tomer Shani, Matan Eyal, Anuj Khare, Shreyas Rammohan Belle, Lei Wang, Chetan Tekur, Mihir Sanjay Kale, Jinliang Wei, Ruoxin Sang, Brennan Saeta, Tyler Liechty, Yi Sun, Yao Zhao, Stephan Lee, Pandu Nayak, Doug Fritz, Manish Reddy Vuyyuru, John Aslanides, Nidhi Vyas, Martin Wicke, Xiao Ma, Evgenii Eltyshev, Nina Martin, Hardie Cate, James Manyika, Keyvan Amiri, Yelin Kim, Xi Xiong, Kai Kang, Florian Luisier, Nilesh Tripuraneni, David Madras, Mandy Guo, Austin Waters, Oliver Wang, Joshua Ainslie, Jason Baldridge, Han Zhang, Garima Pruthi, Jakob Bauer, Feng Yang, Riham Mansour, Jason Gelman, Yang Xu, George Polovets, Ji Liu, Honglong Cai, Warren Chen, Xiang-Hai Sheng, Emily Xue, Sherjil Ozair, Christof Angermueller, Xiaowei Li, Anoop Sinha, Weiren Wang, Julia Wiesinger, Emmanouil Koukoumidis, Yuan Tian, Anand Iyer, Madhu Gurumurthy, Mark Goldenson, Parashar Shah, MK Blake, Hongkun Yu, Anthony Urbanowicz, Jennimaria Palomaki, Chrisantha Fernando, Ken Durden, Harsh Mehta, Nikola Momchev, Elahe Rahimtoroghi, Maria Georgaki, Amit Raul, Sebastian Ruder, Morgan Redshaw, Jinhyuk Lee, Denny Zhou, Komal Jalan, Dinghua Li, Blake Hechtman, Parker Schuh, Milad Nasr, Kieran Milan, Vladimir Mikulik, Juliana Franco, Tim Green, Nam Nguyen, Joe Kelley, Aroma Mahendru, Andrea Hu, Joshua Howland, Ben Vargas, Jeffrey Hui, Kshitij Bansal, Vikram Rao, Rakesh Ghiya, Emma Wang, Ke Ye, Jean Michel Sarr, Melanie Moranski Preston, Madeleine Elish, Steve Li, Aakash Kaku, Jigar Gupta, Ice Pasupat, Da-Cheng Juan, Milan Someswar, Tejvi M., Xinyun Chen, Aida Amini, Alex Fabrikant, Eric Chu, Xuanyi Dong, Amruta Muthal, Senaka Buthpitiya, Sarthak Jauhari, Nan Hua, Urvashi Khandelwal, Ayal Hitron, Jie Ren, Larissa Rinaldi, Shahar Drath, Avigail Dabush, Nan-Jiang Jiang, Harshal Godhia, Uli Sachs, Anthony Chen, Yicheng Fan, Hagai Taitelbaum, Hila Noga, Zhuyun Dai, James Wang, Chen Liang, Jenny Hamer, Chun-Sung Ferng, Chenel Elkind, Aviel Atias, Paulina Lee, Vít Listík, Mathias Carlen, Jan van de Kerkhof, Marcin Pikus, Krunoslav Zaher, Paul Müller, Sasha Zykova, Richard Stefanec, Vitaly Gatsko, Christoph Hirnschall, Ashwin Sethi, Xingyu Federico Xu, Chetan Ahuja, Beth Tsai, Anca Stefanoiu, Bo Feng, Keshav Dhandhania, Manish Katyal, Akshay Gupta, Atharva Parulekar, Divya Pitta, Jing Zhao, Vivaan Bhatia, Yashodha Bhavnani, Omar Alhadlaq, Xiaolin Li, Peter Danenberg, Dennis Tu, Alex Pine, Vera Filippova, Abhipso Ghosh, Ben Limonchik, Bhargava Urala, Chaitanya Krishna Lanka, Derik Clive, Yi Sun, Edward Li, Hao Wu, Kevin Hongtongsak, Ianna Li, Kalind Thakkar, Kuanysh Omarov, Kushal Majmundar, Michael Alverson, Michael Kucharski, Mohak Patel, Mudit Jain, Maksim Zabelin, Paolo Pelagatti, Rohan Kohli, Saurabh Kumar, Joseph Kim, Swetha Sankar, Vineet Shah, Lakshmi Ramachandruni, Xiangkai Zeng, Ben Bariach, Laura Weidinger, Tu Vu, Alek Andreev, Antoine He, Kevin Hui, Sheleem Kashem, Amar Subramanya, Sissie Hsiao, Demis Hassabis, Koray Kavukcuoglu, Adam Sadovsky, Quoc Le, Trevor Strohman, Yonghui Wu, Slav Petrov, Jeffrey Dean, and Oriol Vinyals. Gemini: A family of highly capable multimodal models, 2025. URL https://arxiv.org/abs/2312.11805.

Lu Wang, Fangkai Yang, Chaoyun Zhang, Junting Lu, Jiaxu Qian, Shilin He, Pu Zhao, Bo Qiao, Ray Huang, Si Qin, et al. Large action models: From inception to implementation. *arXiv preprint arXiv:2412.10047*, 2024.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-image technical report, 2025. URL https://arxiv.org/abs/2508.02324.

Jiannan Xiang, Yun Zhu, Lei Shu, Maria Wang, Lijun Yu, Gabriel Barcik, James Lyon, Srinivas Sunkara, and Jindong Chen. Uisim: An interactive image-based ui simulator for dynamic mobile environments. *arXiv preprint arXiv:2509.21733*, 2025.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh J Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *Advances in Neural Information Processing Systems*, 37:52040–52094, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*, 2022.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Chaoyun Zhang, Shilin He, Jiaxu Qian, Bowen Li, Liqun Li, Si Qin, Yu Kang, Minghua Ma, Guyue Liu, Qingwei Lin, et al. Large language model-brained gui agents: A survey. *arXiv preprint arXiv:2411.18279*, 2024.

Chaoyun Zhang, He Huang, Chiming Ni, Jian Mu, Si Qin, Shilin He, Lu Wang, Fangkai Yang, Pu Zhao, Chao Du, et al. Ufo2: The desktop agentos. *arXiv preprint arXiv:2504.14603*, 2025a.

Chaoyun Zhang, Liqun Li, Shilin He, Xu Zhang, Bo Qiao, Si Qin, Minghua Ma, Yu Kang, Qingwei Lin, Saravan Rajmohan, et al. Ufo: A ui-focused agent for windows os interaction. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 597–622, 2025b.

Chaoyun Zhang, Liqun Li, He Huang, Chiming Ni, Bo Qiao, Si Qin, Yu Kang, Minghua Ma, Qingwei Lin, Saravan Rajmohan, et al. Ufo^3: Weaving the digital agent galaxy. *arXiv preprint arXiv:2511.11332*, 2025c.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric, 2018. URL https://arxiv.org/abs/1801.03924.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*, 2023.

A Appendix

A.1 Dataset Summarization

Our dataset is constructed based on GUI-360 (Mu et al., 2025). From this dataset, we sampled and processed task trajectories across three widely used Office applications: Word, Excel, and PowerPoint. GUI-360 provides frame-by-frame screenshots for each trajectory together with rich additional metadata, including the corresponding action commands at each step and task completion signals.

During data preprocessing, we paired each current UI screenshot with the executed action and the subsequent UI screenshot, forming a tuple (s_t, a_t, s_{t+1}) as one sample. We select successful trajectories from the original dataset and uniformly sample from those with continuous trajectories and complete, standardized action annotations. In total, we collected 2,876 samples for training and 339 samples for evaluation, serving as an initial dataset that can be readily scaled up in future work. To facilitate efficient and stable training, we standardized image resolutions and removed samples where the pre- and post-action images remained unchanged, the action was invalid, or the data contained excessive noise. We organize the dataset into training, validation, and test splits, and report the number of samples for each split in Table 6.

Table 6: Number of samples in the CUWM dataset for each data split and application.

Data Split	**Word**	**Excel**	**PowerPoint**
Training	797	997	1082
Validation	40	31	27
Test	119	96	124
Total	956	1124	1233

A.2 Training Details

A.2.1 Overview of the training pipeline

Our Computer-Using World Model (CUWM) factorizes UI dynamics into two stages: (i) a *textual state transition model* that predicts a concise transition description Δ_t from the current UI screenshot and action (s_t, a_t), and (ii) a *visual state realization model* that synthesizes the next UI screenshot $\hat{s}_{t+1}$ conditioned on the current UI and the predicted transition (s_t, Δ_t). We train these two stages in a staged manner: **(1) supervised initialization (SFT)** for both Stage 1 and Stage 2, subsequently applying **(2) reinforcement learning refinement (GRPO)** to Stage 1 only.

A.2.2 Stage 1: Qwen2.5-VL supervised training (textual transitions)

In this stage, we use Qwen2.5-VL as a vision-language model to map (s_t, a_t) to a textual transition description Δ_t. The model receives the current UI screenshot s_t and a natural-language action a_t as input, and is trained to generate Δ_t^{GT}. We optimize a standard autoregressive cross-entropy loss on the target transition text: $\mathcal{L}_{\mathrm{SFT}} = -\log p(\Delta_t^{\mathrm{GT}} \mid s_t, a_t)$. We fine-tune the model using LoRA, all Stage 1 hyperparameters are summarized in Table 7.

Table 7: Stage 1 (Qwen2.5-VL) SFT hyperparameters.

Category	Setting
Model	Qwen2.5-VL-7B-Instruct
Train type	LoRA
Target modules	all-linear
LoRA rank	32
LoRA alpha	32
Precision	bfloat16
Learning rate	1e-4
Warmup ratio	0.05
Weight decay	0
Batch size	4
Grad accumulation	4

Table 8: Stage 2 (Qwen-Image-Edit) SFT hyperparameters.

Category	Setting
Model	Qwen-Image-Edit-2509
Training type	LoRA
LoRA base	DiT
Target modules	`to_q`, `to_k`, `to_v`, `add_q_proj`, `add_k_proj`, `add_v_proj`, `to_out.0`, `to_add_out`, `img_mlp.net.2`, `img_mod.1`, `txt_mlp.net.2`, `txt_mod.1`
LoRA rank	32
LoRA alpha	32
Precision	16-mixed
Learning rate	1e-4
Warmup ratio	0
Weight decay	0.01
Batch size	1
Grad accumulation	1
Max pixels	1,048,576

Table 9: Stage 1 GRPO hyperparameters.

Category	Setting
Model	SFT-Qwen2.5-VL
Algorithm	GRPO
Train type	LoRA
Target modules	all-linear
Exclude modules	.*visual.*
KL loss	enabled (coef=0.01)
LoRA rank	64
LoRA alpha	32
Precision	bfloat16
Learning rate	3e-6
Warmup ratio	0
Weight decay	0.01
Train batch size	32
Rollout n	5
ppo mini batch size	32
ppo micro batch size per gpu	8
Length Penalty Weight (β)	1
Min Length Ratio (r_{low})	0.75
Max Length Ratio (r_{up})	1.25
Max Penalty Score (m)	1.0

A.2.3 Stage 2: Qwen-Image-Edit supervised training (visual realization)

In this stage, we use Qwen-Image-Edit as a conditional image editing model to generate the next UI screenshot. The model is conditioned on the current UI screenshot and the predicted textual transition: $\hat{s}_{t+1} = f_{\text{image}}(s_t, \Delta_t)$. We fine-tune the model with a pixel-wise reconstruction objective using the mean squared error (MSE) loss: $\mathcal{L}_{\text{EDIT}} = \|\hat{s}_{t+1} - s_{t+1}\|_2^2$, where s_{t+1} denotes the ground-truth next-state screenshot. We apply LoRA fine-tuning only to the DiT backbone of Qwen-Image-Edit, all Stage 2 hyperparameters are summarized in Table 8.

A.2.4 Stage 1 RL Refinement: Qwen2.5-VL + GRPO

Although SFT provides a faithful initialization, it does not guarantee that the generated textual transition Δ_t consistently captures the most decision-critical UI structures. To address this, we further refine the Stage 1 model using Group Relative Policy Optimization (GRPO), treating the VLM as a policy π_θ that generates Δ_t conditioned on (s_t, a_t).

During training, for each input tuple (s_t, a_t), we sample a group of $K = 5$ candidate descriptions $\{\Delta_t^{(k)}\}_{k=1}^{K}$ from the current policy using a temperature of **1.0** and top-p of **1.0**. GRPO optimizes the relative preference among these K samples without requiring a separate value network (critic), which ensures stability for text generation tasks involving structured metrics.

Reward Formulation. The optimization objective is driven by a composite scalar reward function. For a sampled description Δ_t, the total reward is defined as:

$$R(s_t, a_t, \Delta_t) = R_{\text{judge}}(\Delta_t, \Delta_t^{\text{GT}}) - \beta \cdot R_{\text{len}}(\Delta_t, \Delta_t^{\text{GT}}), \tag{1}$$

where:

- R_{judge} is the semantic consistency score assigned by the LLM-as-a-Judge (as detailed in Section A.3.1);
- R_{len} is a soft length penalty designed to discourage verbosity;
- β is a hyperparameter controlling the strength of the length regularization.

Soft Length Penalty (R_{len}). To prevent the model from generating hallucinations through excessive verbosity or missing details due to brevity, we impose a dynamic length penalty. Let L_{pred} denote the token count of the predicted description and L_{gt} denote the token count of the ground truth. We define a valid length interval $[l_{\text{min}}, l_{\text{max}}]$ relative to the ground truth:

$$l_{\text{min}} = \max(1, \lfloor r_{\text{low}} \cdot L_{\text{gt}} \rfloor), \quad l_{\text{max}} = \max(l_{\text{min}} + 1, \lfloor r_{\text{up}} \cdot L_{\text{gt}} \rfloor), \tag{2}$$

where r_{low} and r_{up} are scaling factors defining the acceptable range. The penalty term R_{len} increases linearly with the deviation from this interval, capped by a maximum penalty m:

$$R_{\text{len}} = \begin{cases} 0, & l_{\text{min}} \le L_{\text{pred}} \le l_{\text{max}}, \\ m \cdot \min\left(1, \frac{l_{\text{min}} - L_{\text{pred}}}{l_{\text{min}}}\right), & L_{\text{pred}} < l_{\text{min}}, \\ m \cdot \min\left(1, \frac{L_{\text{pred}} - l_{\text{max}}}{l_{\text{max}}}\right), & L_{\text{pred}} > l_{\text{max}}. \end{cases} \tag{3}$$

This formulation encourages concise yet structurally complete transition descriptions. Detailed hyperparameters for GRPO training are summarized in Table 9.

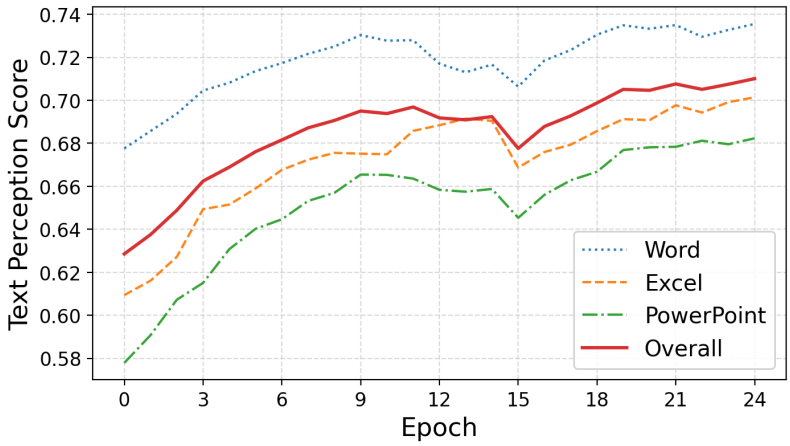

Figure 5: Training curve over epochs for Text Perception Score (↑).

A.2.5 Training Curves and Analysis

Given the inherent stochasticity of diffusion model training, simple loss functions (e.g., MSE) are often insufficient proxies for generation quality. Therefore, we monitor the training progress using the comprehensive set of image-based metrics and the text perception score defined in Section A.3. Figure 5 and Figure 6 illustrate the training dynamics across epochs.

As shown in Figure 6, we observe consistent improvements in visual fidelity: **PSNR** and **SSIM** exhibit an upward trend, while **LPIPS** and **FID** steadily decrease. This trajectory indicates that the model progressively enhances pixel-level accuracy, structural alignment, and perceptual realism. Aligned with these visual gains, Figure 5 demonstrates that the **Text Perception Score** increases steadily across all applications. This confirms that improvements in general image quality effectively translate into more accurate rendering of legible UI text and application-specific patterns. Furthermore, a distinct performance gap emerges across domains: **Word** consistently achieves the highest scores, followed by **Excel**, with **PowerPoint** proving the most challenging. This stratification likely reflects the varying degrees of UI layout complexity and text density inherent to each application.

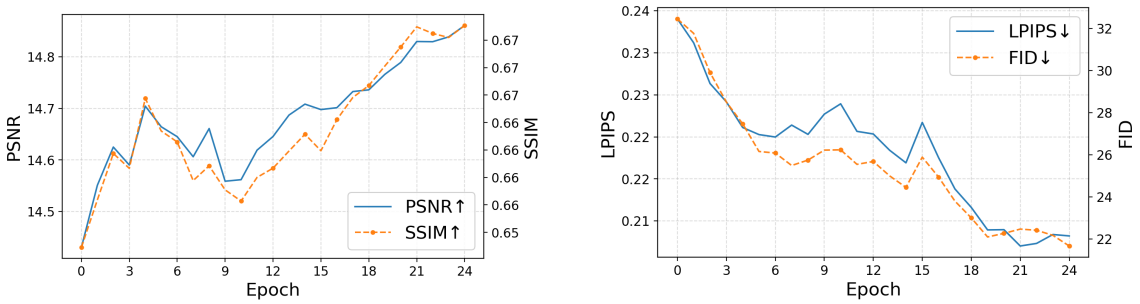

Figure 6: Training curves over epochs for image-level fidelity metrics.

A.3 EVALUATION

A.3.1 LLM-AS-A-JUDGE SCORE

Standard pixel-level similarity metrics often fail to capture the semantic nuances of *what actually changed* in the UI following an action (e.g., updates to the active tab, auxiliary panes, or document content). To address this, we employ an *LLM-as-a-Judge* protocol designed to quantify the semantic consistency between the model's predicted textual description and a reference description.

For each transition, we compare the model-predicted description $\hat{y}$ against a ground-truth description y. We utilize GPT-5 to generate y from the paired screenshots (s_t, s_{t+1}), establishing it as the single source of truth. Given the pair $(\hat{y}, y)$, the judge (GPT-5) is prompted to output a structured JSON containing discrete scores and rationales for specific UI aspects. The complete evaluation prompt is provided in Appendix A.5.3.

Table 10: LLM-as-a-Judge evaluation aspects and their weights.

Key	**UI aspect**	**Weight** w_a
`app_name`	Application name	0.8
`user_action`	Executed user action	1.4
`title_bar`	Title bar state (e.g., document name)	1.0
`ribbon`	Ribbon / toolbar state (e.g., active tab)	1.1
`main_editing_area`	Main editing area / canvas (core content changes)	1.5
`sidebar_pane`	Sidebar / pane state (e.g., open/closed, content)	0.8
`navigation_area`	Navigation area (e.g., thumbnails/outline)	0.6
`status_bar`	Status bar (e.g., page, zoom, mode)	0.8

Per-aspect discrete scoring. Let $\mathcal{A}$ denote the set of UI aspects (e.g., `user_action`, `main_editing_area`) as listed in Table 10, and let $a \in \mathcal{A}$ represent a specific aspect. For a generated description $\hat{y}$ and the ground truth y, the aspect-specific score $s_a(\hat{y}, y)$ is determined as:

$$s_a(\hat{y}, y) = \begin{cases} 1, & \text{if } \hat{y} \text{ aligns with } y \text{ on key elements with high fidelity;} \\ 0.5, & \text{if } \hat{y} \text{ is partially correct but has notable omissions;} \\ 0, & \text{if } \hat{y} \text{ contradicts } y \text{ or introduces hallucinations.} \end{cases} \tag{4}$$

The judge is explicitly instructed to prioritize *content fidelity* over stylistic variations. Omissions of sub-areas not mentioned in the reference are treated neutrally, whereas asserting incorrect changes results in a penalty (score 0).

Aspect weights and final score. To reflect the hierarchical importance of different UI components in downstream reasoning, we compute a weighted average of the aspect scores:

$$\text{JudgeScore}(\hat{y}, y) = \frac{\sum_{a \in \mathcal{A}} w_a \, s_a(\hat{y}, y)}{\sum_{a \in \mathcal{A}} w_a}. \tag{5}$$

As detailed in Table 10, this weighting scheme places higher emphasis on the `main_editing_area` and `user_action`, as these are critical for task progression, while still accounting for the global UI context (e.g., ribbon and title bar).We apply this metric to evaluate performance across different training stages (Base, SFT, SFT+RL), with results summarized in Table 1.

A.3.2 Action Consistency Score

```
  {
    "function": "select_text",
    "args": { "text": "artificial intelligence (AI)" },
    "status": "CONTINUE"
  },
  {
    "function": "click",
    "args": {
      "coordinate": null,
      "button": "left",
      "control_info": {
        "control_type": "Button",
        "control_text": "Text Highlight Color Yellow"
      }
    },
    "status": "FINISH"
  }
]
```

Figure 7: **Example action format.** Two candidate actions illustrating the structured JSON specification used by agents. The first action selects text, while the second clicks a UI control identified by its accessibility label.

A faithful textual state transition model must preserve *decision-relevant* information from the user interface. To quantify this, we introduce the **Action Consistency Score (ACS)**, which measures the functional equivalence between the generated textual state and the ground-truth visual state. The core intuition is that an agent acting solely on the world model's textual prediction should arrive at the same decision as an agent observing the actual screenshot.

Evaluation Protocol. Formally, each evaluation instance consists of a tuple (I, D_{gt}, A, T), where I is the ground-truth current screenshot, D_{gt} is the ground-truth textual description derived from I, A represents accessibility metadata (e.g., control labels, application name), and T is the user instruction. Given a world model, we generate a predicted textual transition D_{wm} for the next state. We employ a *frozen* agent policy π to predict the next action under two distinct conditions:

$$a_{\text{gt}} = \pi\big(I,\ D_{\text{gt}},\ A,\ T\big), \tag{6}$$

$$a_{\text{wm}} = \pi\big(\varnothing,\ D_{\text{wm}},\ A,\ T\big), \tag{7}$$

Here, a_{gt} serves as the *oracle action* derived from the full visual context I, whereas a_{wm} is the *predicted action* derived exclusively from the world model's textual output D_{wm} without visual access.

Scoring Formulation. To compute consistency, we first verify that the agent's output adheres to a structured JSON specification. Specifically, each action a is parsed into three key components: (1) a `function` field denoting the action type (e.g., `click`, `select_text`); (2) an `args` dictionary containing parameters such as target coordinates, control identifiers, or text content; and (3) a `status` field (`CONTINUE` or `FINISH`) indicating task progression. (see Figure 7 for an example))

Based on this structure, we define three atomic matching criteria as detailed in Table 11: **Function Match**, **Status Match**, and **Args Match** (which incorporates spatial tolerance and label matching). The instance-level consistency score is computed as a weighted sum, prioritizing the correctness of action arguments:

$$s(a_{\text{wm}}, a_{\text{gt}}) = 0.25 \cdot \mathbb{1}_{\text{FUNC}} + 0.25 \cdot \mathbb{1}_{\text{STATUS}} + 0.50 \cdot \mathbb{1}_{\text{ARGS}} \tag{8}$$

If the agent fails to produce a valid action in either condition, we assign $s = 0$. The final **Action Consistency Score** is the average score over the entire evaluation dataset $\mathcal{D}$:

$$\text{ACS} = \frac{1}{|\mathcal{D}|} \sum_{(I, D_{\text{gt}}, A, T) \in \mathcal{D}} s(a_{\text{wm}}, a_{\text{gt}})\,. \tag{9}$$

Interpretation. A high ACS indicates that the textual transition D_{wm} successfully retains critical UI cues—such as active tabs, control states, or content updates—necessary for accurate decision-making. Conversely, a low ACS implies that the textual abstraction has lost or distorted information vital for the agent's policy, leading to divergent behavior compared to the visual oracle.

A.3.3 Visual State Realization Evaluation

We assess the fidelity of CUWM-generated next-state UI screenshots using four standard image-quality metrics: PSNR, SSIM, LPIPS, and FID.

PSNR (↑). Peak Signal-to-Noise Ratio measures pixel-wise reconstruction quality based on mean squared error (MSE). Given a generated image $\hat{\mathbf{I}}$ and the ground-truth image $\mathbf{I}$ of size $H \times W$ (with C channels), the MSE is

$$\text{MSE}(\hat{\mathbf{I}}, \mathbf{I}) = \frac{1}{HWC} \sum_{h=1}^{H} \sum_{w=1}^{W} \sum_{c=1}^{C} \left(\hat{I}_{h,w,c} - I_{h,w,c} \right)^2, \tag{10}$$

and PSNR is

$$\text{PSNR}(\hat{\mathbf{I}}, \mathbf{I}) = 10 \log_{10} \left(\frac{\text{MAX}^2}{\text{MSE}(\hat{\mathbf{I}}, \mathbf{I})} \right), \tag{11}$$

Table 11: **Scoring metrics for action evaluation.** Each metric evaluates a specific aspect of action correctness. The Overall Match criterion implies all component metrics are satisfied.

Metric	Description
Function Match	Evaluates whether the predicted action type (e.g., `click`, `type`, `drag`, `select_text`) exactly matches the ground-truth action type.
Status Match	Evaluates whether the predicted task status (`CONTINUE` or `FINISH`) matches the ground-truth status, indicating correct anticipation of task completion.
Args Match	Evaluates whether the action arguments align with the ground truth. For coordinate-based actions, we allow a spatial tolerance of ± 25 pixels or require the point to fall within the target's bounding box. For label-based actions, exact string matching is required. Additional arguments (e.g., `button`, `keys`) must match exactly.
Overall Match	A strict criterion requiring *all three* components (Function Match, Status Match, and Args Match) to be satisfied simultaneously.

where MAX is the maximum possible pixel value (e.g., 255 for 8-bit images or 1 for normalized images).

SSIM (↑). The Structural Similarity Index evaluates perceptual similarity by comparing luminance, contrast, and structure. For local windows (patches) from $\hat{\mathbf{I}}$ and $\mathbf{I}$, SSIM is defined as

$$\mathrm{SSIM}(\hat{\mathbf{I}}, \mathbf{I}) = \frac{(2\mu_{\hat{I}}\mu_I + c_1)(2\sigma_{\hat{I}I} + c_2)}{(\mu_{\hat{I}}^2 + \mu_I^2 + c_1)(\sigma_{\hat{I}}^2 + \sigma_I^2 + c_2)}, \tag{12}$$

where $\mu_{\hat{I}}, \mu_I$ are local means, $\sigma_{\hat{I}}^2, \sigma_I^2$ are local variances, $\sigma_{\hat{I}I}$ is the local covariance, and c_1, c_2 are small constants for numerical stability. We report SSIM averaged over windows (and channels if applicable).

LPIPS (↓). Learned Perceptual Image Patch Similarity measures perceptual distance in deep feature space. Let $\phi^l(\cdot)$ denote the feature map from layer l of a pretrained network, normalized channel-wise as $\hat{\phi}^l$, and let $\mathbf{w}^l$ be learned per-channel weights. LPIPS is computed as

$$\mathrm{LPIPS}(\hat{\mathbf{I}}, \mathbf{I}) = \sum_l \frac{1}{H_l W_l} \sum_{h,w} \left\| \mathbf{w}^l \odot \left(\hat{\phi}^l(\hat{\mathbf{I}})_{h,w} - \hat{\phi}^l(\mathbf{I})_{h,w} \right) \right\|_2^2, \tag{13}$$

where H_l, W_l are the spatial dimensions of layer l, and $\odot$ denotes element-wise multiplication.

FID (↓). Fréchet Inception Distance measures distribution-level discrepancy between generated and real images in the Inception feature space. Let $\{\mathbf{x}_i\}$ and $\{\hat{\mathbf{x}}_j\}$ be Inception features for real and generated images, with empirical means and covariances $(\boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r)$ and $(\boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)$, respectively. FID is

$$\mathrm{FID} = \|\boldsymbol{\mu}_r - \boldsymbol{\mu}_g\|_2^2 + \mathrm{Tr}\Big(\boldsymbol{\Sigma}_r + \boldsymbol{\Sigma}_g - 2\left(\boldsymbol{\Sigma}_r \boldsymbol{\Sigma}_g\right)^{\frac{1}{2}}\Big). \tag{14}$$

A.3.4 Text Perception Score

To evaluate whether a generated next-frame UI screenshot faithfully renders text content (e.g., document text and UI labels), we compute a task-oriented *Text Perception Score* using OmniParser as a screen-text extractor. Given an image, OmniParser returns a list of parsed elements; we keep only items with `type = text` and collect their textual `content` as a multiset of strings. We apply light normalization (lowercasing, removing duplicates, and filtering short/noisy strings) and denote the resulting text sets from the prediction and the ground truth as $\mathcal{P} = \{p_i\}_{i=1}^{|\mathcal{P}|}$ and $\mathcal{G} = \{g_j\}_{j=1}^{|\mathcal{G}|}$, respectively.

Embedding similarity. We embed each text string with a sentence encoder $\phi(\cdot)$ and use cosine similarity:

$$s_{ij} \;=\; \cos\big(\phi(p_i),\, \phi(g_j)\big), \quad p_i \in \mathcal{P},\; g_j \in \mathcal{G}. \tag{15}$$

Symmetric max-match. We measure how well each side can be explained by the other via maximum matching in the embedding space. For the prediction-to-GT direction:

$$M_{\mathcal{P}\to\mathcal{G}} \;=\; \frac{1}{|\mathcal{P}|}\sum_{i=1}^{|\mathcal{P}|} \max_{j}\; s_{ij}, \tag{16}$$

and for the GT-to-prediction direction:

$$M_{\mathcal{G}\to\mathcal{P}} \;=\; \frac{1}{|\mathcal{G}|}\sum_{j=1}^{|\mathcal{G}|} \max_{i}\; s_{ij}. \tag{17}$$

The final Text Rendering Score is the symmetric average:

$$\mathrm{TRS}(\mathcal{P},\mathcal{G}) \;=\; \frac{1}{2}\Big(M_{\mathcal{P}\to\mathcal{G}} + M_{\mathcal{G}\to\mathcal{P}}\Big). \tag{18}$$

Interpretation. $M_{\mathcal{P}\to\mathcal{G}}$ penalizes hallucinated or irrelevant text in the prediction that cannot find a close match in the ground truth, while $M_{\mathcal{G}\to\mathcal{P}}$ penalizes missing or incorrectly rendered text from the ground truth that cannot be recovered from the prediction. Thus, the symmetric formulation rewards both *precision* (avoiding spurious text) and *recall* (preserving all important text). If either side is empty, we define $\mathrm{TRS}(\emptyset,\emptyset) = 1$ and $\mathrm{TRS}(\mathcal{P},\emptyset) = \mathrm{TRS}(\emptyset,\mathcal{G}) = 0$. We report the dataset-level score by averaging TRS over all evaluation samples, and optionally report per-application scores (Word/Excel/PowerPoint) by averaging within each subset.

A.4 Supplementary Agent Evaluation Results

A.4.1 Agent Evaluation Protocol

We evaluate agent performance using a structured protocol designed to isolate the contribution of the world model from the agent policy. During evaluation, each VLM backbone agent (listed in Table 5) receives the current UI screenshot along with a task instruction. The agent is prompted to generate five diverse candidate actions matching the structured JSON format illustrated in Figure 7, using the prompt provided in Appendix A.5. For each candidate action, we pass the task instruction, the action details, and the current screenshot to the CUWM textual transition model, which generates a textual description of the predicted UI changes. This textual transition is then fed into an image-based world model, either the CUWM visual realization model, the base Qwen-Image-Edit, or GPT-Image-1.5, to render the predicted next-state. Finally, the agent selects the action whose predicted outcome best aligns with the task goal.

Table 12: Agent task completion rates with VLM-specific failures removed to isolate world model contribution. No GT indicates samples where the ground-truth action was absent from the VLM's candidate proposals. **Bold** indicates best performance per agent

Agent	No GT	None	Text	Qwen-Edit		CUWM (ours)		GPT-Image	
				Image	Image+Text	Image	Image+Text	Image	Image+Text
Qwen3-VL-8B	159	0.7336	0.7725	0.7629	0.7759	**0.7889**	0.7791	0.7791	0.7684
GPT-4.1-mini	115	0.6600	0.6476	0.6350	0.6486	**0.6686**	0.6188	0.6340	0.6246
GPT-4o	127	0.7288	0.7396	0.7205	0.7464	**0.7548**	0.7394	0.7218	0.7335
Gemini-2.0-Flash	135	0.6519	0.6660	0.6735	0.6195	**0.6768**	0.6064	0.6654	0.6350

A.4.2 SCORING METHODOLOGY

For each test sample, we compare the agent-selected action against the ground-truth action from the GUI-360 dataset using four metrics: *Function Match*, *Status Match*, *Args Match*, and *Overall Match*, summarized in Table 11. The Overall Match criterion requires that all three preceding metrics agree with the ground truth. The final agent task score is computed as the proportion of Overall Matches across all 339 evaluation samples as described in Section 3.4.

A.4.3 GROUND-TRUTH ACTION COVERAGE

It is important to note that not all candidate action sets generated by the agent include the ground-truth action. We observe that approximately 35% of evaluation samples fall into this category, though this proportion varies by VLM backbone, each agent independently fails to propose the ground-truth action on different subsets of samples. While this limitation is unrelated to the CUWM framework and reflects inherent agent proposal quality, we retain these samples in the evaluation to provide a faithful measure of end-to-end agent performance. Table 12 separately reports results excluding these samples, evaluating only cases where the ground-truth action appeared in the candidate set.

A.4.4 EFFECTIVENESS OF RL REFINEMENT ON AGENT TASK SCORE

Table 13 presents the downstream task accuracy averaged over 100 randomly selected samples. Consistent with the Action Consistency Score (ACS) analysis in Section 4.2.1, the RL-refined CUWM consistently outperforms the SFT baseline across both GPT-4.1-mini and Gemini-2.0-Flash agents. Specifically, the RL variant achieves higher accuracy on CUWM-generated images (0.4317 and 0.4700, respectively), surpassing the SFT variant. This validates our hypothesis that the improved semantic preservation achieved via GRPO (indicated by higher ACS) directly translates into higher-fidelity visual realizations that effectively guide agent decision-making.

Table 13: **Agent performance across input modalities.** We report accuracy scores averaged over the last 6 runs with standard deviations where available. **Bold** indicates best performance per agent. This was tested on 100 randomly selected samples.

Model	Method	Text	Base Image	CUWM Image
GPT-4.1-mini	RL	0.4467 ± 0.0075	0.4167 ± 0.0149	**0.4317 ± 0.0107**
	SFT	0.4483 ± 0.0121	0.4250 ± 0.0126	0.4283 ± 0.0069
Gemini-2.0-Flash	RL	0.4433 ± 0.0047	0.4584 ± 0.0075	**0.4700** ± 0.0100
	SFT	0.4410 ± 0.0068	0.4452 ± 0.0085	0.4561 ± 0.0094

A.5 PROMPTS

Agent Evaluation - Action Option Generation Prompt

```
You are an expert in Office Application automation and graphical user
    interfaces with accessibility support.

You will be provided with the following inputs:

1. **Current screenshot**: An image of the current state of an Office
    Application.
```

```
2. **Annotated current screenshot**: The same screenshot annotated with
    numeric markers corresponding to accessibility elements.2. **Annotated
    current screenshot**: The same screenshot annotated with numeric markers
     corresponding to accessibility elements.
3. **Accessibility (a11y) information**: This includes a list of control
    element labels and the textual name of the currently active Office
    Application.
4. **Task instruction**: A description of the action or goal to be
    completed.
5. **Supported actions**: A list of all actions that can be performed in
    this environment.

The annotated screenshot contains numbers that correspond directly to
    entries in the accessibility information. Each number identifies a
    specific UI control element, allowing you to reliably locate and
    reference interface components.

The accessibility information contains control labels that correspond to UI
     control elements in the current screenshot, allowing you to locate and
    reference specific interface components.

Your objective is to generate multiple diverse and plausible next actions
    that could be taken to accomplish the given task instruction, based on
    the current screenshot of the Office Application, the annotated current
    screenshot, and the available accessibility information. The desired
    number of options will be specified in the user instructions or input.

Use all the provided information-including the current screenshot,
    annotated screenshot, accessibility data, task instruction, and
    supported actions-to reason about the next appropriate actions
    accurately.


**IMPORTANT: When possible, prioritize using control_label over coordinate
    for actions. Control labels refer to unique identifiers provided by the
    accessibility (a11y) system for UI elements (such as buttons, text
    fields, or menu items). These identifiers are more reliable and
    accessible than raw screen coordinates, which may vary across layouts,
    resolutions, or UI states.**

For each candidate action, explain your reasoning process-describe how you
    analyze the current screenshot and the annotated screenshot, interpret
    the accessibility information, understand the current UI state, and
    determine what action could be taken next to move toward completing the
    task instruction.

Then, output the next actions in JSON format as a JSON array. Each element
    of the array must be an object with the keys "thoughts" and "tool_call".
     Both fields MUST be present in every element.

Please think very hard and carefully about the current state and the task
    instruction before making a decision, and output your reasoning in each
    element's "thoughts" field as detailed as possible, including:
- Your analysis of the current screenshot and annotated screenshot
- Your interpretation of the accessibility information
```

```
- How you identified the target control using control labels or visual
    elements
- The reason for your tool call and argument selection
- Your assessment of task progress based on the current state

The "tool_call" field in each element should contain:
- "function": str, The function/action type to execute
- "args": Dict, The arguments/parameters for the function
- "status": str, The status after performing this action (either "CONTINUE"
    or "FINISH")

For click operations, prioritize control_label over coordinate:
```json
{
  "thoughts": "The screenshot shows an Excel spreadsheet. From the
      accessibility information, there is a Save button with control_label
      =15. Using the control_label provides a more reliable interaction than
       estimating screen coordinates.",
  "tool_call": {
    "function": "click",
    "args": {"control_label": 15, "coordinate": null, "button": "left"},
    "status": "CONTINUE"
  }
}
````

If control_label is not available, fall back to coordinate:

```json
{
  "thoughts": "The target UI element does not have a corresponding
      control_label in the accessibility information. Based on the visual
      analysis of the annotated screenshot, I estimate the appropriate
      screen coordinates to interact with it.",
  "tool_call": {
    "function": "click",
    "args": {"control_label": null, "coordinate": [150, 30], "button": "
        left"},
    "status": "CONTINUE"
  }
}
```

For type operations, prioritize control_label over coordinate:

```json
{
  "thoughts": "The accessibility information indicates a text input field
      with control_label=8. Typing via the control_label ensures the correct
       field is targeted.",
  "tool_call": {
    "function": "type",
    "args": {"control_label": 8, "coordinate": null, "keys": "Hello World",
        "clear_current_text": true},
    "status": "CONTINUE"
```
```

```
    }
  }
  ```

  For drag operations, coordinates are required:

  ```json
  {
    "thoughts": "This task requires dragging an element from one location to
        another. Drag actions require explicit start and end coordinates to
        describe the spatial movement.",
    "tool_call": {
      "function": "drag",
      "args": {"start_coordinate": [100, 100], "end_coordinate": [200, 200],
          "button": "left"},
      "status": "CONTINUE"
    }
  }
  ```

  If you think the task is finished, output status as "FINISH":

  ```json
  {
    "thoughts": "Based on the current state of the Office Application and the
        task instruction, no further actions are required.",
    "tool_call": {
      "function": "",
      "args": {},
      "status": "FINISH"
    }
  }
  ```

  Only **ONE** action should be taken per option. You may output multiple
      options (as requested by num_options), but each option must correspond
      to exactly one action. If the task instruction could apply to multiple
      elements, choose the most relevant one for each option based on the
      current screenshot and accessibility information, and ensure the options
       are diverse.

  Your response MUST be a valid JSON array with no additional text outside
      the JSON structure.

  Task instruction:
  {instruction}

  Accessibility Information:
  {a11y}

  Supported actions:
  {actions}

  The current screenshot and the annotated current screenshot are provided as
       images.
```
```

```
Please analyze the current state using both visual information and
    accessibility data to generate {num_options} diverse and plausible next
    actions that could be taken to move toward completing the task
    instruction.

Please provide your reasoning and the next actions below in JSON array
    format without any additional text.
```

A.5.1 AGENT EVALUATION - ACTION SELECTION PROMPT

```
You are an AI assistant tasked with selecting the best next action in an
    Office application (e.g., Microsoft Word, Excel, or PowerPoint) based on
     a GUI screenshot and several candidate action outcomes.

The task instruction is:
[Instruction]

The current screenshot is: See the image below.

[Current State Screenshot]

You are given a list of action options, each with:
- action: the structured action command to be executed
- predicted_state_image: a visual prediction (mockup) showing the expected
    GUI state after executing the action

The `predicted_state` fields are simulations. Use them as references, but
    **ignore diffusion artifacts** (e.g., garbled text, noisy details). If a
     prediction is unreliable or inconsistent with the task, you may ignore
    it.

### Analysis Instructions:
1. **Analyze Every Option:** specific `predicted_state` simulations (images
    /text) are provided for every action. You must look at all of them first
    .
2. **When to use the World Model:**
   - **Uncertainty:** Use the predicted image when you are uncertain based
       purely on the action name or your internal knowledge.
   - **Discovery:** Use the predicted image if it clearly shows a better
       way of advancing the goal that you didn't initially think of.
   - **Efficiency:** Use the predicted image if it shows a way to skip
       intermediary steps and advance to the goal faster.
3. **Decide:** Combine your internal knowledge with these visual signals to
     pick the single best action.


Please return your final answer as a JSON object with the following format:
{
  "action_idx": <index of selected action>,
  "thought": "<brief explanation of why this action was selected>"
}
```

```
Do NOT include anything else outside the JSON object.
Here are the candidate action options:

Action Option 1:
 - Action:
 - Predicted State Image: see below.
[Predicted Image 1]

Action Option 2:
 - Action:
 - Predicted State Image: see below.
[Predicted Image 2]

Action Option 3:
 - Action:
 - Predicted State Image: see below.
[Predicted Image 3]

Action Option 4:
 - Action:
 - Predicted State Image: see below.
[Predicted Image 4]

Action Option 5:
 - Action:
 - Predicted State Image: see below.
[Predicted Image 5]

Now, analyze all options and return only the selected 'action_idx' and a
    short 'thought' in JSON format. 'action_idx' MUST be an integer between
    1 and 5 (inclusive). Do NOT output any text outside the JSON object.
```

A.5.2 TEXTUAL STATE TRANSITION PROMPT

```
Your task is to predict and describe the likely appearance of the Next UI
    Screenshot, with a strong emphasis on how the UI would change compared
    to the current state, and where these changes would be located within
    the interface.

You are acting as a World Model assistant for Office applications such as {
    app_name}. Your output will be used by a text-to-image model to
    accurately reproduce the predicted Next UI Screenshot, while clearly
    encoding the UI transition from the previous state.

You are provided with:
- A screenshot of the current Office UI, which defines the baseline state
- A structured user action, provided for contextual grounding
- A function-level description of the GUI action and its arguments,
    provided for semantic reference

Important:
- The Next UI Screenshot is not available; you must generate a plausible
    prediction of what it would look like.
```

```
- Use the current screenshot to understand the baseline state.
- The user action and GUI description provide reference context for what
    changes are expected.
- Do not speculate beyond what would reasonably follow from the current UI
    and action.

---

Reference Information:

Current UI Screenshot:
{image}

Action:
{action}

GUI Action Description:
{gui_description}

---

In your response, follow this structure:

1. Start by stating which Office application this is (e.g., "This is
    Microsoft PowerPoint.").
2. Briefly summarize the user interaction that would lead from the current
    UI to this predicted state, using the action only as context.
3. Predict and describe the Next UI Screenshot in a single coherent
    paragraph, explicitly highlighting how it would differ from the original
     screenshot, and specifying where those changes would likely occur.

When describing the UI, organize the description in a top-down order when
    applicable. Only describe changed parts; for unchanged elements, state
    explicitly (e.g., "Ribbon unchanged"):

- Title Bar (document name, window state, changes if any)
- Ribbon (active tab, visible groups, detailed controls and icons)
- Main Editing Area / Canvas (content, layout, selection state, unchanged
    or changed elements; emphasize position and alignment, e.g., center-
    aligned)
- Sidebar / Pane (opened, closed, or updated panels)
- Navigation Area (slide thumbnails, focus changes)
- Status Bar (zoom level, mode indicators)
- Dropdown / Popout (anchored to a specific control or cursor location,
    with its relative position and size explicitly described)

Explicitly indicate predicted changes and their locations using clear
    language, such as:
- "In the Ribbon, the 'Insert' tab is expected to become active..."
- "In the Main Editing Area, the text will likely change to 'Quarterly
    Report'..."
- "A new panel labeled 'Design Ideas' is likely to appear on the right
    sidebar..."
```

```
All visible text in the UI should be enclosed in double quotes (e.g., "Home
    ", "File", "New Slide").

Ensure that your description:
- Clearly encodes the transition from the current UI to the next UI state
- Specifies where changes are likely to occur in the UI
- Uses terminology and layout conventions consistent with {app_name}

Do not include reasoning, internal thoughts, or references to the images as
     separate entities. Do not answer in bullet points.
Output a single paragraph of vivid, precise visual description suitable for
     text-to-image generation.
```

A.5.3 TEXTUAL STATE TRANSITION EVALUATION - LLM-AS-A-JUDGE PROMPT

```
You are an impartial LLM-as-a-Judge. Your task is to grade a model
    prediction (PRED) against the ground truth (GT) for describing the ``
    Next UI Screenshot'' of an Office application (e.g., Microsoft Word).

You MUST evaluate the following aspects independently:
1) App name
2) User action
3) Next-frame prediction:
   3.1) Title Bar
   3.2) Ribbon
   3.3) Main Editing Area / Canvas
   3.4) Sidebar / Pane
   3.5) Navigation Area
   3.6) Status Bar

Scoring rule for EACH aspect (use ONLY these values):
- 0   = completely incorrect / contradicts GT / missing when GT contains it
- 0.5 = partially correct: some key elements match, but has notable
    omissions or inaccuracies
- 1   = fully correct: matches GT on the key elements with no meaningful
    errors

Critical evaluation guidelines:
- Use GT as the single source of truth.
- Judge content fidelity, not writing quality.
- Be strict about factual UI elements (active tab name, document title,
    zoom %, panes open/closed, specific text edits).
- Penalize hallucinations: if PRED adds UI changes or elements not
    supported by GT, deduct in the relevant aspect(s).
- If GT does NOT mention a sub-area (e.g., Navigation Area), then:
  - If PRED also does not mention it -> score 1 (no contradiction).
  - If PRED claims a specific change/state that GT does not support ->
      score 0.5 or 0 depending on how strong/incorrect it is.
- When scoring 0.5 vs 1, treat the following as ``key elements'':
  - Title Bar: document name, saved/unsaved indicator, window state if
      mentioned
  - Ribbon: active tab, visible groups, important controls/menus if
      mentioned
```

```
    - Dropdown / Popout: presence, anchor, relative position, size, and
        visible content
    - Main Editing Area: the actual document text changes, formatting (bold/
        center-aligned), cursor/selection state, layout
    - Sidebar/Pane: which pane is open, its content list/state
    - Navigation Area: thumbnails/outline focus changes if present
    - Status Bar: page number, zoom, mode toggles (Track Changes, etc.)

Output format requirements:
- Output ONLY valid JSON.
- No markdown, no extra text.
- Include per-aspect scores.


Return JSON with exactly this structure:
{{
  "scores": {{
    "app_name": <0|0.5|1>,
    "user_action": <0|0.5|1>,
    "title_bar": <0|0.5|1>,
    "ribbon": <0|0.5|1>,
    "main_editing_area": <0|0.5|1>,
    "sidebar_pane": <0|0.5|1>,
    "navigation_area": <0|0.5|1>,
    "status_bar": <0|0.5|1>
  }},
  "notes": {{
    "app_name": "<one short sentence rationale>",
    "user_action": "<one short sentence rationale>",
    "title_bar": "<one short sentence rationale>",
    "ribbon": "<one short sentence rationale>",
    "main_editing_area": "<one short sentence rationale>",
    "sidebar_pane": "<one short sentence rationale>",
    "navigation_area": "<one short sentence rationale>",
    "status_bar": "<one short sentence rationale>"
  }}
}}

Now perform the evaluation.

PRED:
<<<
{PRED}
>>>

GT:
<<<
{GT}
>>>
```