AGUVIS: UNIFIED PURE VISION AGENTS FOR AUTONOMOUS GUI INTERACTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Graphical User Interfaces (GUIs) are critical to human-computer interaction, yet automating GUI tasks remains challenging due to the complexity and variability of visual environments. Existing approaches often rely on textual representations of GUIs, which introduce limitations in generalization, efficiency, and scalability. In this paper, we introduce AGUVIS, a unified pure vision-based framework for autonomous GUI agents that operates across various platforms. Our approach leverages image-based observations, and grounding instructions in natural language to visual elements, and employs a consistent action space to ensure cross-platform generalization. To address the limitations of previous work, we integrate explicit planning and reasoning within the model, enhancing its ability to autonomously navigate and interact with complex digital environments. We construct a large-scale dataset of GUI agent trajectories, incorporating multimodal reasoning and grounding, and employ a two-stage training pipeline that first focuses on general GUI grounding, followed by planning and reasoning. Through comprehensive experiments, we demonstrate that AGUVIS surpasses previous state-of-the-art methods in both offline and real-world online scenarios, achieving, to our knowledge, the first fully autonomous pure vision GUI agent capable of performing tasks independently without collaboration with external closed-source models. We will open-source all datasets, models, and training recipes to facilitate future research.

1 INTRODUCTION

Graphical User Interfaces (GUIs) are a cornerstone of human-computer interaction, providing a structured yet intuitive platform for users to accomplish tasks across various digital environments: website, desktop, and mobile devices (Deng et al., 2023; Zhou et al., 2024; Xie et al., 2024; Rawles et al., 2024b). Automating GUI operations through autonomous agents can revolutionize productivity by enabling seamless task execution on various applications using existing human-centric tools. Moreover, this approach lays the groundwork for advanced AI systems that can interact with and learn from rich digital environments in ways that mirror human behavior.

To effectively perform GUI tasks autonomously, a GUI agent requires three core competencies: understanding, grounding, and planning & reasoning. For GUI understanding, the agent must first comprehend high-resolution and complex interfaces designed for human users, enabling it to grasp the context and perform subsequent reasoning tasks. GUI grounding involves mapping natural language instructions to visual observations of the interface. For planning and reasoning, the agent must synthesize and analyze the current multimodal observations of the environment with previous observations and action histories, enabling it to generate coherent and effective next steps to ultimately achieve the task goal. Although recent advances in large vision-language models (LVLMs) (OpenAI, 2024; Reid et al., 2024; Li et al., 2024a; Wang et al., 2024a) have significantly enhanced the ability of AI systems to interpret complex visual interfaces, there remain critical challenges in grounding and reasoning specifically tailored for GUI tasks. We identify three primary challenges that must be addressed to advance the capabilities of GUI agents:

Enhancing Pure Vision Framwork. Previous approaches (Gur et al., 2024; Kim et al., 2023; Deng et al., 2023; Zhou et al., 2024; Xie et al., 2024) predominantly focus on mapping natural language instructions to textual representations of GUIs, such as HTML or accessibility trees. This method presents several limitations. Firstly, GUIs are inherently visual, and leveraging image-based representations aligns more closely with human cognitive processes. Secondly, textual representations can vary widely across different environments, complicating the generalization of the model and limiting the availability of consistent training data. Finally, these textual representations are often verbose and complex, leading to increased inference times compared to more compact image encodings (Figure 2). By unifying observations across platforms as images and grounding instructions to image coordinates, GUI agents can generalize more effectively across diverse environments with varying resolutions.

Unification Across GUI Environments. The action spaces and control APIs for GUI interactions vary significantly across diverse environments, particularly when the observations are textual. Even within the same platform, the action space can differ greatly. This heterogeneity limits the amount of training data available for each environment, impeding the development of a model that can generalize effectively across different platforms and scale further. A unified action space that abstracts these environmental differences is crucial for creating robust and adaptable GUI agents. Previous work (Chen et al., 2024b; Zeng et al., 2024) has attempted to unify digital agent data across diverse environments, such as combining GUI, game, and CLI interfaces for joint training. However, these interfaces do not share the same interaction logic. In contrast, GUIs on desktop, web, and mobile platforms naturally share similar human-computer interaction (HCI) logic. This commonality facilitates their unification, enabling consistent visual observations and action spaces that mutually benefit both visual grounding and reasoning.

Integrating Planning and Reasoning with Grounding. Current methodologies (Zheng et al., 2024a) often depend on the reasoning capabilities of closed-source large language models (LLMs) (OpenAI, 2024) to plan the completion of GUI tasks or, alternatively, train agents to make direct action decisions through grounding without an explicit reasoning process. This dichotomy results in either a lack of grounding abilities or a lack of comprehensive reasoning abilities. Recently, some works (Gou et al., 2024; Lu et al., 2024) attempt to use closed-source LLMs with specialized GUI grounding models together and communicate with natural language instruction to utilize both abilities. However, on the one hand, natural language communication between the two models usually results in information loss. On the other hand, most importantly, this approach is not further scalable to solve GUI interaction since grounding has been improved close to the upper bound with data synthesis, and most remaining problems are planning related. However, the GUI planning and reasoning ability of closed-source LLMs cannot be further improved.

To address these challenges, we introduce a unified framework for GUI agents that harmonizes pure vision observation and consistent action spaces across diverse environments. Our approach leverages vision-based grounding to improve generalization and reduce inference costs while employing a standardized action space with a plugin system to facilitate consistent learning and interaction across various platforms. After a unified GUI grounding training stage, we demonstrate that unified augmented datasets can effectively build a model capable of executing complex GUI grounding instructions on various platforms. In addition, we integrate explicit visual planning and reasoning into the same model, enabling autonomous navigation and interaction within complex digital environments. Since existing GUI agent trajectories do not fully support these demands, we have unified the existing planning datasets on different platforms and constructed a large-scale, pure vision, cross-platform, multi-step dataset of agent trajectories, featuring comprehensive multimodal reasoning and grounding. Through extensive experiments across various scenarios, we demonstrate the effectiveness of our approach in advancing the state-of-the-art for pure vision-based autonomous GUI agents. To our knowledge, this is the first model that can autonomously complete tasks in real-world online environments without relying on higher reasoning abilities from closed-source models.

Our contributions are as follows:

- We introduce a unified pure vision framework for building generalizable GUI agents that operate with vision-based observations and a plugin-enabled action system, enhancing cross-platform adaptability.
- We develop a comprehensive data pipeline that unifies existing GUI grounding annotations and integrates explicit planning and reasoning. This enables the construction of large-scale datasets for grounding and multi-step agent trajectory datasets across platforms.
- Starting with a VLM, we present a two-stage training process—first for GUI grounding, followed by planning and reasoning—resulting in AGUVIS, the first cross-platform au-

tonomous GUI agent capable of performing complex tasks independently without relying on closed-source models. All data, models, and training resources will be open-sourced.

2 AGUVIS

2.1 PROBLEM FORMULATION

We model the autonomous GUI agent's interaction with the environment as a Partially Observable Markov Decision Process (POMDP), characterized by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{O}, T, O)$. In this formulation, $\mathcal{S}$ represents the set of possible states of the environment, $\mathcal{A}$ denotes the set of actions the agent can take, and $\mathcal{O}$ refers to the set of observations the agent can receive. The state transition function, $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$, defines the probability of transitioning from one state to another given an action, while the observation function, $O : \mathcal{S} \times \mathcal{A} \times \mathcal{O} \to [0, 1]$, specifies the probability of receiving a particular observation given a state and an action.

At each time step t, the agent receives an image observation o_t from the GUI environment, updates its belief state b_t based on its previous actions and observations, and generates an inner monologue (Huang et al., 2022). This inner monologue consists of three components: a natural language description of the current observation (d_t), internal reasoning (h_t) based on the high-level goal G, the observation description d_t, and previous thoughts h_{t-1}, and finally, a low-level action instruction (a_t^{instr}) in natural language that specifies the next action. The agent then executes the action a_t based on the instruction a_t^{instr}, receives a new observation o_{t+1}, and repeats this process until it either achieves the goal G or reaches a terminal state.

2.2 UNIFIED PURE VISION FRAMEWORK

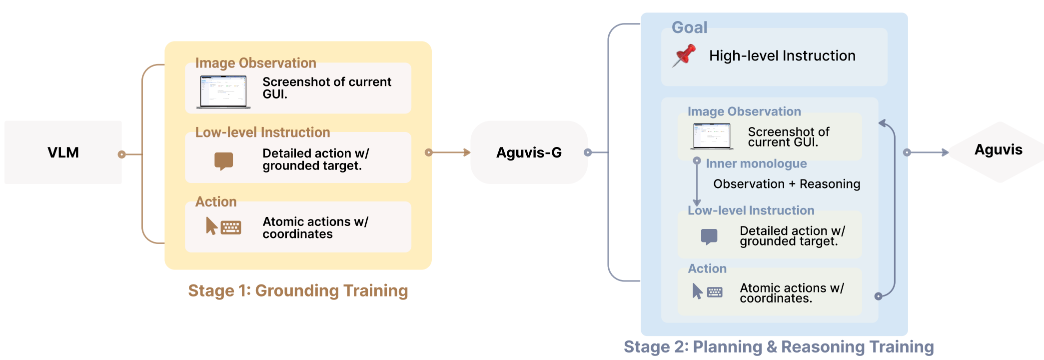


Figure 1: Overview of the two-stage training paradigm for autonomous GUI agents.

In this work, we propose to unify observation and action space via pure vision and standard `pyautogui` commands with a pluggable action system (Table 8). For observation, pure vision does not require the model to understand different UI source codes of the interfaces of different platforms, such as HTML of the webpage, and accessibility tree of desktop and mobile operating systems, which can help improve the generalization. Meanwhile, pure vision can reduce the input token length. Generally, the input length of accessibility tree observation is 6k tokens, and HTML is 4k tokens (Figure 2), depending on the complexity of the interface. Compared with relatively long input, the token cost of image observation does not vary across different interfaces but only depends on model design, which in our case is 1200 tokens for 720p image observation.

For unified action space, we choose the widely used standard `pyautogui` action space with a pluggable action system. This library leverages the high-level programming language Python to replicate and replay various human inputs into computers through code, allowing us to construct a universal and complete representation of actions. We show the action space in Table 8. We use `pyautogui` commands to unify basic GUI operations of all platforms including web, desktop, and mobile. Over this action space, an agent model can then learn to generate actions in order to control GUI without any action space description.

While mouse and keyboard inputs form the core of GUI interactions, they are not comprehensive. Certain platforms require additional actions. For example: (1) specific actions on mobile platforms such as swiping; (2) shortcuts that efficiently perform a series of actions like opening apps; (3) communication actions such as providing answers or terminating after completion. To address these extended requirements, we introduce a pluggable action system. This system allows us to expand the action space by aligning new actions with the existing `pyautogui` commands where possible. For actions that cannot be directly mapped, the pluggable system provides the flexibility to incorporate them with detailed action descriptions. This enables the model to generalize effectively to environments where new actions are introduced. By combining pure vision observations with a unified action space and a flexible pluggable system, our framework enables the training of a single model that can operate across diverse platforms. This setup not only simplifies the training process but also ensures the model can generalize and adapt to novel environments and tasks.

2.3 The AugVis Collection

GUI agent trajectories are a low-resource data source compared with its challenges. This is because the observation and action space vary across different environments even on the same platform. Fortunately, GUI environments share the same operation logic and similar action space. We can efficiently unify existing data to scale the training set. Therefore, we propose The AugVis Collection, a large-scale GUI agent training dataset collected and augmented with existing GUI agent data. This data collection consists of two splits: grounding split (Table 9) and planning & reasoning split (Table 10), corresponding to the two important GUI abilities.

Template-augmented Grounding Data. Vision-based grounding requires the model to ground the natural language intent to the image observation with coordinates. On one hand, there are several previous works that have built datasets on different platforms, including natural language instructions and corresponding target elements. We collected and unified them into `pyautogui` commands format. On the other hand, we found that there are many datasets proposed for user interfaces on different platforms that contain a large amount of metadata, including the positions of all text/icons/widgets in the current interface. Using this type of data we constructed templates for `pyautogui` actions. We randomly generated grounding data pairs through these templates to train models to ground these elements based on images. This operation greatly expanded the amount of data we could use.

VLM-augmented Planning & Reasoning Trajectories. High-quality GUI agent trajectories contain several key components: a high-level goal, a sequence of interleaved observations, natural language reasoning, and grounded actions. Existing approaches typically rely on human annotation to collect these trajectories (Deng et al., 2023; Rawles et al., 2024b; Li et al., 2024c). Most of the agent trajectory data contains high-level goals, observations, and grounded actions. However, the intermediate reasoning process and low-level action instructions are not included. This makes it difficult for existing data to train agents to perform chain-of-thought or inner monologue reasoning to help the model plan the next action, resulting in poor agent performance.

To augment the agent trajectories with detailed reasoning and low-level action instructions, we employ a vision-language model (VLM) to generate the inner monologue for each step in the trajectory. Specifically, for each time step t, given the high-level goal G, the current image observation o_t, and the grounded action a_t, we prompt the VLM to produce the inner monologue components: observation description d_t, thoughts h_t, and low-level action instruction a_t^{instr}. To assist the VLM in generating accurate and contextually relevant monologues, we highlight the target element associated with the grounded action a_t on the image observation o_t. This visual cue helps the model focus on the relevant part of the interface. Additionally, we include the previous low-level action instructions $a_1^{\text{instr}}, a_2^{\text{instr}}, \ldots, a_{t-1}^{\text{instr}}$ to provide the VLM with the action history, ensuring continuity and coherence in the generated reasoning.

The prompting strategy is carefully crafted to guide the VLM in generating inner monologues that are predictive and goal-oriented, without relying on hindsight or revealing future actions. By simulating the agent's thought process in a first-person perspective, we encourage the generation of actionable instructions that align with the high-level goal and current observation. This approach results in a large-scale dataset of agent trajectories enriched with detailed reasoning and instructions.

2.4 MODEL ARCHITECTURE

Unlike grounding agents that rely on structured UI representations (such as accessibility trees) as their textual input, vision-based grounding requires the model to map intents directly to visual observations. This means the model needs to encode high-resolution images while preserving their original aspect ratios. Recent advances in VLMs have made these capabilities possible. We choose Qwen2-VL (Wang et al., 2024b) as our starting VLM. It uses NaViT as an image encoder with native dynamic resolution support (Dehghani et al., 2023). Unlike its predecessor, Qwen2-VL can now process images of any resolution, dynamically converting them into a variable number of visual tokens. To support this feature, ViT is modified by removing the original absolute position embeddings and introducing 2D-RoPE (Su et al., 2024) to capture the two-dimensional positional information of images. Based on these unique features, Qwen2-VL is highly suitable for GUI agents' needs. It can encode high-resolution images of any ratio with relatively fewer image token costs. Therefore, we chose Qwen2-VL as our starting VLM to build our GUI agent.

LLaVA-OneVision (Li et al., 2024a) is another suitable VLM as it also supports high-resolution any ratio image encoding, although its image token cost is relatively higher than Qwen2-VL. We also apply our data recipe and training strategy to LLaVA and show that our framework is model-independent and generally works for high-resolution VLMs details are shown in Section 4.2..

2.5 TRAINING PARADIGM

We begin with a Vision-Language Model (VLM) that possesses advanced image understanding capabilities, and the training process is divided into two main stages: Grounding Training and Planning & Reasoning Training. Each stage utilizes a distinct data split from our THE AUGVIS COLLECTION to progressively enhance the VLM's abilities.

Stage 1: Grounding Training In this stage, we focus on enabling the model to understand and interact with objects within a single GUI screenshot. GUI environments typically feature multiple interactable objects within a single screenshot, generating a large volume of grounding data but leading to shorter, less diverse interaction sequences, which can limit training efficiency.

We train our model with a grounding packing strategy where multiple instruction-action pairs are bundled into a single image, resulting in a single-image-multiple-turn format. This technique allows the model to process several grounding examples from one screenshot, reducing redundant training overhead while retaining a high level of grounding performance. This approach significantly accelerates training by maximizing the use of each image without compromising accuracy. Upon completing this stage, the model is referred to as AGUVIS-G. After this training stage, we assume the agent model AGUVIS-G possesses a robust capability for GUI understanding and grounding, which is the foundation of further planning and reasoning.

Stage 2: Planning & Reasoning Training Building on the foundation of AGUVIS-G, the second stage introduces more complex decision-making and reasoning processes. This phase is designed to teach the model how to execute multi-step tasks by reasoning through agent trajectories that vary in complexity and environments, encompassing diverse reasoning modes.

Thanks to our detailed inner monologue trajectory data, we implement a reasoning mixture approach, where the model is exposed to various levels of cognitive complexity, from straightforward low-level action instructions to full inner monologues that include observation descriptions, thoughts, and detailed action plans. By dynamically adjusting the complexity of these trajectories, we train the model to be adaptable, fostering step-by-step reasoning and high-level decision-making abilities. This diversity in reasoning ensures that the model can handle a wide range of tasks with nuanced understanding and precision. After this stage, the fully trained model is called AGUVIS, which can be employed in both offline and online GUI tasks across diverse environments.

3 EXPERIMENTS

To evaluate the effectiveness of GUI agent models on various platforms, we conduct experiments on several GUI benchmarks: GUI Grounding Evaluation and Offline/Online GUI Agent Evaluation.

3.1 GUI GROUNDING EVALUATION

Table 1: Comparison of various planners and grounding methods on ScreenSpot across various device and input modalities. The top part of table shows the results on *original instructions* evaluation setting while the bottom part shows results on *self-plan* evaluation setting. Best results are in bold.

Planner	Grounder	Mobile		Desktop		Web		Avg
		Text	Icon/Widget	Text	Icon/Widget	Text	Icon/Widget	
-	GPT-4	22.6	24.5	20.2	11.8	9.2	8.8	16.2
	GPT-4o	20.2	24.9	21.1	23.6	12.2	7.8	18.3
	CogAgent	67.0	24.0	74.2	20.0	70.4	28.6	47.4
	SeeClick	78.0	52.0	72.2	30.0	55.7	32.5	53.4
	Qwen2-VL	75.5	60.7	76.3	54.3	35.2	25.7	55.3
	UGround	82.8	60.3	82.5	63.6	80.4	70.4	73.3
	AGUVIS-G-7B	**88.3**	**78.2**	**88.1**	**70.7**	**85.7**	**74.8**	**81.8**
GPT-4	SeeClick	76.6	55.5	68.0	28.6	40.9	23.3	48.8
	OmniParser	93.9	57.0	91.3	63.6	81.3	51.0	73.0
	UGround	90.1	70.3	87.1	55.7	85.7	64.6	75.6
GPT-4o	SeeClick	81.0	59.8	69.6	33.6	43.9	26.2	52.3
	UGround	93.4	76.9	92.8	67.9	88.7	68.9	81.4
AGUVIS-7B		**95.6**	77.7	93.8	67.1	88.3	75.2	**84.4**
AGUVIS-72B		94.5	**85.2**	**95.4**	**77.9**	**91.3**	**85.9**	**89.2**

ScreenSpot. We first assess the performance of GUI grounding, which is a foundational capability of GUI agent models. Following previous work (Cheng et al., 2024; Gou et al., 2024), we evaluate models on ScreenSpot (Cheng et al., 2024). This dataset encompasses a variety of grounding instructions tailored for mobile, desktop, and website platforms, and is assessed under two distinct settings: (1) *Original Instructions*: models perform grounding actions directly following the original instructions; and (2) *Self-plan*: models are required to generate plans in natural language based on the original instructions before executing grounding actions.

The performance illustrated in Table 1 demonstrates that AGUVIS exhibits impressive GUI grounding capabilities under two settings across various platforms. We observe that with the proposed grounding training, AGUVIS-G-7B significantly outperforms existing models with the original instructions, suggesting that AGUVIS has strong universal GUI grounding capability. After training on high-quality planning trajectory data, AGUVIS shows strong planning capability and outperforms previous models that rely on external closed-source LLMs (like GPT-4o). Moreover, further scaling parameters, AGUVIS-72B achieves state-of-the-art performance, attaining an average score of 89.2.

Table 2: Performance comparison on Multimodal Mind2Web across different settings. We report element accuracy (Ele.Acc), Operation F1 (Op.F1), and step success rate (Step SR). Best results are in bold. “T” means the textual HTML code as inputs. “I” means the GUI images as inputs.

Obs.	Planner	Grounder	Cross-Task			Cross-Website			Cross-Domain		
			Ele.Acc	Op.F1	Step SR	Ele.Acc	Op.F1	Step SR	Ele.Acc	Op.F1	Step SR
T	GPT-3.5	Choice	19.4	59.2	16.8	14.9	56.5	14.1	25.2	57.9	24.1
	GPT-4	Choice	40.8	63.1	32.3	30.2	61.0	27.0	35.4	61.9	29.7
T + I	GPT-4	Choice	46.4	73.4	40.2	38.0	67.8	32.4	42.4	69.3	36.8
	GPT-4	SoM	29.6	-	20.3	20.1	-	13.9	27.0	-	23.7
I	-	SeeClick	23.8	-	-	15.3	-	-	16.2	-	-
	-	CogAgent	54.2	-	-	50.0	-	-	54.7	-	-
I	GPT-4o	SeeClick	32.1	-	-	33.1	-	-	33.5	-	-
	GPT-4V	OmniParser	42.4	87.6	39.4	41.0	84.8	36.5	45.5	85.7	42.0
	GPT-4o	UGround	47.7	-	-	46.0	-	-	46.6	-	-
I	AGUVIS-7B		64.2	89.8	60.4	60.7	88.1	54.6	60.4	**89.2**	56.6
	AGUVIS-72B		**69.5**	**90.8**	**64.0**	**62.6**	**88.6**	**56.5**	**63.5**	88.5	**58.2**

3.2 OFFLINE GUI AGENT EVALUATION

Multimodal-Mind2Web. We utilize Multimodal-Mind2Web (Zheng et al., 2024a) for evaluating the offline planning capabilities of GUI agents on websites, which builds on the original Mind2Web (Deng et al., 2023). We compare with previous work including closed LLMs taking text-only (Deng et al., 2023) or SoM as inputs (Zheng et al., 2024a) and recent prue vision-based agent models. Following previous work (Cheng et al., 2024; Gou et al., 2024), AGUVIS only use the GUI screenshot as observation. We report element accuracy (Ele.Acc), Operation F1 (Op.F1), and step success rate (Step SR). As shown in Table 2, AGUVIS consistently achieves superior performance, with a notable improvement in Step SR ($+51.9\%$ averaged), indicating enhanced reasoning capabilities regarding planning.

AndroidControl. We assess the planning performance of GUI agent models on mobile devices using AndroidControl (Li et al., 2024d). Following the setting in Li et al. (2024d), we randomly sample 500 step-actions to create a subset, and we report the step accuracy on out-of-domain (OOD) data within both high-level and low-level tasks. The high-level task setting necessitates that the model plans and executes actions, whereas the low-level task setting requires the model to simply adhere to human-labeled instructions for executing the next-step action. We compare with baselines that take textual accessibility tree or images as GUI observations. Table 3 shows that AGUVIS achieves the best performance under both settings.

Table 3: Step Accuracy of out-of-domain (OOD) data on AndroidControl under high-level tasks and low-level tasks. Best performance is in bold. “Acc.Tree” means the textual accessibility tree of GUI interface as inputs.

Observation	Planner	Grounder	Step Accuracy High-Level	Step Accuracy Low-Level
Acc. Tree	GPT-4-Turbo	Choice	42.1	55.0
	PaLM 2S (Specialized)	Choice	58.5	77.5
Image	GPT-4-Turbo	SeeClick	39.4	47.2
	GPT-4-Turbo	UGround	46.2	58.0
	GPT-4o	SeeClick	41.8	52.8
	GPT-4o	UGround	48.4	62.4
Image	AGUVIS-7B		**61.5**	**80.5**
	AGUVIS-72B		**66.4**	**84.4**

3.3 ONLINE GUI AGENT EVALUATION

Beyond offline planning, we test AGUVIS on real-time interaction benchmarks: Mind2Web-Live (Pan et al., 2024b), AndroidWorld (Rawles et al., 2024a) and MobileMiniWob (Rawles et al., 2024b). We introduce each benchmark below and more details are shown in C.3

Mind2Web-Live. Mind2Web-Live is a dynamic dataset in a real web-based environment derived from the original Mind2Web. The benchmark evaluates whether each required step within a task has been completed and uses the task success rate (Task SR) as the reported metric.

AndroidWorld. AndroidWorld is a benchmark operating on an Android virtual environment, capable of dynamically instantiating with randomly generated parameters to generate unique tasks for automatic evaluation. To assess the pure vision agent models, we follow the instructions in Rawles et al. (2024b), installing a Pixel 6 phone simulator on our computers to serve as the experimental environment. The AndroidWorld benchmark incorporates a fully automated task-level evaluation system that automatically assesses whether a state has successfully completed a designated task.

MobileMiniWob. MobileMiniWob is the instantiation of 92 tasks from MiniWob++ (Zheng et al., 2024b) in AndroidWorld environment. Thus, we adopt the same observation and action space utilized in AndroidWorld and use a real-time evaluation function to determine task success rate.

Table 4: Task Success Rate (SR) and efficiency costs on Mind2Web-Live. USD Efficiency is calculated by dividing the model's total inference cost in USD by the number of successful steps.

Inputs	Planner	Grounder	Task SR	USD Efficiency
HTML	GPT-4-Turbo	Choice	21.1	-
	GPT-4o	Choice	22.1	0.142
	Llama-3.1-405B	Choice	24.0	0.174
	Llama-3.1-70B	Choice	20.2	0.031
	GPT-3.5-turbo	Choice	17.3	0.092
Image	GPT-4-Turbo	UGround	23.1	-
	GPT-4o	UGround	19.2	-
	GPT-4o	AGUVIS-7B	**24.0**	**0.106**
Image	AGUVIS-72B		**27.1**	**0.012**

Figure 2: Comparison of Input Tokens per Step and USD Efficiency in GUI Interaction. The bar chart shows the input tokens required per step during GUI interactions, while the line graph illustrates USD Efficiency for all models.

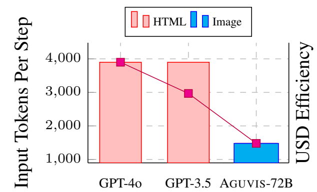


Table 5: Task Success Rates (SR) on AndroidWorld and MobileMiniWob. Best results are in bold.

Input	Planner	Grounding	AndroidWorld$_{SR}$	MobileMiniWob$_{SR}$
AXTree	GPT-4-Turbo	Choice	30.6	59.7
	Gemini 1.5 Pro	Choice	19.4	57.4
Image + AXTree	GPT-4-Turbo	SoM	25.4	67.7
	Gemini 1.5 Pro	SoM	22.8	40.3
Image	GPT-4-Turbo	UGround	31.0	-
	GPT-4o	UGround	32.8	-
	GPT-4o	AGUVIS-7B	**37.1**	**55.0**
Image	AGUVIS-72B		**26.1**	**66.0**

In our online experiments, we explore two distinct configurations. The first configuration employs GPT-4o as the planner, collaborating with our AGUVIS, which serves as the grounder. The second setup utilizes our model in a dual role, acting as both the planner and the grounder. We compare the performance of these configurations with existing SOTA methods that use GPT-4(o) models as planners. Unlike existing methods that rely on Set-of-Mark (SoM) or textual HTML/AXTree information, AGUVIS uses only screenshots as observations and is restricted to `pyautogui` actions $\mathcal{A}$ in all environments: We set the screenshot viewport to a resolution of 1280×720 and disabled all actions based on HTML/AXTree selection.

As shown in Table 4 and Table 5, when incorporating the GPT-4o as planner, AGUVIS-7B outperforms existing work in task success rate across various benchmarks. We further adopt our AGUVIS-72B both as the planner and grounder, achieving the best performance on Mind2Web-Live and MobileMiniWob, which demonstrates the advantage potential of employing purely visual agent models for autonomous GUI interactions. By employing AGUVIS-72B as both the planner and the grounder, we achieve the best performance on Mind2Web-Live and MobileMiniWob. This underscores the advantages of utilizing a unified purely visual agent model for autonomous GUI interactions. Furthermore, we observe that our model demonstrates a significant advantage in terms of efficiency costs compared to both closed-source and open-source models (as discussed below), demonstrating that there is considerable potential for applying purely visual agents in real-world online scenarios.

4 ANALYSIS

4.1 ABLATION

To assess the impact of each stage in the training pipeline of AGUVIS, we conduct ablation experiments. Specifically, we evaluate the performance of the following variants: (a) a model trained without the second stage (planning training), referred to as AGUVIS-G-7B, and (b) a base model,

Table 6: Ablation on AGUVIS-7B on MM-Mind2Web and AndroidControl benchmarks. We report the step success rate.

Settings	Multimodal-Mind2Web			AndroidControl	
	Cross-Task	Cross-Website	Cross-Domain	High-Level	Low-Level
AGUVIS-7B	58.5	55.4	54.8	61.5	80.5
(a) - Stage 2 Training	50.9	45.2	45.3	58.0	75.6
(b) - Stage 1 & 2 Training	50.9	44.9	47.7	59.1	59.2

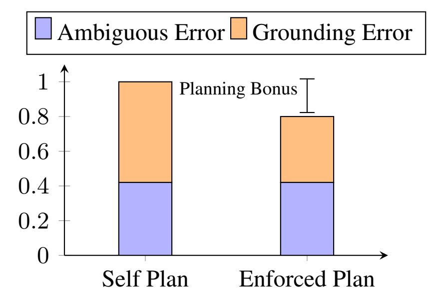
Qwen2-VL (Wang et al., 2024a), without both stages of our specialized training. We report the results of these ablations on two key benchmarks, Multimodal-Mind2Web and AndroidControl, focusing on the step success rate as the evaluation metric (Table 6). The findings show a clear decline in performance when either training stage is omitted. Notably, omitting the second stage (planning and reasoning) has a more significant negative effect on the model's step success rate, indicating that planning training is critical for enhancing the agent's ability to handle complex GUI tasks.

4.2 GENERAZATION ON OTHER VLM BACKBONE

Table 7: Performance of AGUVIS based on LLaVA-OneVision backbone. We report the average score on ScreenSpot and the step success rate of each split in Multimoda-Mind2Web. These results demonstrate that our framework and data recipe are model independent and the planning stage can largely improve the performance of both grounding and planning ability.

Models	ScreenSpot	MM-Mind2Web		
	Average	Task	Website	Domain
Previous SOTA	73.3	39.4	36.5	42.0
AGUVIS$_{\text{OV}}$-G-7B	70.0	-	-	-
AGUVIS$_{\text{OV}}$-7B	81.2	55.3	50.0	50.8

Figure 3: Error analysis on Screenspot dataset under the self-plan setting.



In our experiments, we also implement a version of AGUVIS based on another typical VLM LLaVA-OneVision (Li et al., 2024a), named AGUVIS$_{\text{OV}}$-7B, to explore the generalizability of AGUVIS. We report the average score of ScreenSpot and the step success rate of Multimoda-Mind2web. These results demonstrate that our framework and data recipe are model-independent and the planning training stage can largely improve the performance of both grounding and planning ability.

4.3 EFFICIENCY

We investigate the efficiency costs of AGUVIS on the online planning benchmark Mind2Web-Live. Following Pan et al. (2024a), we adopt the USD Efficiency Score to evaluate the efficiency of our model in completing tasks. Specifically, this Score is calculated as the total dollar cost of tokens used by the model to complete all tasks in the dataset divided by the total Success Steps. A lower USD Efficiency Score indicates that the model requires fewer USD to complete a successful step. In addition to the USD Efficiency Score, we calculated the number of tokens consumed during the completion of the whole dataset divided by the total number of steps taken by agent models. This reflects the average number of tokens consumed per step.

As shown in Figure 2, AGUVIS significantly reduces the efficiency costs by reducing 93% USD costs and 70% input tokens per step compared to GPT-4o, which indicates considerable potential for applying purely visual agents in practical applications.

4.4 ERROR ANALYSIS

We conduct an error analysis of AGUVIS on 50 samples from the ScreenSpot dataset under the self-plan setting to understand the impact of planning on performance. As shown in Figure 3, our findings reveal that 40% of errors are due to ambiguous instructions that could refer to multiple grounding targets, while the remaining 60% are grounding errors. We observe that in these error cases, the model tends to perform direct grounding action rather than planning explicitly before acting. Notably, when we enforce planning by prompting the agent model to generate low-level instructions before execution, it resolved 20% of the grounding errors. This suggests that while the agent model possesses strong grounding capabilities, there remains significant potential for improvement in effectively leveraging planning and reasoning. These insights highlight opportunities for future work, including improving instruction clarity through the agent model itself, developing adaptive planning mechanisms, and refining training data to include more diverse planning scenarios. Addressing these aspects could further enhance our GUI agent model's robustness on various tasks and environments.

5 Related Work

5.1 Benchmarks and datasets for GUI agent

Recent advancements in autonomous GUI agents have led to the development of numerous benchmarks and datasets. Web-based benchmarks such as Mind2Web (Deng et al., 2023), WebArena (Zhou et al., 2024; Koh et al., 2024a), WebLINX (Lù et al., 2024), WorkArena (Drouin et al., 2024) and WebCanvas (Pan et al., 2024b) focus on evaluating agents' performance in web environments. For desktop and mobile platforms, datasets like OSWorld (Xie et al., 2024), WindowsAgentArena (Bonatti et al., 2024), AitW (Rawles et al., 2024b), AitZ (Zhang et al., 2024b), AMEX (Chai et al., 2024), GUI-Odyssey (Lu et al., 2024) and AndroidControl (Li et al., 2024b) have been introduced to assess agents' capabilities across different operating systems and device types. Cross-platform datasets such as ScreenSpot (Cheng et al., 2024), OmniACT (Kapoor et al., 2024), GUICourse (Chen et al., 2024a), and CRAB (Xu et al., 2024a) aim to provide comprehensive evaluation frameworks spanning multiple devices and interfaces. Evaluations on specialized applications have also emerged, such as WonderBread (Wornow et al., 2024)'s focus on business process management tasks and Spider-2V (Cao et al., 2024)'s on data science and engineering workflows. In this work, we extensively test benchmarks under both online and offline task settings to thoroughly evaluate and demonstrate the model's planning and grounding capabilities.

5.2 Models and approaches for GUI agent

In parallel with dataset development, significant progress has been made in creating more capable GUI agents. Models like WebGPT (Nakano et al., 2021), Lemur (Xu et al., 2024b), AgentLumos (Yin et al., 2024), CogAgent (Hong et al., 2024), AutoWebGLM (Lai et al., 2024) and xLAM (Zhang et al., 2024a) have demonstrated improved performance in web navigation tasks. Auto-GUI (Zhang & Zhang, 2024), AppAgent (Zhang et al., 2023), and ScreenAgent (Niu et al., 2024) propose novel approaches for direct GUI interaction without relying on application-specific APIs. SearchAgent (Koh et al., 2024b) introduces an inference-time search algorithm to enhance multi-step reasoning and planning in interactive web environments. These advancements collectively contribute to developing more sophisticated and capable GUI agents, pushing the boundaries of what's possible in automated task completion across various digital platforms.

6 Conclusion

In this paper, we introduced Aguvis, a unified pure vision-based framework for building autonomous GUI agents that operate across diverse platforms. By only leveraging vision-based observations and a consistent action space, Aguvis addresses the key challenges of GUI grounding, planning, and reasoning. Our framework unifies and augments existing datasets, enabling more effective cross-platform generalization while reducing inference costs. Extensive experiments demonstrate that Aguvis outperforms existing methods in both offline and online GUI tasks, showcasing the first fully autonomous pure vision GUI agent capable of completing real-world tasks without reliance on closed-source models. We will open-source all data, models, and training recipes to facilitate future research in this exciting domain.

REFERENCES

Chongyang Bai, Xiaoxue Zang, Ying Xu, Srinivas Sunkara, Abhinav Rastogi, Jindong Chen, and Blaise Agüera y Arcas. Uibert: Learning generic multimodal representations for UI understanding. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, 2021.

Rogerio Bonatti, Dan Zhao, Francesco Bonacci, Dillon Dupont, Sara Abdali, Yinheng Li, Yadong Lu, Justin Wagle, Kazuhito Koishida, Arthur Fender C. Bucker, Lawrence Jang, and Zack Hui. Windows agent arena: Evaluating multi-modal os agents at scale. 2024. URL `https://api.semanticscholar.org/CorpusID:272600411`.

Ruisheng Cao, Fangyu Lei, Haoyuan Wu, Jixuan Chen, Yeqiao Fu, Hongcheng Gao, Xinzhuang Xiong, Hanchong Zhang, Yuchen Mao, Wenjing Hu, Tianbao Xie, Hongshen Xu, Danyang Zhang, Sida Wang, Ruoxi Sun, Pengcheng Yin, Caiming Xiong, Ansong Ni, Qian Liu, Victor Zhong, Lu Chen, Kai Yu, and Tao Yu. Spider2-v: How far are multimodal agents from automating data science and engineering workflows? *CoRR*, abs/2407.10956, 2024. URL `https://arxiv.org/abs/2407.10956`.

Yuxiang Chai, Siyuan Huang, Yazhe Niu, Han Xiao, Liang Liu, Dingyu Zhang, Peng Gao, Shuai Ren, and Hongsheng Li. Amex: Android multi-annotation expo dataset for mobile gui agents. *arXiv preprint arXiv:2407.17490*, 2024.

Wentong Chen, Junbo Cui, Jinyi Hu, Yujia Qin, Junjie Fang, Yue Zhao, Chongyi Wang, Jun Liu, Guirong Chen, Yupeng Huo, et al. Guicourse: From general vision language models to versatile gui agents. *arXiv preprint arXiv:2406.11317*, 2024a.

Zehui Chen, Kuikun Liu, Qiuchen Wang, Wenwei Zhang, Jiangning Liu, Dahua Lin, Kai Chen, and Feng Zhao. Agent-flan: Designing data and methods of effective agent tuning for large language models. In *Findings of the Association for Computational Linguistics*, 2024b.

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

Mostafa Dehghani, Basil Mustafa, Josip Djolonga, Jonathan Heek, Matthias Minderer, Mathilde Caron, Andreas Steiner, Joan Puigcerver, Robert Geirhos, Ibrahim Alabdulmohsin, Avital Oliver, Piotr Padlewski, Alexey Gritsenko, Mario Lučić, and Neil Houlsby. Patch n' pack: Navit, a vision transformer for any aspect ratio and resolution, 2023. URL `https://arxiv.org/abs/2307.06304`.

Biplab Deka, Zifeng Huang, Chad Franzen, Joshua Hibschman, Daniel Afergan, Yang Li, Jeffrey Nichols, and Ranjitha Kumar. Rico: A mobile app dataset for building data-driven design applications. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, 2017.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samual Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. In *Advances in Neural Information Processing Systems*, 2023.

Alexandre Drouin, Maxime Gasse, Massimo Caccia, Issam H. Laradji, Manuel Del Verme, Tom Marty, Léo Boisvert, Megh Thakkar, Quentin Cappart, David Vazquez, Nicolas Chapados, and Alexandre Lacoste. Workarena: How capable are web agents at solving common knowledge work tasks?, 2024. URL `https://arxiv.org/abs/2403.07718`.

Boyu Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. Uground: Navigating the digital world as humans do: Universal visual grounding for gui agents, 2024. URL `https://github.com/OSU-NLP-Group/UGround/blob/gh-pages/static/papers/UGround_paper.pdf`. Preprint.

Izzeddin Gur, Hiroki Furuta, Austin V. Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. A real-world webagent with planning, long context understanding, and program synthesis. In *International Conference on Learning Representations*, 2024.

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, et al. Cogagent: A visual language model for gui agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14281–14290, 2024.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022.

Raghav Kapoor, Yash Parag Butala, Melisa Russak, Jing Yu Koh, Kiran Kamble, Waseem Alshikh, and Ruslan Salakhutdinov. Omniact: A dataset and benchmark for enabling multimodal generalist autonomous agents for desktop and web. *arXiv preprint arXiv:2402.17553*, 2024.

Geunwoo Kim, Pierre Baldi, and Stephen McAleer. Language models can solve computer tasks. In *Advances in Neural Information Processing Systems*, 2023.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Russ Salakhutdinov, and Daniel Fried. Visualwebarena: Evaluating multimodal agents on realistic visual web tasks. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, 2024a.

Jing Yu Koh, Stephen McAleer, Daniel Fried, and Ruslan Salakhutdinov. Tree search for language model agents. *arXiv preprint arXiv:2407.01476*, 2024b.

Hanyu Lai, Xiao Liu, Iat Long Iong, Shuntian Yao, Yuxuan Chen, Pengbo Shen, Hao Yu, Hanchen Zhang, Xiaohan Zhang, Yuxiao Dong, et al. Autowebglm: Bootstrap and reinforce a large language model-based web navigating agent. *arXiv preprint arXiv:2404.03648*, 2024.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024a.

Wei Li, William Bishop, Alice Li, Chris Rawles, Folawiyo Campbell-Ajala, Divya Tyamagundlu, and Oriana Riva. On the effects of data scale on computer control agents. *arXiv preprint arXiv:2406.03679*, 2024b.

Wei Li, William Bishop, Alice Li, Chris Rawles, Folawiyo Campbell-Ajala, Divya Tyamagundlu, and Oriana Riva. On the effects of data scale on computer control agents, 2024c. URL https://arxiv.org/abs/2406.03679.

Wei Li, William W. Bishop, Alice Li, Christopher Rawles, Folawiyo Campbell-Ajala, Divya Tyamagundlu, and Oriana Riva. On the effects of data scale on computer control agents. *CoRR*, abs/2406.03679, 2024d.

Yang Li, Jiacong He, Xin Zhou, Yuan Zhang, and Jason Baldridge. Mapping natural language instructions to mobile ui action sequences. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020a.

Yang Li, Gang Li, Luheng He, Jingjie Zheng, Hong Li, and Zhiwei Guan. Widget captioning: Generating natural language description for mobile user interface elements. In *Conference on Empirical Methods in Natural Language Processing*, 2020b.

Quanfeng Lu, Wenqi Shao, Zitao Liu, Fanqing Meng, Boxuan Li, Botong Chen, Siyuan Huang, Kaipeng Zhang, Yu Qiao, and Ping Luo. Gui odyssey: A comprehensive dataset for cross-app gui navigation on mobile devices. *arXiv preprint arXiv:2406.08451*, 2024.

Xing Han Lù, Zdeněk Kasner, and Siva Reddy. Weblinx: Real-world website navigation with multi-turn dialogue. *ArXiv*, abs/2402.05930, 2024. URL https://api.semanticscholar.org/CorpusID:267547883.

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

Microsoft. Playwright for python documentation. `https://playwright.dev/python/`, 2024.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

Runliang Niu, Jindong Li, Shiqi Wang, Yali Fu, Xiyu Hu, Xueyuan Leng, He Kong, Yi Chang, and Qi Wang. Screenagent: A vision language model-driven computer control agent, 2024. URL `https://arxiv.org/abs/2402.07945`.

OpenAI. Hello gpt-4o, 2024. URL `https://openai.com/index/hello-gpt-4o`.

Yichen Pan, Dehan Kong, Sida Zhou, Cheng Cui, Yifei Leng, Bing Jiang, Hangyu Liu, Yanyi Shang, Shuyan Zhou, Tongshuang Wu, and Zhengyang Wu. Webcanvas: Benchmarking web agents in online environments. *ArXiv*, abs/2406.12373, 2024a. URL `https://api.semanticscholar.org/CorpusID:270562249`.

Yichen Pan, Dehan Kong, Sida Zhou, Cheng Cui, Yifei Leng, Bing Jiang, Hangyu Liu, Yanyi Shang, Shuyan Zhou, Tongshuang Wu, et al. Webcanvas: Benchmarking web agents in online environments. *arXiv preprint arXiv:2406.12373*, 2024b.

Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William Bishop, Wei Li, Folawiyo Campbell-Ajala, Daniel Toyama, Robert Berry, Divya Tyamagundlu, Timothy Lillicrap, and Oriana Riva. Androidworld: A dynamic benchmarking environment for autonomous agents, 2024a. URL `https://arxiv.org/abs/2405.14573`.

Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy Lillicrap. Androidinthewild: A large-scale dataset for android device control. *Advances in Neural Information Processing Systems*, 36, 2024b.

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024a.

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

Michael Wornow, Avanika Narayan, Ben T Viggiano, Ishan S. Khare, Tathagat Verma, Tibor Thompson, Miguel Angel Fuentes Hernandez, Sudharsan Sundar, Chloe Trujillo, Krrish Chawla, Rongfei Lu, Justin Shen, Divya Nagaraj, Joshua Martinez, Vardhan Agrawal, Althea Hudson, Nigam H. Shah, and Christopher Re. Do multimodal foundation models understand enterprise workflows? a benchmark for business process management tasks. *ArXiv*, abs/2406.13264, 2024. URL `https://api.semanticscholar.org/CorpusID:270620942`.

Jason Wu, Siyan Wang, Siman Shen, Yi-Hao Peng, Jeffrey Nichols, and Jeffrey Bigham. Webui: A dataset for enhancing visual ui understanding with web semantics. *ACM Conference on Human Factors in Computing Systems (CHI)*, 2023.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *arXiv preprint arXiv:2404.07972*, 2024.

Tianqi Xu, Linyao Chen, Dai-Jie Wu, Yanjun Chen, Zecheng Zhang, Xiang Yao, Zhiqiang Xie, Yongchao Chen, Shilong Liu, Bochen Qian, Philip H. S. Torr, Bernard Ghanem, and G. Li. Crab: Cross-environment agent benchmark for multimodal language model agents. *ArXiv*, abs/2407.01511, 2024a. URL `https://api.semanticscholar.org/CorpusID:270869926`.

Yiheng Xu, Hongjin Su, Chen Xing, Boyu Mi, Qian Liu, Weijia Shi, Binyuan Hui, Fan Zhou, Yitao Liu, Tianbao Xie, Zhoujun Cheng, Siheng Zhao, Lingpeng Kong, Bailin Wang, Caiming Xiong, and Tao Yu. Lemur: Harmonizing natural language and code for language agents. In *International Conference on Learning Representations*, 2024b.

Da Yin, Faeze Brahman, Abhilasha Ravichander, Khyathi Raghavi Chandu, Kai-Wei Chang, Yejin Choi, and Bill Yuchen Lin. Agent lumos: Unified and modular training for open-source language agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, 2024.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. Agenttuning: Enabling generalized agent abilities for llms. In *Findings of the Association for Computational Linguistics*. Association for Computational Linguistics, 2024.

China. Xiaoyan Zhang, Zhao Yang, Jiaxuan Liu, Yucheng Han, Xin Chen, Zebiao Huang, Bin Fu, and Gang Yu. Appagent: Multimodal agents as smartphone users. *ArXiv*, abs/2312.13771, 2023. URL `https://api.semanticscholar.org/CorpusID:266435868`.

Jianguo Zhang, Tian Lan, Ming Zhu, Zuxin Liu, Thai Hoang, Shirley Kokane, Weiran Yao, Juntao Tan, Akshara Prabhakar, Haolin Chen, Zhiwei Liu, Yihao Feng, Tulika Awalgaonkar, Rithesh Murthy, Eric Hu, Zeyuan Chen, Ran Xu, Juan Carlos Niebles, Shelby Heinecke, Huan Wang, Silvio Savarese, and Caiming Xiong. xlam: A family of large action models to empower ai agent systems. 2024a. URL `https://api.semanticscholar.org/CorpusID:272424184`.

Jiwen Zhang, Jihao Wu, Yihua Teng, Minghui Liao, Nuo Xu, Xiao Xiao, Zhongyu Wei, and Duyu Tang. Android in the zoo: Chain-of-action-thought for gui agents. *arXiv preprint arXiv:2403.02713*, 2024b.

Zhuosheng Zhang and Aston Zhang. You only look at screens: Multimodal chain-of-action agents. In *Findings of the Association for Computational Linguistics*, 2024.

Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. Gpt-4v(ision) is a generalist web agent, if grounded. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, 2024a.

Longtao Zheng, Rundong Wang, Xinrun Wang, and Bo An. Synapse: Trajectory-as-exemplar prompting with memory for computer control. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024b.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, 2024.

A Details of Action Space in Aguvis

In this section, we introduce our unified action space of our pure vision agent framework Aguvis. As shown in Table 8, we use default standard `pyautogui` actions with pluggable actions as the action space of Aguvis, which ensures the agent model's universality across environments as well as its flexibility in the specific environment.

Table 8: Default standard `pyautogui` actions $\mathcal{A}$ with pluggable actions.

Category	Action Space

Basic Actions	`pyautogui.moveTo(x, y)`
	`pyautogui.click(x, y)`
	`pyautogui.write('text')`
	`pyautogui.press('enter')`
	`pyautogui.hotkey('ctrl', 'c')`
	`pyautogui.scroll(200)`
	`pyautogui.dragTo(x, y)`

Pluggable Actions	`browser.select_option(x, y, value)`
	`mobile.swipe(from, to)`
	`mobile.home()`
	`mobile.back()`
	`mobile.open_app(name)`
	`terminate(status)`
	`answer(text)`

...	...

B DATA CURATION OF THE AUGVIS COLLECTION

B.1 DETAILED DATASET STATISTICS

We present the detailed statistical information of all training datasets utilized in both the grounding and planning & reasoning stages. The statistics are shown in Table 9 and Table 10, respectively.

Table 9: The grounding split of THE AUGVIS COLLECTION. Each example in this split consists of a single-step trajectory.

Data source	Platform	Instruction	#Trajectory
SeeClick (Cheng et al., 2024)	Website	Augmented	271K
GUIEnv (Chen et al., 2024a)	Website	Augmented	328K
GUIAct (Chen et al., 2024a)	Website	Original	67K
WebUI (Wu et al., 2023)	Website	Augmented	57K
Widget Captioning (Li et al., 2020b)	Mobile	Original	101K
RicoSCA (Li et al., 2020a)	Mobile	Original	173K
UI RefExp (Bai et al., 2021)	Mobile	Original	16K
RICO Icon (Deka et al., 2017)	Mobile	Augmented	16K
OmniACT (Kapoor et al., 2024)	Desktop & Website	Original	7K
Total			1.036M

Table 10: The planning & reasoning split of THE AUGVIS COLLECTION.

Data source	Platform	Inner Monologue	Avg. Steps	#Trajectory
MM-Mind2Web (Zheng et al., 2024a)	Website	Generated	7.7	1,009
GUIAct (Chen et al., 2024a)	Website	Generated	6.7	2,482
MiniWoB++ (Zheng et al., 2024b)	Website	Generated	3.6	2,762
AitZ (Zhang et al., 2024b)	Mobile	Original	6.0	1,987
AndroidControl (Li et al., 2024d)	Mobile	Original	5.5	13,594
GUI Odyssey (Lu et al., 2024)	Mobile	Generated	15.3	7,735
AMEX (Chai et al., 2024)	Mobile	Generated	11.9	2,991
AitW (Rawles et al., 2024b)	Mobile	Generated	8.1	2,346
Total				35K

C Evaluation Benchmarks

In this section, we introduce more details of evaluation benchmarks used in our work.

C.1 GUI Grounding Evaluation

ScreenSpot. ScreenSpot (Cheng et al., 2024)is a typical benchmark designed specifically for GUI visual grounding, consisting of 1.2K single-step instructions and coordinates of the target elements. This dataset encompasses a variety of grounding instructions tailored for mobile, desktop, and website platforms, and categorizes element types into text and icons/widgets. The benchmark is assessed under two distinct settings: (1) *Original Instructions*: models perform grounding actions directly following the original instructions; and (2) *Self-plan*: models are required to generate plans in natural language based on the original instructions before executing grounding actions.

C.2 Offline GUI Agent Evaluation

Multimodal-Mind2Web. We utilize Multimodal-Mind2Web (Zheng et al., 2024a) for evaluating the offline planning capabilities of GUI agents on websites, which builds on the original Mind2Web (Deng et al., 2023). We report element accuracy (Ele.Acc), Operation F1 (Op.F1), and step success rate (Step SR).

AndroidControl. Following the setting in Li et al. (2024d), we randomly sample 500 step-actions from AndroidControl full test set to create a subset, and we report the step accuracy on out-of-domain (OOD) data within both high-level and low-level tasks. The high-level task setting necessitates that the model plans and executes actions, whereas the low-level task setting requires the model to simply adhere to human-labeled instructions for executing the next-step action.

C.3 Online GUI Agent Evaluation

Mind2Web-Live. We adopt Mind2Web-Live (Pan et al., 2024b) to evaluate GUI agents' online planning, a derived dynamic data set from Mind2Web, comprising 104 real-time interactive web tasks. It evaluates whether each required step within a task has been successfully completed and uses the task success rate (Task SR) as the reported metric. The original Mind2Web-Live is built with WebCavas (Pan et al., 2024a), which is a text-based agent framework. To better accommodate the unified observation and action space of pure vision models, we utilize BrowserGym (Drouin et al., 2024) as the evaluation environment for online web tasks which provide support for pure vision-based agent models. BrowserGym is a browser testing environment built on the Playwright (Microsoft, 2024) engine. We incorporate all Mind2Web-Live tasks and evaluation into BrowserGym, involving registering all Mind2Web-Live tasks, setting up the entry points for these tasks, and porting the Mind2Web-Live evaluation functions to BrowserGym.

As Mind2Web-Live is a text-based benchmark, we have to adapt its evaluation function to suit our pure vision-based model. To achieve this, we introduce the two modifications following:

- For the Mind2Web-Live benchmark's click verification, we adapt our coordinate-based approach by comparing the ground truth CSS selector's bounding box (when available) with our click coordinates, as we cannot directly identify HTML elements.
- Similarly, for input validation, we retrieve and compare the value of the ground truth input element (if present) with the expected value, circumventing the need for precise HTML element identification based on CSS selectors.

The Mind2Web-Live environment relies on real-world websites, many of which implement detection systems for automated browser testing and reCAPTCHA challenges. These factors created difficulties during evluation on the Mind2Web-Live dataset, resulting in a lower task success rate (Task SR). Specifically, we observed the following websites to have significant issues with automation detection:

- **kohls**. Model using the search functionality on the Kohls website through Playwright directly results in a 502 Bad Gateway error.

- **target**. We are unable to open target's job website using Playwright due to network connection error.
- **united**. We are unable to open united website using Playwright due to network connection error.

In addition to the websites that were consistenly prone to failure, several other sites intermittently blocked our Playwright access during testing. In total, we encountered 18 network errors and 6 reCAPTCHA tasks that the model was unable to complete, preventing our model from scoring on these 24 tasks.

AndroidWorld. AndroidWorld (Rawles et al., 2024b) is a benchmark operating on an Android virtual environment, capable of dynamically instantiating with randomly generated parameters to generate unique tasks for automatic evaluation. It spans 20 real-world applications, encompassing 116 diverse tasks. To assess the pure vision agent models, we follow the instructions in Rawles et al. (2024b), installing a Pixel 6 phone simulator on our computers to serve as the experimental environment. The benchmark incorporates a fully automated task-level evaluation system that automatically assesses whether a state has successfully completed a designated task. The AndroidWorld environment supports optional inputs such as Set-of-Mark (SoM) and textual AXTree information, which most multimodal models currently rely on to complete tasks. However, we solely use raw screenshots as the observation input and restrict the model to coordinate-level actions and basic mobile functions.

MobileMiniWob. MobileMiniWob (Rawles et al., 2024b) is the instantiation of 92 tasks from MiniWob++ (Zheng et al., 2024b) in the AndroidWorld environment. Thus, we adopt the same observation and action space used in AndroidWorld and use a real-time evaluation function to determine task success.