TongUI: Internet-Scale Trajectories from Multimodal Web Tutorials for Generalized GUI Agents

Bofei Zhang[2*], Zirui Shang[1,2*], Zhi Gao[1,2,3*], Wang Zhang[2], Rui Xie[2,4], Xiaojian Ma[2], Tao Yuan[2], Xinxiao Wu[1], Song-Chun Zhu[2,3,5], Qing Li[2†]

[1]Beijing Key Laboratory of Intelligent Information Technology, School of Computer Science & Technology, Beijing Institute of Technology
[2]State Key Laboratory for General Artificial Intelligence, BIGAI
[3]School of Intelligence Science and Technology, Peking University
[4]Shanghai Jiao Tong University]
[5]Department of Automation, Tsinghua University

Abstract

Building Graphical User Interface (GUI) agents is a promising research direction, which simulates human interaction with computers or mobile phones to perform diverse GUI tasks. However, a major challenge in developing generalized GUI agents is the lack of sufficient trajectory data across various operating systems and applications, mainly due to the high cost of manual annotations. In this paper, we propose the TongUI framework that transforms millions of multimodal web tutorials into GUI trajectories for generalized GUI agents. Concretely, we crawl GUI videos and articles from the Internet and process them into GUI agent trajectory data. Based on this, we construct the GUI-Net-1M dataset, which contains 1 million trajectories across five operating systems and over 280 applications. To the best of our knowledge, this is the **largest open-source GUI trajectory dataset**. We develop the TongUI agent by fine-tuning Qwen2.5-VL-3B/7B/32B models on GUI-Net-1M, which shows consistent performance improvements on commonly used grounding and navigation benchmarks, outperforming baseline agents by 10% on multiple benchmarks, showing the effectiveness of the GUI-Net-1M dataset and underscoring the significance of our TongUI framework.

Project — https://computer-use-agents.github.io/tongui/
Code — https://github.com/TongUI-agent/TongUI-agent
Dataset — https://huggingface.co/datasets/Bofeee5675/GUI-Net-1M

Introduction

Graphical User Interface (GUI) agents based on large foundation models are designed to automate tasks on digital devices by emulating human interactions with a variety of operating systems and applications (Nguyen et al. 2024; Wang et al. 2024c; Zhang et al. 2024a). These agents utilize large language models (LLMs) or vision-language models (VLMs) to process visual inputs (screenshots) and textual inputs (accessibility tree and HTML code), and produce corresponding actions, such as clicking buttons, filling forms, and scrolling, to complete GUI tasks (Lee et al. 2023; Hong et al. 2024; You et al. 2024). GUI agents significantly enhance human-computer interaction, providing potential applications to various domains, such as software testing, financial services, office assistance, and industrial automation, improving work efficiency and user experience.

Recently, notable efforts have been made for GUI agents by fine-tuning the foundation models (Lin et al. 2024b; Qin et al. 2025; Xu et al. 2025). However, collecting sufficient trajectory data for fine-tuning is challenging. Most existing approaches rely on either manually annotated interaction trajectories (Liu et al. 2024b) that are high-quality but costly to obtain, or synthetic trajectories generated from large open-source or proprietary LLMs/VLMs, which may lack diversity and accuracy (Qin et al. 2023; Liu et al. 2024c; Gao et al. 2024). Desirable GUI agents need to perform actions across a wide variety of applications, each with unique interaction sequences, while collecting comprehensive and diverse data across different applications and operating systems remains a major challenge. Thus, the lack of large-scale, diverse, and well-structured GUI trajectory data continues to be a key bottleneck in developing robust and generalized GUI agents.

In this paper, we propose the TongUI framework that transforms millions of multimodal web tutorials into GUI trajectories for generalized GUI agents, where 'Tong' means generalization in Chinese. Our key observation is that there are readily available multimodal web tutorials on the Internet about how to control computers and smart mobile phones, offering a wealth of information on interacting with various applications and operating systems. These tutorials, in either videos or articles with screenshot formats, provide detailed step-by-step instructions on interacting with GUI. Compared to the aforementioned manually collected or synthetic data pipeline, online tutorials offer easy accessibility, rich information content, and good quality. Thus, it is a natural idea to convert the diverse multimodal web tutorials into task-solving trajectories for GUI agent tuning.

In doing so, we design the TongUI framework composed

*These authors contributed equally.
†Qing Li is the corresponding author.

of four steps: tutorial crawling, tutorial processing, trajectory generation, and data filtering. In the tutorial crawling step, we write some seed tasks and use LLMs to extend them into a wider collection of tasks. The generated tasks serve as keywords for retrieving content from hosts for online GUI tutorials (such as articles from WikiHow and videos from YouTube). The tutorial processing step aims to extract textual descriptions and screenshots of multimodal tutorials. We first obtain textual descriptions of multimodal tutorials via audio transcription or captioning, through which task queries and plans are produced using LLMs on the obtained textual descriptions. Then, we extract salient frames from videos as screenshots of each step, while the images in articles are directly regarded as screenshots. In the trajectory generation step, we leverage a zero-shot GUI agent to automatically recognize trajectories, including reasoning thoughts and actions between two steps. In the data filtering step, we apply a multi-stage pipeline to ensure data quality and GUI relevance, including duplicate tutorial removal, LLM-based content filtering, and trajectory-level filtering using GUI agents. Based on the TongUI framework, we construct a GUI-Net-1M dataset that contains 1M trajectories across five operating systems with more than 280 applications. As far as we know, GUI-Net-1M is the biggest open-source GUI trajectory dataset. Notably, our dataset contains tutorials from different time periods, where the same application may have evolving layouts and task-solving solutions, helping the model generalize to diverse GUI environments.

Based on GUI-Net-1M, we develop the TongUI agent using Qwen2.5-VL-3B/7B models. We evaluate the TongUI agents on both the offline and online settings, and the results show that the TongUI agent exhibits consistent improvements in grounding and navigation capabilities. The results demonstrate underscores the effectiveness of the TongUI framework and the collected GUI-Net-1M dataset that improve the agents in a low-cost manner without the need for expensive manual annotation.

In summary, our contributions are three-fold: (1) We construct GUI-Net-1M, having 1M trajectories across five operating systems and over 280 applications. As far as we know, it is the biggest open-source GUI trajectory dataset. (2) We propose the TongUI framework that enables GUI agents to automatically learn from rich web resources, leading to better generalization. (3) We develop the TongUI agent by finetuning Qwen2.5-VL models on GUI-Net-1M, achieving improvements on multiple popular benchmarks.

Related Work

GUI Agent

The advancements of LLMs and VLMs accelerate the development of GUI agents. In the early state, only closed-source models are used (*e.g.*, GPT-4, claude 3.5, and ChatGLM). Recently, more and more open-source models are released, such as Show-UI (Lin et al. 2024b), UI-TARS (Qin et al. 2025), and CogVLM (Wang et al. 2023). Grounding and planning are two important capabilities in GUI agents. The grounding capability means whether the GUI could identify correct buttons or regions for operations. Some methods improve the grounding capability by adding more prompts (accessibility trees, HTML, and set-of-mask) (Zhou et al. 2023; Deng et al. 2023; Gur et al. 2023; Zhang et al. 2024b; Lu et al. 2024b) and collecting grounding data for finetuning (Yang et al. 2024). As for the planning capability of GUI agents, existing efforts improve it by prompt engineering (Jia et al. 2024; Agashe et al. 2024), supervised finetuning (Lin et al. 2024b; Xu et al. 2025; Hong et al. 2024; Qin et al. 2025), or reinforcement learning (Lai et al. 2024; Putta et al. 2024). In addition, some efforts focus on evaluating the GUI agents from multiple aspects, including action sequence (Rawles et al. 2024b; He et al. 2024), grounding precision (Cheng et al. 2024; Liu et al. 2024a), trajectory effectiveness (Lin et al. 2024a; Li et al. 2025b), and task completion (Rawles et al. 2024a; Trivedi et al. 2024; Xie et al. 2025). The evaluation performance in turn guides the research of intelligent agents.

Agent Data Collection

Compared with data collection for LLMs or VLMs, collecting high-quality data for agents is more challenging, since agents usually require long trajectories that solve complex tasks in different domains (*e.g.*, GUI, multimodal reasoning, embodied AI). Commonly used schemes include human annotation (Liu et al. 2024b; Wang et al. 2025) and model synthesis (Qin et al. 2023; Liu et al. 2024c; Gao et al. 2024). including the planning, action sequence, and action position of GUI agents (Deng et al. 2023; Rawles et al. 2024b; Chen et al. 2024; Qin et al. 2025; Cheng et al. 2024; Lu et al. 2024a). Note that the recent state-of-the-art method UI-TARS (Qin et al. 2025) has not released its data. In contrast, we have fully released our data.

In addition to the two schemes, considering that there are abundant resources for GUI operations on the Internet, some efforts focus on collecting trajectory data of GUI agents from the Web, to preserve the data quality and reduce costs simultaneously (Ou et al. 2024; Xu et al. 2024). They collect textual tutorials from the Internet, and convert them into trajectories by using LLMs or exploring in simulators. Unlike them, we directly collect multimodal tutorials from the Internet and propose a multimodal tutorial process pipeline to convert them into trajectories without any simulators. Meanwhile, compared with synthetic data in simulators, our data is closer to tasks in the real world with practical purposes, benefiting to the generalization capability of GUI agents. In table 1, we show the comparisons of GUI-Net-1M with existing GUI trajectory datasets, including GUI Odessey (Lu et al. 2024a), AgentTrek (Xu et al. 2024), AndroidControl (Li et al. 2025b), AGUVIS (Xu et al. 2025), LBI (Su et al. 2025), E-ANT (Wang et al. 2024a), ShowUI (Lin et al. 2024b), and AITW (Rawles et al. 2023), where GUI-Net-1M has obvious advantages in data size, platform, and operating system numbers.

Method

Formulation

We formulate GUI tasks as a sequential decision process, and we adopt ReAct (Yao et al. 2023) as our agent

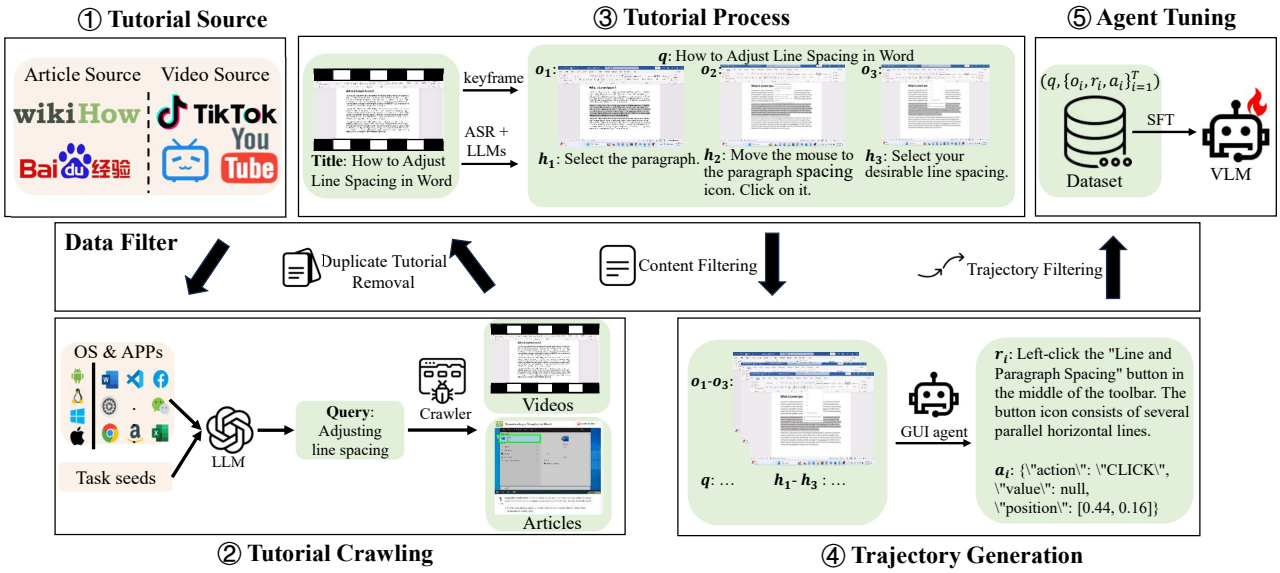

Figure 1: Illustration of the TongUI framework.

Datasets	Size	Platform	OS
GUI Odyssey	7K	W+M	A
AgentTrek	10.4K	W	A+W+L+M+I
AndroidControl	15.3K	W+M	A
AGUVIS	35K	W+M	A+W+L
LBI	42.6K	D+W	L
E-ANT	49K	M	A
ShowUI	137K	W+M	W+L+I
AITW	715K	W+M	A
GUI-Net-1M	1M	D+W+M	A+W+L+M+I

Table 1: Comparison of GUI-Net-1M with other GUI trajectory datasets. For platform, 'D', 'W', and 'M' mean the desktop, web, and mobile, respectively. For operating systems (OS), 'A, 'W', 'L', 'M', and 'I' denote Android, Windows, Linux, MacOS, and iOS, respectively.

framework. At one time step i, given the observation of previous n steps $o_{i-n}, \cdots, o_{i-1}, o_i$ (*e.g.*, a screenshot of GUI), the thoughts r and actions a of previous n steps $(r_{i-n}, a_{i-n}, \cdots, r_{i-1}, a_{i-1})$, and the query q, the agent generates a new thought r_i and an executable action a_i from the action space, such as clicking on a specific UI element, entering text, or scrolling through the interface. Then, executing the action a_i leads to a new observation o_{i+1} (*e.g.*, an updated screenshot). This interaction loop continues, with the agent repeatedly observing the environment, selecting actions, and receiving updated observations until either a termination condition is met (*e.g.*, task completes or fails) or the maximum number of steps is reached. We parameterize the GUI agent using a vision-language model M_θ,

$$r_i^\star, a_i^\star = \arg\max_{r_i, a_i} M_\theta(r_i, a_i \mid q, o_{i-n}, r_{i-n}, a_{i-n}, \ldots, o_{i-1}, r_{i-1}, a_{i-1}, o_i). \quad (1)$$

We propose the TongUI framework that aims to tune M_θ by learning from multimodal web tutorials, as shown in fig. 1. Such tutorials are usually formatted as $\{v, e\}$, where v denotes the visual information (such as images or videos), and e denotes the textual information (such as an introduction to a video). Our goal is to collect multimodal tutorials $\{v, e\}$, and convert them into training data $\{q, o_1, r_1, a_1, \cdots, o_T, r_T, a_T\}$ with the query q and the trajectory $(o_1, r_1, a_1, \cdots, o_T, r_T, a_T)$ in T steps.

Tutorial Source

To collect multimodal web tutorials, we carefully select multiple data sources via brainstorming, ensuring that we cover a wide range of applications, operating systems, and task types. Concretely, we choose YouTube, Bilibili, Wikihow, and Baidu Experience, which host much user-generated content on GUI tasks. YouTube and Bilibili have a variety of videos about GUI tasks. For video tutorials, the visual information v is the video itself, and the textual information e includes the title, the brief introduction to the video, and the caption or audio of the video. Wikihow and Baidu Experience are two widely used platforms that provide step-by-step articles with images in each step across applications and operating systems. Here, the visual information v is the image sequence among all steps, and the textual information e includes the title and textual content in the article. In this case, we could get diverse multimodal web tutorials.

Tutorial Crawling

We use a keyword-based search approach. The search keywords are constructed by combining the name of the applications or website with the task content, that is "app/web + task". Here, "app/web" refers to the operating system or application being used (*e.g.*, a mobile app or website), while "task" has a specific objective (*e.g.*, changing font size in Word, or browsing in Chrome). We identify task seeds across multiple applications via brainstorming, and

expanded them as search keywords using LLMs for more diverse and relevant tasks.

With these keywords, we crawl multimodal web tutorials from source websites. We employ platform-specific methods to retrieve tutorials, including subtitles and audio. For YouTube, we utilize the Google API and YouTube's official API to search for videos, and download the relevant content. For Bilibili and TikTok, we use the unofficial API to search for videos and retrieve both video and audio streams for download. For Wikihow and Baidu Experience, articles usually have multiple tags to indicate the main category of the tutorial, such as Windows, Chrome, Word, and *etc.* We crawl text-image tutorials using tags.

Tutorial Process

Textual processing. Given a crawled multimodal tutorial $\{v, e\}$, this process extracts the task q and rough descriptions $\{h_1, \cdots, h_T\}$ in T steps. For video tutorials from Youtube, Tiktok, and Bilibili, the task guidance is usually in the video's audio information without any textual introductions. To solve this issue, we begin by applying the open-source speech recognition model, Whisper (Radford et al. 2022), to transcribe the audio streams into the text information e. For image-text articles on Wikihow and Baidu Experience, we parse websites based on their article structures for the textual information e.

We use LLMs to identify key task-related verbs and nouns, such as "clicking a button", "filling out a form", or "selecting a menu". Then, we leverage both the title and extracted text to classify tutorials into three categories: mobile, desktop, and others. The data with category "others" is discarded. Finally, we use LLMs to extract the task q and summarize this process into rough descriptions $\{h_1, \cdots, h_T\}$ of T steps from the filtered textual information.

Visual processing. This process extracts the observation $\{o_1, \cdots, o_T\}$ in the T steps. For text-image articles, we could directly obtain the image sequences as the observation $\{o_1, \cdots, o_T\}$ from the crawled data. In practice, there are some noisy data in Baidu Experience and Wikihow, which might be diagrams, comics, or natural images rather than GUI screenshots. To address this issue, we prompt GPT-4o-mini to classify each image as a screenshot or not.

For video tutorials, we need to extract key frames that correspond to critical actions, representing meaningful moments in the task-solving process. We observe that the audio transcript of the videos often contains valuable planning information, such as the description of task steps. Therefore, when audio is available, we first segment the video based on the audio transcript timestamps, treating each segment as a potential task step. We then extract key frames from each of these segments. In the absence of audio, we treat the entire video as a single segment and extract key frames throughout the video. In doing so, we adopt the MOG2 algorithm (Zivkovic 2004a,b; Zivkovic and van der Heijden 2006) to detect significant changes.

After textual and visual processing, tutorials from different sources are structured into a task q with text and image pairs of T steps, denoted as $\big(\{o_1, h_1\}, \cdots, \{o_i, h_i\}, \cdots, \{o_T, h_T\}\big)$, where o_i is the image and h_i is the rough description in the i-th step.

Trajectory Generation

This process aims to generate the trajectory $(o_1, r_1, a_1, \cdots, o_T, r_T, a_T)$ in T steps using the task q, and image o_i and rough description h_i in T steps. Concretely, we use a pretrained GUI agent (such as UI-TARS (Qin et al. 2025)) to generate the thought r_i and the action a_i using the same action space defined by the GUI agent. For the thought r_i and action a_i, we feed the observation o_i with h_i as the query to the zero-shot agent. Here, we use h_i as the query for agents instead of using the task q, because q usually contains an abstract goal instead of specific instructions about what to do on the observation. We empirically find that using h_i leads to better performance. In this case, each step of text and image pairs can be written as $\{o_i, r_i, a_i\}$. In practice, the model might fail to generate well-formatted actions. We discard this step and make the next action the beginning of a new trajectory. For example, for a 4-step trajectory which has a failed generation in the second step, we split it into two training trajectories: $(\{o_1, r_1, a_1\})$ and $(\{o_3, r_3, a_3\}, \{o_4, r_4, a_4\})$. Finally, we combine the trajectory with the task q as $(q, \{o_i, r_i, a_i\}_{i=1}^T)$. We collect these trajectories into a GUI-Net-1M dataset that contains 1M trajectories. The data is on five operating systems with more than 280 applications.

Data Filtering

We apply a multi-stage data filtering pipeline for data quality, as shown in fig. 1. **(1) Duplicate Tutorial Removal.** To eliminate redundancy of crawling, we directly remove exact duplicate tutorials based on their unique identifiers (*e.g.*, video IDs and URLs), ensuring uniqueness. **(2) Content Filtering.** After tutorial processing, we employ an LLM to semantically filter out tutorials that are irrelevant to the GUI tasks. Concretely, based on the article content, titles, and audio transcriptions, the LLM evaluates whether the tutorial is about GUI interactions. **(3) Trajectory Filtering.** In the trajectory generation stage, since we use UI-TARS (Qin et al. 2025), if a step is unrelated to GUI interaction, the agent typically predicts the action `wait` or `call_user`. We use this as a signal to discard such observations. Then, we feed the screenshot and trajectory of a task into a Qwen2.5-VL-7B model, and prompt it to filter out low-quality data.

Agent Tuning

Given a data point $(q, \{o_i, r_i, a_i\}_{i=1}^T)$ of T-step, we train a VLM M_θ via supervised fine-tuning (SFT),

$$\min \ \mathbb{E}_{(q,\{o_i,r_i,a_i\}_{i=1}^T)\sim\mathbb{D}}\Big[-\sum_{i=1}^{T} M_\theta(r_i, a_i \mid q, o_{i-n}, r_{i-n}, a_{i-n}, \ldots, o_{i-1}, r_{i-1}, a_{i-1}, o_i)\Big] \tag{2}$$

where $\mathbb{D}$ is the collected GUI-Net-1M dataset, and we sum the loss values of the T steps in the trajectory. After training, we obtain the TongUI agent.

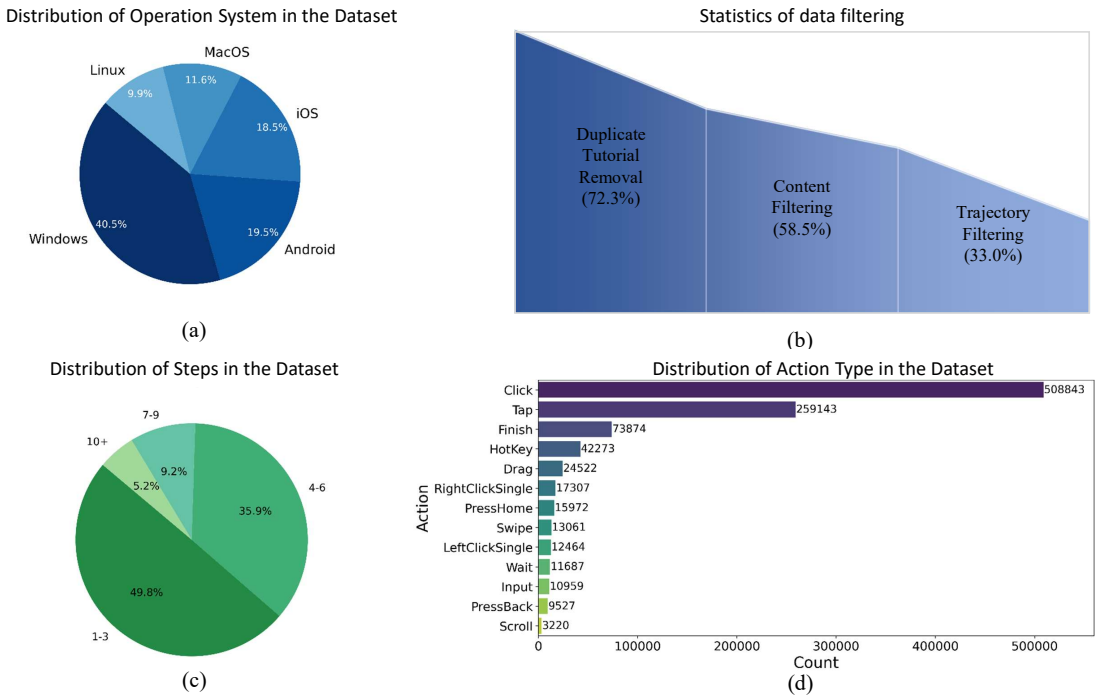

Figure 2: Data statistics on the GUI-Net-1M dataset.

Dataset Statistics

We provide four key statistics to show the diversity of the collected GUI-Net-1M dataset. We show the **operating system distribution** in fig. 2(a). Our dataset covers a diverse range of operating systems, including Windows, Android, iOS, MacOS, and Linux, ensuring a broad representation of GUI interactions across both desktop and mobile environments. We provide statistics on the data filtering flow in fig. 2(b). Throughout the data filtering pipeline, we observe a gradual reduction in data volume, reflecting the progressive refinement of the dataset. After applying duplicate tutorial removal, content filtering, and trajectory filtering, 33.0% of the original data is retained. This three-step filtering process helps to include high-quality and GUI-relevant interactions in the final dataset. We show the **step distribution** in fig. 2(c). Our dataset includes GUI interactions with varying step lengths, ranging from single-step actions to 9-step tasks. The distribution shows that shorter tasks (1-3 steps) are more frequent, while longer tasks gradually decrease in numbers. The higher proportion of trajectories of 1-3 steps is partly due to the trajectory splitting strategy described in Section 3.5. The **action distribution** is shown in fig. 2(d). Click and Tap are the most frequent, reflecting their central role in both desktop and mobile interactions.

Experiments

Setting

Evaluation. We use Qwen2.5-VL-3B/7B/32B as our base VLM model. We evaluate the TongUI agent on offline benchmarks: ScreenSpot (Cheng et al. 2024), ScreenSpot-V2 (Wu et al. 2024), ScreenSpot-Pro (Li et al. 2025a), AITW (Rawles et al. 2023), AndroidControl (Li et al. 2025b), Mind2Web (Deng et al. 2023), and UI-Vision (Nayak et al. 2025). We also evaluate the TongUI agent on an online benchmark: MiniWob (Shi et al. 2017), following the same setting in SeeClick(Cheng et al. 2024).

Compared Methods. We mainly compare the TongUI agent with open-source methods, including ShowUI (Lin et al. 2024b), SeeClick (Cheng et al. 2024), OS-Atlas (Wu et al. 2024), CogAgent (Hong et al. 2024), AGUVIS (Xu et al. 2025), AgentTrek (Xu et al. 2024), *etc*. We also compare TongUI with UI-TARS (Qin et al. 2025) that only releases models without training details and training data. In contrast, we have fully open-sourced our data, code, and models.

Main Results

Grounding In table 2, we show the zero-shot grounding performance of TongUI on ScreenSpot, ScreenSpot-V2, and ScreenSpot-pro. Similar to previous works (Lin et al. 2024b), grounding on icons is much harder than grounding on text. The collected data leads to significant improvements on the baseline Qwen2.5-VL model. Compared with ShowUI, our method has about 5 % - 20 % improvements. TongUI has a comparable performance to UI-TARS, but UI-TARS only releases the models without training details and data. These results demonstrate that GUI-Net-1M can indeed improve the grounding capability of GUI agents.

Offline Navigation We evaluate the offline navigation capability of GUI agents on the AITW, AndroidControl, Mind2Web, and UI-Vision datasets, and results are shown in table 3, table 4, table 6, and table 5, respectively. No matter whether on a small model (3B) or a big model (32B),

Agent Model	ScreenSpot Avg			ScreenSpot-V2 Avg			ScreenSpot-Pro Avg		
	Text	Icon	Avg	Text	Icon	Avg	Text	Icon	Avg
GPT-4o	17.3	18.8	18.3	-	-	-	1.3	0.0	0.8
SeeClick-9B (Cheng et al. 2024)	68.6	38.2	53.4	67.9	37.5	55.1	1.8	0.0	1.1
OS-Atlas-4B (Wu et al. 2024)	86.1	62.1	76.8	81.9	56.4	71.9	5.0	1.7	3.7
OS-Atlas-7B (Wu et al. 2024)	92.2	75.1	85.1	92.2	72.2	84.1	28.1	4.0	18.9
ShowUI-2B (Lin et al. 2024b)	83.4	66.7	75.1	-	-	-	10.8	2.6	7.7
CogAgent-18B (Hong et al. 2024)	-	-	-	-	-	-	12.0	0.8	7.7
Aria-UI (Yang et al. 2024)	90.7	71.4	82.4	-	-	-	17.1	2.0	11.3
AGUVIS-7B (Xu et al. 2025)	92.6	73.3	84.4	-	-	-	-	-	-
UGround-7B (Gou et al. 2024)	79.5	68.1	70.4	-	-	-	25.0	2.8	16.5
UI-TARS-2B (Qin et al. 2025)	89.3	73.0	82.3	91.0	75.3	84.7	39.6	8.4	27.7
UI-TARS-7B (Qin et al. 2025)	93.5	84.8	**89.5**	95.3	86.4	91.6	47.8	16.2	35.7
UI-TARS-72B (Qin et al. 2025)	91.1	**85.4**	88.4	92.5	**87.3**	90.3	**50.9**	**17.5**	**38.1**
Qwen2.5-VL-3B †	68.4	40.8	56.5	73.6	50.1	64.2	9.1	3.3	6.9
Qwen2.5-VL-7B †	87.7	66.4	78.6	92.0	72.3	84.0	17.4	4.5	12.5
TongUI-3B	90.9	76.5	83.6	91.6	77.5	85.5	26.4	4.1	18.0
TongUI-7B	91.6	80.4	86.0	93.2	83.0	88.7	35.1	8.0	24.7
TongUI-32B	**94.1**	82.9	88.5	**95.8**	86.7	**92.1**	45.9	12.6	33.1

Table 2: Results on ScreenSpot, and ScreenSpot-V2, and ScreenSpot-Pro. † means the results are reproduced. The best method is marked in bold, and the second-best method is underlined.

Method	*General*	*Single*	*Web Shopping*	*Install*	*Google Apps*	*Average*
PaLM2-CoT (Zhang and Zhang 2024)	–	–	–	–	–	39.6
OmniParser (Lu et al. 2024b)	48.3	57.8	51.6	77.4	52.9	57.7
SeeClick-9.6B (Cheng et al. 2024)	54.0	73.7	57.6	66.4	54.9	59.3
ShowUI-2B (Lin et al. 2024b)	63.9	77.5	66.6	72.5	69.7	70.0
Qwen2.5-VL-3B†	20.7	31.4	17.1	16.3	16.8	20.5
Qwen2.5-VL-7B†	39.4	41.1	35.8	43.2	42.3	40.4
Qwen2.5-VL-3B-ShowUI	66.0	74.4	65.0	74.5	70.3	70.1
TongUI-3B	65.6	77.0	65.8	75.1	**74.5**	71.6
TongUI-7B	**67.6**	**79.9**	**69.1**	**76.3**	73.5	**73.3**
TongUI-32B	64.0	78.4	65.0	74.2	73.5	71.0

Table 3: Results on the AITW. We report results on five splits of AITW and the average scores.

Method	Model	*High*	*Low*
AITW	PaLM 2L	19.5	56.7
SeeClick	GPT-4-Turbo	33.9	54.3
M3A	GPT-4-Turbo	42.1	55.0
ER	PALM-2S	19.5	45.5
ER	PALM 2L	33.0	45.9
ER	GPT-4	32.1	51.7
ER	Gemini 1.5 Pro	24.4	50.2
AGUVIS	AGUVIS-7B	61.5	80.5
AGUVIS	AGUVIS-72B	66.4	84.4
UI-TARS	UI-TARS-2B	68.9	89.3
UI-TARS	UI-TARS-7B	72.5	90.8
TongUI	TongUI-3B	73.3	91.5
TongUI	TongUI-7B	**76.0**	**91.9**

Table 4: Step accuracy on AndroidControl.

Model	*B*asic	*Functional*	*Spatial*
GPT-4o	1.6	1.5	1.0
Geimni-Flash-2.0	0.5	0.4	0.1
Claude-3.7-Sonnet	9.5	7.7	7.6
ShowUI-2B (Lin et al. 2024b)	8.1	7.7	2.1
AriaUI25-3B (Yang et al. 2024)	12.2	14.0	4.0
UGround-7B (Gou et al. 2025)	15.4	17.1	6.3
AGUVIS-7B (Xu et al. 2025)	17.8	18.3	5.1
CogAgent-9B (Hong et al. 2024)	12.0	12.2	2.6
UI-TARS-7B (Qin et al. 2025)	20.1	24.3	8.4
Qwen2.5-VL-7B	1.2	0.8	0.5
TongUI-3B	22.4	17.4	6.5
TongUI-7B	24.4	22.5	7.2
TongUI-32B	**24.5**	**24.8**	**11.3**

Table 5: Results on UI-Vision

using GUI-Net-1M data leads to competitive performance. For example, TongUI-3B achieves better performance compared to ShowUI-2B by 1.6% on AITW and more than 10% on UI-Vision. The larger model, TongUI-7B, gains better performance compared to TongUI-3B, which is consistent with common sense. On the AndroidControl and UI-Vision datasets, TongUI achieves even better performance than UI-TARS. This highlights that the collected data improves the generalization capability of GUI agents. We argue that the reason is that the collected data involves diverse applications and operating systems, improving generalization.

Method	Cross-Task			Cross-Website			Cross-Domain		
	Elem. Acc	*OP. F1*	*Step SR*	*Elem. Acc*	*OP. F1*	*Step SR*	*Elem. Acc*	*OP. F1*	*Step SR*
CogAgent (Hong et al. 2024)	22.4	53.0	17.6	18.4	42.4	13.4	20.6	42.0	15.5
MindAct (Deng et al. 2023)	55.1	75.7	52.0	42.0	65.2	38.9	42.1	66.5	39.6
OmniParser (Lu et al. 2024b)	42.4	87.6	39.4	41.0	84.8	36.5	45.5	85.7	42.0
ShowUI-2B (Lin et al. 2024b)	39.9	88.6	37.2	41.6	83.5	35.1	39.4	86.8	35.2
SeeClick-9.6B (Cheng et al. 2024)	28.3	87.0	25.5	21.4	80.6	16.4	23.2	84.8	20.8
AgentTrek (Xu et al. 2024)	45.5	84.9	40.9	40.8	82.8	35.1	48.6	84.1	42.1
UI-TARS-2B (Qin et al. 2025)	62.3	90.0	56.3	58.5	87.2	50.8	58.8	89.6	52.3
UI-TARS-7B (Qin et al. 2025)	73.1	92.2	67.1	68.2	90.9	61.7	66.6	90.9	60.5
UI-TARS-72B (Qin et al. 2025)	**74.7**	**92.5**	**68.6**	**72.4**	**91.2**	**63.5**	**68.9**	**91.8**	**62.1**
Qwen2.5-VL-3B †	2.5	14.5	0.4	2.7	12.6	1.0	3.3	24.2	1.7
Qwen2.5-VL-7B †	6.2	72.8	5.0	6.3	68.2	4.5	8.4	73.6	7.2
Qwen2.5-VL-3B-ShowUI	43.2	88.7	39.7	41.3	86.7	35.5	45.1	86.1	40.7
TongUI-3B	53.4	89.0	48.8	54.2	86.4	48.1	53.8	88.2	49.5
TongUI-7B	58.1	88.7	53.4	55.6	87.2	49.0	57.6	88.7	52.9
TongUI-32B	57.2	88.1	52.4	57.4	85.8	50.6	59.2	87.8	54.1

Table 6: Results on Mind2Web. We report results on three types of tasks: cross-task, cross-website, and cross-domain. "Elem. Acc" means whether the element is selected correctly, "OP. F1" denotes the F1 score for the predicted action, and "Step SR" counts successful steps.

Model	Finetuned	*Score*
CC-Net(SL) (Humphreys et al. 2022)	✓	23.4
Pix2Act (Shaw et al. 2023)	✓	55.2
AGUVIS-72B (Xu et al. 2025)	✓	66.0
SeeClick-9.6B (Cheng et al. 2024)	✓	67.0
Qwen2-VL-2B (Wang et al. 2024b)	✓	66.8
ShowUI-2B (Lin et al. 2024b)	✓	71.5
Qwen2.5-VL-3B	×	0.3
Qwen2.5-VL-3B-ShowUI	✓	67.7
TongUI-3B	✓	72.7
TongUI-7B	✓	73.9
TongUI-32B	✓	**74.3**

Table 7: Results on MiniWob

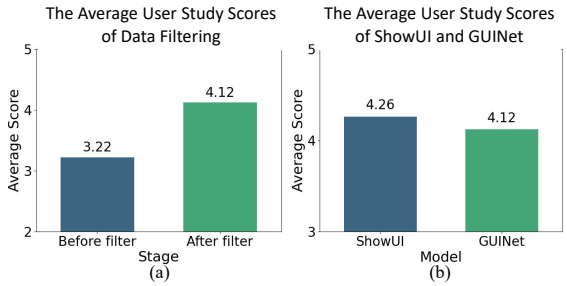

Figure 3: Average scores from humans

Online Navigation We evaluate the online navigation performance on MiniWob, as shown in table 7. Compared with offline navigation, online navigation is more challenging, since it processes dynamic environments, handles unexpected obstacles, and adapts to changes in the navigation path. In this case, TongUI, which learns from multimodal tutorials, achieves improvements again. Considering that the multimodal tutorials are offline data, the improvements confirm the generalization of TongUI.

User Study for Data Quality

To validate the effectiveness of our data filtering strategy and evaluate the data quality of the GUI-Net-1M dataset, we conduct two user studies. In both studies, five participants with substantial experience in GUI agent research (but not involved in our project) are asked to rate the quality of each data point on a scale from 0 (very poor) to 5 (excellent). As shown in fig. 3(a), we evaluate the impact of our Trajectory Filtering step. Specifically, we randomly sample 100 data points before filtering and 100 after filtering, and mix them. As shown in fig. 3(a), the average score increases from 3.22 to 4.12 after filtering, demonstrating that Trajectory Filtering significantly improves data quality by effectively removing low-quality or GUI-irrelevant steps. We also randomly sample 100 data points from the ShowUI dataset for comparison. The results are shown in fig. 3(b), and the ShowUI dataset receives an average score of 4.26. This suggests that data quality in GUI-Net-1M is comparable to that in the ShowUI dataset, while our data leads to further improvements.

Conclusion

In this paper, we present the TongUI framework that converts multimodal web tutorials for an 1M trajectory dataset for GUI agents. By defining suitable and rich sources, we can crawl multimodal tutorials about diverse applications in different operating systems. The proposed tutorial processing method can extract tasks and trajectories from tutorials, and we obtain the GUI-Net-1M dataset. Using this dataset, the tuned Qwen2.5-VL model achieves improvements on multiple commonly used benchmarks, demonstrating the effectiveness of the TongUI framework and the collected GUI-Net-1M dataset. In this method, we have to collect all tutorials and train the model once, which may overlook the potential for continual learning. In the future, we will explore the continual learning capability of GUI agents, enabling the agents to better adapt to new environments.

Acknowledgments

This work was supported by National Science and Technology Major Project (2022ZD0114900), the Natural Science Foundation of China (NSFC) under Grants (No. 62406009), and Opening Project of the State Key Laboratory of General Artificial Intelligence (SKLAGI2024OP01, SKLAGI2024OP14).

References

Agashe, S.; Han, J.; Gan, S.; Yang, J.; Li, A.; and Wang, X. E. 2024. Agent s: An open agentic framework that uses computers like a human. *arXiv preprint arXiv:2410.08164*.

Chen, W.; Cui, J.; Hu, J.; Qin, Y.; Fang, J.; Zhao, Y.; Wang, C.; Liu, J.; Chen, G.; Huo, Y.; et al. 2024. Guicourse: From general vision language models to versatile gui agents. *arXiv preprint arXiv:2406.11317*.

Cheng, K.; Sun, Q.; Chu, Y.; Xu, F.; YanTao, L.; Zhang, J.; and Wu, Z. 2024. SeeClick: Harnessing GUI Grounding for Advanced Visual GUI Agents. 9313–9332.

Deng, X.; Gu, Y.; Zheng, B.; Chen, S.; Stevens, S.; Wang, B.; Sun, H.; and Su, Y. 2023. Mind2web: Towards a generalist agent for the web. 36: 28091–28114.

Gao, Z.; Zhang, B.; Li, P.; Ma, X.; Yuan, T.; Fan, Y.; Wu, Y.; Jia, Y.; Zhu, S.-C.; and Li, Q. 2024. Multi-modal Agent Tuning: Building a VLM-Driven Agent for Efficient Tool Usage. *arXiv preprint arXiv:2412.15606*.

Gou, B.; Wang, R.; Zheng, B.; Xie, Y.; Chang, C.; Shu, Y.; Sun, H.; and Su, Y. 2024. Navigating the digital world as humans do: Universal visual grounding for gui agents. *arXiv preprint arXiv:2410.05243*.

Gou, B.; Wang, R.; Zheng, B.; Xie, Y.; Chang, C.; Shu, Y.; Sun, H.; and Su, Y. 2025. Navigating the Digital World as Humans Do: Universal Visual Grounding for GUI Agents. In *ICLR*.

Gur, I.; Furuta, H.; Huang, A.; Safdari, M.; Matsuo, Y.; Eck, D.; and Faust, A. 2023. A real-world webagent with planning, long context understanding, and program synthesis. *arXiv preprint arXiv:2307.12856*.

He, H.; Yao, W.; Ma, K.; Yu, W.; Dai, Y.; Zhang, H.; Lan, Z.; and Yu, D. 2024. WebVoyager: Building an end-to-end web agent with large multimodal models. *arXiv preprint arXiv:2401.13919*.

Hong, W.; Wang, W.; Lv, Q.; Xu, J.; Yu, W.; Ji, J.; Wang, Y.; Wang, Z.; Dong, Y.; Ding, M.; et al. 2024. Cogagent: A visual language model for gui agents. 14281–14290.

Humphreys, P. C.; Raposo, D.; Pohlen, T.; Thornton, G.; Chhaparia, R.; Muldal, A.; Abramson, J.; Georgiev, P.; Santoro, A.; and Lillicrap, T. 2022. A data-driven approach for learning to control computers. 9466–9482. PMLR.

Jia, C.; Luo, M.; Dang, Z.; Sun, Q.; Xu, F.; Hu, J.; Xie, T.; and Wu, Z. 2024. Agentstore: Scalable integration of heterogeneous agents as specialized generalist computer assistant. *arXiv preprint arXiv:2410.18603*.

Lai, H.; Liu, X.; Iong, I. L.; Yao, S.; Chen, Y.; Shen, P.; Yu, H.; Zhang, H.; Zhang, X.; Dong, Y.; et al. 2024. AutoWebGLM: A Large Language Model-based Web Navigating Agent. 5295–5306.

Lee, S.; Choi, J.; Lee, J.; Wasi, M. H.; Choi, H.; Ko, S. Y.; Oh, S.; and Shin, I. 2023. Explore, select, derive, and recall: Augmenting llm with human-like memory for mobile task automation. *arXiv preprint arXiv:2312.03003*.

Li, K.; Meng, Z.; Lin, H.; Luo, Z.; Tian, Y.; Ma, J.; Huang, Z.; and Chua, T.-S. 2025a. Screenspot-pro: Gui grounding for professional high-resolution computer use. *arXiv preprint arXiv:2504.07981*.

Li, W.; Bishop, W. E.; Li, A.; Rawles, C.; Campbell-Ajala, F.; Tyamagundlu, D.; and Riva, O. 2025b. On the effects of data scale on ui control agents. 37: 92130–92154.

Lin, K. Q.; Li, L.; Gao, D.; Wu, Q.; Yan, M.; Yang, Z.; Wang, L.; and Shou, M. Z. 2024a. VideoGUI: A Benchmark for GUI Automation from Instructional Videos. *arXiv preprint arXiv:2406.10227*.

Lin, K. Q.; Li, L.; Gao, D.; Yang, Z.; Wu, S.; Bai, Z.; Lei, W.; Wang, L.; and Shou, M. Z. 2024b. Showui: One vision-language-action model for gui visual agent. *arXiv preprint arXiv:2411.17465*.

Liu, J.; Song, Y.; Lin, B. Y.; Lam, W.; Neubig, G.; Li, Y.; and Yue, X. 2024a. Visualwebbench: How far have multimodal llms evolved in web page understanding and grounding? *arXiv preprint arXiv:2404.05955*.

Liu, X.; Zhang, T.; Gu, Y.; Iong, I. L.; Xu, Y.; Song, X.; Zhang, S.; Lai, H.; Liu, X.; Zhao, H.; et al. 2024b. Visualagentbench: Towards large multimodal models as visual foundation agents. *arXiv preprint arXiv:2408.06327*.

Liu, Z.; Hoang, T.; Zhang, J.; Zhu, M.; Lan, T.; Kokane, S.; Tan, J.; Yao, W.; Liu, Z.; Feng, Y.; et al. 2024c. Apigen: Automated pipeline for generating verifiable and diverse function-calling datasets. *arXiv preprint arXiv:2406.18518*.

Lu, Q.; Shao, W.; Liu, Z.; Meng, F.; Li, B.; Chen, B.; Huang, S.; Zhang, K.; Qiao, Y.; and Luo, P. 2024a. Gui odyssey: A comprehensive dataset for cross-app gui navigation on mobile devices. *arXiv preprint arXiv:2406.08451*.

Lu, Y.; Yang, J.; Shen, Y.; and Awadallah, A. 2024b. Omniparser for pure vision based gui agent. *arXiv preprint arXiv:2408.00203*.

Nayak, S.; Jian, X.; Lin, K. Q.; Rodriguez, J. A.; Kalsi, M.; Awal, R.; Chapados, N.; Özsu, M. T.; Agrawal, A.; Vazquez, D.; et al. 2025. Ui-vision: A desktop-centric gui benchmark for visual perception and interaction. *arXiv preprint arXiv:2503.15661*.

Nguyen, D.; Chen, J.; Wang, Y.; Wu, G.; Park, N.; Hu, Z.; Lyu, H.; Wu, J.; Aponte, R.; Xia, Y.; et al. 2024. Gui agents: A survey. *arXiv preprint arXiv:2412.13501*.

Ou, T.; Xu, F. F.; Madaan, A.; Liu, J.; Lo, R.; Sridhar, A.; Sengupta, S.; Roth, D.; Neubig, G.; and Zhou, S. 2024. Synatra: Turning indirect knowledge into direct demonstrations for digital agents at scale. *arXiv preprint arXiv:2409.15637*.

Putta, P.; Mills, E.; Garg, N.; Motwani, S.; Finn, C.; Garg, D.; and Rafailov, R. 2024. Agent q: Advanced reasoning and learning for autonomous ai agents. *arXiv preprint arXiv:2408.07199*.

Qin, Y.; Liang, S.; Ye, Y.; Zhu, K.; Yan, L.; Lu, Y.; Lin, Y.; Cong, X.; Tang, X.; Qian, B.; et al. 2023. Toolllm: Facilitating large language models to master 16000+ real-world apis. *arXiv preprint arXiv:2307.16789*.

Qin, Y.; Ye, Y.; Fang, J.; Wang, H.; Liang, S.; Tian, S.; Zhang, J.; Li, J.; Li, Y.; Huang, S.; et al. 2025. UI-TARS: Pioneering Automated GUI Interaction with Native Agents. *arXiv preprint arXiv:2501.12326*.

Radford, A.; Kim, J. W.; Xu, T.; Brockman, G.; McLeavey, C.; and Sutskever, I. 2022. Robust Speech Recognition via Large-Scale Weak Supervision. 28492–28518. PMLR.

Rawles, C.; Clinckemaillie, S.; Chang, Y.; Waltz, J.; Lau, G.; Fair, M.; Li, A.; Bishop, W.; Li, W.; Campbell-Ajala, F.; et al. 2024a. Androidworld: A dynamic benchmarking environment for autonomous agents. *arXiv preprint arXiv:2405.14573*.

Rawles, C.; Li, A.; Rodriguez, D.; Riva, O.; and Lillicrap, T. 2023. Androidinthewild: A large-scale dataset for android device control. 36: 59708–59728.

Rawles, C.; Li, A.; Rodriguez, D.; Riva, O.; and Lillicrap, T. 2024b. Androidinthewild: A large-scale dataset for android device control. 36.

Shaw, P.; Joshi, M.; Cohan, J.; Berant, J.; Pasupat, P.; Hu, H.; Khandelwal, U.; Lee, K.; and Toutanova, K. N. 2023. From pixels to ui actions: Learning to follow instructions via graphical user interfaces. 34354–34370.

Shi, T.; Karpathy, A.; Fan, L.; Hernandez, J.; and Liang, P. 2017. World of bits: An open-domain platform for web-based agents. 3135–3144. PMLR.

Su, H.; Sun, R.; Yoon, J.; Yin, P.; Yu, T.; and Arık, S. Ö. 2025. Learn-by-interact: A data-centric framework for self-adaptive agents in realistic environments. *arXiv preprint arXiv:2501.10893*.

Trivedi, H.; Khot, T.; Hartmann, M.; Manku, R.; Dong, V.; Li, E.; Gupta, S.; Sabharwal, A.; and Balasubramanian, N. 2024. AppWorld: A controllable world of apps and people for benchmarking interactive coding agents. *arXiv preprint arXiv:2407.18901*.

Wang, K.; Xia, T.; Gu, Z.; Zhao, Y.; Shen, S.; Meng, C.; Wang, W.; and Xu, K. 2024a. E-ant: A large-scale dataset for efficient automatic gui navigation. *arXiv preprint arXiv:2406.14250*.

Wang, P.; Bai, S.; Tan, S.; Wang, S.; Fan, Z.; Bai, J.; Chen, K.; Liu, X.; Wang, J.; Ge, W.; et al. 2024b. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Wang, S.; Liu, W.; Chen, J.; Zhou, Y.; Gan, W.; Zeng, X.; Che, Y.; Yu, S.; Hao, X.; Shao, K.; et al. 2024c. Gui agents with foundation models: A comprehensive survey. *arXiv preprint arXiv:2411.04890*.

Wang, W.; Lv, Q.; Yu, W.; Hong, W.; Qi, J.; Wang, Y.; Ji, J.; Yang, Z.; Zhao, L.; Song, X.; Xu, J.; Xu, B.; Li, J.; Dong, Y.; Ding, M.; and Tang, J. 2023. CogVLM: Visual Expert for Pretrained Language Models. arXiv:2311.03079.

Wang, X.; Wang, B.; Lu, D.; Yang, J.; Xie, T.; Wang, J.; Deng, J.; Guo, X.; Xu, Y.; Wu, C. H.; Shen, Z.; Li, Z.; Li, R.; Li, X.; Chen, J.; Zheng, B.; Li, P.; Lei, F.; Cao, R.; Fu, Y.; Shin, D.; Shin, M.; Hu, J.; Wang, Y.; Chen, J.; Ye, Y.; Zhang, D.; Du, D.; Hu, H.; Chen, H.; Zhou, Z.; Wang, Y.; Wang, H.; Yang, D.; Zhong, V.; Sung, F.; Charles, Y.; Yang, Z.; and Yu, T. 2025. OpenCUA: Open Foundations for Computer-Use Agents. arXiv:2508.09123.

Wu, Z.; Wu, Z.; Xu, F.; Wang, Y.; Sun, Q.; Jia, C.; Cheng, K.; Ding, Z.; Chen, L.; Liang, P. P.; et al. 2024. Os-atlas: A foundation action model for generalist gui agents. *arXiv preprint arXiv:2410.23218*.

Xie, T.; Zhang, D.; Chen, J.; Li, X.; Zhao, S.; Cao, R.; Toh, J. H.; Cheng, Z.; Shin, D.; Lei, F.; et al. 2025. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. 37: 52040–52094.

Xu, Y.; Lu, D.; Shen, Z.; Wang, J.; Wang, Z.; Mao, Y.; Xiong, C.; and Yu, T. 2024. AgentTrek: Agent Trajectory Synthesis via Guiding Replay with Web Tutorials. *arXiv preprint arXiv:2412.09605*.

Xu, Y.; Wang, Z.; Wang, J.; Lu, D.; Xie, T.; Saha, A.; Sahoo, D.; Yu, T.; and Xiong, C. 2025. Aguvis: Unified Pure Vision Agents for Autonomous GUI Interaction.

Yang, Y.; Wang, Y.; Li, D.; Luo, Z.; Chen, B.; Huang, C.; and Li, J. 2024. Aria-UI: Visual Grounding for GUI Instructions. *arXiv preprint arXiv:2412.16256*.

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; and Cao, Y. 2023. React: Synergizing reasoning and acting in language models.

You, K.; Zhang, H.; Schoop, E.; Weers, F.; Swearngin, A.; Nichols, J.; Yang, Y.; and Gan, Z. 2024. Ferret-ui: Grounded mobile ui understanding with multimodal llms. 240–255. Springer.

Zhang, C.; He, S.; Qian, J.; Li, B.; Li, L.; Qin, S.; Kang, Y.; Ma, M.; Lin, Q.; Rajmohan, S.; et al. 2024a. Large language model-brained gui agents: A survey. *arXiv preprint arXiv:2411.18279*.

Zhang, C.; Li, L.; He, S.; Zhang, X.; Qiao, B.; Qin, S.; Ma, M.; Kang, Y.; Lin, Q.; Rajmohan, S.; et al. 2024b. Ufo: A ui-focused agent for windows os interaction. *arXiv preprint arXiv:2402.07939*.

Zhang, Z.; and Zhang, A. 2024. You Only Look at Screens: Multimodal Chain-of-Action Agents. 3132–3149.

Zhou, S.; Xu, F. F.; Zhu, H.; Zhou, X.; Lo, R.; Sridhar, A.; Cheng, X.; Ou, T.; Bisk, Y.; Fried, D.; et al. 2023. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*.

Zivkovic, Z. 2004a. Improved adaptive Gaussian mixture model for background subtraction. In *Proceedings of the 17th International Conference on Pattern Recognition (ICPR)*, volume 2, 28–31.

Zivkovic, Z. 2004b. Recursive unsupervised learning of finite mixture models. 26(5): 651–656.

Zivkovic, Z.; and van der Heijden, F. 2006. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognition Letters*, 27(7): 773–780.