Navigating the Digital World as Humans Do: UNIVERSAL VISUAL GROUNDING FOR GUI AGENTS

像人类一样导航数字世界: 面向 GUI 代理的通用视觉定位

Boyu Gou [1] Ruohan Wang [1] Boyuan Zheng [1] Yanan Xie [2] Cheng Chang [2] Yiheng Shu [1] Huan Sun [1] Yu Su [1]

勾博宇 [1] 王若涵 [1] 郑博远 [1] 谢雅楠 [2] 常成 [2] 舒一恒 [1] 孙欢 [1] 苏宇 [1]

[1] The Ohio State University [2] Orby AI

[1] 俄亥俄州立大学 [2] Orby AI

{gou.43, sun.397, su.809}@osu.edu, yanan@orby.ai

{gou.43, sun.397, su.809}@osu.edu, yanan@orby.ai

https://osu-nlp-group.github.io/UGround/

ABSTRACT

摘要

Multimodal large language models (MLLMs) are transforming the capabilities of graphical user interface (GUI) agents, facilitating their transition from controlled simulations to complex, real-world applications across various platforms. However, the effectiveness of these agents hinges on the robustness of their grounding capability. Current GUI agents predominantly utilize text-based representations such as HTML or accessibility trees, which, despite their utility, often introduce noise, incompleteness, and increased computational overhead. In this paper, we advocate a human-like embodiment for GUI agents that perceive the environment entirely visually and directly perform pixel-level operations on the GUI. The key is visual grounding models that can accurately map diverse referring expressions of GUI elements to their coordinates on the GUI across different platforms. We show that a simple recipe, which includes web-based synthetic data and slight adaptation of the LLaVA architecture, is surprisingly effective for training such visual grounding models. We collect the largest dataset for GUI visual grounding so far, containing 10M GUI elements and their referring expressions over 1.3M screenshots, and use it to train UGround, a strong universal visual grounding model for GUI agents. Empirical results on six benchmarks spanning three categories (grounding, offline agent, and online agent) show that 1) UGround substantially outperforms existing visual grounding models for GUI agents, by up to 20% absolute, and 2) agents with UGround outperform state-of-the-art agents, despite the fact that existing agents use additional text-based input while ours only uses visual perception. These results provide strong support for the feasibility and promise of GUI agents that navigate the digital world as humans do.

多模态大语言模型 (MLLMs) 正在改变图形用户界面 (GUI) 代理的能力，促进其从受控模拟向跨平台的复杂现实应用转变。然而，这些代理的有效性依赖于其定位能力的鲁棒性。目前的 GUI 代理主要利用基于文本的表示，如 HTML 或辅助功能树，尽管有用，但常常引入噪声、不完整性和增加计算开销。本文主张为 GUI 代理赋予类人化的体现，使其完全通过视觉感知环境，并直接在 GUI 上执行像素级操作。关键在于视觉定位模型，能够准确地将 GUI 元素的多样化指称表达映射到不同平台 GUI 上的坐标。我们展示了一种简单方案，包括基于网络的合成数据和对 LLaVA 架构的轻微调整，在训练此类视觉定位模型上效果出乎意料地好。我们收集了迄今为止最大的 GUI 视觉定位数据集，包含 1,300,000 张截图中 1,000 万个 GUI 元素及其指称表达，并用其训练了 UGround，一种强大的通用 GUI 视觉定位模型。六个涵盖三类任务 (定位、离线代理和在线代理) 的基准测试的实证结果表明:1)UGround 在 GUI 代理视觉定位模型中表现显著优于现有模型，最高提升 20 个百分点；2) 搭载 UGround 的代理即使仅使用视觉感知，也优于使用额外文本输入的最先进代理。这些结果有力支持了类人方式导航数字世界的 GUI 代理的可行性和前景。

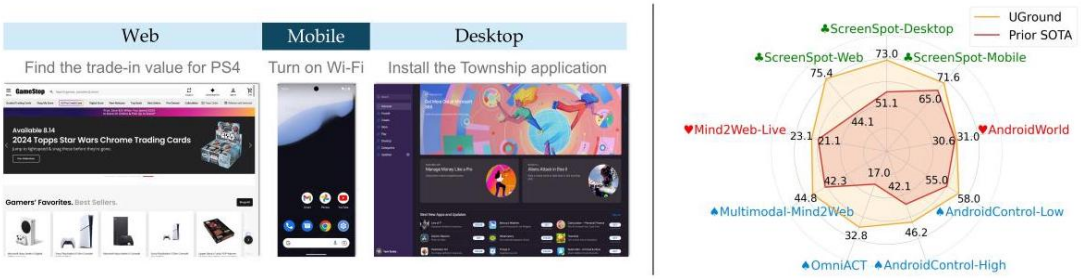


Figure 1: Examples of agent tasks across platforms and performance on GUI grounding (ScreenSpot), offline agent (♠: Multimodal-Mind2Web, AndroidControl, and OmniACT), and online agent benchmarks (♥: Mind2Web-Live and AndroidWorld) when using GPT-4 as the planner.

图 1: 跨平台代理任务示例及使用 GPT-4 作为规划器时在 GUI 定位 (ScreenSpot)、离线代理 (♠:Multimodal-Mind2Web、AndroidControl 和 OmniACT) 和在线代理基准 (♥:Mind2Web-Live 和 AndroidWorld) 上的性能表现。

1 INTRODUCTION

1 引言

GUI (graphical user interface) agents, which are autonomous agents acting in the digital world via operating on GUIs, have been rapidly co-evolving with large language models (LLMs). On the

GUI(图形用户界面) 代理是通过操作 GUI 在数字世界中自主行动的代理, 正与大型语言模型 (LLMs) 快速协同进化。在

one hand, the general multimedia understanding and generation capabilities of (multimodal) LLMs empower GUI agents to generalize beyond simple simulated settings (Shi et al. 2017; Humphreys et al. 2022) to diverse and complex real-world environments, including the web (Deng et al. 2023) Zhou et al. 2024; Yao et al. 2022), desktop (Xie et al. 2024; Wu et al. 2024) and mobile operating systems (Rawles et al. 2023; Yan et al. 2023; Rawles et al. 2024). On the other hand, GUI agents have become an important testbed for LLMs, providing both

the necessary breadth and depth for driving continued development as well as a pathway to many commercially viable automation applications.

一方面，(多模态)LLMs 的通用多媒体理解与生成能力使 GUI 代理能够超越简单模拟环境 (Shi 等，2017；Humphreys 等，2022)，推广到多样且复杂的现实环境，包括网页 (Deng 等，2023；Zhou 等，2024；Yao 等，2022)、桌面 (Xie 等，2024；Wu 等，2024) 和移动操作系统 (Rawles 等，2023；Yan 等，2023；Rawles 等，2024)。另一方面，GUI 代理已成为 LLMs 的重要试验场，既提供了推动持续发展的必要广度和深度，也为众多商业自动化应用开辟了路径。

Most humans perceive the digital world visually and act via keyboards, mice, or touchscreens. In principle, the embodiment of a GUI agent should already be complete if it can 1) visually perceive the GUI renderings, and 2) have effectors equivalent to a keyboard for typing and equivalent to a mouse or touchscreen for pixel-level operations like clicking and hovering [1] However, current GUI agents assume more than that. For perception, most current agents rely on reading the underlying text-based representations such as HTML or accessibility (a11y) trees (Deng et al. 2023) Gur et al. 2024, Zhou et al. 2024) Only with the recent advances in multimodal LLMs (MLLMs) does visual perception become broadly viable, but text-based representations are still used jointly (Zheng et al. 2024, Koh et al. 2024; Zhang et al. 2024a). For effectors, most current agents act via selecting from a list of options, e.g., HTML elements (Deng et al. 2023; Zheng et al. 2024) or labeled bounding boxes (He et al. 2024; Zhang et al. 2024a), instead of pixel-level operations directly on the GUI. Obtaining those options in turn often requires access to text-based representations and/or separate models for detecting objects and text (Wang et al. 2024a, Kapoor et al. 2024).

大多数人类通过视觉感知数字世界，并通过键盘、鼠标或触摸屏进行操作。原则上，如果一个 GUI 代理能够 1) 视觉感知 GUI 渲染，2) 拥有相当于键盘的输入装置和相当于鼠标或触摸屏的像素级操作装置 (如点击和悬停)，那么其具身化 (embodiment) 就已基本完成 [1] 然而，目前的 GUI 代理假设的条件更多。就感知而言，大多数现有代理依赖于读取底层的基于文本的表示，如 HTML 或辅助功能 (a11y) 树 (Deng 等，2023；Gur 等，2024；Zhou 等，2024)。只有随着多模态大型语言模型 (MLLMs) 的最新进展，视觉感知才变得广泛可行，但仍然联合使用基于文本的表示 (Zheng 等，2024；Koh 等，2024；Zhang 等，2024a)。就执行器而言，大多数现有代理通过从选项列表中选择操作，例如 HTML 元素 (Deng 等，2023；Zheng 等，2024) 或带标签的边界框 (He 等，2024；Zhang 等，2024a)，而非直接在 GUI 上进行像素级操作。获取这些选项通常需要访问基于文本的表示和/或用于检测对象和文本的独立模型 (Wang 等，2024a；Kapoor 等，2024)。

However, there is no free lunch, and those additional requirements come with their limitations. On the one hand, text-based representations are noisy and incomplete. Full HTML documents contain a considerable amount of irrelevant information. A11y trees are more compact and mainly contain semantic information, but similar to other semantic annotations that rely on voluntary participation, they widely suffer from incomplete and incorrect annotations [3] In contrast, visual renderings, by design, are information-complete and only contain information relevant to users. On the other hand, the additional input increases latency and inference costs. Zheng et al. (2024) found that HTML can consume up to 10 times more tokens to encode than the corresponding visual. Meanwhile, obtaining an ally tree can be time-consuming in itself, especially in desktop or mobile environments. The added latency and cost at every step are further compounded in the long-horizon agent tasks, compromising user experience and practicality.

然而，没有免费的午餐，这些额外的需求也带来了局限性。一方面，基于文本的表示存在噪声且不完整。完整的 HTML 文档包含大量无关信息。辅助功能树更为紧凑，主要包含语义信息，但类似于依赖自愿参与的其他语义注释，它们普遍存在不完整和错误的注释 [3] 相比之下，视觉渲染本质上是信息完整的，仅包含与用户相关的信息。另一方面，额外的输入增加了延迟和推理成本。Zheng 等 (2024) 发现，HTML 编码所需的 token 数量可达到对应视觉编码的 10 倍。同时，获取辅助功能树本身可能耗时，尤其是在桌面或移动环境中。每一步增加的延迟和成本在长时间任务中进一步累积，影响用户体验和实用性。

In this work, we are interested in how far GUI agents with a human-like embodiment, i.e., only visual observation of environments and pixel-level operations, can go. There have been a few attempts (Shaw et al. (2023) Hong et al. (2024) Cheng et al. (2024), but they are rarely adopted in state-of-the-art solutions. We find that a major bottleneck is grounding, i.e., mapping textual plans generated by an (M)LLM to the precise locations on the GUI. There are three desiderata for a GUI agent grounding model: 1) High accuracy. A single grounding error can get an agent stuck and fail the whole task. 2) Strong generalization. It should work on different GUIs: desktop (Windows, Linux, macOS), mobile (Android, iOS), different websites, etc. 3) Flexibility. It should plug and play in different MLLMs instead of being tightly coupled with a certain model. Existing visual grounding methods for GUI agents (Shaw et al. 2023; Hong et al. 2024; Cheng et al. 2024) fail to meet these desiderata, hindering the advances towards GUI agents with human-like embodiment.

本研究关注具有人类般具身化的 GUI 代理，即仅通过视觉观察环境并进行像素级操作，能够达到何种程度。已有少量尝试 (Shaw 等, 2023；Hong 等, 2024；Cheng 等, 2024)，但它们很少被最先进的解决方案采用。我们发现一个主要瓶颈是定位 (grounding)，即将 (多模态) 大型语言模型 ((M)LLM) 生成的文本计划映射到 GUI 上的精确位置。GUI 代理定位模型有三个期望:1) 高准确性。一次定位错误可能导致代理卡住，任务失败。2) 强泛化能力。应适用于不同 GUI: 桌面 (Windows、Linux、macOS)、移动 (Android、iOS)、不同网站等。3) 灵活性。应能即插即用于不同 MLLM，而非与某一模型紧耦合。现有针对 GUI 代理的视觉定位方法 (Shaw 等，2023；Hong 等，2024；Cheng 等，2024) 未能满足这些期望，阻碍了具有人类般具身化 GUI 代理的发展。

The main contributions of this work are three-fold:

本研究的主要贡献有三点:

1. We make careful arguments and a strong case for GUI agents with human-like embodiment that perceive the digital world entirely visually and take pixel-level operations on GUIs, and propose a generic framework, SeeAct-V, for building such agents by adapting from the popular SeeAct framework (Zheng et al. 2024).

1. 我们提出了有力论证，支持具有人类般具身化的 GUI 代理，即完全通过视觉感知数字世界并在 GUI 上进行像素级操作，并基于流行的 SeeAct 框架 (Zheng 等, 2024) 提出了一个通用框架 SeeAct-V，用于构建此类代理。

---

[1] Except for auditory perception, which is beyond the scope of this study.

[1] 除了听觉感知，因其超出本研究范围。

[2] The ally tree is a compact yet informative representation intended for assistive technologies to facilitate people with disabilities, e.g., visual impairment.

[2] 辅助功能树是一种紧凑而信息丰富的表示，旨在为辅助技术服务，帮助残障人士，如视力障碍者。

[3] A 2024 survey over the top one million websites found that 95.9% of the home pages had accessibility conformance errors such as missing alternative text for images or missing form input labels, with an average of 56.8 errors per page (WebAIM 2024).

[3] 2024 年对前一百万个网站的调查发现，95.9% 的首页存在辅助功能符合性错误，如图片缺少替代文本或表单输入标签缺失，平均每页有 56.8 个错误 (WebAIM 2024)。

---

2. We show that a simple recipe, which includes web-based synthetic data and slight adaptation of the LLaVA architecture (Liu et al. 2024c), is surprisingly effective for GUI visual grounding. Using this recipe, we construct and release the largest GUI visual grounding dataset to date, covering 10M GUI elements and their referring expressions over 1.3M GUI screenshots. We also train and release a universal visual grounding model, UGround, on the dataset.

2. 我们展示了一种简单的方法，该方法包括基于网络的合成数据和对 LLaVA 架构 (Liu 等，2024c) 的小幅调整，出人意料地有效于 GUI 视觉定位。利用该方法，我们构建并发布了迄今为止最大的 GUI 视觉定位数据集，涵盖了 130 万张 GUI 截图中的 1000 万个 GUI 元素及其指代表达。我们还在该数据集上训练并发布了一个通用视觉定位模型 UGround。

3. We conduct the most comprehensive evaluation for GUI agents to date, covering six benchmarks spanning three categories (Figure 1): grounding (desktop, mobile, and web), offline agent evaluation (desktop, mobile, and web), and online agent evaluation (mobile and web). The results demonstrate: 1) UGround substantially outperforms existing visual grounding models for GUI agents across the board, by up to 20% absolute. 2) SeeAct-V agents with UGround can achieve at least comparable and often much better performance than state-of-the-art agents that use additional text-based input. These results provide strong support for the feasibility and promises of GUI agents that navigate the digital world as humans do.

3. 我们进行了迄今为止最全面的 GUI 代理评估，涵盖了三个类别的六个基准 (图 1): 定位 (桌面、移动和网页)、离线代理评估 (桌面、移动和网页) 以及在线代理评估 (移动和网页)。结果表明:1)UGround 在所有方面显著优于现有的 GUI 代理视觉定位模型，最高提升达 20 个百分点。2) 结合 UGround 的 SeeAct-V 代理在性能上至少与使用额外文本输入的最先进代理相当，且常常表现更优。这些结果有力支持了 GUI 代理以人类方式导航数字世界的可行性和前景。

2 METHOD

2 方法

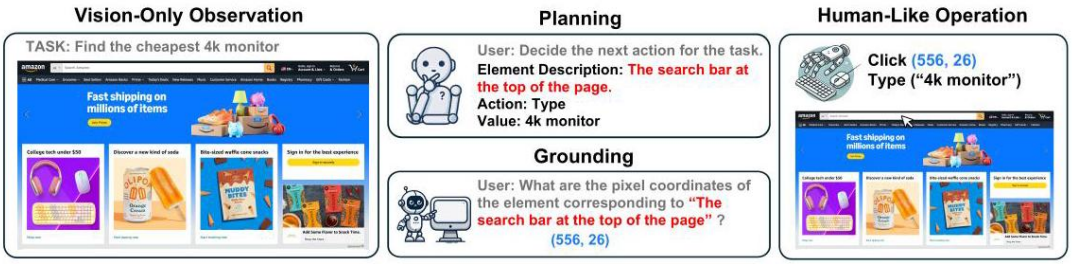


Figure 2: SeeAct-V, which uses screenshots as the only environmental observation (task instructions are input as text), without relying on HTML or a11y trees. It includes an MLLM that generates textual plans and a visual grounding model to map textual plans into coordinates on the screenshot. Note: "Click" is always automatically inserted before "Type."

图 2:SeeAct-V 仅使用截图作为环境观察 (任务指令以文本形式输入)，不依赖 HTML 或辅助功能树 (a11y trees)。它包含一个生成文本计划的多模态大模型 (MLLM) 和一个将文本计划映射到截图坐标的视觉定位模型。注:“点击”操作总是自动插入在“输入”之前。

2.1 OVERVIEW

2.1 概述

We adapt the popular SeeAct framework (Zheng et al. 2024) to one in which agents only take visual observation of the environment and directly conduct pixel-level operations, denoted as SeeAct-V (Figure 2). The original SeeAct has two stages: planning and grounding, both handled by an MLLM. At each step, the MLLM first generates a textual plan, then selects grounding candidates from a short list. The grounding candidates are either filtered HTML elements or labels of Set-of-Mark (SoM; Yang et al. (2023)) annotations on the screenshot, both of which require HTMLs or a11y trees as additional input. In contrast, SeeAct-V only uses screenshots for environmental observation. For grounding, SeeAct-V uses a separate model specialized for visual grounding that directly produces the coordinates on the current screen where the agent should act. We provide our philosophy behind the modular design of SeeAct-V in Appendix B

我们将流行的 SeeAct 框架 (Zheng 等，2024) 调整为仅让代理获取环境的视觉观察并直接执行像素级操作的版本，称为 SeeAct-V(图 2)。原始 SeeAct 包含两个阶段: 规划和定位，均由 MLLM 处理。每一步，MLLM 先生成文本计划，再从候选列表中选择定位目标。定位候选是经过筛选的 HTML 元素或截图上的 Set-of-Mark(SoM；Yang 等，2023) 标注标签，均需 HTML 或辅助功能树作为额外输入。相比之下，SeeAct-V 仅使用截图进行环境观察。定位方面，SeeAct-V 采用专门的视觉定位模型，直接输出代理应操作的当前屏幕坐标。我们在附录 B 中阐述了 SeeAct-V 模块化设计的理念。

A strong visual grounding model therefore becomes the key for making SeeAct-V a compelling framework. Ideally, it should generalize across platforms (e.g., web, desktop, and mobile) and handle diverse ways of referring to GUI elements. Considering the rapid evolution of MLLMs, this grounding model should be easily pluggable into different MLLMs to help ground their plans into different GUI environments. Finally, GUI screenshots can vary drastically in resolution and orientation, therefore the grounding model should handle a wide range of input resolutions. The main technical contribution of this work is a surprisingly simple recipe (incl. data and modeling)

for training such universal visual grounding models. We introduce our simple data synthesis strategy in §2.2 followed by modeling considerations in §2.3 With this simple recipe, we construct the largest training data for GUI grounding to date and train UGround, a strong universal visual grounding model for GUI agents.

因此，强大的视觉定位模型成为使 SeeAct-V 成为有吸引力框架的关键。理想情况下，该模型应能跨平台 (如网页、桌面和移动) 泛化，并处理多样的 GUI 元素指代表达。鉴于多模态大模型 (MLLM) 的快速发展，该定位模型应易于插入不同 MLLM，帮助将其计划映射到不同 GUI 环境。最后，GUI 截图在分辨率和方向上差异巨大，定位模型应能处理广泛的输入分辨率。本工作的主要技术贡献是一套出人意料的简单方案 (包括数据和建模) 用于训练此类通用视觉定位模型。我们在 §2.2 介绍简单的数据合成策略，随后在 §2.3 讨论建模考虑。借助该方案，我们构建了迄今最大的 GUI 定位训练数据，并训练了强大的通用视觉定位模型 UGround。

2.2 DATA CONSTRUCTION

2.2 数据构建

We synthesize a large, high-quality, and diverse set of □screenshot, referring expression, coordinates□ triplets as training data for visual grounding, where we use the center point coordinates of an element as the expected output. Our data synthesis is fully based on webpages. Webpages are ideal for grounding data synthesis because of their dual representation-we can easily get the full HTML, the visual rendering, and fine-grained correspondences between the two (e.g., HTML elements to precise bounding boxes). HTML elements also contain rich metadata such as CSS or accessibility attributes, opening numerous opportunities for synthesizing diverse referring expressions (REs). Finally, since GUI designs share many similarities across platforms, we hypothesize that visual grounding models trained only on web data will generalize to other platforms like desktop and mobile UIs.

我们合成了大量高质量且多样化的 □ 截图，指代表达，坐标 □ 三元组作为视觉定位的训练数据，其中元素的中心点坐标作为期望输出。我们的数据合成完全基于网页。网页因其双重表示特性非常适合定位数据合成——我们可以轻松获取完整 HTML、视觉渲染及两者之间的精细对应关系 (如 HTML 元素到精确边界框)。HTML 元素还包含丰富的元数据，如 CSS 或辅助功能属性，为合成多样的指代表达 (REs) 提供了诸多可能。最后，由于 GUI 设计在各平台间有许多相似之处，我们假设仅用网页数据训练的视觉定位模型能泛化到桌面和移动 UI 等其他平台。

Common RE Types for GUIs. People use diverse ways to refer to GUI elements (Figure 3). Previous visual grounding works (Hong et al. 2024, Cheng et al. 2024) have not sufficiently considered this dimension of diversity. We categorize common REs for GUI elements into three types: 1) Visual REs, i.e., salient visual features like text or image content, element types (e.g., buttons or input fields), shapes, colors, etc. 2) Positional REs, including both absolute (e.g., ”at the top left of the page”) and relative positions (e.g., ”to the right of element X ”) to other elements. Besides straightforward positional information, contextual references (e.g., ”for Item A,” ”under the section X”) are more challenging for grounding because they require understanding both positional relationships and semantic relationships between elements (e.g., a like button is associated with a product). 3) Functional REs, i.e., referring to elements by their main functions (e.g., ”Navigate to Home,” ”Go to My Cart”). Composite types that combine two or more of these types are also common, especially when stronger disambiguation is needed, e.g., ”click the heart button under the Pokémon shirt to add to favorite.”

GUI 的常见指称表达类型。人们使用多种方式来指代 GUI 元素 (见图 3)。以往的视觉定位研究 (Hong 等，2024；Cheng 等，2024) 未充分考虑这一多样性维度。我们将 GUI 元素的常见指称表达 (RE) 分为三类:1) 视觉指称，即显著的视觉特征，如文本或图像内容、元素类型 (例如按钮或输入框)、形状、颜色等；2) 位置指称，包括绝对位置 (例如“页面左上角”) 和相对位置 (例如“位于元素 *X* 右侧”) 两种。除了直接的位置描述，语境引用 (例如“针对项目 A”，“在 X 部分下方”) 更具挑战性，因为它们需要理解元素间的位置关系和语义关系 (例如点赞按钮与产品的关联)；3) 功能指称，即通过元素的主要功能来指代 (例如“导航到首页”，“进入我的购物车”)。组合类型，即结合两种或以上指称方式的情况也很常见，尤其在需要更强区分时，例如“点击宝可梦衬衫下方的心形按钮以添加到收藏”。

1. Red icon labeled ”UNIQLO”

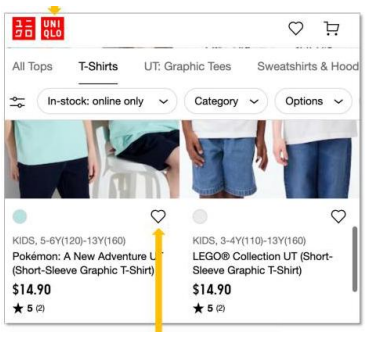
1. 标有“UNIQLO”的红色图标

2. Button at the top left corner

2. 左上角的按钮

3. Navigate back to the homepage

3. 返回主页



1. Hollow heart button

1. 空心心形纽扣

2. Button below the Pokémon shirt

2. 宝可梦衬衫下方的纽扣

3. Favor the Pokémon shirt

3. 偏爱宝可梦衬衫

Figure 3: Examples of visual, positional, and functional REs.

图 3: 视觉、位置和功能指称表达的示例。

Hybrid RE Synthesis from Web. We propose a novel

来自网络的混合关系抽取合成。我们提出了一种新颖的

hybrid synthesis pipeline, orchestrating both carefully curated rules as well as LLMs to generate diverse REs for HTML elements: 1) Primary Descriptors: We extract abundant visual and functional information that are embedded in the attributes of HTML elements. For example, HTML attributes like inner-text and alt provide visual clues (including text content), while accessibility attributes like aria-label reveal more functional aspects of an HTML element. However, HTML attributes are often incomplete. To harvest visual and functional signals beyond HTML attributes, we use an open MLLM, LLaVA-NeXT-13B (Liu et al. 2024b). We input the visual rendering of an HTML element along with its available attributes to the MLLM and prompt it to generate diverse REs. This process often yields composite REs that combine some HTML attributes with visual features (e.g., "hollow heart") or new knowledge from the MLLM (e.g., a blue bird icon represents Twitter). Similar to Lai et al. (2023), we also employ an LLM (Llama-3-8B-Instruct; AI@Meta (2024)) to make these generated REs more concise. We randomly select an HTML attribute (that may contain functional or visual information) or the synthesized description by LLMs as the primary descriptor of an element. 2) Positional Expressions: We curate rules to generate positional REs according to the absolute position of an element in the screenshot as well as its spatial relationship to neighboring elements (e.g., "at the top of the page," "between element A and B"). We also create multiple rules to generate contextual references. For example, we identify elements of certain types in the screenshot (e.g., radio buttons, checkboxes, input fields), and generate REs for them based on their spatial and structural relationship (e.g., hierarchical structure of the DOM tree) to others (e.g., "the input field labeled Birthday").

混合合成流程，结合精心设计的规则和大型语言模型 (LLMs) 生成 HTML 元素的多样化引用表达 (REs):1) 主要描述符: 我们提取嵌入在 HTML 元素属性中的丰富视觉和功能信息。例如，HTML 属性如 inner-text 和 alt 提供视觉线索 (包括文本内容)，而无障碍属性如 aria-label 揭示 HTML 元素的更多功能方面。然而，HTML 属性往往不完整。为了获取超出 HTML 属性的视觉和功能信号，我们使用开放多模态大型语言模型 (MLLM)LLaVA-NeXT-13B(Liu 等，2024b)。我们将 HTML 元素的视觉渲染及其可用属性输入 MLLM，并提示其生成多样化的引用表达。该过程通常产生结合部分 HTML 属性与视觉特征 (如“空心心形”) 或来自 MLLM 的新知识 (如蓝色鸟标志代表 Twitter) 的复合引用表达。类似于 Lai 等 (2023)，我们还采用 LLM(Llama-3-8B-Instruct；AI@Meta(2024)) 使生成的引用表达更简洁。我们随机选择一个可能包含功能或视觉信息的 HTML 属性或由 LLMs 合成的描述作为元素的主要描述符。2) 位置表达: 我们制定规则，根据元素在截图中的绝对位置及其与邻近元素的空间关系 (如“页面顶部”，“元素 A 与 B 之间”) 生成位置引用表达。我们还创建多条规则生成上下文引用。例如，我们识别截图中某些类型的元素 (如单选按钮、复选框、输入字段)，并基于它们与其他元素的空间和结构关系 (如 DOM 树的层级结构) 生成引用表达 (如“标记为生日的输入字段”)。

We collect screenshots (mix of portrait and landscape views in various resolutions) and metadata of web elements (salient HTML attributes, bounding box coordinates) from Common Crawl 4 and then

我们收集了 Common Crawl 4 中的网页元素截图 (包括多种分辨率的纵向和横向视图混合) 及其元数据 (显著的 HTML 属性、边界框坐标)，然后

---

https://commoncrawl.org/

Table 1: Overview of training datasets used for UGround.

表 1:UGround 使用的训练数据集概览。

Dataset	Annotation	#of Elements	#of Screenshots	Platform
Web-Hybrid (Ours)	Rule + LLM	9M	773K	Web
Web-Direct (Ours)	GPT	408K	408K	Web
GUIAct (Chen et al., 2024)	GPT + Human	140K	13K	Web
AndroidControl (L1 et al., 2024b)	Human	47K	47K	Android
Widget Caption (L1 et al., 2020b)	Human	41K	15K	Android
UIBert (Bai et al. 2021	Human	16K	5K	Android
AITZ (Zhang et al. 2024c)	GPT + Human	8K	8K	Android
Total		10M	1.3M	Web + Android

数据集	标注	元素数量	截图数量	平台
Web-混合 (本研究)	规则 + 大型语言模型 (LLM)	9M	773K	网页
Web-直接 (本研究)	GPT	408K	408K	网页
GUIAct(Chen 等，2024)	GPT + 人工	140K	13K	网页
AndroidControl(L1 等，2024b)	人工	47K	47K	安卓
控件标题 (L1 等，2020b)	人工	41K	15K	安卓
UIBert(Bai 等，2021)	人工	16K	5K	安卓
AITZ(Zhang 等，2024c)	GPT + 人工	8K	8K	安卓
总计		10M	1.3M	网页 + 安卓

apply our data synthesis pipeline to get our main training dataset (Web-Hybrid). We leave more details to Appendix E.1

应用我们的数据合成流程以获得主要训练数据集 (Web-Hybrid)。更多细节见附录 E.1。

Supplementary Data. There have been multiple prior efforts on constructing grounding data for Android, so we incorporate the existing datasets as well. We also use GPT-40 to directly synthesize a small set of REs for web elements, with a focus on more open-ended REs (no constraints on the type) and functional REs (Web-Direct). These additions help provide more diverse REs and cover elements in Android, especially those not commonly found on the web (e.g., toggle buttons).

补充数据。此前已有多项关于构建 Android 定位数据的工作，因此我们也整合了现有数据集。我们还使用 GPT-40 直接合成了一小部分针对网页元素的引用表达 (REs)，重点是更开放式的引用表达 (无类型限制) 和功能性引用表达 (Web-Direct)。这些补充有助于提供更多样化的引用表达，并覆盖 Android 中的元素，尤其是那些网页上不常见的元素 (如切换按钮)。

In total, we compile a dataset totaling 10M UI elements, with the majority (90%) from our hybrid synthesis pipeline (Table 1). Elements on the same screenshot are batched to accelerate training.

总计，我们编制了一个包含 1000 万个 UI 元素的数据集，其中大部分 (90%) 来自我们的混合合成流程 (见表 1)。同一截图上的元素被批处理以加速训练。

2.3 MODEL DESIGN

2.3 模型设计

We adopt a widely used open-source model architecture, 7B LLaVA-NeXT (Liu et al. 2024b), as our backbone model for visual grounding. We make a few adaptations to tailor it for GUI grounding.

我们采用广泛使用的开源模型架构 7B LLaVA-NeXT(Liu et al. 2024b) 作为视觉定位的骨干模型，并对其进行了少量调整以适应 GUI 定位任务。

Input-Output Formulation. We always instruct the model to answer ”In the screenshot, what are the pixel element coordinates corresponding to {Description}?” Following recent work in visual grounding (Cheng et al. 2024), we represent the answer in natural language so we can directly use autoregressive decoding. Specifically, we opt for coordinates in the numerical form (e.g., ”(1344, 1344)”) to precisely point to an element without any normalization.

输入-输出形式。我们始终指示模型回答“在截图中，与 {Description} 对应的像素元素坐标是什么?”遵循视觉定位的最新研究 (Cheng et al. 2024)，我们用自然语言表示答案，以便直接使用自回归解码。具体来说，我们选择数值形式的坐标 (例如“(1344, 1344)”)，以精确指向元素，无需归一化。

Image Resolution. GUI screenshots are much larger than typical natural images, often requiring a resolution above 1,000px for legibility. LLaVA (Liu et al. 2024c a) was initially built for 336px images, and was later scaled up to at most 772px via the AnyRes technique (Cheng et al. 2023; Gao et al. 2024; Liu et al. 2024b; Guo et al. 2024; Dong et al. 2024). It resizes and splits a large image into small slices, encodes each slice independently with the vision encoder, and adds a special token at the end of each row to help the language model keep track of the image shape. AnyRes allows easy scaling up of input resolution. However, it is always a trade-off between the diversity of supported resolutions and the speed of training and inference. To strike a balance and avoid meaningless excessive resolutions, we enlarge the allowed input sizes to 36 ViT (Dosovitskiy et al. 2021) slices, and use CLIP@224px (Radford et al. 2021) as the image encoder for more flexible splitting, pushing the maximum supported resolution to 1,344 × 1,344 (landscape) and 896 × 2,016 (portrait). Additionally, we use Vicuna-1.5-7b-16k (Zheng et al., 2023) with 16K context length to handle long visual contexts. Finally, there is a low-resolution image fusion module commonly used in AnyRes. However, we find it ineffective for GUI grounding, as 224px is too small to provide informative global context, so we leave it out from our model. More details are in Appendix F.

图像分辨率。GUI 截图远大于典型自然图像，通常需要超过 1000 像素的分辨率以保证可读性。LLaVA(Liu et al. 2024c a) 最初针对 336 像素图像构建，后通过 AnyRes 技术 (Cheng et al. 2023；Gao et al. 2024；Liu et al. 2024b；Guo et al. 2024；Dong et al. 2024) 扩展至最多 772 像素。该技术将大图像调整大小并切分为小片段，使用视觉编码器独立编码每片段，并在每行末尾添加特殊标记，帮助语言模型跟踪图像形状。AnyRes 便于输入分辨率的扩展，但支持分辨率的多样性与训练推理速度之间存在权衡。为平衡两者并避免无意义的过高分辨率，我们将允许的输入尺寸扩大到 36 个 ViT(Dosovitskiy et al. 2021) 切片，采用 CLIP@224px(Radford et al. 2021) 作为图像编码器以实现更灵活的切分，将最大支持分辨率推至 1344×1344(横屏) 和 896×2016(竖屏)。此外，我们使用支持 16K 上下文长度的 Vicuna-1.5-7b-16k(Zheng et al., 2023) 处理长视觉上下文。最后，AnyRes 中常用的低分辨率图像融合模块在 GUI 定位中效果不佳，因为 224 像素过小，无法提供有用的全局上下文，因此我们未将其纳入模型。更多细节见附录 F。

3 EXPERIMENTS

3 实验

Most existing studies on GUI agents typically evaluate on one or two benchmarks. In contrast, we conduct a much more comprehensive evaluation on GUI agents to show the universality of our method. Our evaluation employs six benchmarks that span all three major platforms (i.e., web, desktop, and mobile) and cover three settings: visual grounding (§3.1), offline agent evaluation on cached environment states (§3.2), and online agent evaluation in live environments (§3.3). The visual grounding setting focuses on the grounding performance of UGround, while the agent settings test the end-to-end effectiveness of the SeeAct-V framework with UGround integrated. On the agent

现有大多数关于 GUI 代理的研究通常只在一两个基准上进行评估。相比之下，我们对 GUI 代理进行了更全面的评估，以展示我们方法的通用性。我们的评估涵盖六个基准，涉及三大主流平台 (即网页、桌面和移动端)，并覆盖三种设置: 视觉定位 (§3.1)、基于缓存环境状态的离线代理评估 (§3.2) 和实时环境中的在线代理评估 (§3.3)。视觉定位设置侧重于 UGround 的定位性能，而代理设置则测试集成 UGround 的 SeeAct-V 框架的端到端效果。在代理

Table 2: Grounding accuracy on ScreenSpot (Standard Setting). Results for GPT-4, CogAgent, and SeeClick are from Cheng et al. (2024).

表 2:ScreenSpot 上的定位准确率 (标准设置)。GPT-4、CogAgent 和 SeeClick 的结果来自 Cheng et al.(2024)。

Grounding Model	Mobile		Desktop		Web		Average
	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-40	20.2	24.9	21.1	23.6	12.2	7.8	18.3
CogAgent (Hong et al., 2024)	67.0	24.0	74.2	20.0	70.4	28.6	47.4
SeeClick (Cheng et al., 2024)	78.0	52.0	72.2	30.0	55.7	32.5	53.4
UGround	82.8	60.3	82.5	63.6	80.4	70.4	73.3
UGround-V1-2B	89.4	72.0	88.7	65.7	81.3	68.9	77.7
UGround-V1-7B	93.0	79.9	93.8	76.4	90.9	84.0	86.3
UGround-V1-72B	94.1	83.4	94.9	85.7	90.4	87.9	89.4

定位模型	移动端		桌面端		网页端		平均值
	文本	图标/小部件	文本	图标/小部件	文本	图标/小部件	
GPT-4	22.6	24.5	20.2	11.8	9.2	8.8	16.2
GPT-40	20.2	24.9	21.1	23.6	12.2	7.8	18.3
CogAgent (Hong et al., 2024)	67.0	24.0	74.2	20.0	70.4	28.6	47.4
SeeClick (Cheng et al., 2024)	78.0	52.0	72.2	30.0	55.7	32.5	53.4
UGround	82.8	60.3	82.5	63.6	80.4	70.4	73.3
UGround-V1-2B	89.4	72.0	88.7	65.7	81.3	68.9	77.7
UGround-V1-7B	93.0	79.9	93.8	76.4	90.9	84.0	86.3
UGround-V1-72B	94.1	83.4	94.9	85.7	90.4	87.9	89.4

Table 3: Grounding accuracy on ScreenSpot (Agent Setting) with planner-generated REs.

表 3: 使用规划器生成的 REs 时，ScreenSpot(代理设置) 上的定位准确率。

Planner	Grounding	Mobile		Desktop		Web		Avg.
		Text	Icon/Widget	Text	Icon/Widget	Text	Icon/Widget	
GPT-4	SeeClick	76.6	55.5	68.0	28.6	40.9	23.3	48.8
	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
	UGround-V1-2B	94.1	77.7	92.8	63.6	90.0	70.9	81.5
	UGround-V1-7B	94.1	79.9	93.3	73.6	89.6	73.3	84.0
	UGround-V1-72B	94.5	79.9	93.8	75.0	88.7	75.2	84.5

规划器	定位	移动端		桌面端		网页端		平均
		文本	图标/小部件	文本	图标/小部件	文本	图标/小部件	
GPT-4	SeeClick	76.6	55.5	68.0	28.6	40.9	23.3	48.8
	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
	UGround-V1-2B	94.1	77.7	92.8	63.6	90.0	70.9	81.5
	UGround-V1-7B	94.1	79.9	93.3	73.6	89.6	73.3	84.0
	UGround-V1-72B	94.5	79.9	93.8	75.0	88.7	75.2	84.5

benchmarks, we compare the vision-only SeeAct-V framework with prior SOTA methods that usually require additional text-based representations (HTML or a11y tree) as input. Within SeeAct-V, we also compare UGround with existing visual grounding models whenever possible. To provide a more comparable baseline to models based on Qwen2-VL (Wang et al. 2024b) or newer base models, we also include the results of UGround-V1 series based on Qwen2-VL in this section, which is trained with the same data mixture.

在基准测试中，我们将仅视觉的 SeeAct-V 框架与通常需要额外基于文本的表示 (HTML 或 a11y 树) 作为输入的先前 SOTA 方法进行比较。在 SeeAct-V 中，我们还尽可能将 UGround 与现有的视觉定位模型进行比较。为了提供一个更具可比性的基线，针对基于 Qwen2-VL(Wang 等，2024b) 或更新基础模型的模型，我们还在本节中包含了基于 Qwen2-VL 训练的 UGround-V1 系列结果，该系列使用相同的数据混合训练。

3.1 GUI VISUAL GROUNDING

3.1 图形用户界面视觉定位

We first evaluate UGround on the ScreenSpot benchmark (Cheng et al., 2024), which is specifically designed for visual grounding on GUIs. The benchmark consists of 1,272 single-step instructions and the corresponding bounding boxes of the target elements across mobile (e.g., iOS and Android), desktop (e.g., macOS and Windows), and web environments. These elements vary between text-based elements, icons (e.g., the trash can icon) and widgets (e.g., to-do lists), representing diverse GUI element types.

我们首先在 ScreenSpot 基准 (Cheng 等，2024) 上评估 UGround，该基准专门为图形用户界面 (GUI) 的视觉定位设计。该基准包含 1272 条单步指令及对应目标元素的边界框，涵盖移动端 (如 iOS 和 Android)、桌面端 (如 macOS 和 Windows) 和网页环境。这些元素包括基于文本的元素、图标 (如垃圾桶图标) 和控件 (如待办事项列表)，代表了多样的 GUI 元素类型。

We evaluate under two settings: 1) Standard Setting. In the standard setting of ScreenSpot, the instructions are written by human annotators with a primary focus on functional description of the target elements, e.g., simply ”close” to refer to the ’X’ button that closes a window or ”set an alarm for 7:40” when the input image shows the iPhone clock app with a list of inactive alarms. 2) Agent Setting. For GUI agents, a grounding model needs to work with a planning model (e.g., an MLLM) and ground the REs it generates, which includes not only functional REs but also visual and positional REs (see §2.2). To provide a more comprehensive evaluation on visual grounding for GUI agents, we input each ScreenSpot example to an MLLM, which acts as a planning model, and asks it to generate diverse REs for the target element. This setting is therefore more representative of the grounding challenges in GUI agents. We mainly compare UGround with SeeClick (Cheng et al., 2024), the state-of-the-art visual grounding model on ScreenSpot, and another visual grounding model CogAgent (Hong et al. 2024). To show the challenge of visual grounding for general-purpose models, we also compare with GPT-4 and GPT-4o.

我们在两种设置下进行评估:1) 标准设置。在 ScreenSpot 的标准设置中，指令由人工标注者编写，主要关注目标元素的功能描述，例如简单地用“关闭”指代关闭窗口的“X”按钮，或当输入图像显示 iPhone 时钟应用及一系列未激活的闹钟时，指令为“设置 7:40 的闹钟”。2) 代理设置。对于 GUI 代理，定位模型需要与规划模型 (如多模态大语言模型，MLLM) 协作，定位其生成的引用表达 (REs)，这些 REs 不仅包括功能性 RE，还包括视觉和位置 RE(见 §2.2)。为了对 GUI 代理的视觉定位进行更全面的评估，我们将每个 ScreenSpot 示例输入 MLLM，作为规划模型，要求其为目标元素生成多样的 REs。因此，该设置更能代表 GUI 代理中定位的挑战。我们主要将 UGround 与 ScreenSpot 上的最先进视觉定位模型 SeeClick(Cheng 等，2024) 及另一视觉定位模型 CogAgent(Hong 等，2024) 进行比较。为展示通用模型在视觉定位上的挑战，我们还与 GPT-4 和 GPT-4o 进行了比较。

Results. As shown in Table 2 and Table 3, UGround outperforms all existing models across all the settings and platforms by a substantial margin, about an absolute improvement of 20% on average under the standard setting and 29% under the agent setting. Interestingly, UGround performs remarkably well on desktop UIs, despite the fact that it is never trained on desktop screenshots (Table 1). Compared with existing models, UGround performs especially well on icons and widgets,

结果。如表 2 和表 3 所示，UGround 在所有设置和平台上均大幅超越现有模型，标准设置下平均绝对提升约 20%，代理设置下提升 29% 。有趣的是，尽管 UGround 从未在桌面截图上训练 (见表 1)，但其在桌面用户界面上的表现依然非常出色。与现有模型相比，UGround 在图标和控件上的表现尤为突出，

Table 4: Element accuracy on Multimodal-Mind2Web. Results by Choice and SoM are from Zheng et al. (2024). The SoM results are on subsets of 30 tasks for each split.

表 4:Multimodal-Mind2Web 上的元素准确率。Choice 和 SoM 的结果来自 Zheng 等 (2024)。SoM 结果基于每个划分的 30 个任务子集。

Input	Planner	Grounding	Cross-Task	Cross-Website	Cross-Domain	Avg.
Image + Text	GPT-4	Choice	46.4	38.0	42.4	42.3
		SoM	29.6	20.1	27.0	25.6
Image (SeeAct-V)	GPT-4	SeeClick	29.7	28.5	30.7	29.6
		UGround	45.1	44.7	44.6	44.8
	GPT-40	SeeClick	32.1	33.1	33.5	32.9
		UGround	47.7	46.0	46.6	46.8
		UGround-V1-2B	48.6	47.6	47.7	48.0
		UGround-V1-7B	50.7	48.1	48.5	49.1

输入	规划器	定位	跨任务	跨网站	跨领域	平均
图像 + 文本	GPT-4	选择	46.4	38.0	42.4	42.3
		SoM	29.6	20.1	27.0	25.6
图像 (SeeAct-V)	GPT-4	SeeClick	29.7	28.5	30.7	29.6
		UGround	45.1	44.7	44.6	44.8
	GPT-40	SeeClick	32.1	33.1	33.5	32.9
		UGround	47.7	46.0	46.6	46.8
		UGround-V1-2B	48.6	47.6	47.7	48.0
		UGround-V1-7B	50.7	48.1	48.5	49.1

which are generally more challenging for grounding because that requires deeper understanding of the contextual (e.g., positional) and semantic (e.g., functional) information. Overall, the strong results on ScreenSpot clearly demonstrates UGround's universal grounding capability across platforms and planners as well as the remarkable effectiveness of our simple data synthesis and modeling recipe.

这类任务通常对定位更具挑战性，因为它需要对上下文 (如位置) 和语义 (如功能) 信息有更深入的理解。总体而言，ScreenSpot 上的优异表现清晰地展示了 UGround 在不同平台和规划器上的通用定位能力，以及我们简单数据合成和建模方案的显著有效性。

3.2 OFFLINE AGENT EVALUATION

3.2 离线代理评估

We discuss the experimental setup for three offline agent evaluation benchmarks followed by result discussion. Concrete examples from each benchmark are given in Appendix D

我们讨论了三个离线代理评估基准的实验设置，随后进行结果讨论。每个基准的具体示例见附录 D。

Web: Multimodal-Mind2Web. We use Multimodal-Mind2Web (Zheng et al. 2024), the multimodal extension of Mind2Web (Deng et al., 2023), for our evaluation on realistic web tasks. The test split consists of 1,013 tasks spanning over 100 different websites. Each task contains a high-level task instruction and a sequence of actions, with a screenshot of the webpage before each action, as the golden trajectory. All the webpages along the golden trajectory are cached to support offline evaluation. The tasks are crowdsourced with a focus on ensuring real-world meaningfulness (i.e., what real users would need on those websites).

Web:Multimodal-Mind2Web。我们使用 Multimodal-Mind2Web(Zheng 等, 2024), 这是 Mind2Web(Deng 等，2023) 的多模态扩展，用于评估现实网页任务。测试集包含 1013 个任务，涵盖 100 多个不同网站。每个任务包含高级任务指令和一系列动作，每个动作前有网页截图，作为黄金轨迹。黄金轨迹中的所有网页均被缓存以支持离线评估。任务通过众包完成，重点确保其现实意义 (即真实用户在这些网站上的需求)。

Zheng et al. (2024) have clearly demonstrated the necessity of visual perception for web agents, so we mainly compare with zero-shot methods that use MLLMs as planners and omit text-only LLMs. Zheng et al. (2024) have also identified grounding as the main challenge and proposed several grounding strategies, including 1) Choice,

where the planner is asked to choose from a short list of filtered HTML elements, and 2) SoM, where the input screenshot is superposed with Set-of-Mark (Yang et al. 2023) labels and the planner is asked to select from the labels. Both strategies require additional text-based representations (i.e., HTML) to obtain the candidates and/or locate the elements in the screenshot to label. We report element accuracy, i.e., accuracy of selecting the correct element, and omit operation scores because they are orthogonal to grounding comparisons.

Zheng 等 (2024) 明确展示了视觉感知对网页代理的重要性，因此我们主要与使用多模态大语言模型 (MLLMs) 作为规划器的零样本方法比较，省略仅文本大语言模型 (LLMs)。Zheng 等 (2024) 还指出定位是主要挑战，并提出了几种定位策略，包括 1)Choice，规划器需从过滤后的 HTML 元素短列表中选择；2)SoM，输入截图叠加 Set-of-Mark(Yang 等，2023) 标签，规划器需从标签中选择。两种策略均需额外的基于文本的表示 (即 HTML) 以获取候选项和/或定位截图中的元素以标注。我们报告元素准确率，即选择正确元素的准确率，省略操作得分，因为其与定位比较无关。

Mobile: AndroidControl. We use AndroidControl (Li et al., 2024b), a large-scale Android dataset comprising 15K unique tasks over 833 Apps. Screenshots, action sequences, and a11y trees are cached from human demonstrations as golden trajectories for training and evaluation purposes. Each action is also labeled by a corresponding low-level instruction (e.g., "set the hours to 6"). Following Li et al. (2024b), we use 500 random steps from the test set. We compare with the SOTA zero-shot method, the text-only version of M3A (Rawles et al. 2024), which instructs GPT-4 to generate textual actions as well as select elements from the a11y tree (Choice). We adopt the two task settings in Li et al. (2024b): high-level tasks, where only the high-level intent is provided, and low-level tasks, where both the high-level intent and the corresponding low-level instruction for each step are available. We use the standard metric, step-wise accuracy, where a step is considered successful only if all the predicted actions, elements, and arguments (if applicable) are correct.

移动端:AndroidControl。我们使用 AndroidControl(Li 等，2024b)，一个包含 833 个应用中 1.5 万个独特任务的大规模安卓数据集。截图、动作序列和辅助功能树 (a11y 树) 均从人工演示中缓存，作为训练和评估的黄金轨迹。每个动作还标注有对应的低级指令 (如“将小时设置为 6”)。遵循 Li 等 (2024b)，我们使用测试集中的 500 个随机步骤。我们与最先进的零样本方法——仅文本版本的 M3A(Rawles 等，2024) 比较，该方法指令 GPT-4 生成文本动作并从辅助功能树中选择元素 (Choice)。我们采用 Li 等 (2024b) 中的两种任务设置: 高级任务，仅提供高级意图；低级任务，同时提供高级意图和每步对应的低级指令。使用标准指标逐步准确率，只有当所有预测的动作、元素和参数 (如适用) 均正确时，该步骤才算成功。

Desktop: OmniACT. We use OmniACT (Kapoor et al. 2024) to evaluate the accuracy of UGround on desktop tasks. The dataset consists of 9,802 tasks covering 38 desktop applications and 27 websites across different desktop platforms (macOS, Windows, and Linux). Each task requires the generation of a PyAutoGUI script, which is a sequence of actions to complete the task on a single screenshot. The SOTA method, DetACT (Kapoor et al. 2024), extracts UI elements and their coordinates through a combination of OCR (optical character recognition), icon matching, and color detection modules. These elements are filtered by task relevance and then passed to LLMs or MLLMs to generate the PyAutoGUI script with the appropriate coordinates for interaction.

桌面端:OmniACT。我们使用 OmniACT(Kapoor 等，2024) 评估 UGround 在桌面任务上的准确率。该数据集包含 9802 个任务，覆盖 38 个桌面应用和 27 个网站，跨 macOS、Windows 和 Linux 等不同桌面平台。每个任务需生成 PyAutoGUI 脚本，即在单张截图上完成任务的动作序列。最先进方法 DetACT(Kapoor 等，2024) 通过 OCR(光学字符识别)、图标匹配和颜色检测模块提取 UI 元素及其坐标。元素经任务相关性过滤后，传递给 LLMs 或 MLLMs 生成带有适当交互坐标的 PyAutoGUI 脚本。

Table 5: Step accuracy on AndroidControl over 500 random actions from the test split. Baseline results are from Li et al. (2024b).

表 5:AndroidControl 测试集中 500 个随机动作的逐步准确率。基线结果来自 Li 等 (2024b)。

Input	Planner	Grounding	Step Accuracy	
			High	Low
Text	GPT-4	Choice	42.1	55.0
Image (SeeAct-V)	GPT-4	SeeClick	39.4	47.2
		UGround	46.2	58.0
	GPT-4o	SeeClick	41.8	52.8
		UGround	48.4	62.4
		UGround-V1-2B	50.0	65.0
		UGround-V1-7B	49.8	66.2

输入	规划器	定位	步骤准确率	
			高	低
文本	GPT-4	选择	42.1	55.0
图像 (SeeAct-V)	GPT-4	SeeClick	39.4	47.2
		UGround	46.2	58.0
	GPT-4o	SeeClick	41.8	52.8
		UGround	48.4	62.4
		UGround-V1-2B	50.0	65.0
		UGround-V1-7B	49.8	66.2

Table 6: Action scores (AS) on OmniACT. Baseline results are from Kapoor et al. (2024).

表 6:OmniACT 上的动作得分 (AS)。基线结果来自 Kapoor 等人 (2024)。

Inputs	Planner	Grounding	AS
Text Image + Text	GPT-4	DetACT	11.6
		DetACT	17.0
Image (SeeAct-V)	GPT-4	SeeClick	28.9
		UGround	31.1
	GPT-40	SeeClick	29.6
		UGround	32.8
		UGround-V1-2B	32.9
		UGround-V1-7B	34.0

输入	规划器	定位	AS
文本图像 + 文本	GPT-4	DetACT	11.6
		DetACT	17.0
图像 (SeeAct-V)	GPT-4	SeeClick	28.9
		UGround	31.1
	GPT-40	SeeClick	29.6
		UGround	32.8
		UGround-V1-2B	32.9
		UGround-V1-7B	34.0

For SeeAct-V, we replace the input of the DetACT pipeline with only screenshots and instruct MLLMs to generate element descriptions rather than directly generate coordinates. We then employ UGround to obtain the coordinates of the elements, which are subsequently integrated into the PyAutoGUI scripts. To ensure a fair comparison, we strictly follow the approach in Kapoor et al. (2024), including the same prompt and retrieval strategy that selects five in-context examples from the training set based on task similarity. We report the action score, which measures the accuracy of the action sequences while penalizing errors in generated arguments.

对于 SeeAct-V，我们将 DetACT 流水线的输入替换为仅包含截图，并指示多模态大模型 (MLLMs) 生成元素描述，而非直接生成坐标。随后，我们采用 UGround 获取元素的坐标，这些坐标随后被整合进 PyAutoGUI 脚本中。为确保公平比较，我们严格遵循 Kapoor 等人 (2024) 的方法，包括相同的提示和检索策略，该策略基于任务相似性从训练集中选择五个上下文示例。我们报告动作得分，该得分衡量动作序列的准确性，同时对生成参数中的错误进行惩罚。

Results. As shown in Table 4, Table 5, and Table 6, SeeAct-V with UGround outperforms all the baselines across the board, despite only using raw screenshots as input while baselines use additional input. UGround also consistently outperforms a strong GUI grounding model, SeeClick. These results provide solid support for human-like vision-only embodiment for GUI agents, a position this work aims to make a case for. The results also further validate UGround's efficacy as a universal grounding model for GUI agents.

结果。如表 4、表 5 和表 6 所示，尽管 SeeAct-V 仅使用原始截图作为输入，而基线方法使用额外输入，SeeAct-V 结合 UGround 仍在各方面均优于所有基线。UGround 也持续优于强大的 GUI 定位模型 SeeClick。这些结果为基于视觉的人类式 GUI 代理体现提供了有力支持，正是本工作旨在论证的观点。结果进一步验证了 UGround 作为通用 GUI 代理定位模型的有效性。

3.3 ONLINE AGENT EVALUATION

3.3 在线代理评估

We further evaluate our approach in an end-to-end manner on two online agent benchmarks that closely resemble the offline web and Android benchmarks in §3.2, but involve interactions with live websites and mobile applications. Due to the high cost of online evaluation, we only use UGround for grounding.

我们进一步在两个在线代理基准上以端到端方式评估我们的方法，这些基准与 §3.2 中的离线网页和安卓基准高度相似，但涉及与实时网站和移动应用的交互。鉴于在线评估成本较高，我们仅使用 UGround 进行定位。

Web: Mind2Web-Live. We use the test set from Mind2Web-Live (Pan et al. 2024). The benchmark is built on Mind2Web (Deng et al. 2023) by adding functional evaluation to the tasks that makes automated evaluation possible on live websites. Specifically, it defines and annotates key nodes for each task, which are critical steps that must be completed for a task to be considered successful, regardless of which trajectory an agent takes. The baseline agent from Pan et al. (2024) is text-only, perceives and interacts with webpages by hundreds of HTML elements at a time. For SeeAct-V, we change the observation to be screenshots only, and make necessary changes to the original action space to fully eliminate the dependency on HTML during planning, grounding, and execution (details in Appendix G.5). We use standard metrics: micro completion rate, which measures the proportion of completed key nodes across all the tasks, and task success rate, which measures the proportion of fully completed tasks.

网页:Mind2Web-Live。我们使用 Mind2Web-Live(Pan 等人，2024) 的测试集。该基准基于 Mind2Web(Deng 等人，2023) 构建，通过为任务添加功能性评估，使得在实时网站上实现自动化评估成为可能。具体而言，它为每个任务定义并标注关键节点，这些节点是任务成功完成的关键步骤，无论代理采取何种路径。Pan 等人 (2024) 的基线代理仅基于文本，通过数百个 HTML 元素感知并交互网页。对于 SeeAct-V，我们将观察改为仅截图，并对原始动作空间进行必要调整，以完全消除规划、定位和执行过程中对 HTML 的依赖 (详见附录 G.5)。我们使用标准指标: 微观完成率，衡量所有任务中完成关键节点的比例；任务成功率，衡量完全完成任务的比例。

Mobile: AndroidWorld. We use AndroidWorld (Rawles et al. 2024), an online mobile agent benchmark running in Android emulators. It includes 116 tasks across 20 Apps, with evaluation based on the final states of the device. We compare with the SOTA agent M3A and its text-only variant from Rawles et al. (2024). They receives both raw and SoM images, together with textual UI elements, or only the textual UI elements as the observation respectively. Both variants employ a ReAct-style reasoning process (Yao et al. 2023) to select the next target element from a list of UI elements. Additionally, they integrate self-reflection (Shinn et al. 2024) for the agent to summarize its current action and improve decision-making in subsequent steps. We report task success rate, which measures the percentage of fully completed tasks.

移动端:AndroidWorld。我们使用 AndroidWorld(Rawles 等人，2024)，这是一个在安卓模拟器中运行的在线移动代理基准。它包含 20 个应用中的 116 个任务，评估基于设备的最终状态。我们与 Rawles 等人 (2024) 提出的最先进代理 M3A 及其仅文本变体进行比较。它们分别接收原始图像和 SoM 图像，以及文本 UI 元素，或仅接收文本 UI 元素作为观察输入。两种变体均采用 ReAct 风格的推理过程 (Yao 等人，2023)，从 UI 元素列表中选择下一个目标元素。此外，它们集成了自我反思机制 (Shinn 等人，2024)，使代理能够总结当前动作并改进后续决策。我们报告任务成功率，衡量完全完成任务的百分比。

Results. SeeAct-V with UGround gets comparable or higher performance in online agent evaluation, as shown in Table 7 and Table 8 Particularly, it achieves a much higher success rate compared with the SoM variant of M3A, even though Android environments have less dense UI layouts and

结果。SeeAct-V 结合 UGround 在在线代理评估中取得了相当或更高的性能，如表 7 和表 8 所示。特别是，与 M3A 的 SoM 变体相比，尽管安卓环境的 UI 布局较为稀疏，SeeAct-V 仍实现了更高的成功率。

Table 7: Completion rate (CR) and task success rate (SR) on Mind2Web-Live. Baseline results are from Pan et al. (2024).

表 7:Mind2Web-Live 上的完成率 (CR) 和任务成功率 (SR)。基线结果来自 Pan 等人 (2024)。

Inputs	Planner	Grounding	CR	SR
Text	GPT-4	Choice	44.3	21.1
	GPT-40		47.6	22.1
Image (SeeAct-V)	GPT-4 GPT-40	UGround	50.7	
50.8	23.1			
19.2				

输入	规划器	定位	CR	SR
文本	GPT-4	选择	44.3	21.1
	GPT-40		47.6	22.1
图像 (SeeAct-V)	GPT-4 GPT-40	UGround	50.7	
50.8	23.1			
19.2				

Table 8: Task success rate (SR) on Android-World. Baseline results are from Rawles et al. (2024).

表 8:Android-World 上的任务成功率 (SR)。基线结果来自 Rawles 等人 (2024)。

Input	Planner	Grounding	SR
Text	GPT-4	Choice	30.6
Image + Text		SoM	25.4
Image (SeeAct-V)	GPT-4GPT-40GPT-40	UGround	31.0
			32.8
		UGround-V1-7B	44.0

输入	规划器	基础	SR
文本	GPT-4	选择	30.6
图像 + 文本		SoM	25.4
图像 (SeeAct-V)	GPT-4GPT-40GPT-40	UGround	31.0
			32.8
		UGround-V1-7B	44.0

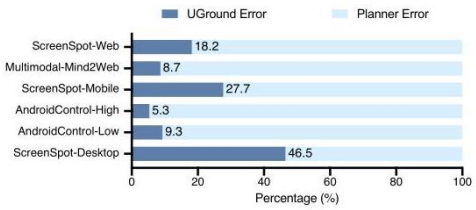
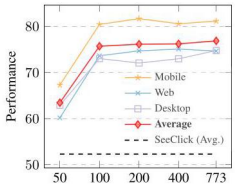


Figure 4: Error distribution from manual analysis.

图 4: 手动分析的错误分布。



#Web Synthetic Training Data (K) (# Screenshots)

网络合成训练数据 (千)(# 截图数)

Figure 5: Scaling curve of UGround on ScreenSpot w.r.t. Web-Hybrid data size.

图 5:UGround 在 ScreenSpot 上相对于 Web-Hybrid 数据规模的扩展曲线。

are generally more suitable for SoM (i.e., less obstruction by the SoM labels). These results again provide solid support for the feasibility and promises of human-like vision-only embodiment for GUI agents and the effectiveness of UGround.

通常更适合 SoM(即，较少被 SoM 标签遮挡)。这些结果再次有力支持了类人视觉单一感知 (vision-only embodiment)GUI 代理的可行性和前景，以及 UGround 的有效性。

3.4 ERROR ANALYSIS

3.4 错误分析

We conduct a manual error analysis of the best performing method, SeeAct-V with UGround, to understand the bottleneck for further improvement. We randomly sample 60 failure cases from each split of ScreenSpot (agent setting with GPT-4o), AndroidControl, and Multimodal-Mind2Web. Except for data annotation errors, errors from the models can be categorized into planning errors, i.e., generating plans with incorrect element descriptions, and grounding errors, i.e., predicting incorrect coordinates for a correct element description from the planner.

我们对表现最佳的方法 SeeAct-V 结合 UGround 进行了手动错误分析，以了解进一步改进的瓶颈。我们从 ScreenSpot(使用 GPT-4o 的代理设置)、AndroidControl 和 Multimodal-Mind2Web 的每个划分中随机抽取 60 个失败案例。除数据标注错误外，模型错误可分为规划错误，即生成了错误元素描述的计划；以及定位错误，即对规划器正确描述的元素预测了错误的坐标。

As shown in Figure 4, planning errors are the dominant cause of failures across all benchmarks, further confirming the strong grounding capability of UGround. The most frequent error is that the planner generates (otherwise correct) description of an incorrect element on the screen, indicating a lack of correct understanding of either the task and/or the elements. Other common planning errors include hallucinating non-existent elements or producing overly generic descriptions that are too vague to uniquely locate the target element, even for human evaluators.

如图 4 所示，规划错误是所有基准测试中失败的主要原因，进一步证实了 UGround 强大的定位能力。最常见的错误是规划器生成了 (其他方面正确的) 错误元素描述，表明对任务和/或元素缺乏正确理解。其他常见规划错误包括虚构不存在的元素或产生过于笼统的描述，导致即使是人工评估者也难以唯一定位目标元素。

On the other hand, on ScreenSpot-Mobile and ScreenSpot-Desktop, a considerable portion of the failures do stem from grounding errors. Both desktop and mobile UIs feature a pervasive use of icons with idiosyncratic meaning. For example, a stylized dollar sign represents the Zelle App, or an icon with two cartoon people represents one's contact list in Microsoft Outlook. We find that pretrained MLLMs and our web-centric grounding training are effective in capturing the semantics of popular icons (e.g., icons representing Google) or commonsense meaning (e.g., clock icons usually represent time-related functions like alarms). However, it is challenging to capture the idiosyncratic semantics of icons in the long tail, which arguably requires either additional documentation or more targeted exploration to learn. This is a major cause of the grounding errors. Interestingly, when tested on more realistic agent tasks, e.g., in AndroidControl, AndroidWorld, and OmniACT, UGround still proves to be relatively robust. This is because most of the agent tasks concern things in the head of the distribution; things in the long tail are naturally rare (though still important). This explains the strong performance of UGround on mobile and desktop agent benchmarks. Nonetheless, how to capture idiosyncratic semantics in the long tail is still an open challenge for grounding.

另一方面，在 ScreenSpot-Mobile 和 ScreenSpot-Desktop 上，相当一部分失败确实源于定位错误。桌面和移动 UI 普遍使用具有特殊含义的图标。例如，风格化的美元符号代表 Zelle 应用，或两个卡通人物的图标代表 Microsoft Outlook 中的联系人列表。我们发现预训练的多模态大语言模型 (MLLMs) 和我们以网络为中心的定位训练能有效捕捉流行图标 (如代表 Google 的图标) 或常识性含义 (如时钟图标通常代表闹钟等时间相关功能)。然而，捕捉长尾中图标的特殊语义仍具挑战性，这可能需要额外文档或更有针对性的探索学习。这是定位错误的主要原因。有趣的是，在更现实的代理任务中测试时，如 AndroidControl、AndroidWorld 和 OmniACT，UGround 仍表现出较强的鲁棒性。这是因为大多数代理任务涉及分布头部的内容；长尾内容自然较少 (但仍重要)。这解释了 UGround 在移动和桌面代理基准上的优异表现。尽管如此，如何捕捉长尾中的特殊语义仍是定位领域的开放挑战。

Table 9: Training data ablations for UGround on ScreenSpot (Agent Setting).

表 9:UGround 在 ScreenSpot(代理设置) 上的训练数据消融实验。

Training Data	Mobile		Desktop		Web		Average
	Text	Icon/Widget	Text	Icon/Widget	Text	Icon/Widget	
Web-Hybrid	89.0	73.4	88.1	61.4	84.8	64.6	76.9
Others	92.3	71.2	84.5	46.4	87.0	59.2	73.4
All	93.4	76.9	92.8	67.9	88.7	68.9	81.4

训练数据	移动端		桌面端		网页端		平均值
	文本	图标/小部件	文本	图标/小部件	文本	图标/小部件	
网页混合	89.0	73.4	88.1	61.4	84.8	64.6	76.9
其他	92.3	71.2	84.5	46.4	87.0	59.2	73.4
全部	93.4	76.9	92.8	67.9	88.7	68.9	81.4

3.5 TRAINING DATA ANALYSIS: SCALING AND ABLATIONS

3.5 训练数据分析: 规模扩展与消融研究

We conduct scaling analysis and ablation studies on our training data to better understand the contribution of different data for UGround's strong performance, and use the agent setting of ScreenSpot for the evaluation (with GPT-40 as the planner). Further ablations around data, model design, and RE types are provided in Appendix C

我们对训练数据进行了规模扩展分析和消融研究，以更好地理解不同数据对 UGround 卓越性能的贡献，并采用 ScreenSpot 的代理设置进行评估 (规划器为 GPT-40)。关于数据、模型设计和关系抽取 (RE) 类型的进一步消融研究详见附录 C。

Scaling Curve on Web-Hybrid. We investigate the scaling of our primary synthetic dataset, Web-Hybrid, which consists of 9M data instances over 773K web screenshots in total. The scaling results in Figure 5 show that the average performance consistently improves as the data scales up, though the return starts diminishing after 100 K screenshots. Notably, with just 50 K screenshots (about 600 K elements) as training data, UGround surpasses SeeClick by more than 10%, which is trained on about 3M web and Android elements from about 400 K screenshots. The results clearly show the high data quality and the effectiveness for grounding training of our data synthesis pipeline. Upon manual inspection, we observe that additional data after 100K screenshots primarily enhances understanding of less frequent elements such as radio buttons, checkboxes, or very small text elements. As data increases, the model can point to the center of element bounding boxes more accurately and better handle tiny hyperlinks.

Web-Hybrid 上的规模曲线。我们研究了主要合成数据集 Web-Hybrid 的规模扩展，该数据集包含总计 773K 网页截图中的 900 万数据实例。图 5 中的规模扩展结果显示，随着数据规模的增加，平均性能持续提升，尽管在 100 K 张截图后收益开始递减。值得注意的是，仅用 50 K 张截图 (约 600 K 个元素) 作为训练数据，UGround 的表现就超过了 SeeClick 超过 10%，而 SeeClick 是基于约 300 万网页和安卓元素、约 400 K 张截图训练的。结果清晰表明了我们数据合成流程的数据高质量及其对视觉定位训练的有效性。人工检查发现，100K 张截图之后的额外数据主要提升了对较少见元素 (如单选按钮、复选框或非常小的文本元素) 的理解。随着数据增加，模型能更准确地指向元素边界框中心，并更好地处理微小的超链接。

Training Data Ablations. To further investigate the impact of training data sources, we compare the performance of UGround trained on only Web-Hybrid, only the supplementary data, or both (see Table 1). Results in Table 9 further validate the necessity of Web-Hybrid. Training on other data without Web-Hybrid often underperforms training on Web-Hybrid alone. This is most evident on icons and widgets, which require understanding more diverse aspects, such as visual features and functions, than text-based elements. Finally, these two data sources are complementary and their combination yield the best performance across the board.

训练数据消融。为进一步探究训练数据来源的影响，我们比较了仅用 Web-Hybrid、仅用补充数据或两者结合训练的 UGround 性能 (见表 1)。表 9 的结果进一步验证了 Web-Hybrid 的必要性。仅用非 Web-Hybrid 数据训练通常不及单独用 Web-Hybrid 训练的效果。这一点在图标和控件上尤为明显，这些元素需要理解比基于文本的元素更丰富的视觉特征和功能。最后，这两种数据源具有互补性，结合使用能在各方面取得最佳表现。

4 CONCLUSIONS AND LIMITATIONS

4 结论与局限性

We introduce UGround, a universal GUI visual grounding model developed with large-scale web-based synthetic data. UGround shows strong cross-platform generalization and substantially outperforms the prior models. We propose a vision-only framework SeeAct-V that allows pixel-level interactions based solely on visual input. Comprehensive evaluation on both offline and online agent benchmarks demonstrates that SeeAct-V agents with UGround can achieve comparable and often better performance than prior SOTA agents that rely on additional textual inputs like HTML or a11y trees for observation or grounding.

我们提出了 UGround，一种基于大规模网页合成数据开发的通用 GUI 视觉定位模型。UGround 展现了强大的跨平台泛化能力，显著优于先前模型。我们提出了纯视觉框架 SeeAct-V，允许基于视觉输入实现像素级交互。在离线和在线代理基准上的全面评估表明，搭载 UGround 的 SeeAct-V 代理能够达到甚至超越依赖额外文本输入 (如 HTML 或辅助功能树) 进行观察或定位的先前最先进 (SOTA) 代理的性能。

Nevertheless, there are still some limitations that could be addressed in future work to advance visual grounding in GUI applications and visually grounded GUI agents. First, UGround is trained on very large-scale synthetic data. Considering the similarity and repetition of elements between web pages, there is room to improve on data efficiency during training, for example by better data grouping and deduplication. On the other hand, despite the cross-platform generalization shown in our experiment results, the issue of long-tail elements remains underaddressed in this work. Mobile UIs and desktop UIs often feature specific icons with idiosyncratic semantics, and it can be impractical to account for every long-tail element in a training set. Additionally, no desktop UI data is incorporated in the training of this work, which limits the performance on desktop UIs. Given the scarcity of training datasets for desktop UIs, we anticipate the development of more comprehensive datasets in this domain. Lastly, UGround depends on an external planner; it is not meant to function independently as a GUI agent. Nonetheless, we hope that our datasets, model, and framework can contribute to future studies of vision-only agents, as well as contribute to advancing the grounding capabilities of end-to-end models, as strong grounding data has been shown to improve end-to-end models (Cheng et al. 2024; Hong et al. 2024; Chen et al. 2024).

尽管如此，仍存在一些局限性，未来工作可针对这些问题推动 GUI 应用中的视觉定位及视觉定位 GUI 代理的发展。首先，UGround 是在极大规模合成数据上训练的。考虑到网页元素的相似性和重复性，训练中的数据效率仍有提升空间，例如通过更好的数据分组和去重。另一方面，尽管实验结果显示了跨平台泛化能力，长尾元素问题在本工作中仍未充分解决。移动和桌面 UI 常包含具有特殊语义的特定图标，训练集中涵盖所有长尾元素几乎不现实。此外，本工作未纳入任何桌面 UI 数据，限制了在桌面 UI 上的表现。鉴于桌面 UI 训练数据集的稀缺，我们期待该领域更全面数据集的发展。最后，UGround 依赖外部规划器，非独立 GUI 代理。尽管如此，我们希望我们的数据集、模型和框架能助力未来纯视觉代理的研究，并推动端到端模型的定位能力提升，已有研究表明高质量定位数据能提升端到端模型性能 (Cheng et al. 2024；Hong et al. 2024；Chen et al. 2024)。

ETHICS STATEMENT

伦理声明

This work employs web-based data synthesis to develop visual grounding models for GUIs. The synthesis pipeline and data collection presented in this paper are intended solely for research purposes related to GUI grounding and GUI agents, in line with prior works in the field (Hong et al. 2024) Cheng et al. 2024).

本工作采用基于网页的数据合成方法开发 GUI 视觉定位模型。本文所述的合成流程和数据收集仅用于与 GUI 定位和 GUI 代理相关的研究目的，符合该领域先前工作 (Hong et al. 2024；Cheng et al. 2024)。

The webpages utilized in our work are sourced from the Common Crawl datase[5] which is a publicly available Internet archive for research and non-commercial use. We use only a small subset of it and strictly adhere to Common Crawl's terms of use throughout our study.

我们使用的网页数据来源于 Common Crawl 数据集 [5]，这是一个公开的互联网档案，供研究和非商业用途。我们仅使用了其中一小部分，并在整个研究过程中严格遵守 Common Crawl 的使用条款。

Our use and dissemination of the data are exclusively for academic research and fully comply with Section 107 of the U.S. Copyright Law regarding Fair Use. Prior to release, the data undergoes rigorous content moderation. We acknowledge full responsibility for any legal issues arising from our data collection and accept all associated risks. Furthermore, the distribution of the data is managed in strict accordance with applicable regulations and guidelines to ensure compliance with AI ethics standards and non-commercial usage.

我们对数据的使用和传播仅限于学术研究，完全符合美国版权法第 107 条关于合理使用的规定。数据在发布前经过严格的内容审核。我们承认对因数据收集引发的任何法律问题承担全部责任，并接受所有相关风险。此外，数据的分发严格遵守适用的法规和指南，以确保符合人工智能伦理标准和非商业用途。

ACKNOWLEDGMENTS

致谢

We are grateful for the collaboration with the Orby AI team (particularly Sanjari Srivastava, Peng Qi, Gang Li, and Will Lu) for their contribution on data collection and analysis, as well as for providing computing resources. We would also like to extend our appreciation to colleagues from the OSU NLP group and Kanzhi Cheng, Yulu Guo, Lizi Yang for their insightful comments. Special thanks to Yichen Pan, Christopher Rawles, Dehan Kong, Alice Li, and Raghav Kapoor for their assistance with evaluation. This work is supported in part by Orby AI, ARL W911NF2220144, and NSF CAREER #1942980. The views and conclusions contained herein are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. government. The U.S. government is authorized to reproduce and distribute reprints for government purposes notwithstanding any copyright notice herein.

我们感谢 Orby AI 团队 (特别是 Sanjari Srivastava、Peng Qi、Gang Li 和 Will Lu) 在数据收集和分析方面的合作，以及提供计算资源。我们还要感谢俄亥俄州立大学自然语言处理 (NLP) 小组的同事们以及 Kanzhi Cheng、Yulu Guo、Lizi Yang 的宝贵意见。特别感谢 Yichen Pan、Christopher Rawles、Dehan Kong、Alice Li 和 Raghav Kapoor 在评估方面的协助。本工作部分由 Orby AI、ARL W911NF2220144 和 NSF CAREER #1942980 资助。文中观点和结论仅代表作者本人，不应被解读为美国政府的官方政策，无论明示或暗示。尽管本文含有版权声明，美国政府有权为政府用途复制和分发本文。

REFERENCES

参考文献

AI@Meta.Llama 3 model card, 2024. URL https://github.com/meta-llama/llama3/blob/ main/MODEL_CARD.md

AI@Meta.Llama 3 模型卡,2024。网址 https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md

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, IJCAI-21, pp. 1705-1712, 2021.

Chongyang Bai, Xiaoxue Zang, Ying Xu, Srinivas Sunkara, Abhinav Rastogi, Jindong Chen, 和 Blaise Agüera y Arcas. UIBert: 学习通用多模态表示以理解用户界面。在第三十届国际人工智能联合会议 (IJCAI-21) 论文集，页 1705-1712，2021 年。

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-VL: A versatile vision-language model for understanding, localization, text reading, and beyond. arXiv preprint arXiv:2308.12966, 2023.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, 和 Jingren Zhou. Qwen-VL: 一款多功能视觉-语言模型，用于理解、定位、文本阅读及更多。arXiv 预印本 arXiv:2308.12966，2023 年。

Pratyay Banerjee, Shweti Mahajan, Kushal Arora, Chitta Baral, and Oriana Riva. Lexi: Self-supervised learning of the ui language. In Findings of the Association for Computational Linguistics: EMNLP 2022, 2022.

Pratyay Banerjee, Shweti Mahajan, Kushal Arora, Chitta Baral, 和 Oriana Riva. Lexi: 用户界面语言的自监督学习。在计算语言学协会会议成果:EMNLP 2022，2022 年。

Ruisheng Cao, Fangyu Lei, Haoyuan Wu, Jixuan Chen, Yeqiao Fu, Hongcheng Gao, Xinzhuang Xiong, Hanchong Zhang, Yuchen Mao, Wenjing Hu, et al. Spider2-V: How far are multimodal agents from automating data science and engineering workflows? In The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track, 2024.

Ruisheng Cao, Fangyu Lei, Haoyuan Wu, Jixuan Chen, Yeqiao Fu, Hongcheng Gao, Xinzhuang Xiong, Hanchong Zhang, Yuchen Mao, Wenjing Hu 等。Spider2-V: 多模态代理距离自动化数据科学与工程工作流还有多远？在第三十八届神经信息处理系统会议数据集与基准轨道，2024 年。

’https://commoncrawl.org/

’https://commoncrawl.org/

’https://commoncrawl.org/terms-of-use

’https://commoncrawl.org/terms-of-use

Jun Chen, Deyao Zhu, Xiaoqian Shen, Xiang Li, Zechun Liu, Pengchuan Zhang, Raghuraman Krishnamoorthi, Vikas Chandra, Yunyang Xiong, and Mohamed Elhoseiny. MiniGPT-v2: large language model as a unified interface for vision-language multi-task learning. arXiv preprint arXiv:2310.09478, 2023a.

Jun Chen, Deyao Zhu, Xiaoqian Shen, Xiang Li, Zechun Liu, Pengchuan Zhang, Raghuraman Krishnamoorthi, Vikas Chandra, Yunyang Xiong, 和 Mohamed Elhoseiny. MiniGPT-v2: 大型语言模型作为视觉-语言多任务学习的统一接口。arXiv 预印本 arXiv:2310.09478，2023a。

Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. Shikra: Unleashing multimodal llm’s referential dialogue magic. arXiv preprint arXiv:2306.15195, 2023b.

Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, 和 Rui Zhao. Shikra: 释放多模态大型语言模型的指称对话魔力。arXiv 预印本 arXiv:2306.15195，2023b。

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, 2024.

Wentong Chen, Junbo Cui, Jinyi Hu, Yujia Qin, Junjie Fang, Yue Zhao, Chongyi Wang, Jun Liu, Guirong Chen, Yupeng Huo 等。GUICourse: 从通用视觉语言模型到多功能图形用户界面代理。arXiv 预印本 arXiv:2406.11317，2024 年。

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, Jianbing Zhang, and Zhiyong Wu. SeeClick: Harnessing GUI grounding for advanced visual GUI agents. In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2024.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, Jianbing Zhang, 和 Zhiyong Wu. SeeClick: 利用图形用户界面定位打造高级视觉 GUI 代理。在第 62 届计算语言学协会年会论文集 (第一卷: 长篇论文)，2024 年。

Siyuan Cheng, Bozhong Tian, Qingbin Liu, Xi Chen, Yongheng Wang, Huajun Chen, and Ningyu Zhang. Can we edit multimodal large language models? In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pp. 13877-13888, 2023.

程思远，田博中，刘清斌，陈曦，王永恒，陈华军，张宁宇。我们能编辑多模态大型语言模型吗? 发表于 2023 年自然语言处理实证方法会议论文集，第 13877-13888 页，2023 年。

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, pp. 845-854, 2017.

比普拉布·德卡，黄子峰，查德·弗兰岑，约书亚·希布施曼，丹尼尔·阿弗根，李阳，杰弗里·尼科尔斯，兰吉塔·库马尔。Rico: 用于构建数据驱动设计应用的移动应用数据集。发表于第 30 届 ACM 用户界面软件与技术年会论文集，第 845-854 页，2017 年。

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2Web: Towards a generalist agent for the web. In Advances in Neural Information Processing Systems, volume 36, pp. 28091-28114, 2023.

邓翔，顾宇，郑博远，陈世杰，萨姆·史蒂文斯，王博时，孙焕，苏宇。Mind2Web: 迈向通用网络智能体。发表于神经信息处理系统进展，第 36 卷，第 28091-28114 页，2023 年。

Xiaoyi Dong, Pan Zhang, Yuhang Zang, Yuhang Cao, Bin Wang, Linke Ouyang, Songyang Zhang, Haodong Duan, Wenwei Zhang, Yining Li, Hang Yan, Yang Gao, Zhe Chen, xinyue zhang, Wei Li, Li Jingwen, Wenhai Wang, Kai Chen, Conghui He, Xingcheng ZHANG, Jifeng Dai, Yu Qiao, Dahua Lin, and Jiaqi Wang. InternLM-XComposer2-4KHD: A pioneering large vision-language model handling resolutions from 336 pixels to 4K HD. In Advances in Neural Information Processing Systems, volume 37, pp. 42566-42592, 2024.

董晓毅，张攀，臧宇航，曹宇航，王斌，欧阳林可，张松阳，段浩东，张文伟，李一宁，严航，高阳，陈哲，张欣悦，李伟，景文立，王文海，陈凯，何聪辉，张星成，戴继峰，乔宇，林大华，王佳琦。InternLM-XComposer2-4KHD: 首个支持 336 像素至 4K 高清分辨率的大型视觉语言模型。发表于神经信息处理系统进展，第 37 卷，第 42566-42592 页，2024 年。

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In International Conference on Learning Representations, 2021.

阿列克谢·多索维茨基，卢卡斯·拜耶，亚历山大·科列斯尼科夫，迪尔克·魏森博恩，翟晓华，托马斯·温特希纳，莫斯塔法·德赫加尼，马蒂亚斯·明德勒，乔治·海戈尔德，西尔万·格利等。图像价值 16x16 词: 大规模图像识别的 Transformer 模型。发表于国际学习表征会议，2021 年。

Peng Gao, Renrui Zhang, Chris Liu, Longtian Qiu, Siyuan Huang, Weifeng Lin, Shitian Zhao, Shijie Geng, Ziyi Lin, Peng Jin, et al. SPHINX-X: Scaling data and parameters for a family of multi-modal large language models. arXiv preprint arXiv:2402.05935, 2024.

高鹏，张仁睿，刘克里斯，邱龙天，黄思远，林伟峰，赵世天，耿世杰，林子怡，金鹏等。SPHINX-X: 多模态大型语言模型系列的数据与参数扩展。arXiv 预印本 arXiv:2402.05935，2024 年。

Zonghao Guo, Ruyi Xu, Yuan Yao, Junbo Cui, Zanlin Ni, Chunjiang Ge, Tat-Seng Chua, Zhiyuan Liu, and Gao Huang. LLaVA-UHD: An LMM perceiving any aspect ratio and high-resolution images. In Computer Vision - ECCV 2024 - 18th European Conference, Milan, Italy, September 29-October 4, 2024, Proceedings, Part LXXXIII, volume 15141, pp. 390-406, 2024.

郭宗浩，徐如意，姚远，崔俊博，倪赞林，葛春江，蔡达生，刘志远，黄高。LLaVA-UHD: 感知任意长宽比和高分辨率图像的大型多模态模型。发表于计算机视觉 - ECCV 2024 - 第 18 届欧洲会议，意大利米兰，2024 年 9 月 29 日至 10 月 4 日，论文集第 LXXXIII 部分，第 15141 卷，第 390-406 页，2024 年。

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 The Twelfth International Conference on Learning Representations, 2024.

伊泽丁·古尔，古田浩树，奥斯汀·V·黄，穆斯塔法·萨夫达里，松尾丰，道格拉斯·埃克，亚历山德拉·福斯特。具备规划、长上下文理解与程序合成的真实世界网络智能体。发表于第十二届国际学习表征会议，2024 年。

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. WebVoyager: Building an end-to-end web agent with large multimodal models. In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2024.

何洪亮，姚文林，马凯鑫，余文浩，戴勇，张洪明，兰振中，余东。WebVoyager: 基于大型多模态模型构建的端到端网络智能体。发表于第 62 届计算语言学协会年会论文集 (第一卷: 长文)，2024 年。

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.

洪文义，王伟涵，吕庆松，徐家正，于文萌，季俊辉，王岩，王子涵，董宇霄，丁明等。CogAgent: 面向图形用户界面代理的视觉语言模型。发表于 IEEE/CVF 计算机视觉与模式识别会议论文集，第 14281-14290 页，2024 年。

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu

Chen. LoRA: Low-rank adaptation of large language models. In International Conference on Learning Representations, 2022.

爱德华·J·胡，沈业龙，菲利普·沃利斯，艾伦-朱泽元，李元志，王善，王璐，陈伟柱。LoRA: 大型语言模型的低秩适配。发表于国际学习表征会议，2022 年。

Peter C Humphreys, David Raposo, Tobias Pohlen, Gregory Thornton, Rachita Chhaparia, Alistair Muldal, Josh Abramson, Petko Georgiev, Adam Santoro, and Timothy Lillicrap. A data-driven approach for learning to control computers. In International Conference on Machine Learning, pp. 9466-9482. PMLR, 2022.

Peter C Humphreys, David Raposo, Tobias Pohlen, Gregory Thornton, Rachita Chhaparia, Alistair Muldal, Josh Abramson, Petko Georgiev, Adam Santoro, 和 Timothy Lillicrap. 一种基于数据驱动的计算机控制学习方法。发表于国际机器学习大会，页 9466-9482。PMLR, 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.

Raghav Kapoor, Yash Parag Butala, Melisa Russak, Jing Yu Koh, Kiran Kamble, Waseem Alshikh, 和 Ruslan Salakhutdinov. OmniACT: 一个用于支持桌面和网页多模态通用自主代理的数据集和基准。arXiv 预印本 arXiv:2402.17553, 2024。

Andrej Karpathy, Armand Joulin, and Li F Fei-Fei. Deep fragment embeddings for bidirectional image sentence mapping. In Advances in Neural Information Processing Systems, volume 27, 2014.

Andrej Karpathy, Armand Joulin, 和 Li F Fei-Fei. 用于双向图像-句子映射的深度片段嵌入。发表于神经信息处理系统进展，卷 27，2014。

Geunwoo Kim, Pierre Baldi, and Stephen McAleer. Language models can solve computer tasks. In Advances in Neural Information Processing Systems, volume 36, pp. 39648-39677, 2023.

Geunwoo Kim, Pierre Baldi, 和 Stephen McAleer. 语言模型能够解决计算机任务。发表于神经信息处理系统进展，卷 36，页 39648-39677，2023。

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan 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 (Volume 1: Long Papers), pp. 881-905, 2024.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, 和 Daniel Fried. VisualWebArena: 评估多模态代理在真实视觉网页任务中的表现。发表于第 62 届计算语言学协会年会论文集 (第一卷: 长篇论文)，页 881-905，2024。

Zhengfeng Lai, Haotian Zhang, Wentao Wu, Haoping Bai, Aleksei Timofeev, Xianzhi Du, Zhe Gan, Jiulong Shan, Chen-Nee Chuah, Yinfei Yang, et al. From scarcity to efficiency: Improving clip training via visual-enriched captions. arXiv preprint arXiv:2310.07699, 2023.

Zhengfeng Lai, Haotian Zhang, Wentao Wu, Haoping Bai, Aleksei Timofeev, Xianzhi Du, Zhe Gan, Jiulong Shan, Chen-Nee Chuah, Yinfei Yang 等. 从稀缺到高效: 通过视觉丰富的字幕改进 CLIP 训练。arXiv 预印本 arXiv:2310.07699, 2023。

Bo Li, Hao Zhang, Kaichen Zhang, Dong Guo, Yuanhan Zhang, Renrui Zhang, Feng Li, Ziwei Liu, and Chunyuan Li. LLaVA-NeXT: What else influences visual instruction tuning beyond data?, May 2024a. URL https://llava-vl.github.io/blog/2024-05-25-llava-next-ablations/

Bo Li, Hao Zhang, Kaichen Zhang, Dong Guo, Yuanhan Zhang, Renrui Zhang, Feng Li, Ziwei Liu, 和 Chunyuan Li. LLaVA-NeXT: 除了数据之外，视觉指令调优还受哪些因素影响? 2024 年 5 月 a。网址 https://llava-vl.github.io/blog/2024-05-25-llava-next-ablations/

Gang Li and Yang Li. Spotlight: Mobile ui understanding using vision-language models with a focus. In The Eleventh International Conference on Learning Representations, 2022.

Gang Li 和 Yang Li. Spotlight: 利用视觉-语言模型聚焦实现移动界面理解。发表于第十一届国际学习表征会议，2022。

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, 和 Oriana Riva. 数据规模对计算机控制代理影响的研究。arXiv 预印本 arXiv:2406.03679, 2024b。

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, pp. 8198-8210, 2020a.

Yang Li, Jiacong He, Xin Zhou, Yuan Zhang, 和 Jason Baldridge. 将自然语言指令映射到移动界面操作序列。发表于第 58 届计算语言学协会年会，页 8198-8210，2020a。

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 Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 5495-5510, 2020b.

Yang Li, Gang Li, Luheng He, Jingjie Zheng, Hong Li, 和 Zhiwei Guan. 界面组件描述生成: 为移动用户界面元素生成自然语言描述。发表于 2020 年自然语言处理实证方法会议 (EMNLP)，页 5495-5510，2020b。

Zhangheng Li, Keen You, Haotian Zhang, Di Feng, Harsh Agrawal, Xiujun Li, Mohana Prasad Sathya Moorthy, Jeff Nichols, Yinfei Yang, and Zhe Gan. Ferret-UI 2: Mastering universal user interface understanding across platforms. arXiv preprint arXiv:2410.18967, 2024c.

Zhangheng Li, Keen You, Haotian Zhang, Di Feng, Harsh Agrawal, Xiujun Li, Mohana Prasad Sathya Moorthy, Jeff Nichols, Yinfei Yang, 和 Zhe Gan. Ferret-UI 2: 跨平台通用用户界面理解的掌握。arXiv 预印本 arXiv:2410.18967, 2024c。

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 26296-26306, 2024a.

Haotian Liu, Chunyuan Li, Yuheng Li, 和 Yong Jae Lee. 通过视觉指令调优改进基线模型。发表于 IEEE/CVF 计算机视觉与模式识别会议，页 26296-26306，2024a。

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. LLaVA-NeXT: Improved reasoning, OCR, and world knowledge, January 2024b. URL https: //llava-vl.github.io/blog/2024-01-30-llava-next/

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, 和 Yong Jae Lee. LLaVA-NeXT: 改进的推理、光学字符识别 (OCR) 和世界知识，2024 年 1 月 b。网址 https://llava-vl.github.io/blog/2024-01-30-llava-next/

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. Advances in neural information processing systems, 36, 2024c.

刘昊天，李春元，吴庆阳，李永宰。视觉指令调优。神经信息处理系统进展，第 36 卷，2024 年。

Yadong Lu, Jianwei Yang, Yelong Shen, and Ahmed Awadallah. Omniparser for pure vision based gui agent. arXiv preprint arXiv:2408.00203, 2024.

卢亚东，杨建伟，沈业龙，艾哈迈德·阿瓦达拉。基于纯视觉的 GUI 代理的全能解析器。arXiv 预印本 arXiv:2408.00203，2024 年。

Chuofan Ma, Yi Jiang, Jiannan Wu, Zehuan Yuan, and Xiaojuan Qi. Groma: Localized visual tokenization for grounding multimodal large language models. arXiv preprint arXiv:2404.13013, 2024.

马楚凡，姜毅，吴建南，袁泽焕，齐晓娟。Groma: 用于多模态大型语言模型定位的局部视觉标记化。arXiv 预印本 arXiv:2404.13013，2024 年。

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 11-20, 2016.

毛俊华，乔纳森·黄，亚历山大·托舍夫，奥安娜·坎布鲁，艾伦·L·尤伊尔，凯文·墨菲。生成与理解无歧义的物体描述。载于 IEEE 计算机视觉与模式识别会议论文集，第 11-20 页，2016 年。

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. In Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24, 2024.

牛润良，李金东，王世奇，傅雅丽，胡希宇，冷学元，孔贺，常毅，王琦。ScreenAgent: 基于视觉语言模型的计算机控制代理。载于第三十三届国际人工智能联合会议 (IJCAI-24)，2024 年。

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, 2024.

潘一辰，孔德涵，周思达，崔成，冷一飞，姜冰，刘航宇，尚彦毅，周书妍，吴通爽，等。WebCanvas: 在线环境中网页代理的基准测试。arXiv 预印本 arXiv:2406.12373，2024 年。

Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, and Furu Wei. Kosmos-2: Grounding multimodal large language models to the world. arXiv preprint arXiv:2306.14824, 2023.

彭志良，王文辉，董力，郝雅茹，黄少涵，马书明，魏茹。Kosmos-2: 将多模态大型语言模型接地于现实世界。arXiv 预印本 arXiv:2306.14824，2023 年。

Yijun Qian, Yujie Lu, Alexander G Hauptmann, and Oriana Riva. Visual grounding for user interfaces. In Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track), pp. 97-107, 2024.

钱一军，卢宇杰，亚历山大·G·豪普特曼，奥里安娜·里瓦。用户界面的视觉定位。载于 2024 年北美计算语言学协会人类语言技术会议 (第 6 卷: 产业轨道)，第 97-107 页，2024 年。

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In International conference on machine learning, pp. 8748-8763. PMLR, 2021.

亚历克·拉德福德，金钟旭，克里斯·哈拉西，阿迪提亚·拉梅什，加布里埃尔·高，桑德希尼·阿加瓦尔，吉里什·萨斯特里，阿曼达·阿斯凯尔，帕梅拉·米什金，杰克·克拉克，等。从自然语言监督中学习可迁移的视觉模型。载于国际机器学习会议，第 8748-8763 页。PMLR，2021 年。

Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy Lillicrap. Android in the wild: A large-scale dataset for android device control. In Advances in Neural Information Processing Systems, volume 36, pp. 59708-59728, 2023.

克里斯托弗·罗尔斯, 爱丽丝·李, 丹尼尔·罗德里格斯, 奥里安娜·里瓦, 蒂莫西·利利克拉普。Android in the wild: 用于安卓设备控制的大规模数据集。载于神经信息处理系统进展, 第 36 卷, 第 59708-59728 页，2023 年。

Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William Bishop, Wei Li, Folawiyo Campbell-Ajala, et al. AndroidWorld: A dynamic benchmarking environment for autonomous agents. arXiv preprint arXiv:2405.14573, 2024.

克里斯托弗·罗尔斯，莎拉·克林克迈利，张一凡，乔纳森·沃尔茨，加布里埃尔·劳，玛丽贝丝·费尔，爱丽丝·李，威廉·毕晓普，李伟，福拉维奥·坎贝尔-阿贾拉，等。AndroidWorld: 自主代理的动态基准环境。arXiv 预印本 arXiv:2405.14573，2024 年。

Peter Shaw, Mandar Joshi, James Cohan, Jonathan Berant, Panupong Pasupat, Hexiang Hu, Urvashi Khandelwal, Kenton Lee, and Kristina N Toutanova. From pixels to UI actions: Learning to follow instructions via

graphical user interfaces. In Advances in Neural Information Processing Systems, volume 36, pp. 34354-34370, 2023.

彼得·肖，曼达尔·乔希，詹姆斯·科汉，乔纳森·贝兰特，帕努蓬·帕苏帕特，胡鹤翔，乌尔瓦希·坎德尔瓦尔，肯顿·李，克里斯蒂娜·N·图塔诺娃。从像素到 UI 操作: 通过图形用户界面学习执行指令。载于神经信息处理系统进展，第 36 卷，第 34354-34370 页，2023 年。

Tianlin Shi, Andrej Karpathy, Linxi Fan, Jonathan Hernandez, and Percy Liang. World of bits: An open-domain platform for web-based agents. In International Conference on Machine Learning, pp. 3135-3144. PMLR, 2017.

史天麟，安德烈·卡帕西，范林曦，乔纳森·埃尔南德斯，珀西·梁。World of bits: 基于网页代理的开放域平台。载于国际机器学习会议，第 3135-3144 页。PMLR，2017 年。

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. Advances in Neural Information Processing Systems, 36, 2024.

诺亚·辛恩，费德里科·卡萨诺，阿什温·戈皮纳特，卡尔蒂克·纳拉西姆汉，姚顺宇。Reflexion: 具备语言强化学习的语言代理。神经信息处理系统进展，第 36 卷，2024 年。

Junyang Wang, Haiyang Xu, Jiabo Ye, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. Mobile-Agent: Autonomous multi-modal mobile device agent with visual perception. In ICLR 2024 Workshop on Large Language Model (LLM) Agents, 2024a.

王俊阳，徐海洋，叶嘉博，闫明，沈伟舟，张骥，黄飞，桑吉涛。Mobile-Agent: 具备视觉感知的自主多模态移动设备代理。载于 ICLR 2024 大型语言模型 (LLM) 代理研讨会，2024 年。

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.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, 等. Qwen2-vl: 提升视觉-语言模型对任意分辨率世界的感知能力. arXiv 预印本 arXiv:2409.12191, 2024b.

Weiyun Wang, Min Shi, Qingyun Li, Wenhai Wang, Zhenhang Huang, Linjie Xing, Zhe Chen, Hao Li, Xizhou Zhu, Zhiguo Cao, et al. The all-seeing project: Towards panoptic visual recognition and understanding of the open world. In The Twelfth International Conference on Learning Representations, 2024c.

Weiyun Wang, Min Shi, Qingyun Li, Wenhai Wang, Zhenhang Huang, Linjie Xing, Zhe Chen, Hao Li, Xizhou Zhu, Zhiguo Cao, 等. 全视项目: 迈向开放世界的全景视觉识别与理解. 载于第十二届国际学习表征会议, 2024c.

Wenhai Wang, Zhe Chen, Xiaokang Chen, Jiannan Wu, Xizhou Zhu, Gang Zeng, Ping Luo, Tong Lu, Jie Zhou, Yu Qiao, and Jifeng Dai. VisionLLM: Large language model is also an open-ended decoder for vision-centric tasks. In Advances in Neural Information Processing Systems, volume 36, pp. 61501-61513, 2023.

Wenhai Wang, Zhe Chen, Xiaokang Chen, Jiannan Wu, Xizhou Zhu, Gang Zeng, Ping Luo, Tong Lu, Jie Zhou, Yu Qiao, 和 Jifeng Dai. VisionLLM: 大型语言模型也是面向视觉任务的开放式解码器. 载于神经信息处理系统进展, 第 36 卷, 页 61501-61513, 2023.

WebAIM. The WebAIM Million. https://webaim.org/projects/million/, 2024. Accessed: 2024-08-04.

WebAIM. The WebAIM Million. https://webaim.org/projects/million/, 2024. 访问日期:2024-08-04.

Zhiyong Wu, Chengcheng Han, Zichen Ding, Zhenmin Weng, Zhoumianze Liu, Shunyu Yao, Tao Yu, and Lingpeng Kong. OS-Copilot: Towards generalist computer agents with self-improvement. In ICLR 2024 Workshop on Large Language Model (LLM) Agents, 2024.

Zhiyong Wu, Chengcheng Han, Zichen Ding, Zhenmin Weng, Zhoumianze Liu, Shunyu Yao, Tao Yu, 和 Lingpeng Kong. OS-Copilot: 迈向具备自我提升能力的通用计算机代理. 载于 ICLR 2024 大型语言模型 (LLM) 代理研讨会, 2024.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Jing Hua Toh, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In Advances in Neural Information Processing Systems, volume 37, pp. 52040-52094, 2024.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Jing Hua Toh, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, 和 Tao Yu. Osworld: 真实计算机环境中开放式任务多模态代理的基准测试. 载于神经信息处理系统进展, 第 37 卷, 页 52040-52094, 2024.

An Yan, Zhengyuan Yang, Wanrong Zhu, Kevin Lin, Linjie Li, Jianfeng Wang, Jianwei Yang, Yiwu Zhong, Julian McAuley, Jianfeng Gao, et al. GPT-4V in wonderland: Large multimodal models for zero-shot smartphone gui navigation. arXiv preprint arXiv:2311.07562, 2023.

An Yan, Zhengyuan Yang, Wanrong Zhu, Kevin Lin, Linjie Li, Jianfeng Wang, Jianwei Yang, Yiwu Zhong, Julian McAuley, Jianfeng Gao, 等. GPT-4V 奇境: 面向零样本智能手机 GUI 导航的大型多模态模型. arXiv 预印本 arXiv:2311.07562, 2023.

Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. Set-of-Mark prompting unleashes extraordinary visual grounding in GPT-4v. arXiv preprint arXiv:2310.11441, 2023.

Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, 和 Jianfeng Gao. Set-of-Mark 提示释放 GPT-4v 卓越的视觉定位能力. arXiv 预印本 arXiv:2310.11441, 2023.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. Advances in Neural Information Processing Systems, 35:20744-20757, 2022.

Shunyu Yao, Howard Chen, John Yang, 和 Karthik Narasimhan. Webshop: 迈向具备基础语言代理的可扩展真实网络交互. 神经信息处理系统进展, 35:20744-20757, 2022.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In International Conference on Learning Representations (ICLR), 2023.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, 和 Yuan Cao. ReAct: 语言模型中推理与行动的协同. 载于国际学习表征会议 (ICLR), 2023.

Keen You, Haotian Zhang, Eldon Schoop, Floris Weers, Amanda Swearngin, Jeffrey Nichols, Yinfei Yang, and Zhe Gan. Ferret-UI: Grounded mobile ui understanding with multimodal llms. ArXiv, abs/2404.05719, 2024.

Keen You, Haotian Zhang, Eldon Schoop, Floris Weers, Amanda Swearngin, Jeffrey Nichols, Yinfei Yang, 和 Zhe Gan. Ferret-UI: 基于多模态大型语言模型的移动界面理解. ArXiv, abs/2404.05719, 2024.

Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. Modeling context in referring expressions. In Computer Vision-ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14, pp. 69-85. Springer, 2016.

Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, 和 Tamara L Berg. 指代表达中的上下文建模. 载于计算机视觉-ECCV 2016: 第 14 届欧洲会议, 荷兰阿姆斯特丹, 2016 年 10 月 11-14 日, 论文集, 第二部分 14, 页 69-85. Springer, 2016.

Zhuosheng Zhan and Aston Zhang. You only look at screens: Multimodal chain-of-action agents. arXiv preprint arXiv:2309.11436, 2023.

Zhuosheng Zhan 和 Aston Zhang. 你只需看屏幕: 多模态动作链代理. arXiv 预印本 arXiv:2309.11436, 2023.

Chaoyun Zhang, Liqun Li, Shilin He, Xu Zhang, Bo Qiao, Si Qin, Minghua Ma, Yu Kang, Qingwei Lin, Saravan Rajmohan, et al. UFO: A UI-focused agent for windows os interaction. arXiv preprint arXiv:2402.07939, 2024a.

张超云，李立群，何世林，张旭，乔博，秦思，马明华，康宇，林清伟，Saravan Rajmohan 等。UFO: 一个面向 Windows 操作系统交互的用户界面代理。arXiv 预印本 arXiv:2402.07939，2024a。

Haotian Zhang, Mingfei Gao, Zhe Gan, Philipp Dufter, Nina Wenzel, Forrest Huang, Dhruti Shah, Xianzhi Du, Bowen Zhang, Yanghao Li, et al. MM1.5: Methods, analysis & insights from multimodal llm fine-tuning. arXiv preprint arXiv:2409.20566, 2024b.

张昊天，高明飞，甘哲，Philipp Dufter，Nina Wenzel，Huang Forrest，Dhruti Shah，杜贤志，张博文，李阳浩等。MM1.5: 多模态大语言模型微调的方法、分析与见解。arXiv 预印本 arXiv:2409.20566，2024b。

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. In Findings of the Association for Computational Linguistics: EMNLP 2024, 2024c.

张继文，吴继豪，滕一华，廖明辉，徐诺，肖潇，魏中宇，唐杜宇。Android in the zoo: 面向 GUI 代理的链式行动思维。在计算语言学协会会议论文集:EMNLP 2024，2024c。

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, 2024.

郑博远，苟博宇，Kil Jihyung，孙欢，苏宇。GPT-4V(ision) 是一个通用的网络代理，前提是有基础支持。在第 41 届国际机器学习大会，2024 年。

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In Advances in Neural Information Processing Systems, volume 36, pp. 46595-46623, 2023.

郑连民，蒋伟林，盛颖，庄思远，吴章浩，庄永浩，林子，李卓翰，李大成，Eric Xing，张昊，Joseph E Gonzalez，Ion Stoica。用 MT-Bench 和 Chatbot Arena 评估大语言模型作为裁判的能力。在神经信息处理系统进展，卷 36，第 46595-46623 页，2023 年。

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. WebArena: A realistic web environment for building autonomous agents. In The Twelfth International Conference on Learning Representations, 2024.

周淑妍，Frank F Xu，朱浩，周旭辉，Robert Lo，Abishek Sridhar，程贤义，欧天岳，Yonatan Bisk，Daniel Fried 等。WebArena: 一个用于构建自主代理的真实网络环境。在第十二届国际学习表征会议，2024 年。

Table of Contents in Appendix

附录目录

A RELATED WORK

相关工作

GUI Agents. LLMs and MLLMs have demonstrated great capabilities and potentials in GUI automation, working as digital agents in various GUI environments (Yan et al. 2023; Kim et al. 2023; Wang et al. 2024a; Zheng et al. 2024; Xie et al. 2024). Despite the growing number of studies focused on building multimodal agents (Koh et al. 2024; Zhou et al. 2024; Cao et al. 2024), most work still relies on HTML or a11y trees for grounding, even when they are not used for observation. In this work, we advance an alternative line of research: pixel-level visually grounded GUI agents (Shaw et al. 2023; Zhan & Zhang, 2023; Hong et al. 2024; Cheng et al. 2024; Niu et al. 2024). Unlike nearly all previous work of this line, we propose a generic two-stage approach that separates planning and visual grounding to build vision-only GUI agents, which perform remarkably well on realistic agent benchmarks with vision-only input, and offers the flexibility to the choices of planning and grounding models.

GUI 代理。大型语言模型 (LLMs) 和多模态大型语言模型 (MLLMs) 在 GUI 自动化方面展现了强大的能力和潜力，作为数字代理在各种 GUI 环境中工作 (Yan 等，2023；Kim 等，2023；Wang 等，2024a；Zheng 等，2024；Xie 等，2024)。尽管越来越多的研究致力于构建多模态代理 (Koh 等，2024；Zhou 等，2024；Cao 等，2024)，但大多数工作仍依赖于 HTML 或辅助功能树 (a11y trees) 进行定位，即使它们未被用于观察。在本工作中，我们推进了一条替代研究路线: 基于像素级视觉定位的 GUI 代理 (Shaw 等，2023；Zhan & Zhang，2023；Hong 等，2024；Cheng 等，2024；Niu 等，2024)。与该领域几乎所有先前工作不同，我们提出了一种通用的两阶段方法，将规划与视觉定位分离，构建仅依赖视觉输入的 GUI 代理，在现实代理基准测试中表现出色，并为规划和定位模型的选择提供了灵活性。

Visual Grounding. Visual grounding has been long studied on natural images (Karpathy et al. 2014, Mao et al. 2016; Yu et al. 2016). More recently, with the advancements of MLLMs, their visual grounding capabilities on natural images have attracted significant attention (Bai et al. 2023; Chen et al. 2023a b. Peng et al. 2023; Wang et al. 2024c; 2023; Ma et al. 2024). However, due to significant gaps in image resolution and GUI understanding, these models trained on natural contexts work poorly on GUI visual grounding (Cheng et al. 2024). One of the most popular approaches, SoM (Yang et al. 2023), proposes a visual prompting method that adds marks such as boxes and numbers to images and instructs MLLMs to identify the referred objects by the labels. It is widely adopted in GUI scenarios (Yan et al. 2023; He et al. 2024; Koh et al. 2024), but still suffers from problems including reliance on complete object information or object segmentation. Only few studies have been conducted for visual grounding on GUI screenshots. Based on Rico (Deka et al. 2017), Bai et al. (2021) annotates referring expressions by humans; RicoSCA (Li et al. 2020a) generates a larger synthetic referring expression dataset; and Li et al. (2020b) collect human-labeled captions of UI elements. They have been primary resources for GUI grounding for a long time (Li & Li 2022) Banerjee et al. 2022). Later on, Qian et al. (2024) synthesize referring expressions from Rico by heuristic rules and train a vision language model by a new layout-aware contrastive learning technique. CogAgent (Hong et al. 2024) compiles HTML documents and screenshots from real websites to GUI grounding data for the pretraining stage, and finetunes on open-source and in-house human-labeled data, to build a 18B MLLM with strong pixel-level GUI grounding capabilities. Ferret-UI (You et al. 2024) develops a UI generalist MLLM trained on a series of UI-related tasks including grounding. Omniparser (Lu et al. 2024) trains an element detection model for GUIs, which highlight another direction for GUI visual grounding. The most similar effort to ours is SeeClick (Cheng et al. 2024), which enhances Qwen-VL (Bai et al. 2023) by finetuning on GUI grounding data, including simplistic synthetic data compiled from real websites. It still falls short of the small image resolution of Qwen-VL, as well as the simplistic nature of the training data. Cheng et al. (2024) also create a new grounding benchmark for GUIs, which benefits our evaluation and analysis.

视觉定位。视觉定位长期以来一直在自然图像上进行研究 (Karpathy 等，2014；Mao 等，2016；Yu 等，2016)。近年来，随着多模态大型语言模型 (MLLMs) 的进步，其在自然图像上的视觉定位能力引起了广泛关注 (Bai 等，2023；Chen 等，2023a b；Peng 等，2023；Wang 等，2024c；Ma 等，2024)。然而，由于图像分辨率和 GUI 理解上的显著差异，这些在自然场景中训练的模型在 GUI 视觉定位上表现不佳 (Cheng 等，2024)。一种最流行的方法 SoM(Yang 等，2023) 提出了一种视觉提示方法，通过在图像上添加框和数字等标记，指导 MLLMs 通过标签识别所指对象。该方法被广泛应用于 GUI 场景 (Yan 等，2023；He 等，2024；Koh 等，2024)，但仍存在依赖完整对象信息或对象分割的问题。针对 GUI 截图的视觉定位研究较少。基于 Rico 数据集 (Deka 等，2017)，Bai 等 (2021) 由人工标注指代表达；RicoSCA(Li 等，2020a) 生成了更大规模的合成指代表达数据集；Li 等 (2020b) 收集了 UI 元素的人为标注描述。这些长期以来一直是 GUI 定位的主要资源 (Li & Li，2022；Banerjee 等，2022)。随后，Qian 等 (2024) 通过启发式规则从 Rico 合成指代表达，并采用新颖的布局感知对比学习技术训练视觉语言模型。CogAgent(Hong 等，2024) 汇编了真实网站的 HTML 文档和截图作为 GUI 定位数据用于预训练阶段，并在开源及内部人工标注数据上微调，构建了具备强大像素级 GUI 定位能力的 18B MLLM。Ferret-UI(You 等，2024) 开发了一个 UI 通用型 MLLM，训练涵盖包括定位在内的一系列 UI 相关任务。Omniparser(Lu 等，2024) 训练了一个 GUI 元素检测模型，开辟了 GUI 视觉定位的另一方向。与我们最相似的工作是 SeeClick(Cheng 等，2024)，其通过在 GUI 定位数据上微调 Qwen-VL(Bai 等，2023)，包括从真实网站编译的简易合成数据，但仍受限于 Qwen-VL 较小的图像分辨率及训练数据的简单性。Cheng 等 (2024) 还创建了一个新的 GUI 定位基准，有助于我们的评估与分析。

B PHILOSOPHY BEHIND SEEACT-V AND UGROUND

SeeAct-V 与 UGround 背后的理念

When it comes to agent designs, the current wisdom, by and large, is to train a monolithic LLM (e.g., CogAgent (Hong et al. 2024), SeeClick (Cheng et al. 2024), along with several recent supervised fine-tuning endeavors aimed at enhancing ”agentic behaviors”). At a philosophical level, part of the goal of SeeAct-V is to challenge that status quo and advocate a modular design for language agents instead.

谈及代理设计，目前的主流观点大致是训练一个整体式的大型语言模型 (如 CogAgent(Hong 等，2024)、SeeClick(Cheng 等，2024) 以及近期若干旨在增强“代理行为”的监督微调工作)。在哲学层面上，SeeAct-V 的部分目标是挑战这一现状，倡导语言代理的模块化设计。

A fundamental challenge of language agents arises from the complexity, dynamism, and inherent idiosyncrasies of the environments in which they operate. For instance, consider web agents: the internet comprises over one billion websites, each of which can exhibit an extremely large and dynamic number of states, and each can be constantly changing (for example, due to frequent updates in backend databases). Furthermore, there is a considerable amount of highly idiosyncratic semantics in each environment, e.g., uncommon icons, jargon, and counter-intuitive designs.

语言代理面临的一个根本挑战源于其所处环境的复杂性、动态性及固有的特殊性。例如，考虑网络代理: 互联网包含超过十亿个网站，每个网站可能呈现极其庞大且动态变化的状态数量，并且这些状态不断变化 (例如，由于后端数据库的频繁更新)。此外，每个环境中存在大量高度特殊的语义，例如不常见的图标、行话和反直觉的设计。

As a result, although we are still at the early stage of agent research, we posit that a monolithic model, regardless of its future scale and capabilities, is unlikely to fully encapsulate the diverse complexities and idiosyncrasies across all environments. Therefore, developing a generalist agent that reliably generalizes across various contexts necessitates a modular system design. This involves synergistically orchestrating a foundation model (e.g., GPT-4o) with multiple specialized modules, each tailored to specific functionalities.

因此，尽管我们仍处于代理研究的早期阶段，我们认为单一模型，无论其未来规模和能力如何，都不太可能完全涵盖所有环境中多样的复杂性和特殊性。因此，开发一个能够可靠地跨多种情境泛化的通用代理，需要采用模块化系统设计。这涉及协同调度一个基础模型 (例如 GPT-4o) 与多个专门模块，每个模块针对特定功能进行定制。

Grounding, in particular, is a capability for which a dedicated module is highly advantageous. Fundamentally, grounding involves interpreting domain-specific semantics and creating a map between that and natural language representations understood by a generic LLM. A specialized grounding module simplifies the capture of idiosyncratic semantics and facilitates easier adaptation across different domains (for example, by fine-tuning the grounding model rather than the entire foundation model). Consequently, the grounding module provides domain-specific semantic input to the foundation model. This constitutes a central motivation for the design of SeeAct-V and the work presented herein.

特别地，Grounding(语义落地) 是一项极其适合专门模块实现的能力。根本上，语义落地涉及解释领域特定的语义，并在该语义与通用大语言模型 (LLM) 理解的自然语言表示之间建立映射。专门的语义落地模块简化了对特殊语义的捕捉，并促进了跨不同领域的适应 (例如，通过微调语义落地模型而非整个基础模型)。因此，语义落地模块为基础模型提供领域特定的语义输入。这构成了 SeeAct-V 设计及本文工作的核心动机。

Our design also offers several practical advantages:

我们的设计还提供了若干实际优势:

Modularity: It permits the independent study and enhancement of UGround as a standalone grounding model, decoupled from specific planning modules.

模块化: 允许将 UGround 作为独立的语义落地模型进行独立研究和改进，且与具体的规划模块解耦。

Flexibility: It is compatible with diverse multimodal LLMs and grounding models without requiring specialized fine-tuning on downstream benchmarks.

灵活性: 兼容多种多模态大语言模型和语义落地模型，无需在下游基准上进行专门微调。

Comparative Consistency: By standardizing the planning stage, the design minimizes confounding variables, thereby facilitating a clearer assessment of how various grounding models and methods influence agent performance.

比较一致性: 通过标准化规划阶段，设计最大限度地减少了混杂变量，从而更清晰地评估不同语义落地模型和方法对代理性能的影响。

Empirical results demonstrate that SeeAct-V, when integrated with UGround, outperforms end-to-end MLLMs (whether employing textual or SoM grounding). This is particularly noteworthy considering that training end-to-end models demands extensive high-quality data on agent trajectories (which combine both planning and grounding), which is both challenging and costly.

实证结果表明，SeeAct-V 与 UGround 集成时，性能优于端到端多模态大语言模型 (无论采用文本还是 SoM 语义落地)。这尤其值得注意，因为训练端到端模型需要大量高质量的代理轨迹数据 (结合规划和语义落地)，这既具有挑战性又成本高昂。

C FURTHER ABLATION STUDIES

C 进一步消融研究

In addition to the studies in §3.5, we present further ablation experiments to investigate both model design choices and the effectiveness of our web-based synthetic dataset. We report grounding accuracy on ScreenSpot (Agent Setting), with GPT-40 as the planner.

除了 §3.5 中的研究外，我们还进行了更多消融实验，以探讨模型设计选择及我们基于网络的合成数据集的有效性。我们报告了在 ScreenSpot(代理设置) 上的语义落地准确率，规划器采用 GPT-40。

C.1 CONTROLLED COMPARISON TO BASELINE MODELS

C.1 与基线模型的受控比较

Both model design and training data contribute critically to the strong performance of UGround. To isolate their individual contributions, we introduce a new variant, UGround-Qwen, which is fine-tuned

模型设计和训练数据均对 UGround 的优异表现起关键作用。为分离它们的个别贡献，我们引入了一个新变体 UGround-Qwen，该模型经过微调

Table C.1: Ablations of data and base models for UGround on ScreenSpot (Agent Setting).

表 C.1:UGround 在 ScreenSpot(代理设置) 上的数据和基础模型消融实验。

Model	Model Design	Continual SFT Data	Mobile		Desktop		Web		Avg
			Text	Icon/Widget	Text	Icon/Widget	Text	Icon/Widget	
Qwen-VL-Chat	Owen-VL	None	21.3	21.4	18.6	10.7	9.1	5.8	14.5
SeeClick	Qwen-VL	Full SeeClick	81.0	59.8	69.6	33.6	43.9	26.2	52.3
UGround-Qwen	Qwen-VL	Web-Hybrid	80.2	57.2	76.3	39.3	74.4	47.1	62.4
UGround	Ours	Web-Hybrid	89.0	73.4	88.1	61.4	84.8	64.6	76.9

模型	模型设计	持续微调数据	移动端		桌面端		网页端		平均
			文本	图标/组件	文本	图标/组件	文本	图标/组件	
Qwen-VL-Chat	Owen-VL	无	21.3	21.4	18.6	10.7	9.1	5.8	14.5
SeeClick	Qwen-VL	完整 SeeClick	81.0	59.8	69.6	33.6	43.9	26.2	52.3
UGround-Qwen	Qwen-VL	网页混合	80.2	57.2	76.3	39.3	74.4	47.1	62.4
UGround	我们的	网页混合	89.0	73.4	88.1	61.4	84.8	64.6	76.9

Table C.2: Ablations of image resolution for UGround on ScreenSpot (Agent Setting).

表 C.2:UGround 在 ScreenSpot(代理设置) 上的图像分辨率消融实验。

Continual SFT Data	Image Resolution	Mobile		Desktop		Web		Avg.
		Text	Icon/Widget	Text	Icon/Widget	Text	Icon/Widget	
Web-Hybrid	Fixed 448 x 448	89.4	65.1	83.5	56.4	77.0	61.7	72.2
	Fixed 896 x 896	86.8	69.0	85.1	62.9	81.4	57.8	73.8
	Fixed 1,344 x 1,344	79.9	68.6	86.1	62.1	79.1	63.6	73.2
	Dynamic (Ours)	89.0	73.4	88.1	61.4	84.8	64.6	76.9

持续微调数据	图像分辨率	移动端		桌面端		网页端		平均
		文本	图标/小部件	文本	图标/小部件	文本	图标/小部件	
网页混合	固定 448 x 448	89.4	65.1	83.5	56.4	77.0	61.7	72.2
	固定 896 x 896	86.8	69.0	85.1	62.9	81.4	57.8	73.8
	固定 1,344 x 1,344	79.9	68.6	86.1	62.1	79.1	63.6	73.2
	动态 (本方法)	89.0	73.4	88.1	61.4	84.8	64.6	76.9

from Qwen-VL-Chat (the same backbone used in SeeClick), using only our main web-based synthetic dataset, Web-Hybrid The results are presented in Table C.1

来自 Qwen-VL-Chat(与 SeeClick 使用相同的骨干网络)，仅使用我们的主要基于网络的合成数据集 Web-Hybrid，结果见表 C.1

Training Data: When using the same backbone (Qwen-VL-Chat), UGround-Qwen trained solely on Web-Hybrid achieves an average absolute improvement of 10.1% over SeeClick, even though SeeClick incorporates additional open-source mobile UI data. This result underscores both the high quality of our synthetic web data and its capability to generalize across platforms.

训练数据: 在使用相同骨干网络 (Qwen-VL-Chat) 的情况下，UGround-Qwen 仅在 Web-Hybrid 上训练，平均绝对提升 10.1%，优于 SeeClick，尽管 SeeClick 整合了额外的开源移动 UI 数据。该结果强调了我们合成网络数据的高质量及其跨平台泛化能力。

Model Design: UGround demonstrates a 14.5% absolute improvement over UGround-Qwen, thereby highlighting the effectiveness of our model design.

模型设计:UGround 相较于 UGround-Qwen 实现了 14.5% 的绝对提升，凸显了我们模型设计的有效性。

We omit comparisons with CogAgent due to its inferior performance relative to SeeClick, despite its substantially larger model size (18B parameters) and dataset (140M grounding samples).

由于 CogAgent 性能不及 SeeClick，尽管其模型规模 (180 亿参数) 和数据集 (1.4 亿定位样本) 显著更大，我们省略了与其的比较。

C.2 MODEL DESIGN

C.2 模型设计

We analyze the effect of image resolution on performance, focusing on two key aspects: (1) the impact of increasing image resolution using scaled-up AnyRes grid settings, and (2) the benefits of dynamic resolution and aspect ratio adjustments compared to fixed square configurations.

我们分析了图像分辨率对性能的影响，重点关注两个方面:(1) 使用放大 AnyRes 网格设置提升图像分辨率的影响；(2) 动态分辨率和纵横比调整相较于固定正方形配置的优势。

Scaling of Image Resolution. We scale up image resolution with fixed square sizes for convenience (448 x 448 $\rightarrow$ 896 x 896 $\rightarrow$ 1
.

图像分辨率的放大。为方便起见，我们采用固定正方形尺寸放大图像分辨率 (448 x 448 $\rightarrow$ 896 x 896 $\rightarrow$ 1,344 x 1,344) 。

As shown in Table C.2, larger image resolution generally improves the model performance, particularly on web and desktop UIs that often contain small links and icons. However, mobile UIs, as being less dense, do not benefit as significantly from increased resolution.

如表 C.2 所示，更高的图像分辨率通常提升模型性能，尤其是在包含小链接和图标的网页及桌面 UI 上。然而，移动 UI 因密度较低，分辨率提升带来的益处不明显。

Dynamic Image Resolution and Aspect Ratio. As shown in Table C.2, UGround benefits from dynamic image resolution supported by AnyRes, effectively adapting to varied resolutions and aspect ratios (for example, to mobile UIs or desktop UIs). This flexibility results in improved performance across platforms. For example, on desktop and web UIs, UGround achieves comparable or superior results using approximately 2/3 of the tokens required by the fixed 1,344 x 1,344 model in 16:9 scenarios.

动态图像分辨率与纵横比。如表 C.2 所示，UGround 受益于 AnyRes 支持的动态图像分辨率，能有效适应多样的分辨率和纵横比 (例如移动 UI 或桌面 UI)，这种灵活性提升了跨平台性能。例如，在桌面和网页 UI 上，UGround 在 16:9 场景中使用约固定 1,344 x 1,344 模型所需令牌的 2/3，实现了相当或更优的结果。

Similar findings around these two aspects are also discussed in general domains (Li et al., 2024a) Zhang et al. 2024b), as well as some concurrent GUI works (Chen et al. 2024) Li et al. 2024c).

关于这两个方面的类似发现也在通用领域 (Li et al., 2024a；Zhang et al., 2024b) 及部分同期 GUI 研究 (Chen et al., 2024；Li et al., 2024c) 中有所讨论。

C.3 RE TYPES

C.3 RE 类型

The taxonomy for REs introduced in this work represents a novel contribution and has not been addressed in prior studies (Li et al. 2020b) Hong et al. 2024; Cheng et al. 2024). In this section, we present ablation studies focused on the role of positional REs. We omit detailed studies on

本工作提出的 RE 分类法是一项创新贡献，先前研究 (Li et al., 2020b；Hong et al., 2024；Cheng et al., 2024) 未涉及。本节展示了关于位置 RE 作用的消融研究，省略了详细研究

---

[7] The data is converted to the format used in SeeClick. Given the maximum sequence length used in the training of Qwen-VL and SeeClick, we reduce the elements to a maximum of 30 for each page.

[7] 数据已转换为 SeeClick 使用的格式。鉴于 Qwen-VL 和 SeeClick 训练时的最大序列长度，我们将每页元素数量限制为最多 30 个。

Table C.3: RE ablations for UGround on ScreenSpot (Agent Setting).

表 C.3:ScreenSpot(代理设置) 上 UGround 的 RE 消融实验。

Training Data	Mobile		Desktop		Web		Average
	Text	Icon/Widget	Text	Icon/Widget	Text	Icon/Widget	
Web-Hybrid (w/o Pos REs)	86.5	73.4	87.1	61.4	82.2	65.5	76.0
Web-Hybrid	89.0	73.4	88.1	61.4	84.8	64.6	76.9

训练数据	移动端		桌面端		网页端		平均值
	文本	图标/小部件	文本	图标/小部件	文本	图标/小部件	
网页混合 (无位置关系)	86.5	73.4	87.1	61.4	82.2	65.5	76.0
网页混合	89.0	73.4	88.1	61.4	84.8	64.6	76.9

Table C.4: RE ablations for UGround on ScreenSpot (Standard Setting).

表 C.4:UGround 在 ScreenSpot(标准设置) 上的关系抽取消融实验。

Training Data	Mobile		Desktop		Web		Average
	Text	Icon/Widget	Text	Icon/Widget	Text	Icon/Widget	
Web-Hybrid (w/o Pos REs)	72.2	52.0	72.7	55.0	76.5	61.2	64.9
Web-Hybrid	75.5	54.2	79.9	58.6	77.0	68.0	68.8

训练数据	移动端		桌面端		网页端		平均值
	文本	图标/小部件	文本	图标/小部件	文本	图标/小部件	
网页混合 (无位置正则表达式)	72.2	52.0	72.7	55.0	76.5	61.2	64.9
网页混合	75.5	54.2	79.9	58.6	77.0	68.0	68.8

visual and functional REs because (1) they are interleaved in HTML DOMs and are challenging to fully disentangle, and (2) they have been extensively studied in prior work. For example, an HTML attribute (e.g., aria-label) may convey both visual and functional cues, and the MLLM can exploit different aspects of the input.

视觉和功能性 REs，因为 (1) 它们在 HTML DOM 中交织在一起，难以完全分离，(2) 它们在先前的研究中已被广泛研究。例如，HTML 属性 (如 aria-label) 可能同时传达视觉和功能性线索，MLLM 可以利用输入的不同方面。

We train a new checkpoint with Web-Hybrid, omitting all positional REs while maintaining the overall number of web elements. As shown in Table C.3 and Table C.4, the inclusion of positional REs generally enhances model performance.

我们使用 Web-Hybrid 训练了一个新的检查点，省略了所有位置 REs，同时保持网页元素的总体数量。如表 C.3 和表 C.4 所示，包含位置 REs 通常能提升模型性能。

We hypothesize that the integration of positional and contextual data enables the model to more effectively capture and attend to the spatial relationships among UI elements. This enhanced contextual understanding is crucial for grounding tasks that cannot rely solely on visual or functional cues, especially in challenging cases where those cues alone are insufficient.

我们假设位置和上下文数据的整合使模型能够更有效地捕捉和关注 UI 元素之间的空间关系。这种增强的上下文理解对于依赖视觉或功能线索不足以完成的定位任务尤为关键，尤其是在这些线索单独不足以应对的复杂情况下。

D EXAMPLES

D 示例

D.1 MULTIMODAL-MIND2WEB

D.1 多模态-Mind2Web

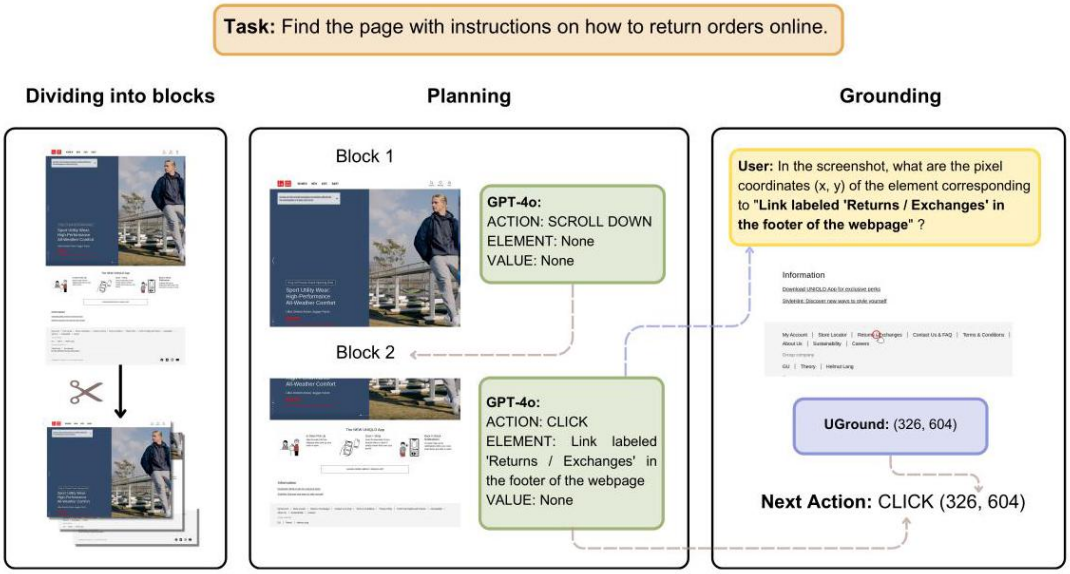


Figure D.1: Example of the Multimodal-Mind2Web evaluation pipeline.

图 D.1: 多模态-Mind2Web 评估流程示例。

D.2 ANDROIDCONTROL

D.2 AndroidControl

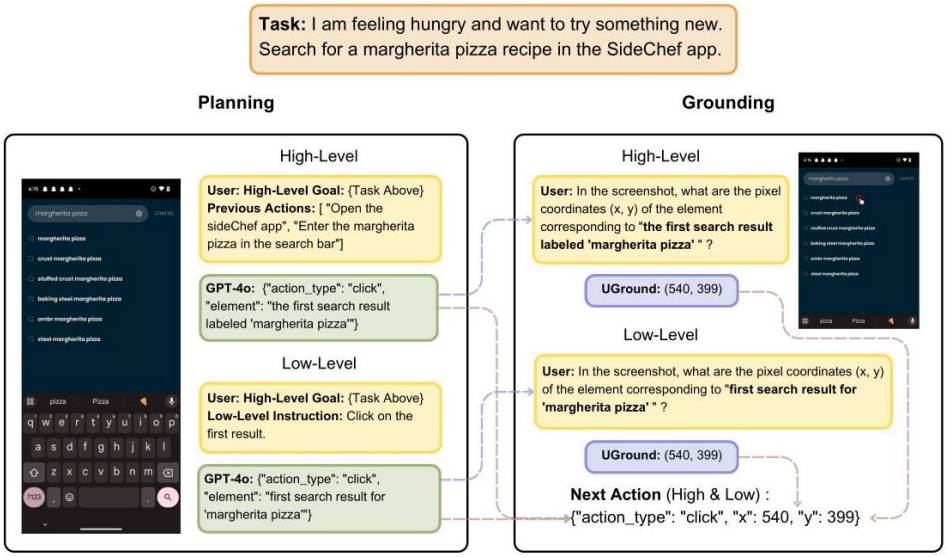


Figure D.2: Example of the AndroidControl evaluation pipeline.

图 D.2:AndroidControl 评估流程示例。

D.3 OMNIACT

D.3 OmniACT

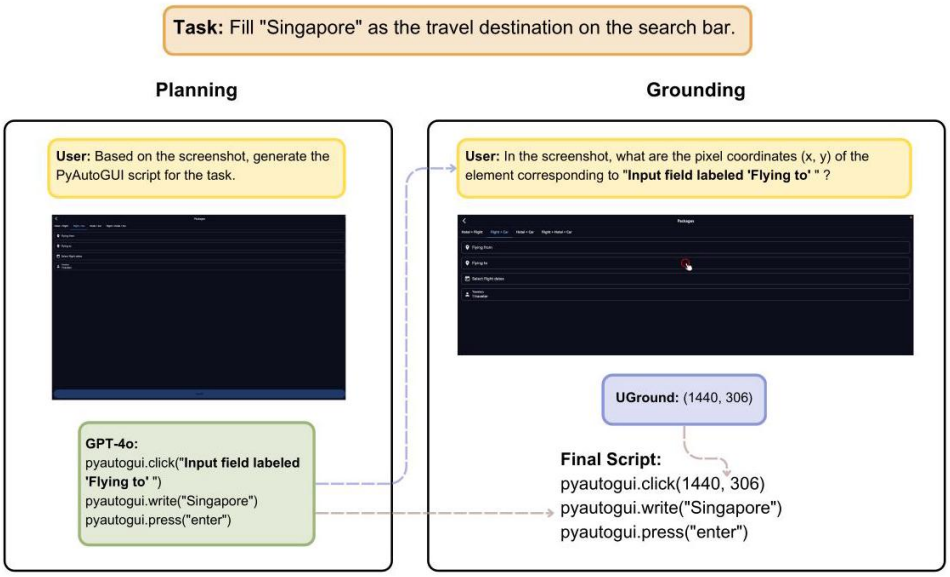


Figure D.3: Example of the OmniACT evaluation pipeline.

图 D.3:OmniACT 评估流程示例。

D.4 TRAINING DATA

D.4 训练数据

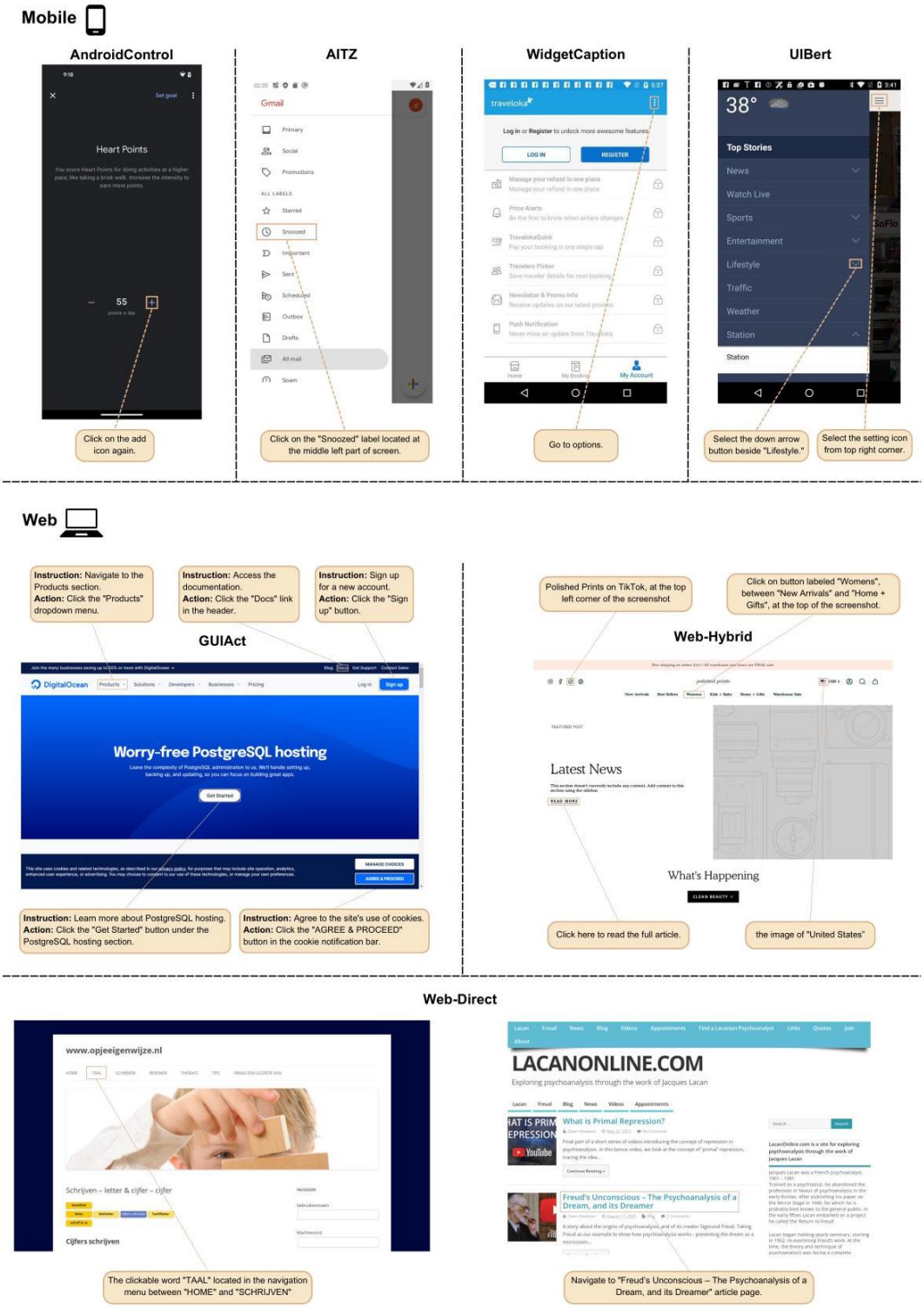


Figure D.4: Examples of training data from different sources.

图 D.4: 来自不同来源的训练数据示例。

E DATA CONSTRUCTION

E 数据构建

We describe the details of our data construction in this section. Illustrative examples of all our training data are provided in Figure D.4

本节介绍我们数据构建的详细内容。所有训练数据的示例见图 D.4。

E.1 WEB-HYBRID

E.1 Web-Hybrid

Following prior work (Hong et al. 2024; Cheng et al. 2024), we download and randomly sample from the latest Common Crawls We apply several filtering methods to exclude non-webpage files based on URL patterns and to remove non-English pages as indicated by the language labels provided by Common Crawl. We employ Playwright to load and render webpages, capture screenshots, and collect metadata for web elements. To ensure a diverse set of data, we simulate vertical scrolling to capture screenshots and elements at various positions on each webpage. The metadata for each element includes bounding box coordinates and relevant HTML attributes, such as the element's tag, inner text (inner_text), and alternative text (e.g., alt).

继承先前工作 (Hong et al. 2024；Cheng et al. 2024)，我们下载并随机抽样最新的 Common Crawl 数据。我们应用多种过滤方法，根据 URL 模式排除非网页文件，并根据 Common Crawl 提供的语言标签去除非英文页面。我们使用 Playwright 加载和渲染网页，截取屏幕截图，并收集网页元素的元数据。为确保数据多样性，我们模拟垂直滚动，捕获网页上不同位置的截图和元素。每个元素的元数据包括边界框坐标和相关 HTML 属性，如元素标签、内部文本 (inner_text) 和替代文本 (如 alt)。

During rendering, we randomly select image sizes to cover a diverse range of resolutions and aspect ratios. Approximately one-third of the data is rendered in mobile-friendly aspect ratios, thereby triggering the mobile version of certain websites and enhancing the coverage of mobile UI environments. For each long webpage, up to three blocks of content within a viewport-sized area are randomly sampled to ensure content diversity. In total, the dataset comprises approximately 773K screenshots from around 700K URLs.

在渲染过程中，我们随机选择图像尺寸以覆盖多样的分辨率和纵横比。约三分之一的数据采用适合移动设备的纵横比，从而触发某些网站的移动版本，增强移动界面环境的覆盖。对于每个长网页，随机采样最多三个视口大小区域内的内容块，以确保内容多样性。整个数据集包含约 77.3 万张截图，来源于约 70 万个 URL。

As detailed in §2.2, we employ a hybrid strategy to generate REs for webpage elements. Below, we first describe how we leverage MLLMs (LLaVA-NeXT-13B) and LLMs (Llama-3-8B) to generate concise, element-level descriptions without positional or contextual information.

如 §2.2 所述，我们采用混合策略生成网页元素的指称表达 (REs)。下面首先介绍如何利用多模态大模型 (MLLMs，LLaVA-NeXT-13B) 和大语言模型 (LLMs，Llama-3-8B) 生成简洁的元素级描述，且不包含位置信息或上下文信息。

We extract the bounding box regions from the webpage screenshots corresponding to the elements and pass these smaller cropped element images along with their salient HTML attributes to LLaVA. Using the prompts outlined below, we prompt LLaVA to generate an element description based on its internal knowledge, the element's image, and relevant HTML attributes:

我们从网页截图中提取对应元素的边界框区域，将这些裁剪后的元素图像及其显著的 HTML 属性一并输入 LLaVA。使用以下提示，促使 LLaVA 基于其内部知识、元素图像及相关 HTML 属性生成元素描述:

Based on the attached image of a web element, please provide a short description of the web element displayed. The goal is to capture the intuitive and visual appearance of the element. Use the accompanying HTML information as context but focus more on describing what is visually observable. Avoid directly referencing HTML attributes; instead, interpret their possible visual implications if they can be inferred from the image. Be cautious of potential inaccuracies in the HTML attributes and use them to enhance understanding only when they align reasonably with what can be inferred visually.

基于所附网页元素图像，请简要描述所显示的网页元素。目标是捕捉元素的直观视觉外观。以随附的 HTML 信息为上下文，但重点描述视觉上可观察到的内容。避免直接引用 HTML 属性；如能从图像推断其可能的视觉含义，可进行解释。注意 HTML 属性可能存在不准确之处，仅当其与视觉推断合理一致时，才用以辅助理解。

HTML: {A list of salient HTML attributes}

HTML: {一组显著的 HTML 属性列表}

We observe that since the input to LLaVA is a small cropped image, the model tends to have less hallucinations compared to directly caption an element with a bounding box overlaid in the image. However, due to the limited language capabilities of the 13B LLaVA model, the generated interpretations tend to be lengthy. To address this, the lengthy output is subsequently processed by Llama-3-8B with the prompt below that instructs it to condense the description into a brief referring expression:

我们观察到，由于输入 LLaVA 的是小幅裁剪图像，模型相较于直接在图像上叠加边界框进行描述时，产生幻觉的情况较少。然而，受限于 13B LLaVA 模型的语言能力，生成的解释往往较长。为此，随后使用 Llama-3-8B 对冗长输出进行处理，采用以下提示将描述压缩为简短的指称表达:

Here is a description of an element in a webpage. Using the detailed description provided, create a concise phrase that captures the essential visual and functional characteristics of the web element. The rephrased description should be straightforward, simple and precise enough to allow humans quickly spot this element in a webpage screenshot. Focus on the most prominent visual features and any critical function indicated by the text.

以下是网页元素的描述。请根据提供的详细描述，创建一个简洁短语，捕捉该网页元素的核心视觉和功能特征。重述的描述应直截了当、简明且精准，便于人类快速在网页截图中识别该元素。重点突出最显著的视觉特征及文本所指示的关键功能。

Description: {}

描述: {}

Leave only your final description in the answer, without any explanation.

答案中仅保留最终描述，不要添加任何解释。

Next, the generation process for each crawled element is as follows.

接下来，每个爬取元素的生成流程如下。

We begin by categorizing the webpage elements based on their tags into two groups: interactive elements (e.g., a, input, select, etc.) and pure text elements (e.g., p, h1, h2, etc.). Referring expressions are generated only

我们首先根据标签将网页元素分为两类: 交互元素 (如 a、input、select 等) 和纯文本元素 (如 p、h1、h2 等)。仅为交互元素生成指称表达

8CC-MAIN-2023-50

8CC-MAIN-2023-50

Table E.1: Statistics of element types (by HTML tags) in Web-Hybrid (%).

表 E.1:Web-Hybrid 中元素类型 (按 HTML 标签划分) 的统计数据 (%)。

a	img	button	input	svg	select	textarea	video
68.99	15.41	6.81	5.32	2.25	0.99	0.18	0.04

a	图像	按钮	输入	矢量图形 (SVG)	选择	多行文本框	视频
68.99	15.41	6.81	5.32	2.25	0.99	0.18	0.04

Table E.2: Statistics of element HTML attributes and MLLM-based synthetic REs used in Web-Hybrid (%). Calculated as the number of elements using an attribute/RE divided by the total number of elements.

表 E.2:Web-Hybrid 中元素 HTML 属性和基于多语言大模型 (MLLM) 的合成正则表达式 (RE) 的统计 (%)。计算方法为使用某属性/正则表达式的元素数量除以元素总数。

MLLM-based RE	inner-text	title	alt	aria-label	aria-describedby	placeholder	value
11.19	43.58	20.01	12.25	11.32	0.21	0.06	0.02

基于多模态大语言模型 (MLLM) 的关系抽取	内部文本	标题	替代文本	无障碍标签	无障碍描述	占位符	数值
11.19	43.58	20.01	12.25	11.32	0.21	0.06	0.02

for interactive elements, as these constitute the primary targets in GUI grounding tasks. In addition, pure text elements are utilized as potential sources for referring expression generation.

对于交互元素，因为它们构成了 GUI 定位任务中的主要目标。此外，纯文本元素被用作生成指称表达的潜在来源。

For each interactive element, we first apply an OCR model (EasyOCR) to extract text from the element's bounding box. If the similarity between the OCR-extracted text and the element's inner_text exceeds a threshold of 0.7, the element is considered textual, and the MLLM-based synthesis pipeline is bypassed. This procedure prevents the generation of trivial data (e.g., "Gray links labeled by link text"). Moreover, for textual elements, those sharing identical text with other elements on the same page are filtered out to avoid grounding ambiguities.

对于每个交互元素，我们首先应用 OCR 模型 (EasyOCR) 从元素的边界框中提取文本。如果 OCR 提取的文本与元素的 inner_text 的相似度超过 0.7 阈值，则该元素被视为文本元素，跳过基于多模态大模型 (MLLM) 的合成流程。此过程防止生成琐碎数据 (例如，“由链接文本标记的灰色链接”)。此外，对于文本元素，过滤掉与同一页面上其他元素文本完全相同的元素，以避免定位歧义。

Based on manually crafted rules, we label each element's neighboring elements in various directions (multiple neighbors are allowed), mark the nearest upper h1, h2, or h3 elements (titles), and determine its absolute position (e.g., center of the screenshot, top, top-left corner) to generate position-based referring expressions. We randomly select up to neighboring elements in different directions and randomly pick elements whose distance from the target is within 500 pixels (empirically, always selecting the closest element does not yield the best performance). These are used to generate relative position descriptions. Some of the relative descriptions are further randomly modified to common terms such as "next to" or "between". For contextual references, if an element is identified as a checkbox or radio button based on its HTML properties, it is assumed to have an associated label (e.g., "radio button for Yes"). If such labels are provided in the HTML attributes, they are used directly; otherwise, the nearest element on the same row (or column, if necessary) is selected as the label. Similar procedures are followed for input fields and select boxes. Additional expressions such as "under," "in," or "under section A" are generated based on the hierarchical structure of titles (primarily h1, h2, and h3). Attributes like title, alt, or aria-label are always considered as potential descriptors, typically contributing functional information.

基于手工制定的规则，我们标注每个元素在各个方向上的邻近元素 (允许多个邻居)，标记最近的上方 h1、h2 或 h3 元素 (标题)，并确定其绝对位置 (例如，截图中心、顶部、左上角) 以生成基于位置的指称表达。我们随机选择不同方向上最多若干邻近元素，并随机挑选距离目标元素在 500 像素以内的元素 (经验表明，总是选择最近元素并不能获得最佳效果)。这些用于生成相对位置描述。部分相对描述进一步随机替换为常用词汇，如“旁边”或“之间”。对于上下文引用，如果根据 HTML 属性识别元素为复选框或单选按钮，则假定其有相关联的标签 (例如，“是的单选按钮”)。如果 HTML 属性中提供了此类标签，则直接使用；否则选择同一行 (或必要时同一列) 上最近的元素作为标签。输入框和选择框也遵循类似流程。基于标题层级结构 (主要是 h1、h2 和 h3)，生成诸如“下方”、“内”或“在 A 节下”等附加表达。标题、alt 或 aria-label 等属性始终被视为潜在描述符，通常提供功能性信息。

Finally, for each element, descriptors from accessibility labels, the element's own text, or MLLM-based descriptions are randomly combined with absolute positional information (included on a random basis) and supplemented by between zero and two relative or contextual descriptions. For interactive elements such as radio buttons, the label is always included. In each webpage, up to 100 elements are selected, prioritizing those with accessibility labels or MLLM annotations. The number of pure text elements is limited to no more than three times the sum of elements with accessibility labels and those annotated via MLLMs (with a minimum of 10, or the total available elements, whichever is lower) to reduce the number of pure text elements. Additionally, unique accessibility labels and their frequencies are counted; labels occurring more than 1,000 times are downsampled to a maximum of 1,000 occurrences. For example, the label "Next" appears 13K times, and is downsampled to 1K occurrences in our training data.

最后，对于每个元素，将来自无障碍标签、元素自身文本或基于多模态大模型 (MLLM) 描述的描述符随机与绝对位置信息 (随机包含) 组合，并补充零到两个相对或上下文描述。对于单选按钮等交互元素，标签始终包含在内。每个网页最多选择 100 个元素，优先选择带有无障碍标签或 MLLM 注释的元素。纯文本元素数量限制为无障碍标签元素与 MLLM 注释元素总和的三倍以内 (至少 10 个，或可用元素总数中较小者)，以减少纯文本元素数量。此外，统计唯一无障碍标签及其出现频率；出现超过 1000 次的标签将下采样至最多 1000 次。例如，“下一步”标签出现 1.3 万次，在训练数据中下采样至 1000 次。

To illustrate the primary data distribution, we provide statistics about HTML element types, as well as attributes and positional RE types used in the final REs within Web-Hybrid. The statistics are shown in Table E.1 [Table E.2] and Table E.3 We omit exact percentages of visual and functional REs because they are often interleaved in HTML DOMs and MLLM-based synthetic REs, and generally are hard to distinguish.

为说明主要数据分布，我们提供了 HTML 元素类型、属性及 Web-Hybrid 最终指称表达中使用的位置指称表达类型的统计数据。统计见表 E.1 [表 E.2] 和表 E.3。我们省略了视觉和功能指称表达的具体百分比，因为它们在 HTML DOM 和基于多模态大模型的合成指称表达中常常交织，且通常难以区分。

E.2 WEB-DIRECT

E.2 WEB-DIRECT

For the Web-Direct dataset, we directly employ GPT-4o to generate referring expressions. We observed that,

due to its limited grounded understanding capabilities, simply enclosing an element in the image with a bounding box often leads to notable hallucinations, particularly when it provides descriptions of nearby elements. To mitigate these hallucinations without incurring the high cost of manual post-verification, we find that annotating an element with both a red bounding box and a red arrow pointing to it substantially reduces hallucinations.

对于 Web-Direct 数据集，我们直接使用 GPT-4o 生成指称表达。我们观察到，由于其有限的定位理解能力，仅用边界框框出图像中的元素常导致显著的幻觉，尤其是在描述附近元素时。为减轻这些幻觉且避免高昂的人工后验验证成本，我们发现用红色边界框加上指向该元素的红色箭头标注，能显著减少幻觉现象。

---

[9] https://github.com/JaidedAI/EasyOCR/

[9] https://github.com/JaidedAI/EasyOCR/

Table E.3: Statistics of relative positional REs, absolute Positional REs, and contextual REs used in Web-Hybrid (%). Contextual References are also counted as relative positional REs. Calculated as the number of elements using an RE divided by the total number of elements.

表 E.3:Web-Hybrid 中使用的相对位置指称表达、绝对位置指称表达和上下文指称表达的统计 (百分比)。上下文引用也计入相对位置指称表达。计算方式为使用某指称表达的元素数量除以元素总数。

Relative Positional RE	Contextual RE	Absolute Positional RE
23.49	8.43	3.05

相对位置关系抽取	上下文关系抽取	绝对位置关系抽取
23.49	8.43	3.05

In addition, we explicitly query GPT-40 regarding the identification of the element, which further minimizes potential hallucinations and filters out a small number of crawling errors or occluded elements.

此外，我们明确向 GPT-40 查询元素的识别，这进一步减少了潜在的幻觉现象，并过滤掉少量爬取错误或被遮挡的元素。

Two separate prompts are used in Web-Direct: one to generate free-form referring expressions and another to generate functionally oriented referring expressions:

Web-Direct 中使用了两个独立的提示: 一个用于生成自由形式的指代表达，另一个用于生成功能导向的指代表达:

Here is supposed to be an interactive element (button, link, dropdown, text box, etc.) in the red box pointed by an arrow in the screenshot. Can you find it? Is it visible from the screenshot? Can you write a concise description that is sufficient for humans to locate it from the screenshot? Your response should be a JSON. For example, ”visible”: true, ”description”: ”your description here”.

截图中箭头指向的红框内应有一个交互元素 (按钮、链接、下拉菜单、文本框等)。你能找到它吗? 它在截图中可见吗? 你能写出一个简洁的描述，足以让人从截图中定位它吗? 你的回答应为 JSON 格式。例如，”visible”: true, ”description”: ” 你的描述内容”。

Here is supposed to be an interactive element (button, link, dropdown, text box, etc.) in the red box pointed by an arrow in the screenshot. Can you find it? Is it visible from the screenshot? What unique function does this element enable? Your response should be a JSON. For example, ”visible”: true, ”action”: ”subscribe the latest updates”.

截图中箭头指向的红框内应有一个交互元素 (按钮、链接、下拉菜单、文本框等)。你能找到它吗? 它在截图中可见吗? 该元素具有什么独特功能? 你的回答应为 JSON 格式。例如，”visible”: true, ”action”: ” 订阅最新更新”。

E.3 OPEN-SOURCE DATA

E.3 开源数据

We leverage several high-quality open-source referring expression datasets in Android, as well as the GUIAct dataset, as supplementary sources of web data. Specifically:

我们利用了多个高质量的 Android 开源指代表达数据集，以及 GUIAct 数据集，作为网页数据的补充来源。具体包括:

1. GUIAct: We use the annotated data from GUIAct (web-single). Steps that do not involve coordinates or that are marked as multi-step operations (for example, ”click ... then type”) are filtered out. We use both the Instruction and Action annotations for grounding (i.e., each element is seen in training twice with different expressions).

1. GUIAct: 我们使用 GUIAct(web-single) 中的标注数据。过滤掉不涉及坐标或标记为多步骤操作 (例如，“点击... 然后输入”) 的步骤。我们同时使用 Instruction 和 Action 标注进行定位 (即每个元素在训练中以不同表达出现两次)。

2. AndroidControl: Similarly, we use the human-annotated actions from the training set. We filter out any actions that do not have associated coordinate data, ensuring that only steps with specific visual grounding targets are included in the dataset.

2. AndroidControl: 同样，我们使用训练集中的人工标注动作。过滤掉没有关联坐标数据的动作，确保数据集中仅包含具有具体视觉定位目标的步骤。

3. Widget Caption: For each element in the training set, multiple functional captions are provided. To enhance diversity, two captions per element are randomly selected from the available set of functional captions during data construction.

3. Widget Caption: 对于训练集中的每个元素，提供多个功能性标题。为增强多样性，数据构建时从可用的功能标题集中随机选择每个元素的两个标题。

4. UIBert: We use the training set elements from UIBert without any additional special processing, directly utilizing the referring expressions provided by this dataset.

4. UIBert: 我们直接使用 UIBert 的训练集元素，无需额外特殊处理，直接利用该数据集提供的指代表达。

5. AITZ: We incorporate the annotated actions (Thought) from AITZ, using each step's action annotation for grounding in the dataset. These annotations contribute to a more diverse set of referring expressions, particularly for action-oriented grounding tasks.

5. AITZ: 我们引入 AITZ 中的标注动作 (Thought)，使用每一步的动作标注进行数据集中的定位。这些标注丰富了指代表达的多样性，特别是针对动作导向的定位任务。

F MODEL AND TRAINING DETAILS

F 模型与训练细节

F.1 OVERVIEW

F.1 概述

For flexible investigation of the model architecture, we build the architecture based on LLaVA-NeXT (Liu et al. 2024b), and train from scratch using open-source data from Liu et al. (2024a). We use CLIP-ViT-L-14 (224px) as our base image encoder for more flexible splitting of AnyRes, and keep it frozen during training. We use Vicuna-1.5-7b-16k (Zheng et al. 2023) as the language backbone as a long-context LM backbone for handling long visual contexts.

为了灵活探索模型架构，我们基于 LLaVA-NeXT(Liu et al. 2024b) 构建架构，并使用 Liu et al. (2024a) 的开源数据从零开始训练。我们采用 CLIP-ViT-L-14(224px) 作为基础图像编码器，以便更灵活地拆分 AnyRes，并在训练中保持其冻结状态。语言骨干采用 Vicuna-1.5-7b-16k(Zheng et al. 2023)，作为长上下文语言模型骨干，用于处理长视觉上下文。

F.2 ANYRES

F.2 AnyRes

As described in §2.3, AnyRes allows convenient scaling up of image resolutions, although it's not always beneficial to enlarge image resolutions (Li et al. 2024a). We keep the main pipeline of AnyRes, splitting images into 224px grids. However, to keep the original image aspect ratios, we resize only by width and pad to the bottoms if needed, and use pixel-level coordinates in numbers that are compatible with this design. We allow at most 36 grids, for a maximum resolution of 1,344 x 1,344 and 896 x 2,016. We empirically find AnyRes does not generalize to unseen image resolutions for visual grounding. Therefore, we resize images by width to keep them within the training resolution ranges when needed. We remove the low-resolution image for providing global context, because

it intuitively does not provide informative contexts when images are larger than 1,000px, and we empirically find it slightly hurt the performance.

如 §2.3 所述，AnyRes 允许方便地扩展图像分辨率，尽管放大图像分辨率并非总是有益 (Li et al. 2024a)。我们保持 AnyRes 的主流程，将图像拆分为 224px 网格。但为保持原始图像宽高比，仅按宽度调整大小，必要时在底部填充，并使用与此设计兼容的像素级坐标数值。最多允许 36 个网格，最大分辨率为 1344 x 1344 和 896 x 2016。我们经验发现 AnyRes 对未见过的图像分辨率在视觉定位上泛化能力有限。因此，必要时按宽度调整图像大小，使其保持在训练分辨率范围内。我们移除了用于提供全局上下文的低分辨率图像，因为当图像大于 1000px 时，直观上其不提供有效信息，且经验上发现其略微降低了性能。

F.3 TRAINING

F.3 训练

Our training primarily consists of two stages:

我们的训练主要包括两个阶段:

1. LLaVA-1.5 Pretraining and Finetuning: We follow the exact pretraining in Liu et al. (2024a). Then, in the instruction finetuning stage, we change the grounding data from normalized coordinates to absolute coordinates as we wish, and start to use our modified AnyRes setting.

1. LLaVA-1.5 预训练和微调: 我们严格遵循 Liu 等人 (2024a) 中的预训练方法。然后，在指令微调阶段，我们将定位数据从归一化坐标改为绝对坐标，并开始使用我们修改后的 AnyRes 设置。

2. GUI Visual Grounding: Then we train UGround on our training datasets.

2. GUI 视觉定位: 随后我们在训练数据集上训练 UGround。

Due to the huge computation cost of handling high-resolution images, we use LoRA (Hu et al. 2022) for instruction finetuning in the two stages, with a device batch size of 4.

由于处理高分辨率图像的计算成本极高，我们在两个阶段的指令微调中均采用了 LoRA(Hu 等人, 2022)，设备批量大小为 4。

The first stage takes about 50 hours on a single 4x NVIDIA A100 machine (global batch size 128 with gradient accumulation). For the large-scale GUI data training, we use 112 NVIDIA H100 GPUs and finish the training in about 6 hours (global batch size 448).

第一阶段在单台 4 卡 NVIDIA A100 机器上耗时约 50 小时 (全局批量大小 128，采用梯度累积)。对于大规模 GUI 数据训练，我们使用了 112 块 NVIDIA H100 GPU，训练时间约为 6 小时 (全局批量大小 448)。

G EVALUATION DETAILS

G 评估细节

G.1 MODEL ENDPOINTS

G.1 模型端点

As studied in (Pan et al. 2024), different GPT endpoints could lead to slight differences in the performance of GUI tasks. Hence, we provide the specific endpoint names we use in our evaluation, as well as those of the baselines we use (if available).

如 (Pan 等人，2024) 所研究，不同的 GPT 端点可能导致 GUI 任务性能的细微差异。因此，我们提供了评估中使用的具体端点名称，以及所用基线模型的端点名称 (如有)。

- Ours (across every benchmark): gpt-4-turbo-2024-04-09 and gpt-4o-2024-05-13
  - 我们的模型 (所有基准):gpt-4-turbo-2024-04-09 和 gpt-4o-2024-05-13
- Multimodal-Mind2Web: gpt-4-1106-vision-preview
  - Multimodal-Mind2Web:gpt-4-1106-vision-preview
- OmniACT: gpt-4-0613 and gpt-4-1106-vision-preview
  - OmniACT:gpt-4-0613 和 gpt-4-1106-vision-preview
- Mind2Web-Live: gpt-4-0125-preview and gpt-4o-2024-05-13
  - Mind2Web-Live:gpt-4-0125-preview 和 gpt-4o-2024-05-13
- AndroidWorld: gpt-4-turbo-2024-04-09
  - AndroidWorld:gpt-4-turbo-2024-04-09

G.2 MULTIMODAL-MIND2WEB

G.2 MULTIMODAL-MIND2WEB

Many screenshots in Multimodal-Mind2Web have giant vertical heights (e.g., 1,280 × 10,000 pixels). Similar to Zheng et al. (2024), to avoid overly long screenshots, we divide whole webpage screenshots into viewport-sized blocks, and simulate scrolling down to the next block whenever agents determine that no valid action can be taken or explicitly choose to scroll. Specifically, we divide each full-page screenshot into 1,280 × 1,000 pixel blocks, except for the final block, which may be shorter depending on the page's total height. Most of the target elements are within the first block (about 80%). See Figure D. 1 for an illustrative example of the pipeline.

Multimodal-Mind2Web 中的许多截图具有极高的垂直高度 (例如, 1,280 × 10,000 像素)。类似于 Zheng 等人 (2024), 为避免截图过长, 我们将整页网页截图划分为视口大小的块, 并在代理判断无法执行有效操作或明确选择滚动时，模拟向下滚动至下一块。具体来说，我们将每个整页截图划分为 1,280 × 1,000 像素的块，最后一块可能因页面总高度不同而较短。大多数目标元素位于第一块内 (约 80%)。示意流程见图 D.1。

We report element accuracy on the benchmark, and the grounding is considered to be correct if the output coordinates fall in the box coordinates of the ground truth element.

我们报告了基准测试中的元素准确率，当输出坐标落在真实元素的框坐标内时，视为定位正确。

G.3 ANDROIDCONTROL

G.3 ANDROIDCONTROL

We adopt the M3A (Multimodal Autonomous Agent for Android) prompt (Rawles et al. 2024), the state-of-the-art zero-shot method in Li et al. (2024b). We only make minor modifications to integrate UGround into M3A.

我们采用了 M3A(Multimodal Autonomous Agent for Android，多模态安卓自主代理) 提示 (Rawles 等, 2024)，这是 Li 等 (2024b) 中最先进的零样本方法。我们仅做了少量修改以将 UGround 集成到 M3A 中。

We follow the standard data processing steps outlined in Li et al. (2024b). During evaluation, coordinates generated by grounding models are translated to the smallest visible element that includes the coordinates.

我们遵循 Li 等 (2024b) 中概述的标准数据处理步骤。评估时，定位模型生成的坐标会被转换为包含该坐标的最小可见元素。

G.4 OMNIACT

G.4 OMNIACT

We follow the method in Kapoor et al. (2024) for prompt design and the selection of five in-context examples. The prompt is slightly modified to generate element descriptions as function parameters for PyAutoGUI scripts, instead of directly outputting coordinates. After generating the PyAutoGUI script with element descriptions, we use grounding models to predict the corresponding coordinates and substitute them back into the original script. See Figure D. 3 for an illustrative example of the pipeline.

我们遵循 Kapoor 等 (2024) 的方法进行提示设计和五个上下文示例的选择。提示稍作修改，以生成作为 PyAutoGUI 脚本函数参数的元素描述，而非直接输出坐标。生成带元素描述的 PyAutoGUI 脚本后，我们使用定位模型预测对应坐标，并将其替换回原始脚本。管道示例见图 D.3。

We compare our method with DetACT (Kapoor et al. 2024), the state-of-the-art method in Kapoor et al. (2024), which extracts UI elements and their coordinates through a combination of OCR, icon matching, and color detection. These elements are filtered by task relevance and passed to LLMs or MLLMs to generate the PyAutoGUI script. In contrast, our method does not use a pre-generated elements list. The planner model focuses on generating precise element descriptions based solely on the screenshot. Additionally, we corrected basic errors in the public evaluation scripts (for example, wrong file paths and wrong calculation of distances).

我们将方法与 Kapoor 等 (2024) 中的 DetACT 进行比较，后者通过 OCR、图标匹配和颜色检测结合提取 UI 元素及其坐标。这些元素经过任务相关性筛选后传递给 LLM 或 MLLM 生成 PyAutoGUI 脚本。相比之下，我们的方法不使用预生成的元素列表。规划模型专注于仅基于截图生成精确的元素描述。此外，我们修正了公共评测脚本中的基本错误 (如错误的文件路径和距离计算错误)。

G.5 MinD2WEB-LIVE

G.5 MinD2WEB-LIVE

The baseline agent in Pan et al. (2024) is text-only, perceives and interacts with webpages by hundreds of textual HTML elements at a time. To study vision-only agents, we change the observation to pure screenshots. We also make necessary changes to the standard action space to entirely isolate HTML from the planning, grounding, and execution: 1) We add Scroll_Up and Scroll_Down to the action space to better support vision-only agents with viewport-sized observation. 2) We remove Fill_Form and Fill_Search from the action space, which use an additional judgment model to determine whether to press enter after typing through HTML information. Instead, we use Type and Press_Enter to let the agent make its own decisions autonomously. 3) We disable API-based Select, and force agents to select options merely through clicking and make the action more challenging. We admit some select buttons cannot be easily operated with only Click. We compromise this point to fulfill the motivation of this vision-only study.

Pan 等 (2024) 中的基线代理仅基于文本，通过数百个文本 HTML 元素感知和交互网页。为研究纯视觉代理，我们将观察改为纯截图，并对标准动作空间做必要调整，完全隔离 HTML 与规划、定位及执行:1) 新增 Scroll_Up 和 Scroll_Down 动作，以更好支持视口大小观察的纯视觉代理。2) 移除 Fill_Form 和 Fill_Search 动作，这些动作依赖额外判断模型决定输入后是否按回车，改用 Type 和 Press_Enter 让代理自主决策。3) 禁用基于 API 的 Select，强制代理仅通过点击选择选项，使动作更具挑战性。我们承认部分选择按钮仅靠点击难以操作，但为实现纯视觉研究目标对此做了妥协。

G.6 ANDROIDWORLD

G.6 ANDROIDWORLD

We build SeeAct-V agents based on the M3A agent in Rawles et al. (2024), which receives both raw and SoM images, and reason about the next action in a ReAct style (Yao et al. 2023) and choose the next target element from the element list. It also adopts self-reflection (Shinn et al. 2024) in the agent pipeline to instruct agents to summarize the current move and facilitate the following steps.

我们基于 Rawles 等 (2024) 中的 M3A 代理构建 SeeAct-V 代理，该代理接收原始图像和 SoM 图像，并以 ReAct 风格 (Yao 等，2023) 推理下一步动作，从元素列表中选择目标元素。代理流程中还采用了自我反思 (Shinn 等，2024)，指导代理总结当前动作并促进后续步骤。

We mainly remove SoM images and textual list of elements from the a11y tree in the observation (in both planning and reflection phases), and change element-based actions to pixel-level actions.

我们主要从观察中移除 SoM 图像和 a11y 树中的元素文本列表 (在规划和反思阶段均如此)，并将基于元素的动作改为像素级动作。

H PROMPTS

H PROMPTS

Table H.1: Prompt used for the planning model in Multimodal-Mind2Web, modified from the prompt in (Zheng et al. 2024)

表 H.1:Multimodal-Mind2Web 中规划模型使用的提示，修改自 (Zheng 等，2024) 中的提示

System Role

系统角色

You are imitating humans doing web navigation for a task step by step.

你正在模仿人类逐步完成网页导航任务。

At each stage, you can see the webpage like humans by a screenshot and know the previous actions before the current step through recorded history.

在每个阶段，你可以通过截图像人类一样查看网页，并通过记录的历史了解当前步骤之前的操作。

You need to decide on the first following action to take.

你需要决定接下来要执行的第一个动作。

You can click an element with the mouse, select an option, type text with the keyboard, or scroll down.

你可以用鼠标点击一个元素，选择一个选项，使用键盘输入文本，或向下滚动页面。

Task Description

任务描述

You are asked to complete the following task: {Task description}

请完成以下任务:{任务描述}

Previous Actions: {List of previous actions, if any}

之前的操作:{之前操作列表 (如有)}

The screenshot below shows the webpage you see.

下面的截图显示了你看到的网页。

Useful Guidelines

有用的指导原则

First, observe the current webpage and think through your next step based on the task and previous actions.

首先，观察当前网页，并根据任务和之前的操作思考下一步。

To be successful, it is important to follow the following rules:

要成功，遵循以下规则非常重要:

1. Make sure you understand the task goal to avoid wrong actions.

1. 确保理解任务目标，避免错误操作。

2. Ensure you carefully examine the current screenshot and issue a valid action based on the observation.

2. 仔细检查当前截图，并根据观察发出有效操作。

3. You should only issue one action at a time.

3. 每次只能执行一个操作。

4. The element you want to operate with must be fully visible in the screenshot. If it is only partially visible, you need to SCROLL DOWN to see the entire element.

4. 你要操作的元素必须在截图中完全可见。如果只部分可见，需要向下滚动以查看完整元素。

5. The necessary element to achieve the task goal may be located further down the page. If you don’t want to interact with any elements, simply select SCROLL DOWN to move to the section below.

5. 实现任务目标所需的元素可能位于页面更下方。如果不想与任何元素交互，只需选择向下滚动以移动到下方区域。

Reasoning

推理过程

Explain the action you want to perform and the element you want to operate with (if applicable). Describe your thought process and reason in 3 sentences.

说明您想执行的操作以及您想操作的元素 (如适用)。用三句话描述您的思路和理由。

Output Format

输出格式

Finally, conclude your answer using the format below.

最后，使用以下格式总结您的答案。

Ensure your answer strictly follows the format and requirements provided below, and is clear and precise.

确保您的答案严格遵循以下提供的格式和要求，且清晰准确。

The action, element, and value should each be on three separate lines.

操作、元素和值应分别占三行。

ACTION: Choose an action from CLICK, TYPE, SELECT, SCROLL DOWN. You must choose one of these four, instead of choosing None.

操作: 从 CLICK、TYPE、SELECT、SCROLL DOWN 中选择一个操作。必须选择这四个中的一个，不能选择无操作。

ELEMENT: Provide a description of the element you want to operate. (If ACTION == SCROLL DOWN, this field should be none.)

元素: 提供您想操作的元素描述。(如果操作为 SCROLL DOWN，此字段应为 none。)

It should include the element’s identity, type (button, input field, dropdown menu, tab, etc.), and text on it (if applicable).

应包括元素的标识、类型 (按钮、输入框、下拉菜单、标签页等) 及其上的文本 (如适用)。

Ensure your description is both concise and complete, covering all the necessary information and less than 30 words.

确保您的描述简洁完整，涵盖所有必要信息，且少于 30 个字。

If you find identical elements, specify its location and details to differentiate it from others.

如果发现相同元素，请指明其位置和细节以区分其他元素。

VALUE: Provide additional input based on ACTION.

值: 根据操作提供额外输入。

The VALUE means:

值的含义:

If ACTION == TYPE, specify the text to be typed.

如果操作为 TYPE，指定要输入的文本。

If ACTION == SELECT, specify the option to be chosen.

如果操作为 SELECT，指定要选择的选项。

Otherwise, write 'None'.

否则，填写“None”。

Table H.2: Prompts used for the planning model in AndroidControl, modified from the prompt in (Li et al. 2024b) and (Rawles et al. 2024)

表 H.2:AndroidControl 中用于规划模型的提示，修改自 (Li et al. 2024b) 和 (Rawles et al. 2024) 中的提示。

General Instruction

通用说明

You are an agent who can operate an Android phone on behalf of a user.

你是一个可以代表用户操作安卓手机的代理。

Based on user's goal/request, you may complete some tasks described in the requests/goals by performing actions (step by step) on the phone.

根据用户的目标/请求，你可以通过在手机上逐步执行操作来完成请求/目标中描述的一些任务。

When given a user request, you will try to complete it step by step. At each step, you will be given the current screenshot and a history of what you have done (in text). Based on these pieces of information and the goal, you

must choose to perform one of the action in the following list (action description followed by the JSON format) by outputting the action in the correct JSON format.

当收到用户请求时，你将尝试逐步完成它。每一步，你会获得当前的屏幕截图和你已完成操作的历史 (文本形式)。基于这些信息和目标，你必须选择执行以下列表中的某个操作 (操作描述后跟 JSON 格式)，并以正确的 JSON 格式输出该操作。

- If you think the task has been completed, finish the task by using the status action with complete as goal_status: {”action_type”:”status”,”goal_status”:”successful”}
  - 如果你认为任务已完成，使用状态操作并将 goal_status 设为 complete 来结束任务:{”action_type”:”status”,”goal_status”:”successful”}
- If you think the task is not feasible (including cases like you don’t have enough information or cannot perform some necessary actions), finish by using the ’status’action with infeasible as goal_status: {”action_type”: ”status”, ”goal_status”: ”infeasible”}
  - 如果你认为任务不可行 (包括信息不足或无法执行某些必要操作的情况)，使用状态操作并将 goal_status 设为 infeasible 来结束任务:{”action_type”: ”status”, ”goal_status”: ”infeasible”}
- Click/tap on an element on the screen, describe the element you want to operate with: {”action_type”: ”click”, ”element”: □target_element_description□}
  - 点击/轻触屏幕上的元素，描述你想操作的元素:{”action_type”: ”click”, ”element”: □target_element_description□}
- Long press on an element on the screen, similar with the click action above: {”action_type”: ”long-press”, ”description”: □target_element_description□}
  - 长按屏幕上的元素，类似于上述点击操作:{”action_type”: ”long-press”, ”description”: □target_element_description□}
- Type text into a text field: {”action_type”: ”type_text”, ”text”: □text_input□, ”element”: □target_element_description□}
  - 在文本框中输入文字:{”action_type”: ”type_text”, ”text”: □text_input□, ”element”: □target_element_description□}
- Scroll the screen in one of the four directions: {”action_type”: ”scroll”, ”direction”: □up, down, left, right□ }
  - 向四个方向之一滚动屏幕:{”action_type”: ”scroll”, ”direction”: □up, down, left, right□ }
- Navigate to the home screen: {”action_type”: ”navigate_home”}
  - 返回主屏幕:{”action_type”: ”navigate_home”}

- Navigate back: {”action_type”: ”navigate_back”}

- 返回上一级:{”action_type”: ”navigate_back”}

- Open an app (nothing will happen if the app is not installed): {”action_type”: ”open_app”, ”app_name”: □name□}

- 打开应用 (若未安装则无反应):{”action_type”: ”open_app”, ”app_name”: □name□}

- Wait for the screen to update: {”action_type”: ”wait”}

- 等待屏幕更新:{”action_type”: ”wait”}

Useful Guidelines

有用的指导原则

Here are some useful guidelines you need to follow:

以下是你需要遵循的一些有用指导原则:

General:

通用:

- Usually there will be multiple ways to complete a task, pick the easiest one. Also when something does not work as expected (due to various reasons), sometimes a simple retry can solve the problem, but if it doesn’t (you can see that from the history), SWITCH to other solutions.

- 通常完成任务有多种方法，选择最简单的一种。当某些操作未按预期工作 (原因多样) 时，有时简单重试即可解决问题，但如果重试无效 (可从历史记录中判断)，则切换到其他方案。

- If the desired state is already achieved (e.g., enabling Wi-Fi when it’s already on), you can just complete the task.

- 如果目标状态已达成 (例如 Wi-Fi 已开启时再启用 Wi-Fi)，则可直接完成任务。

Action Related:

操作相关:

- Use the ’open_app’ action whenever you want to open an app (nothing will happen if the app is not installed), do not use the app drawer to open an app unless all other ways have failed.

- 想打开应用时使用“open_app”操作 (若应用未安装则无反应)，除非所有其他方法均失败，否则不要使用应用抽屉打开应用。

- Use the 'type_text' action whenever you want to type something (including password) instead of clicking characters on the keyboard one by one. Sometimes there is some default text in the text field you want to type in, remember to delete them before typing.

- 想输入内容 (包括密码) 时使用“type_text”操作，而非逐个点击键盘字符。若文本框内有默认文字，记得先删除再输入。

- For 'click', 'long-press' and 'type_text', the element you pick must be VISIBLE in the screenshot to interact with it.

- 对于“click”、“long-press”和“type_text”操作，所选元素必须在截图中可见才能进行交互。

- The 'element' field requires a concise yet comprehensive description of the target element in a single sentence, not exceeding 30 words. Include all essential information to uniquely identify the element. If you find identical elements, specify their location and details to differentiate them from others.

- “element”字段需用一句不超过 30 字的简洁且全面的描述，包含所有必要信息以唯一识别目标元素。若存在相同元素，需指明其位置和细节以区分。

- Consider exploring the screen by using the 'scroll' action with different directions to reveal additional content.

- 可通过“scroll”操作配合不同方向探索屏幕，显示更多内容。

- The direction parameter for the 'scroll' action specifies the direction in which the content moves and opposites to swipe; for example, to view content at the bottom, the 'scroll' direction should be set to 'down'.

- “scroll”操作的方向参数指内容移动方向，与滑动方向相反；例如查看底部内容时，scroll 方向应设为“down”。

Text Related Operations:

文本相关操作:

Table H. 2 - Continued from the previous page

表 H.2 - 续前页

- Normally to select certain text on the screen: ⟨i⟩ Enter text selection mode by long pressing the area where the text is, then some of the words near the long press point will be selected (highlighted with two pointers indicating the range) and usually a text selection bar will also appear with options like 'copy', 'paste', 'select all', etc. (ii) Select the exact text you need. Usually the text selected from the previous step is NOT the one you want, you need to adjust the range by dragging the two pointers. If you want to select all text in the text field, simply click the 'select all' button in the bar.

- 通常选取屏幕上的文本: ⟨i⟩ 长按文本区域进入文本选择模式，长按点附近的部分文字会被选中 (用两个指针标示范围)，通常会出现文本选择栏，含“复制”、“粘贴”、“全选”等选项。(ii) 选中所需的准确文本。通常上一步选中的文本不是最终所需，需拖动两个指针调整范围。若想选中文本框内所有文字，直接点击选择栏中的“全选”按钮。

- At this point, you don’t have the ability to drag something around the screen, so in general you cannot select arbitrary text.

- 此时无法在屏幕上拖动内容，因此一般不能任意选择文本。

- To delete some text: the most traditional way is to place the cursor at the right place and use the backspace button in the keyboard to delete the characters one by one (can long press the backspace to accelerate if there are many to delete). Another approach is to first select the text you want to delete, then click the backspace button in the keyboard.

- 删除文本: 最传统的方法是将光标放在正确位置，使用键盘上的退格键逐个删除字符 (若需删除大量字符可长按退格键加速)。另一种方法是先选中要删除的文本，再点击键盘上的退格键。

- To copy some text: first select the exact text you want to copy, which usually also brings up the text selection bar, then click the ’copy’ button in bar.

- 复制文本: 先选中准确的文本，通常会弹出文本选择栏，然后点击栏中的“复制”按钮。

- To paste text into a text box, first long press the text box, then usually the text selection bar will appear with a ’paste’ button in it.

- 要将文本粘贴到文本框中，先长按文本框，通常会出现带有“粘贴”按钮的文本选择栏。

- When typing into a text field, sometimes an auto-complete dropdown list will appear. This usually indicates this is a enum field and you should try to select the best match by clicking the corresponding one in the list.

- 在文本字段中输入时，有时会出现自动完成的下拉列表。这通常表示这是一个枚举字段，你应尝试通过点击列表中对应项来选择最匹配的内容。

High-Level Prompt

高级提示

{General Instruction}

{General Instruction}

The current user goal/request is: {High-level goal}

当前用户目标/请求是:{High-level goal}

Here is a history of what you have done so far: {History}

以下是你迄今为止的操作历史:{History}

The current raw screenshot is given to you.

当前的原始截图已提供给你。

{Useful Guidelines}

{Useful Guidelines}

Now output an action from the above list in the correct JSON format, following the reason why you do that. Your answer should look like:

现在请从上述列表中输出一个动作，格式为正确的 JSON，并说明你这样做的原因。你的回答应如下所示:

Reason: ...

原因:...

Action: {"action_type": ...}

动作:{"action_type": ...}

Your Answer:

你的回答:

Low-Level Prompt

低级提示

{General Instruction}

{General Instruction}

The user's high-level goal/request is: {High-level goal}

用户的高级目标/请求是:{High-level goal}

The current next step's low-level goal is: {Low-level goal}

当前下一步的低级目标是:{Low-level goal}

The current raw screenshot is given to you.

当前给出的是原始截图。

{Useful Guidelines}

{有用的指南}

Now output an action from the above list in the correct JSON format, following the reason why you do that. Your answer should look like:

现在请根据上述列表输出一个动作，格式为正确的 JSON，并说明你这样做的理由。你的回答应如下所示:

Reason: ...

理由:...

Action: {”action_type”: ...}

动作:{”action_type”: ...}

Your Answer:

你的回答:

Table H.3: Prompt used for the planning model in OmniACT, modified from the prompt in (Kapoor et al., 2024)

表 H.3:OmniACT 中规划模型使用的提示，修改自 (Kapoor et al., 2024) 中的提示

General Instruction

通用说明

You are an excellent robotic process automation agent who needs to generate a PyAutoGUI script for the tasks given to you.

你是一名出色的机器人流程自动化代理，需要为分配给你的任务生成 PyAutoGUI 脚本。

You will receive some examples to help with the format of the script that needs to be generated.

你将收到一些示例，帮助你了解需要生成的脚本格式。

There are some actions that require you to provide an element description for the elements you want to operate on. For the description, follow the requirements below:

有些操作需要你提供你想操作的元素描述。描述时请遵循以下要求:

Element Description Requirements:

元素描述要求:

Provide a concise description of the element you want to operate.

提供你想操作元素的简明描述。

It should include the element’s identity, type (button, input field, dropdown menu, tab, etc.), and text on it (if have).

应包括元素的身份、类型 (按钮、输入框、下拉菜单、标签页等) 及其上的文本 (如有)。

If you find identical elements, specify their location and details to differentiate them from others. Ensure your description is both concise and complete, covering all the necessary information and less than 30 words, and organize it into one sentence.

如果发现相同元素，请指明其位置和细节以区分。确保描述简洁完整，涵盖所有必要信息，不超过 30 字，并组织成一句话。

[IMPORTANT!!] Stick to the format of the output scripts in the example.

请严格遵守示例中输出脚本的格式。

[IMPORTANT!!!] Use only the functions from the API docs.

仅使用 API 文档中的函数。

[IMPORTANT!!] Follow the output format strictly. Only write the script and nothing else.

严格遵守输出格式。只写脚本，别无其他。

API Reference

API 参考

Here is the API reference for generating the script:

以下是生成脚本的 API 参考:

def click(element=description):

def click(element=description):

”Moves the mouse to the element corresponding to the description and performs a left click.

” 将鼠标移动到与描述对应的元素并执行左键点击。

Example:

示例:

High Level Goal: Click at the rectangular red button labeled ”Next”.

高级目标: 点击标有“下一步”的矩形红色按钮。

Python script:

Python 脚本:

```
import pyautogui
```

```
import pyautogui
```

```
pyautogui.click(”Rectangular red button labeled ”Next” ”)
```

```
pyautogui.click(” 标有“下一步”的矩形红色按钮”)
```

```
pass
```

```
pass
```

```
def rightClick(element=description):
```

```
def rightClick(element=description):
```

”Moves the mouse to the element corresponding to the description and performs a right click.

” 将鼠标移动到与描述对应的元素并执行右键点击。

Example:

示例:

High Level Goal: Right-click at link labeled ”vacation rentals”under the ”housing”section.

高级目标: 在“住房”部分下标有“度假租赁”的链接上右键点击。

Python script:

Python 脚本:

```
import pyautogui
```

import pyautogui

pyautogui.rightClick(”Link labeled ”vacation rentals”under the ”housing”section”)

pyautogui.rightClick(” 位于“housing”部分下标记为“vacation rentals”的链接”)

pass

pass

def doubleClick(element=description):

def doubleClick(element=description):

”Moves the mouse to the element corresponding to the description and performs a double click.

“将鼠标移动到与描述对应的元素上并执行双击操作。”

Example:

示例:

High Level Goal: Double-click at folder named ”courses”.

高级目标: 双击名为“courses”的文件夹。

Python script:

Python 脚本:

import pyautogui

import pyautogui

pyautogui.doubleClick(”Folder named ”courses” ”)

pyautogui.doubleClick(” 名为“courses”的文件夹”)

pass

pass

def scroll(clicks=amount_to_scroll):

def scroll(clicks=amount_to_scroll):

”’Scrolls the window that has the mouse pointer by float value (amount_to_scroll).

“滚动鼠标指针所在窗口，滚动量为浮点数 (amount_to_scroll)。”

Example:

示例:

High Level Goal: Scroll screen by 30.

高级目标: 屏幕滚动 30。

Python script:

Python 脚本:

```
import pyautogui
```

```
import pyautogui
```

```
pyautogui.scroll(30)
```

```
pyautogui.scroll(30)
```

```
pass
```

```
pass
```

---

def hscroll(clicks=amount_to_scroll):

```
def hscroll(clicks=amount_to_scroll):
```

””Scrolls the window that has the mouse pointer horizontally by float value (amount to scroll).

“”水平滚动鼠标指针所在窗口，滚动量为浮点数 (amount to scroll)。””

Example:

示例:

High Level Goal: Scroll screen horizontally by 30.

高级目标: 水平滚动屏幕 30。

Python script:

Python 脚本:

import pyautogui

import pyautogui

pyautogui.hscroll(30)

pyautogui.hscroll(30)

pass

pass

def dragTo(element=description, button=holdButton):

def dragTo(element=description, button=holdButton):

”Drags the mouse to the element corresponding to the description with (holdButton) pressed. hold-

“按住 (holdButton) 拖动鼠标到与描述对应的元素。

Button can be ’left’, ’middle’, or ’right’.

按键可以是‘left’、‘middle’或‘right’。”

Example:

示例:

High Level Goal: Drag the screen from the current position to recycle bin with the left click of the

高级目标: 用鼠标左键将屏幕从当前位置拖动到回收站

mouse.

鼠标。

Python script:

Python 脚本:

import pyautogui

import pyautogui

pyautogui.dragTo(”Recycle bin with trash can shape”, ”left”)

pyautogui.dragTo(” 带垃圾桶图标的回收站”, ”left”)

III

pass

pass

def moveTo(element = description):

def moveTo(element = description):

”Takes the mouse pointer to the element corresponding to the description.

” 将鼠标指针移动到与描述对应的元素上。

Example:

示例:

High Level Goal: Hover the mouse pointer to search button.

高级目标: 将鼠标指针悬停在搜索按钮上。

Python script:

Python 脚本:

import pyautogui

import pyautogui

pyautogui.moveTo(”Request appointment button”)

pyautogui.moveTo(” 请求预约按钮”)

pass

pass

def write(str=stringType, interval=secs_between_keys):

def write(str=stringType, interval=secs_between_keys):

”Writes the string wherever the keyboard cursor is at the function calling time with

” 在函数调用时键盘光标所在位置输入字符串，间隔为

(secs_between_keys) seconds between characters.

(secs_between_keys) 字符之间的秒数。

Example:

示例:

High Level Goal: Write ”Hello world”with 0.1 seconds rate.

高级目标: 以 0.1 秒的速度输入“Hello world”。

Python script:

Python 脚本:

import pyautogui

import pyautogui

pyautogui.write(”Hello world”, 0.1)

pyautogui.write(”Hello world”, 0.1)

pass

pass

def press(str=string_to_type):

def press(str=string_to_type):

”Simulates pressing a key down and then releasing it up. Sample keys include ’enter’, ’shift’, arrow

“模拟按下键并释放的操作。示例按键包括‘enter’、‘shift’、方向键

keys, ’f1’.

和‘f1’。”

Example:

示例:

High Level Goal: Press the enter key now.

高级目标: 现在按下回车键。

Python script:

Python 脚本:

```
import pyautogui
```

```
import pyautogui
```

```
pyautogui.press(”enter”)
```

```
pyautogui.press(”enter”)
```

```
pass
```

```
pass
```

```
def hotkey(*args = list_of_hotkey):
```

```
def hotkey(*args = list_of_hotkey):
```

```
””Keyboard hotkeys like Ctrl-S or Ctrl-Shift-1 can be done by passing a list of key names to hotkey().
```

”” 键盘快捷键如 Ctrl-S 或 Ctrl-Shift-1 可以通过传递按键名称列表给 hotkey() 来实现。

```
Multiple keys can be pressed together with a hotkey.
```

多个按键可以通过快捷键同时按下。

```
Example:
```

示例:

```
High Level Goal: Use Ctrl and V to paste from clipboard.
```

高级目标: 使用 Ctrl 和 V 从剪贴板粘贴。

```
Python script:
```

Python 脚本:

```
import pyautogui
```

```
import pyautogui
```

Table H.3 - Continued from the previous page

表 H.3 - 续前页

pyautogui.hotkey(”ctrl”, ”v”)

```
pyautogui.hotkey(”ctrl”, ”v”)
```

pass

```
pass
```

Examples

```
示例
```

Here are some examples similar to the tasks you need to complete.

```
以下是一些与你需要完成的任务类似的示例。
```

However, these examples use coordinate format for actions like click, rightClick, doubleClick, moveTo, dragTo, instead of element description.

```
然而，这些示例中点击 (click)、右击 (rightClick)、双击 (doubleClick)、移动 (moveTo)、拖动 (dragTo) 等操作使用的是坐标格式，而非元素描述。
```

You should only refer to the actions in these examples, and for the output format, stick to the content in the API reference.

```
你应仅参考这些示例中的操作，对于输出格式，应遵循 API 参考中的内容。
```

For example, do not output ”pyautogui.click(100,200)”, instead output ”pyautogui.click(”Gray Tools menu button with a downward arrow in the top right corner”) ”.

```
例如，不要输出”pyautogui.click(100,200)”，而应输出”pyautogui.click(” 右上角带向下箭头的灰色工具菜单按钮”)”。
```

Omit ”import pyautogui”, do not include any comments or thoughts. Your output should only contain the script itself.

```
省略”import pyautogui”，不要包含任何注释或想法。你的输出应仅包含脚本本身。
```

{Example list}

```
{示例列表}
```

Task Description

```
任务描述
```

Based on the screenshot, generate the PyAutoGUI script for the following task: {Task description} You should list all the necessary steps to finish the task, which could involve multiple steps. Also, ensure simplifying your steps as much as possible, avoid dividing a single task into multiple steps if it can be completed in one.

根据截图，为以下任务生成 PyAutoGUI 脚本:{任务描述}。请列出完成该任务所需的所有步骤，可能包含多个步骤。同时，尽量简化步骤，避免将单个任务拆分成多个步骤，如果一步即可完成。

Table H.4: Prompt used for the planning model in ScreenSpot (Agent Setting).

表 H.4:ScreenSpot 中规划模型使用的提示 (代理设置)。

Task Description

任务描述

You are an excellent agent for mobile, web, and desktop navigation tasks.

你是一名出色的移动端、网页和桌面导航任务代理。

Describe the target element for this task based on the provided screenshot:

请根据提供的截图描述此任务的目标元素:

Task: {Task description}

任务:{任务描述}

Element Description Requirements

元素描述要求

Provide a concise description of the element you want to operate.

提供你想操作的元素的简明描述。

Ensure your description is both concise and complete, covering all the necessary information in less than 30 words, and organized into one sentence.

确保描述简洁完整，涵盖所有必要信息，不超过 30 个字，并组织成一句话。

If you find identical elements, specify their location and details to differentiate them from others.

如果发现相同元素，请指明其位置和细节以区分其他元素。

Output Format

输出格式

Your output should only include the element description itself and follow the requirements. Do not start with ”the target element” or ”the element”.

输出内容仅包含元素描述本身，且符合要求。不要以“目标元素”或“该元素”开头。