GPT-4V(ision) is a Generalist Web Agent, if Grounded

Boyuan Zheng   Boyu Gou   Jihyung Kil   Huan Sun   Yu Su
The Ohio State University
{zheng.2372, gou.43, kil.5, sun.397, su.809}@osu.edu

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

Abstract

The recent development on large multimodal models (LMMs), especially GPT-4V(ision) and Gemini, has been quickly expanding the capability boundaries of multimodal models beyond traditional tasks like image captioning and visual question answering. In this work, we explore the potential of LMMs like GPT-4V as a generalist web agent that can follow natural language instructions to complete tasks on any given website. We propose SEEACT, *a generalist web agent that harnesses the power of LMMs for integrated visual understanding and acting on the web. We evaluate on the recent* MIND2WEB *benchmark. In addition to standard offline evaluation on cached websites, we enable a new online evaluation setting by developing a tool that allows running web agents on live websites. We show that GPT-4V presents a great potential for web agents—it can successfully complete 50% of the tasks on live websites if we manually ground its textual plans into actions on the websites. This substantially outperforms text-only LLMs like GPT-4 or smaller models (FLAN-T5 and BLIP-2) specifically fine-tuned for web agents. However, grounding still remains a major challenge. Existing LMM grounding strategies like set-of-mark prompting turns out not effective for web agents, and the best grounding strategy we develop in this paper leverages both the HTML text and visuals. Yet, there is still a substantial gap with oracle grounding, leaving ample room for further improvement.*[1]

1. Introduction

Large multimodal models (LMMs; [3, 22, 26]), especially recent ones such as GPT-4V(ision) [31] and Gemini [4], have shown a remarkable capability on standard vision-and-language understanding and reasoning benchmarks [15, 19, 21, 28, 37, 53, 56]. While web content has been a primary source of training data, a largely overlooked part of the web

[1]Code and evaluation tool will be released at https://github.com/OSU-NLP-Group/SeeAct.

Figure 1. SEEACT leverages an LMM like GPT-4V to visually perceive websites and generate plans in textual forms. The textual plans are then grounded onto the HTML elements and operations to act on the website.

is the websites themselves—every website is designed to be rendered visually for easy consumption by human users. This poses a new challenge and a new opportunity for LMMs. On the one hand, screenshots of rendered websites, which could contain thousands of elements with rich relations, are more complex than the images in most existing benchmarks, which are usually object- or scene-centric. On the other hand, if LMMs can accurately comprehend websites, it will open the door for numerous applications on the web.

In this work, we aim to investigate the potential of LMMs as generalist web agents [12]. A generalist web agent, as defined in MIND2WEB [12], is expected to follow natural language instructions and complete tasks on any given real-world website (e.g., Figure 1). The tasks can be fairly diverse and complex, with one task possibly taking 10+ actions across multiple dynamically rendered webpages. Existing work [12, 27] primarily uses large language models (LLMs) such as GPT-4 [31] on the raw HTML input. However, HTML code is noisier than the rendered visuals and has a lower information density. For example, the screenshot in Figure 1 contains 423 HTML elements that would require 186,490 textual tokens with the GPT-2 Tokenizer, while requiring only 1,445 visual tokens using GPT-4V's visual tokenizer. Furthermore, HTML alone provides incomplete information and misses critical semantics from, e.g., embedded images.

To this end, we propose SEEACT, a generalist web agent that harnesses the power of LMMs for integrated visual understanding and acting on the web. We will focus on GPT-4V, the most advanced LMM publicly available to date, and compare it with smaller LMMs such as BLIP-2 [22]. We find that GPT-4V exhibits a strong capability in visually understanding rendered webpages and generate the right plans in textual forms across a wide range of websites and tasks. However, *grounding* [6, 16], *i.e.*, converting the textual plan into precise actions on the website, remains a major challenge. It involves selecting the right HTML element to interact with as well as the right operation (e.g., `Click`, `Type`, or `Select`). We propose multiple grounding methods, including superpositioning bounding boxes and index labels onto the image, similar to set-of-mark prompting [46] that has been shown effective on object- or scene-centric images. However, we find that on complex images with rich semantic and spatial relationships like webpage screenshots, severe hallucination is observed from GPT-4V. The most effective grounding strategy leverages the known correspondence between HTML element and their visual rendering, a unique property for websites compared to natural images.

We evaluate SEEACT on the MIND2WEB dataset [12] and compare it with text-only large language models (LLMs) like GPT-4 [31] as well as smaller models (FLAN-T5 [10] and BLIP-2 [22]) specifically fine-tuned for web agents. In addition to the standard offline evaluation setting on cached websites, we further establish an *online evaluation* setting by developing a new tool that allows for running web agents on live websites. The major findings from our exploration are summarized below:

- SEEACT with GPT-4V is a strong generalist web agent, if oracle grounding is provided. In online evaluation, it can successfully complete 50% of tasks on different websites, substantially outperforming existing methods like GPT-4 (20%) or FLAN-T5 (18%). This strongly demonstrates the potential of LMMs like GPT-4V for web agents.
- However, grounding is still a major challenge. The best grounding strategy still has a 20-25% gap with oracle grounding. Among the various grounding strategies, the best one organically leverages both HTML text and visuals, substantially outperforming image annotation strategies [46] by up to 30%.
- In-context learning with large models (both LMMs and LLMs) show better generalization to unseen websites, while supervised fine-tuning still has an edge on websites seen during training.
- There is a non-negligible discrepancy between online and offline evaluation because there can often be multiple viable plans for completing the same task. Online evaluation is more indicative of a model's true performance.

2. SEEACT

In this section, we first explain the problem formulation of web agent and then introduce our developed SEEACT, a generalist web agent based on GPT-4V.

Specifically, given a web-based task (*e.g.*, "Rent a truck with the lowest rate" in the car rental website), we examine two essential capabilities of GPT-4V as a generalist web agent: (i) **Action Generation** to produce an action description at each step (*e.g.*, "Move the cursor over the 'Find Your Truck' button and perform a click") towards completing the task, and (ii) **Element Grounding** to identify an HTML element (*e.g.*, "[button] Find Your Truck") at the current step on the webpage.

2.1. Formulation

Given a website $\mathcal{S}$ (*e.g.,* a car rental website) and a task T (*e.g., "Rent a truck with the lowest rate"*), the web agent should generate a sequence of executable actions $A = [\boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_n]$ to complete the task. Specifically, at time step t, the agent should generate an action $\boldsymbol{a}_t$ based on the current environment observation s_t, the previous actions $\{\boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_{t-1}\}$, and the task T:

$$\boldsymbol{a}_t = \pi(s_t, T, \{\boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_{t-1}\}) \tag{1}$$

The environment observation s_t comprises an HTML document h_t and a screenshot image i_t. LLMs can only be grounded on the HTML document, while LMMs can be grounded on both the HTML document and the screenshot image. The website status is updated accordingly after each action:

$$s_{t+1} = \mathcal{S}(\boldsymbol{a}_t) = \{h_{t+1}, i_{t+1}\}$$

For simplicity, in subsequent step-wise formulations, the time step notation t is omitted.

An action $\boldsymbol{a}$ corresponds to a browser event provided by the website environment. Therefore, we formulate an action as a triplet of three necessary variables for a browser event (e, o, v). $e \in \mathcal{E}$ identifies the target webpage element to operate on, such as the "Find Your Truck" button in Figure 2. $\mathcal{E}$ represents the set of webpage elements within the environment $\mathcal{S}$. The operation $o \in \mathcal{O}$ is the action to be performed on the target element, with $\mathcal{O}$ encompassing all possible operations in $\mathcal{S}$ (*e.g.*, `Click`, `Type`). The variable v denotes the additional value needed for a certain operation (*e.g.*, the date *12/10/2023* for a `Type` operation).

Figure 2. An example of the element grounding process for a single action during completing the given task with three different methods. In this action step, the model needs to click the "Find Your Truck" button to perform a search. For grounding with textual choices, some element candidates represented with HTML text are given, the model is required to generate the choice index of the target element. For image annotation, bounding boxes and index labels are added to the image. The model is required to generate the label on the bottom-left of the target element. For grounding with element attributes, the model needs to predict the text and type of the target element.

However, agents based on LLMs or LMMs are typically unable to directly generate the three variables (e, o, v) required for a browser event. Instead, they generate a textual description of the intended action $\tilde{\boldsymbol{a}}$, containing information about these variables as $(\tilde{e}, \tilde{o}, \tilde{v})$. This process is referred to as **Action Generation**. To interact with the environment, a further step is required to convert $\tilde{\boldsymbol{a}}$ into $\boldsymbol{a}$, which we refer to as **Action Grounding**.

2.2. Action Generation

We explicitly instruct GPT-4V to imitate humans browsing a webpage and analyze the task, webpage, and previous actions. It is asked to generate an action description $\tilde{a}$ based on its analysis and reasoning. We take the screenshot image i as the visual context without utilizing the HTML document h for action generation.

2.3. Action Grounding

Despite the capability of GPT-4V in identifying and describing the next action to complete the given task in natural language, it is still challenging to convert the action description $\tilde{\boldsymbol{a}}$ into an executable action $\boldsymbol{a}$ within the environment. Deriving operation type $\boldsymbol{o}$ and value $\boldsymbol{v}$ from the action description $\tilde{\boldsymbol{a}}$ can be solved through string parsing reasonably well. The key challenge is to identify the target element $\boldsymbol{e}$ from the generated $\tilde{e}$, which we refer to as **Element Grounding**.

To address this challenge, we explore three approaches using different types of information: Grounding via Element Attributes, Grounding via Textual Choices, and Grounding via Image Annotation, as depicted in Figure 2. The prompting details of action generation and grounding are included in Appendix A.

Grounding via Element Attributes. This approach involves prompting the model to generate as detailed attributes of the target element as possible, thereby providing more information to precisely match with the target HTML element.

Specifically, we prompt the model to not only describe the element $\boldsymbol{e}$, but also specify the target element's type and the textual content in $\tilde{\boldsymbol{e}}$. For example, as illustrated in Figure 2, the model would generate element text as "Find Your Truck" and identify its type as a "BUTTON." Following this, a heuristic search is performed across the DOM elements, using the element text and type to locate matching elements. In cases where a single match is found, it is automatically selected. Otherwise, when multiple matches arise, the model is further prompted to select the final selection.

Grounding via Textual Choices. The above approach demands precise and sufficient attribute descriptions from GPT-4V and accurate matching by the heuristic search, which can be highly demanding. For instance, many elements may have no textual content or have textual information in a nearby element instead of itself.

Alternatively, we provide the model with textual representations of elements as choices to facilitate grounding, which has already been proven effective in MindAct [12]. Specifically, MindAct utilizes a ranker model to select top-k candidate elements $(e_1, e_2, ..., e_k)$ with a pretrained cross-encoder. Each candidate element is represented as a choice in a multi-choice question with its HTML text, as illustrated in Figure 2. After generating the action description $\tilde{a}$, the model is further asked a multi-choice question to choose its intended element from the given multiple choices (including a 'none' option).

Grounding via Image Annotation. Textual representations alone are sometimes insufficient to distinguish similar or identical elements, as illustrated in Appendix E. Therefore, in this approach, we propose to overlay a bounding box for each candidate element e selected by the ranker as well as an alphabet label on the bottom left of the bounding box[2].The model is expected to generate the label corresponding to the target element.

Oracle Action Grounding. Ideally, the action description $\tilde{a}$ must encompass all necessary details to precisely identify each variable (e, o, v) of the action triplet. To assess the performance of action generation, an oracle grounding method, which ensures the variables be identified as long as they are mentioned in the action description, is desired. Here we approximate the oracle grounding method by asking human annotators to identify the model's intended actions.

3. Experiments

3.1. Dataset

We evaluate our methods on MIND2WEB [12], a comprehensive dataset encompassing over 2,000 complex web tasks with annotated actions. This dataset spans 137 websites across 31 low-level domains, categorized into 12 high-level domains. It supports three primary operations: `Click`, `Type`, and `Select`, with `Hover` and `Press Enter` operations integrated into `Click` to avoid ambiguity.

The dataset's test sets aim to measure the generalization of web agents across different tasks, websites, and domains. Specifically, the Cross-Task setting focuses on evaluating agents on tasks that are new to the training data but within included domains and websites. The Cross-Website setting evaluates agents with tasks across 10 new websites for each of the top-level domains in the training data. The Cross-Domain setting assesses agent performance on tasks in two top-level domains held out from the training data.

We align each HTML document in the dataset with its corresponding webpage screenshot image from the MIND2WEB raw dump, which undergoes human verification to confirm correct rendering and element visibility. Given the constraints of the GPT-4V inference quota, our experiments focus on a random subset of 30 tasks from each test split. Detailed statistics of these subsets are presented in Table 1.

[2]We use the Supervision library for image annotation: https://supervision.roboflow.com/

3.2. Methods

SeeAct. In grounding via image annotation and textual choices, we first employ the DeBERTa-base cross-encoder from MindAct [12] to rank the top 50 elements for better comparison with its text-only counterparts. Then, we cluster elements into groups of 17 options for inference. In grounding via element attributes, no candidate element is provided. We access GPT-4V through ChatGPT WebUI, which only allows a zero-shot setting.

To compare with SEEACT, we also implement methods based on text-only LLMs and BLIP-2 [22] following the two-stage strategy of MindAct [12]. Firstly, we employ the ranker above to pick the top 50 elements. Subsequently, the action generation problem is formulated as a multi-choice question answering problem, with the candidate elements as options, including a "None" option if the target element is absent. During inference, elements are clustered into groups of 5 elements, with iterative refinement, until a single choice is made or all options are discarded. We evaluate supervised fine-tuning (SFT) methods using FLAN-T5 [10] and BLIP-2-T5 and in-context learning (ICL) methods using GPT-3.5, GPT-4.

FLAN-T5. We fine-tune FLAN-T5 using a left-to-right language modeling objective with the target sequence of ground-truth actions. The fine-tuned FLAN-T5 then serves as the backbone for inference, enabling action generation in the format defined in Equation 2.1.

BLIP-2-T5. The BLIP-2 model combines a vision encoder and an LLM with a bridging component for modality connection. We jointly fine-tune the LLM and the bridge module on MIND2WEB training data while keeping the vision encoder frozen. For the vision encoder, we leverage the ViT-L/14 pre-trained from CLIP [36] with an image resolution of 2,048. To ensure a fair comparison with the FLAN-T5-based text-only model, we choose FLAN-T5 as the language model and initialize it with the parameters fine-tuned on MIND2WEB.

GPT-3.5 and GPT-4. We also conduct experiments with text-only LLMs, specifically GPT-3.5-turbo-0613 and GPT-4-turbo-1106-preview, using in-context learning in 3-shot settings. We use the same multiple-choice formulation and include three demonstration examples for in-context learning as specified in MindAct.

3.3. Offline Evaluation

We adopt the evaluation metrics utilized in MIND2WEB. **Element Accuracy** (Ele. Acc) compares the predicted element with the ground-truth elements. **Operation F1** (Op. F1) calculates the token-level F1 score for the predicted operation comprised of action and input value. **Step Success Rate** (Step SR) measures the success of each action step. A step is successful only if the selected element and the predicted operation are correct. We report macro averages across tasks for these step-wise metrics. **Success Rate** (SR) measures

Split	# Tasks	# Domains	# Websites	Avg # Actions	Avg # Visual Tokens	Avg # HTML Elements	Avg # HTML Tokens
Train	1,009	31	73	7.3	4,240	602	128,827
Cross-Task	30	13	25	7.5	2,952	630	96,612
Cross-Website	30	8	9	6.9	3,441	635	71,729
Cross-Domain	30	10	19	5.0	3,759	521	72,002

Table 1. Dataset Statistics. The average # of visual tokens is calculated with OpenAI visual token calculator.

the success of an entire task. A task is regarded successful only if all steps have succeeded. This metric is stringent without allowing the model any space for exploration and error correction. Therefore, for offline evaluation, we focus on the first three metrics. However, we also conduct online evaluation on live websites to get a better evaluation on the whole task success rate, as detailed below.

3.4. Online Evaluation

We develop a new online evaluation tool using Playwright[3] to evaluate web agents on live websites (instead of cached websites in offline evaluation). Our tool can convert the predicted action (e, o, v) into a browser event. To adhere to ethical standards, our experiments are restricted to non-login tasks in compliance with user agreements, and we closely monitor agent activities during online evaluation to prevent any actions that have potentially harmful impact, like placing an order or modifying the user profile.

For a fair comparison between offline and online evaluations, we only re-write time-sensitive tasks to ensure they are still valid when the evaluation is conducted. For instance, we update the dates for flight-related tasks. Finally, we conduct the online evaluation on a subset of total 50 tasks from the three test splits and report the mean performance across these tasks.

4. Results and Analysis

4.1. Offline Evaluation Results

GPT-4V can be a Generalist Web Agent with Oracle Action Grounding. Given an effective action grounding method, GPT-4V has the potential to serve as a generalist web agent. Specifically, as described in subsection 2.3, we provide GPT-4V with an oracle action grounding method (SEEACT$_{\text{Oracle}}$) through human annotation, the model achieves a step success rate of 65.7%, 70.0%, and 62.1% across three test splits, respectively. As shown in Table 2, this method substantially outperforms other models under all metrics across three test splits. Specifically, it achieves a 11.9% step success rate improvement over the second-best method in the Cross Task setting. The performance advantage is more pronounced under the Cross-Website and Cross-Domain settings, where it leads by 28.3% and 21.2% step success rates, demonstrating its generality compared with supervised fine-tuning. This observation is further corroborated with in the online evaluation (Table 3).

Element Grounding Method Comparison. However, there is a noticeable gap between oracle grounding and all the three proposed grounding methods. This demonstrates that grounding, especially element grounding, is a major bottleneck. Element grounding via textual choice (SEEACT$_{\text{Choice}}$) demonstrates the best performance under all metrics across all settings, comparable to supervised fine-tuning and showing a substantial improvement over text-only LLMs. Grounding via image annotation (SEEACT$_{\text{Annotation}}$) offers an intuitive approach and shows promising results in recent work that focuses on object- or scene-centric images [46]. However, we find that on complex images with rich semantic and spacial relationships like webpage screenshots, severe hallucination is observed from GPT-4V. Specifically, it often fails to correctly map its generated element description (which is often correct according to oracle grounding) to the right bounding boxe and index label in the image, leading to a low element accuracy. This limitation primarily arises from GPT-4V's weakness in understanding image details and relative spatial location, a topic that we will further delve into in subsection 4.3.

Grounding via element attributes (SEEACT$_{\text{Attribute}}$) also demonstrates inferior performance. This method's effectiveness is primarily limited by its heuristic-based element localization strategy, which depends on textual and locality characteristics. This becomes problematic as not all webpage elements contain text, and sometimes the relevant text is associated with a nearby but distinct element.

LMMs vs. LLMs. The SEEACT$_{\text{Choices}}$ model demonstrates a substantial performance advantage over the text-only GPT-4 under all three metrics across all three test splits. Specifically, it outperforms GPT-4 in step success rate of 8.9%, 14.7%, and 11.2% on three settings, respectively. Interestingly, fine-tuned BLIP-2-T5 does not show a noticeable gain over FLAN-T5, despite having additional visual input. Several factors may contribute to this. First, the CLIP model used as the image encoder may not be sufficiently adept at image details, as explored by Shen et al. [38]. This limitation

[3]https://playwright.dev/python/docs/api/class-playwright

Model	Cross-Task			Cross-Website			Cross-Domain		
	Ele. Acc	Op. F1	Step SR	Ele. Acc	Op. F1	Step SR	Ele. Acc	Op. F1	Step SR
Supervised Fine-Tuning									
FLAN-T5									
– Base	40.5	74.4	37.3	28.7	69.6	27.9	38.2	69.1	36.2
– Large	52.2	70.7	48.8	35.3	65.8	32.7	41.9	64.6	39.5
– XL	**56.8**	74.6	**52.5**	**42.6**	69.9	**39.5**	**43.8**	65.2	**40.7**
BLIP-2-T5									
– Base	39.5	**74.9**	36.1	34.0	70.8	32.2	38.2	72.8	37.5
– Large	50.0	72.1	46.0	39.5	71.5	36.3	40.9	70.1	39.4
– XL	52.9	**74.9**	50.3	41.7	**74.1**	38.3	**43.8**	**73.4**	39.6
In-Context Learning									
GPT-3.5	19.4	59.8	16.8	14.9	56.5	14.1	25.5	57.9	24.2
GPT-4	40.2	63.4	31.7	27.4	61.0	27.0	36.2	61.9	29.7
SEEACT									
– Attributes	4.7	39.5	4.7	9.7	37.8	9.7	16.0	41.4	15.3
– Annotation	15.1	66.5	13.0	11.3	63.4	10.5	16.5	65.1	14.7
– Choice	**48.9**	**69.1**	**40.6**	**48.5**	**70.6**	**41.7**	**44.0**	**70.9**	**40.9**
– Oracle	72.9	80.9	65.7	74.4	83.7	70.0	72.8	73.6	62.1

Table 2. Performance of different models. Except SEEACT grounding with Attributes and Oracle, all other methods use the ranker in the MindAct framework for candidate generation. For SEEACT, "Attributes", "Choices", "Annotation", and "Oracle" refer to element grounding via Element Attributes, Textual Choices, Image Annotation, and Human Annotation, respectively, as described in subsection 2.3.

is particularly relevant for our web navigation task, which demands a high level of image detail comprehension. Second, BLIP-2-T5 utilizes an off-the-shelf CLIP model that may not be optimal for webpage screenshots. Finally, although the screenshots in the test splits are error-free, some of the examples in the training set might contain issues such as rendering failures or inaccuracies in the timing of screenshot capture by annotators.

SFT vs. ICL. We compare SFT and ICL methods to offer insights for developing web agents in different scenarios. ICL (with SEEACT) demonstrate consistent and robust performance across three test splits. ICL is particularly advantageous in scenarios lacking annotations or requiring strong generalization capabilities for new domains and websites. As grounding methods improve towards oracle grounding, ICL is poised to show even stronger performance. On the other hand, SFT methods show better generalization across tasks on websites already seen during training. Considering the high cost of data annotation for web agents and the billions of websites on the internect, ICL offers a more compelling solution for generalist web agents. However, if one only needs to develop a strong web agent for a certain website, SFT is still a competitive solution.

4.2. Online Evaluation Results

In online evaluation, we pair a web agent with a human annotator, where the human was tasked to monitor agent actions that may change real-world states and determine whether each task was successfully completed. For comparative analysis, we include success rates from offline evaluation, denoted as Offline_0 (allowing zero wrong action) and Offline_1 (allowing one wrong action).

	Offline_0	Offline_1	Online
FLAN-T5-XL	2	20	18
GPT-4	2	18	20
$\text{SEEACT}_{\text{Choice}}$	8	22	26
$\text{SEEACT}_{\text{Oracle}}$	18	58	50

Table 3. Whole task success rate (%) under both offline and online evaluation. Offline_0 and Offline_1 refer to no tolerance for error at any step and allowing for error at one step, respectively.

Table 3 shows that the whole task success rate in online evaluation substantially exceeds that of offline evaluation (Offline_0). This finding suggests that the whole task success rate is likely underestimated in the offline evaluation due to the variability in actions and plans. In other words, there may be multiple viable plans for a task, but the reference plan in offline evaluation only captures one of them.

Across all three settings, $\text{SEEACT}_{\text{Choice}}$ outperforms both GPT-4 and FLAN-T5-XL. Using oracle grounding further improves the performance substantially, reaching a remarkable whole task success rate of 50%. Although GPT-4 shows much worse performance than FLAN-T5-XL in step success rate under offline evaluation (Table 2), they exhibit similar

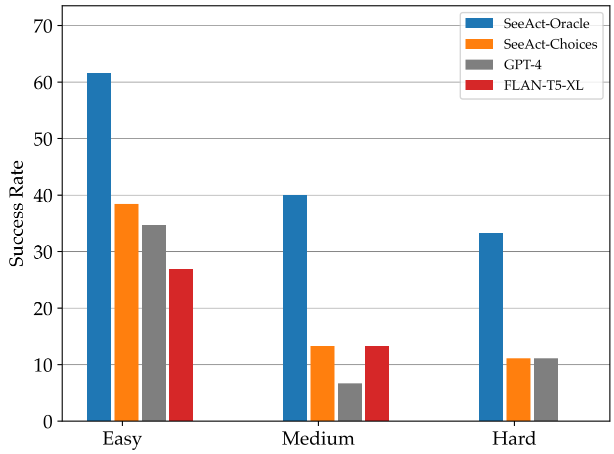


Figure 3. Whole task success rate across task difficulty levels. We categorize tasks based on the number of actions to complete, i.e., Easy: 2-4, Medium: 5-7, and Hard: 8-12, with 26, 15, and 9 tasks in each group, respectively.

success rates in online evaluation. These results further confirm the potential of large models for generalist web agents compared with fine-tuned small models.

Online Success Rate by Task Difficulty. We investigate the performance of web agents on tasks across different difficulty levels. We estimate the task difficulty based on the number of actions taken by annotators during action trace annotation. As shown in Figure 3, the whole task success rate is negatively correlated with the number of actions—it decreases as the number of actions increases across all four methods. SeeAct$_{\text{Oracle}}$ consistently outperforms other methods across all difficulty levels. Interestingly, the gap between SeeAct$_{\text{Oracle}}$ and SeeAct$_{\text{Choice}}$ enlarges on longer-horizon tasks. This is understandable because grounding errors cascade to later steps; nonetheless, it further shows the challenge of grounding for GPT-4V and the need for better grounding methods.

4.3. Further Analysis and Discussion

Error Analysis in Grounding via Image Annotation. Set-of-mark prompting [46] uses a similar method as grounding via image annotation and has been shown effective on object- or scene-centric images [24, 35, 57]. However, this grounding method is suboptimal on webpage screenshot images that are complex and contain rich semantic and spatial relationships.

To analyze the reasons behind the failures, we randomly sample 100 predicted actions and observe the major types of errors as: (1) Wrong action generation; (2) Making up bounding box & label; (3) Failure to link bounding boxes with the correct labels. Illustrative examples are included in Appendix C.

Our analysis reveals that 53% of the errors originate from wrong action generation wherein the model fail to predict the target element within the action description. This error rate is 27% higher than images without annotation, which exhibits a 26% error rate because image annotation often obscures critical content within webpages. Overlaying image annotations on webpages without obscuring important content poses a significant challenge. This is primarily due to the dense layout of webpages, which often feature small elements designed for precise cursor interactions. In contrast, the arrangement of objects in mobile UIs and natural scenes tend to be sparser.

Then 21% of the errors can be attributed to GPT-4V's tendency of visual illusion [17], where the model misinterprets and fabricates content over the image. Specifically, the target element described in action generation does not have a bounding box or a label on the bottom-left, where the model is supposed to generate "None". However, the model falsely assumes the presence of a bounding box and makes up a label as the answer. Another 18% of errors are caused by GPT-4V's limitation in recognizing the relative position within an image. Specifically, the model is capable of identifying both the target element within the bounding box and the labels. However, it struggles to link the bounding box with its corresponding label.

Powerful Web Action Planner. GPT-4V exhibits promising capabilities, ranging from long-range action planning, webpage content reasoning, and error correction to surpassing the limitations of superficial textual similarity matching inherent in fine-tuned, text-only models. GPT-4V exhibits capability in long-range planning involving a series of interconnected low-level actions. As illustrated in Appendix D, besides generating the immediate next step action, the model also anticipates a sequence of subsequent actions in the future to complete the given task.

Error Correction Awareness. GPT-4V also exhibits the awareness of error correction in the previous actions. In the example in Appendix G, GPT-4V is capable of discovering the mobile phone number is invalid due to wrong format and generate the description about the action to correct this error. This highlights the model's potential for adaptation in online settings, where actions may not always follow pre-defined, ideal paths as in offline evaluations. This error-correction capability paves the way for adding robustness and reasonable dynamic planning.

Advantages of Knowledge. GPT-4V demonstrates substantial advantages in tasks that requires certain knowledge over fine-tuned models at a smaller scale. As shown in Appendix F, GPT-4V is able to identify the IATA code of the airport in Los Cabos as SJD. In contrast, models are typically weaker at knowledge intensive tasks and also likely to lose knowledge during the fine-tuning process due to catastrophic forgetting.

5. Related Work

5.1. Large Multimodal Models

The development of LMMs has been characterized by large-scale image-text pretraining to achieve in-context learning capabilities [1, 3, 8, 9, 13, 22, 30, 43, 44, 52]. Further work focuses on improving instruction following [32, 33] capabilities with curated multimodal instruction-following data [11, 26, 59] and alignment with reinforcement learning from human feedback [42]. GPT-4V [31] and Gemini [4] represent significant progress in LMMs. Several studies [2, 31, 45, 46, 49, 54] have highlighted their remarkable multimodal capabilities, emphasizing the advanced and versatile integration of visual and language reasoning abilities. Their performance on a series of benchmarks [15, 19, 21, 28, 37, 53, 56] also showcases remarkable capabilities on vision-and-language understanding and reasoning. Although open-sourced models still exhibit a performance gap with GPT-4V, they have the advantages of controllability and ease of fine-tuning for various applications. For example, in CogAgent [20], LMMs are fine-tuned on HTML and screenshot image pairs to enhance webpage understanding ability and further enhanced with an image encoder for high-resolution image details. Ferret [51] is finetuned to allow visual referring and grounding.

5.2. Web Agent

Driven by the vision of achieving a seamless human-web interaction, considerable initiatives have been invested in web agents that can understand language instruction and complete tasks on the web. Significant efforts have been invested in building benchmarks for web agents with simplified environment simulation [25, 29, 39, 50]. WebArena [58] creates website simulations in the sandbox from four popular categories with functionality and data mimicking their real-world equivalents. Mind2Web [12] instead provides environments in diverse domains, websites, and various tasks on real-world websites by dumping webpages from live websites along with action trajectory annotation. Similar benchmarks have also been proposed for mobile UI automation [5, 23, 41]. To improve web agent performance, WebAgent [18] pretrains a T5 model on HTML data and performs instruction tuning to enable long HTML document summarization and action planning. WebGUM [14] enables web agents to observe both HTML and the rendered screenshot by fine-tuning LMM with a large multimodal corpus for web agents. CogAgent [20] further improves with a high-resolution image encoder to enable the agent to recognize small webpage elements and texts. In a concurrent work [45], GPT-4V exhibits strong performance on mobile UI understanding, which is much simpler than the desktop websites we study.

5.3. Visual Grounding

While LMMs, especially GPT-4V and Gemini, have achieved remarkable vision-language understanding capabilities, they still face challenges in fine-grained visual grounding. Various visual prompting methods have been proposed to augment GPT-4V's image detail grounding ability. Typically, these methods involve overlaying visual marks onto images, facilitating the model's reference and grounding to specific regions. Various types of visual marks have been explored, such as red circles [40], mask areas [47], and combinations of circles, arrows, and hand drawings [48]. SoM [46] involves segmenting the image into semantically meaningful regions and overlaying an array of visual marks like numbers, alphabets, masks, or bounding boxes. Fine-tuning vision-language models with image-annotated data has been shown to be effective. Kosmos-2 [34] represents bounding box locations through textual location tokens. BuboGPT [55] extract entities and find corresponding masks for objects in the image. Shikra [7] handles image detail referring and grounding by applying spatial coordinates as text tokens in inputs and outputs, respectively. Ferret [51] represents regions with both discrete coordinates and continuous features along with a spatial-aware visual sampler to handle diverse spatial characteristics across various shapes.

6. Limitations and Potential Societal Impact

Web agents demonstrate a great promise in automating routine web tasks through natural language commands, thereby enhancing web accessibility, especially for people less familiar with information technology. However, there are still potential concerns and limitations regarding our experiment scale and safety concerns for deployment in the real world.

Limited Experiment Scale. We can only run experiments on a subset of Mind2Web due to the limited GPT-4V inference quota. We will seek to expand the scale of the experiments.

Agent Safety. While a generalist web agent holds the potential to automate routine web tasks, enhance user experiences and promote web accessibility, safety concerns related to their real-world deployment are also critical. These concerns span privacy issues, such as access to users' personal profiles, and sensitive operations, such as financial transactions or application form submissions. During online evaluation, we noticed the possibility for these web agents to generate harmful actions on the web, and we manually validate the safety of all the actions before execution. It is critical for further research to thoroughly assess and mitigate the safety risks associated with web agents, ensuring they are safeguarded against producing and executing harmful actions.

7. Conclusion

In this work, we developed SeeAct, a generalist web agent that harnesses the power of large multimodal models (LMMs) like GPT-4V to integrate visual understanding and acting on the web. We showed that LMMs present a great promise for generalist web agents, with a success rate of 50% on live websites given an oracle grounding method. GPT-4V also exhibits impressive capabilities, such as error correction and long-range planning. However, fine-grained visual grounding is still a major challenge. The most effective grounding strategies we explored in this paper still exhibit a 20-25% performance gap compared to oracle grounding. Future work should better leverage the unique properties of the Web, *e.g.*, the known correspondence between HTML and visual elements, for improving grounding and reducing hallucinations from LMMs.

Furthermore, we show a significant discrepancy between online and offline evaluations, emphasizing the importance of online evaluation for an accurate assessment of a model's capabilities. This discrepancy is largely due to the variability in potential plans for completing the same task, pointing to the dynamic nature of web interactions.

Acknowledgments

The authors would like to thank colleagues from the OSU NLP group for their thoughtful comments. This research was supported in part by ARL W911NF2220144 and Cisco.

References

[1] Armen Aghajanyan, Po-Yao (Bernie) Huang, Candace Ross, Vladimir Karpukhin, Hu Xu, Naman Goyal, Dmytro Okhonko, Mandar Joshi, Gargi Ghosh, Mike Lewis, and Luke Zettlemoyer. Cm3: A causal masked multimodal model of the internet. *ArXiv*, abs/2201.07520, 2022. 8

[2] Syeda Nahida Akter, Zichun Yu, Aashiq Muhamed, Tianyue Ou, Alex Bauerle, Ángel Alexander Cabrera, Krish Dholakia, Chenyan Xiong, and Graham Neubig. An in-depth look at gemini's language abilities. 2023. 8

[3] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andy Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning. *ArXiv*, abs/2204.14198, 2022. 1, 8

[4] Gemini Team Google Rohan Anil, Sebastian Borgeaud, and Yonghui Wu et al. Gemini: A family of highly capable multimodal models. 2023. 1, 8

[5] Andrea Burns, Deniz Arsan, Sanjna Agrawal, Ranjitha Kumar, Kate Saenko, and Bryan A. Plummer. A dataset for interactive vision-language navigation with unknown command feasibility. In *European Conference on Computer Vision*, 2022. 8

[6] Khyathi Raghavi Chandu, Yonatan Bisk, and Alan W Black. Grounding 'grounding'in nlp. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4283–4305, 2021. 2

[7] Ke Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. Shikra: Unleashing multimodal llm's referential dialogue magic. *ArXiv*, abs/2306.15195, 2023. 8

[8] Xi Chen, Xiao Wang, Soravit Changpinyo, A. J. Piergiovanni, Piotr Padlewski, Daniel M. Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer, Alexander Kolesnikov, Joan Puigcerver, Nan Ding, Keran Rong, Hassan Akbari, Gaurav Mishra, Linting Xue, Ashish V. Thapliyal, James Bradbury, Weicheng Kuo, Mojtaba Seyedhosseini, Chao Jia, Burcu Karagol Ayan, Carlos Riquelme, Andreas Steiner, Anelia Angelova, Xiaohua Zhai, Neil Houlsby, and Radu Soricut. Pali: A jointly-scaled multilingual language-image model. *ArXiv*, abs/2209.06794, 2022. 8

[9] Xi Chen, Josip Djolonga, Piotr Padlewski, Basil Mustafa, Soravit Changpinyo, Jialin Wu, Carlos Riquelme Ruiz, Sebastian Goodman, Xiao Wang, Yi Tay, Siamak Shakeri, Mostafa Dehghani, Daniel M. Salz, Mario Lucic, Michael Tschannen, Arsha Nagrani, Hexiang Hu, Mandar Joshi, Bo Pang, Ceslee Montgomery, Paulina Pietrzyk, Marvin Ritter, A. J. Piergiovanni, Matthias Minderer, Filip Pavetic, Austin Waters, Gang Li, Ibrahim M. Alabdulmohsin, Lucas Beyer, Julien Amelot, Kenton Lee, Andreas Steiner, Yang Li, Daniel Keysers, Anurag Arnab, Yuanzhong Xu, Keran Rong, Alexander Kolesnikov, Mojtaba Seyedhosseini, Anelia Angelova, Xiaohua Zhai, Neil Houlsby, and Radu Soricut. Pali-x: On scaling up a multilingual vision and language model. *ArXiv*, abs/2305.18565, 2023. 8

[10] Hyung Won Chung, Le Hou, S. Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Wei Yu, Vincent Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed Huai hsin Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models. *ArXiv*, abs/2210.11416, 2022. 2, 4

[11] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Albert Li, Pascale Fung, and Steven C. H. Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. *ArXiv*, abs/2305.06500, 2023. 8

[12] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. *arXiv preprint arXiv:2306.06070*, 2023. 1, 2, 3, 4, 8

[13] Danny Driess, F. Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Ho Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint,

Klaus Greff, Andy Zeng, Igor Mordatch, and Peter R. Florence. Palm-e: An embodied multimodal language model. In *International Conference on Machine Learning*, 2023. 8

[14] Hiroki Furuta, Ofir Nachum, Kuang-Huei Lee, Yutaka Matsuo, Shixiang Shane Gu, and Izzeddin Gur. Multimodal web navigation with instruction-finetuned foundation models. *ArXiv*, abs/2305.11854, 2023. 8

[15] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. *International Journal of Computer Vision*, 127:398 – 414, 2016. 1, 8

[16] Yu Gu, Xiang Deng, and Yu Su. Don't generate, discriminate: A proposal for grounding language models to real-world environments. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4928–4949, Toronto, Canada, 2023. Association for Computational Linguistics. 2

[17] Tianrui Guan, Fuxiao Liu, Xiyang Wu, Ruiqi Xian, Zongxia Li, Xiaoyu Liu, Xijun Wang, Lichang Chen, Furong Huang, Yaser Yacoob, Dinesh Manocha, and Tianyi Zhou. Hallusionbench: An advanced diagnostic suite for entangled language hallucination&visual illusion in large vision-language models. 2023. 7

[18] Izzeddin Gur, Hiroki Furuta, Austin Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. A real-world webagent with planning, long context understanding, and program synthesis. *ArXiv*, abs/2307.12856, 2023. 8

[19] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Xiaodong Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *ArXiv*, abs/2009.03300, 2020. 1, 8

[20] Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, and Jie Tang. Cogagent: A visual language model for gui agents. 2023. 8

[21] Sahar Kazemzadeh, Vicente Ordonez, Marc andre Matten, and Tamara L. Berg. Referitgame: Referring to objects in photographs of natural scenes. In *Conference on Empirical Methods in Natural Language Processing*, 2014. 1, 8

[22] Junnan Li, Dongxu Li, Silvio Savarese, and Steven C. H. Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *ArXiv*, abs/2301.12597, 2023. 1, 2, 4, 8

[23] Yang Li, Jiacong He, Xiaoxia Zhou, Yuan Zhang, and Jason Baldridge. Mapping natural language instructions to mobile ui action sequences. *ArXiv*, abs/2005.03776, 2020. 8

[24] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, 2014. 7

[25] Evan Zheran Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, and Percy Liang. Reinforcement learning on web interfaces using workflow-guided exploration. In *International Conference on Learning Representations (ICLR)*, 2018. 8

[26] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *ArXiv*, abs/2304.08485, 2023. 1, 8

[27] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Yuxian Gu, Hangliang Ding, Kai Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Shengqi Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. Agentbench: Evaluating llms as agents. *ArXiv*, abs/2308.03688, 2023. 1

[28] Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and A. Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *ArXiv*, abs/2209.09513, 2022. 1, 8

[29] S. Mazumder and Oriana Riva. Flin: A flexible natural language interface for web navigation. *ArXiv*, abs/2010.12844, 2020. 8

[30] Masoud Monajatipoor, Liunian Harold Li, Mozhdeh Rouhsedaghat, Lin F. Yang, and Kai-Wei Chang. Metavl: Transferring in-context learning ability from language models to vision-language models. *ArXiv*, abs/2306.01311, 2023. 8

[31] OpenAI. Gpt-4 technical report. *ArXiv*, abs/2303.08774, 2023. 1, 2, 8

[32] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke E. Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Francis Christiano, Jan Leike, and Ryan J. Lowe. Training language models to follow instructions with human feedback. *ArXiv*, abs/2203.02155, 2022. 8

[33] Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. Instruction tuning with gpt-4. *ArXiv*, abs/2304.03277, 2023. 8

[34] 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*, abs/2306.14824, 2023. 8

[35] Bryan A. Plummer, Liwei Wang, Christopher M. Cervantes, Juan C. Caicedo, J. Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. *International Journal of Computer Vision*, 123:74 – 93, 2015. 7

[36] 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*, pages 8748–8763. PMLR, 2021. 4

[37] Tanik Saikh, Tirthankar Ghosal, Amish Mittal, Asif Ekbal, and Pushpak Bhattacharyya. Scienceqa: a novel resource for question answering on scholarly articles. *International Journal on Digital Libraries*, 23:289 – 301, 2022. 1, 8

[38] Sheng Shen, Liunian Harold Li, Hao Tan, Mohit Bansal, Anna Rohrbach, Kai-Wei Chang, Zhewei Yao, and Kurt Keutzer. How much can clip benefit vision-and-language tasks? *arXiv preprint arXiv:2107.06383*, 2021. 5

[39] Tianlin Shi, Andrej Karpathy, Linxi (Jim) Fan, Josefa Z. Hernández, and Percy Liang. World of bits: An open-domain

platform for web-based agents. In *International Conference on Machine Learning*, 2017. 8

[40] Aleksandar Shtedritski, C. Rupprecht, and Andrea Vedaldi. What does clip know about a red circle? visual prompt engineering for vlms. *ArXiv*, abs/2304.06712, 2023. 8

[41] Liangtai Sun, Xingyu Chen, Lu Chen, Tianle Dai, Zichen Zhu, and Kai Yu. Meta-gui: Towards multi-modal conversational agents on mobile gui. In *Conference on Empirical Methods in Natural Language Processing*, 2022. 8

[42] Zhiqing Sun, Sheng Shen, Shengcao Cao, Haotian Liu, Chunyuan Li, Yikang Shen, Chuang Gan, Liangyan Gui, Yu-Xiong Wang, Yiming Yang, Kurt Keutzer, and Trevor Darrell. Aligning large multimodal models with factually augmented rlhf. *ArXiv*, abs/2309.14525, 2023. 8

[43] Jianfeng Wang, Zhengyuan Yang, Xiaowei Hu, Linjie Li, Kevin Lin, Zhe Gan, Zicheng Liu, Ce Liu, and Lijuan Wang. Git: A generative image-to-text transformer for vision and language. *ArXiv*, abs/2205.14100, 2022. 8

[44] Zirui Wang, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. Simvlm: Simple visual language model pretraining with weak supervision. *ArXiv*, abs/2108.10904, 2021. 8

[45] An Yan, Zhengyuan Yang, Wanrong Zhu, Kevin Lin, Linjie Li, Jianfeng Wang, Jianwei Yang, Yiwu Zhong, Julian McAuley, Jianfeng Gao, Zicheng Liu, and Lijuan Wang. Gpt-4v in wonderland: Large multimodal models for zero-shot smartphone gui navigation. 2023. 8

[46] Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chun yue Li, and Jianfeng Gao. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. *ArXiv*, abs/2310.11441, 2023. 2, 5, 7, 8

[47] Lingfeng Yang, Yue Wang, Xiang Li, Xinlong Wang, and Jian Yang. Fine-grained visual prompting. *ArXiv*, abs/2306.04356, 2023. 8

[48] Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. The dawn of lmms: Preliminary explorations with gpt-4v(ision). *ArXiv*, abs/2309.17421, 2023. 8

[49] Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. The dawn of lmms: Preliminary explorations with gpt-4v (ision). *arXiv preprint arXiv:2309.17421*, 9, 2023. 8

[50] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *ArXiv*, abs/2207.01206, 2022. 8

[51] Haoxuan You, Haotian Zhang, Zhe Gan, Xianzhi Du, Bowen Zhang, Zirui Wang, Liangliang Cao, Shih-Fu Chang, and Yinfei Yang. Ferret: Refer and ground anything anywhere at any granularity. *ArXiv*, abs/2310.07704, 2023. 8

[52] L. Yu, Bowen Shi, Ramakanth Pasunuru, Benjamin Muller, O. Yu. Golovneva, Tianlu Wang, Arun Babu, Binh Tang, Brian Karrer, Shelly Sheynin, Candace Ross, Adam Polyak, Russell Howes, Vasu Sharma, Puxin Xu, Hovhannes Tamoyan, Oron Ashual, Uriel Singer, Shang-Wen Li, Susan Zhang, Rich James, Gargi Ghosh, Yaniv Taigman, Maryam Fazel-Zarandi, Asli Celikyilmaz, Luke Zettlemoyer, and Armen Aghajanyan. Scaling autoregressive multi-modal models: Pretraining and instruction tuning. *ArXiv*, abs/2309.02591, 2023. 8

[53] Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. *ArXiv*, abs/2311.16502, 2023. 1, 8

[54] Xinlu Zhang, Yujie Lu, Weizhi Wang, An Yan, Jun Yan, Lianke Qin, Heng Wang, Xifeng Yan, William Yang Wang, and Linda Ruth Petzold. Gpt-4v(ision) as a generalist evaluator for vision-language tasks. *ArXiv*, abs/2311.01361, 2023. 8

[55] Yang Zhao, Zhijie Lin, Daquan Zhou, Zilong Huang, Jiashi Feng, and Bingyi Kang. Bubogpt: Enabling visual grounding in multi-modal llms. *ArXiv*, abs/2307.08581, 2023. 8

[56] Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied Sanosi Saied, Weizhu Chen, and Nan Duan. Agieval: A human-centric benchmark for evaluating foundation models. *ArXiv*, abs/2304.06364, 2023. 1, 8

[57] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5122–5130, 2017. 7

[58] Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. *ArXiv*, abs/2307.13854, 2023. 8

[59] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *ArXiv*, abs/2304.10592, 2023. 8

A. Offline Experiment Details

The prompt for action generation is shown in Table 1. For grounding via textual choices, image annotation, and element attributes, the prompts are shown in Tabs. 2 to 4, along with specific task and examples in Figs. 1 to 5.

Table 1. Prompt for SEEACT Action Generation with GPT-4V.

System Role	Imagine that you are imitating humans doing web navigation for a task step by step. At each stage, you can see the webpage like humans by a screenshot and know the previous actions before the current step decided by yourself through recorded history. You need to decide on the first following action to take. You can click an element with the mouse, select an option, or type text with the keyboard. (For your understanding, they are like the click(), select_option() and type() functions in playwright respectively) One next step means one operation within the three.
Action Generation	You are asked to complete the following task: {TASK} Previous Actions: {PREVIOUS ACTIONS} The screenshot below shows the webpage you see. Follow the following guidance to think step by step before outlining the next action step at the current stage: (Current Webpage Identification) Firstly, think about what the current webpage is. (Previous Action Analysis) Secondly, combined with the screenshot, analyze each step of the previous action history and their intention one by one. Particularly, pay more attention to the last step, which may be more related to what you should do now as the next step. (Screenshot Details Analysis) Closely examine the screenshot to check the status of every part of the webpage to understand what you can operate with and what has been set or completed. You should closely examine the screenshot details to see what steps have been completed by previous actions even though you are given the textual previous actions. Because the textual history may not clearly and sufficiently record some effects of previous actions, you should closely evaluate the status of every part of the webpage to understand what you have done. (Next Action Based on Webpage and Analysis) Then, based on your analysis, in conjunction with human web browsing habits and the logic of web design, decide on the following action. And clearly outline which element in the webpage users will operate with as the first next target element, its detailed location, and the corresponding operation. To be successful, it is important to follow the following rules: 1. You should only issue a valid action given the current observation. 2. You should only issue one action at a time

Table 2. Prompt for SEEACT grounding via element attributes. We made a slight modification to prompt action generation and only show the modified part here to save space, as well as the prompt in Table 3 and Table 4.

System Role	Same as Table 1
Action Generation	Slightly modified from Table 1 ... (Next Action Based on Webpage and Analysis) Then, based on your analysis, in conjunction with human web browsing habits and the logic of web design, decide on the following action. And clearly outline which element in the webpage users will operate with as the first next target element, its detailed location, and the corresponding operation. Please also closely examine the screenshot to adequately describe its position relative to nearby elements and its textual or visual content (if it has). If you find multiple elements similar to your target element, use a more precise description to ensure people can distinguish your target element from them through your answer. ...
Format Answer	(Final Answer) Finally, conclude your answer using the format below. Ensure your answer is strictly adhering to the format provided below. Please do not leave any explanation in your answers of the final standardized format part, and this final part should be clear and certain. The element, element type, element text, action and value should be in five separate lines. Format: ELEMENT: Please describe which element you need to operate with. Describe it as detailed as possible, including what it is and where it is. ELEMENT TYPE: Please specify its type from these options: BUTTON, TEXTBOX, SELECTBOX, or LINK. ELEMENT TEXT: Please provide the exact text displayed on the element. Do not invent or modify the text; reproduce it as-is from the screenshot. ACTION: Choose an action from CLICK, TYPE, SELECT. VALUE: Provide additional input based on ACTION. The VALUE means: If ACTION == TYPE, specify the text to be typed. If ACTION == SELECT, specify the option to be chosen. If ACTION == CLICK, write "None".

Table 3. Prompt for SeeAct grounding via textual choices.

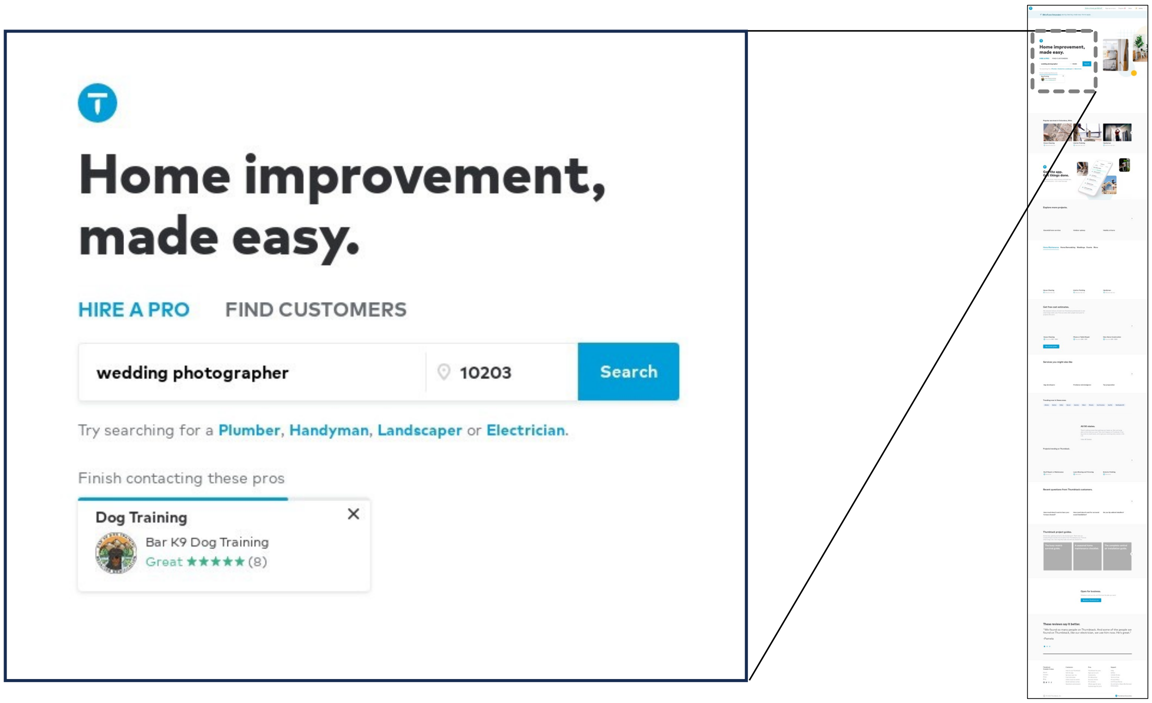
System Role	Same as Table 1
Action Generation	Same as Table 1
Referring Description	(Reiteration) First, reiterate your next target element, its detailed location, and the corresponding operation. (Multichoice Question) Below is a multi-choice question, where the choices are elements in the webpage. From the screenshot, find out where and what each one is on the webpage. Then, determine whether one matches your target element. Please examine the choices one by one. Choose the matching one. If multiple options match your answer, choose the most likely one by re-examining the screenshot, the choices, and your further reasoning. If none of these elements match your target element, please select [None of the other options match the correct element]. A. [CHOICE A] B. [CHOICE B] ...
Format Answer	(Final Answer) Finally, conclude your answer using the format below. Ensure your answer is strictly adhering to the format provided below. Please do not leave any explanation in your answers of the final standardized format part, and this final part should be clear and certain. The element choice, action, and value should be in three separate lines. Format: ELEMENT: The uppercase letter of your choice. ACTION: Choose an action from CLICK, TYPE, SELECT. VALUE: Provide additional input based on ACTION. The VALUE means: If ACTION == TYPE, specify the text to be typed. If ACTION == SELECT, specify the option to be chosen. If ACTION == CLICK, write "None".

Table 4. Prompt for SEEACT grounding via image annotation.

System Role	Same as Table 1
Action Generation	Slightly modified from Table 1 ... The screenshot below shows the webpage you see. In the screenshot, some red bounding boxes and white-on-black uppercase letters at the bottom left corner of the bounding boxes have been manually added. You should ignore them for now. Follow the following guidance to think step by step before outlining the next action step at the current stage: ...
Referring Description	(Reiteration) First, reiterate your next target element, its detailed location, and the corresponding operation. (Verification with the Screenshot) Then, please closely re-examine the screenshot to find whether your target element is marked by a red bounding box and has a white uppercase letter on a black background at the bottom left corner of the bounding box, which is positioned closely next to the bounding box. If yes, use that letter for your final answer. If not, please do not make them up. If it is not marked, please output "NA" as your target element in the following final answer part.
Format Answer	(Final Answer) Finally, conclude your answer using the format below. Ensure your answer is strictly adhering to the format provided below. Please do not leave any explanation in your answers of the final standardized format part, and this final part should be clear and certain. The element choice, action, and value should be in three separate lines. Format: ELEMENT: The uppercase letter of your choice. ACTION: Choose an action from CLICK, TYPE, SELECT. VALUE: Provide additional input based on ACTION. The VALUE means: If ACTION == TYPE, specify the text to be typed. If ACTION == SELECT, specify the option to be chosen. If ACTION == CLICK, write "None".

Example of Grounding via Element Attributes – 1st Turn

Input – 1st turn: Action Generation Prompts
Task: View the profile of a Wedding Photographer near 10203 for a 4 hour wedding on april 13
Previous Actions:
[textbox] Search on Thumbtack -> TYPE: wedding photographer
[textbox] Zip code -> TYPE: 10203



GPT-4V: (Current Webpage Identification)
The current webpage appears to be the homepage of Thumbtack, which is a local services marketplace where users can find professionals for various services, including wedding photography.

(Previous Action Analysis)
The previous actions indicate that the user has typed "wedding photographer" into the search box and entered the zip code "10203" to find a wedding photographer near that location. The intention behind these actions is to search for a wedding photographer in the specified area.

(Screenshot Details Analysis)
The screenshot shows the Thumbtack homepage with the search bar at the top, containing the previously entered search term "wedding photographer" and the zip code "10203." Below the search bar, there are suggestions for different services and popular articles, but these are not relevant to the task at hand. The search button next to the zip code field is visible, indicating that the search has not yet been initiated.

(Next Action Based on Webpage and Analysis)
Based on the analysis, the next logical step is to initiate the search for wedding photographers in the specified zip code area. To do this, the user should click on the "Search" button.

The "Search" button is located to the right of the zip code field at the top of the webpage. It is a rectangular button with rounded corners, and the text "Search" is visible on it. This button is directly adjacent to the zip code field where "10203" has been entered.

The first next action to take is to click the "Search" button to proceed with finding a wedding photographer for the specified requirements.

Figure 1. An example of action generation in grounding via element attributes.

Example of Grounding via Element Attributes – 2^{nd} Turn

Input – 2^{nd} turn: Referring Prompts

GPT-4V: ELEMENT: The "Search" button located to the right of the zip code field at the top of the Thumbtack homepage.

ELEMENT TYPE: BUTTON

ELEMENT TEXT: Search

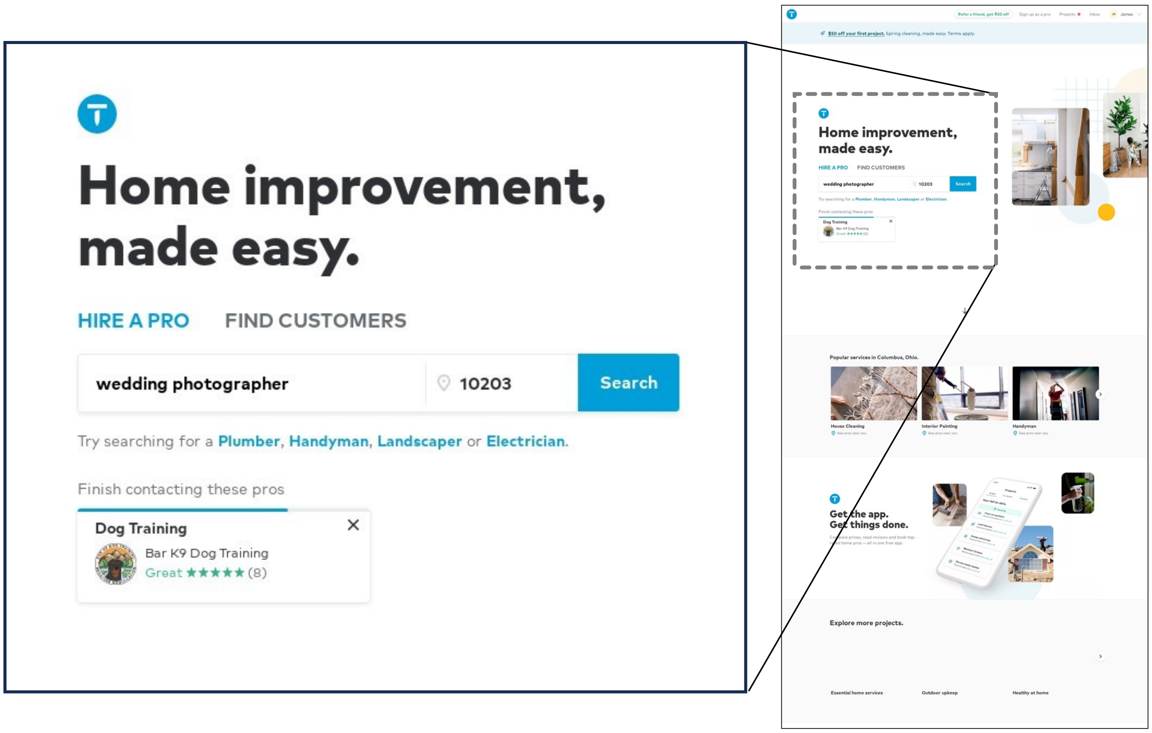
ACTION: CLICK

VALUE: None

Figure 2. An example of grounding via element attributes after action generation.

Example of Grounding via Textual Choices – 1st Turn

Input – 1st turn: Action Generation Prompts
Task: View the profile of a Wedding Photographer near 10203 for a 4 hour wedding on april 13
Previous Actions:
[textbox] Search on Thumbtack -> TYPE: wedding photographer
[textbox] Zip code -> TYPE: 10203



GPT-4V: (Current Webpage Identification)
The current webpage appears to be the homepage of Thumbtack, a platform for hiring local professionals for various services.

(Previous Action Analysis)
The previous actions indicate that the user is looking for a wedding photographer. The user has typed "wedding photographer" into a search box and entered the zip code "10203" into another box, presumably to localize the search to that area.

(Screenshot Details Analysis)
The screenshot shows the Thumbtack homepage with the search fields already populated with "wedding photographer" and "10203." The user has not yet initiated the search, as the search results are not displayed on the screen. The "Search" button is visible next to the zip code field, indicating that the next logical step is to execute the search.

(Next Action Based on Webpage and Analysis)
Based on the analysis, the next action should be to initiate the search for wedding photographers in the specified zip code area. The first next target element is the "Search" button located to the right of the zip code field.

The corresponding operation is to click the "Search" button to proceed with finding a list of wedding photographers near the zip code 10203.

Figure 3. An example of action generation in grounding via textual choices.

Example of Grounding via Textual Choices – 2nd Turn

Input – 2nd turn: Referring Prompts
Choices:
A. <button id=0 button header dropdown button> <div> JS
B.   Refer a friend, get $50 off
C. <div> Profile </div>
D. <button id=3 button> HIRE A PRO </button>
E. <button id=4 button> FIND CUSTOMERS </button>
F. <input id=5 search on thumbtack wedding photographer whats on your
G. <input id=6 zip code zip_code 10203 />
H. <button id=7 submit search> Search </button>
I. ,
J. .
K. <button id=10 button navigation arrow down />
L. <a id=11> <source image/webp /> <div furniture moving and heavy
M. <input id=12 search on thumbtack whats on your to-do list?
N. <input id=13 zip code zip_code 43235 />
O. <button id=14 button> Cancel </button>
P. <button id=15 submit> Search </button>
Q. None of the other options match the correct element

GPT-4V: (Reiteration)
The next target element is the "Search" button located to the right of the zip code field.

(Multichoice Question)
H. <button id=7 submit search> Search </button>

(Final Answer)

ELEMENT: H

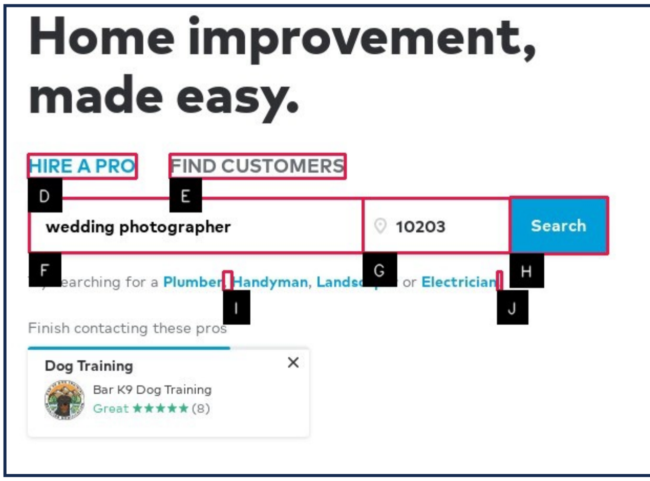
ACTION: CLICK

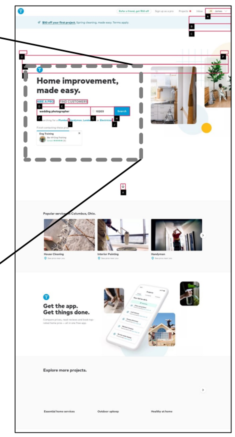
VALUE: None

Figure 4. An example of grounding via textual choices after action generation.

Example of Grounding via Image Annotation

Input – 1st turn: Action Generation Prompts
Task: View the profile of a Wedding Photographer near 10203 for a 4 hour wedding on april 13
Previous Actions:
[textbox] Search on Thumbtack -> TYPE: wedding photographer
[textbox] Zip code -> TYPE: 10203





GPT-4V: (Current Webpage Identification)
The current webpage appears to be the homepage of Thumbtack, a platform for finding local professionals for various services. The page features a search functionality and showcases different categories of home improvement and other services.

(Previous Action Analysis)
The previous actions indicate that the user has typed "wedding photographer" into the search box and entered the zip code "10203" into the zip code field. The intention behind these actions is to search for a wedding photographer near the specified zip code.

(Screenshot Details Analysis)
The screenshot shows that the search term "wedding photographer" has been entered into the search box, and the zip code "10203" has been entered into the zip code field. There is a "Search" button next to the zip code field, which has not yet been activated. This suggests that the user has not yet initiated the search to find wedding photographers near the specified zip code.

(Next Action Based on Webpage and Analysis)
Based on the analysis, the next logical step is to initiate the search for wedding photographers near the zip code "10203." To do this, the user should click on the "Search" button.

The first next target element is the "Search" button located next to the zip code field, which is highlighted with a red bounding box and labeled with the letter "H" in the screenshot.

The corresponding operation is to CLICK on the "Search" button.

Input – 2nd turn: Referring Prompts

GPT-4V: ELEMENT: H

ACTION: CLICK

VALUE: None

Figure 5. An example of grounding via image annotation.

B. Online Experiment Details

We develop an online evaluation tool using Playwright to load webpages and conduct operations generated by web agents. We manually monitor each step of the model and assess whether it finishes the tasks. We explicitly prohibit attempts to log in or perform final submissions to prevent potentially harmful effects.

MindAct. We strictly follow the original settings in MindAct-FLAN-T5 and MindAct-GPT-4. The action space only contains `Click`, `Type`, and `Select`.

SEEACT$_{\text{Oracle}}$ We manually implemented the model's intended actions. The action history was automatically generated by the model, with an added requirement of summarizing the actions into the "Element", "Operation", "Value" format.

SEEACT$_{\text{Choice}}$ We still adopt the top-50 and batch them into three option groups as described in offline experiments. We allow `PRESS ENTER` and `TERMINATE` for the model to make confirmation, and stop the process.

During our tests, pop-up ads on webpages were manually closed. The MindAct model was not trained on pop-up ads and hence lacks the feature to automatically manage ads, which could result in stalling. Conversely, SEEACTcould proactively suggest ad closure through visual analysis and reasoning.

C. Error Examples for Grounding via Image Annotation

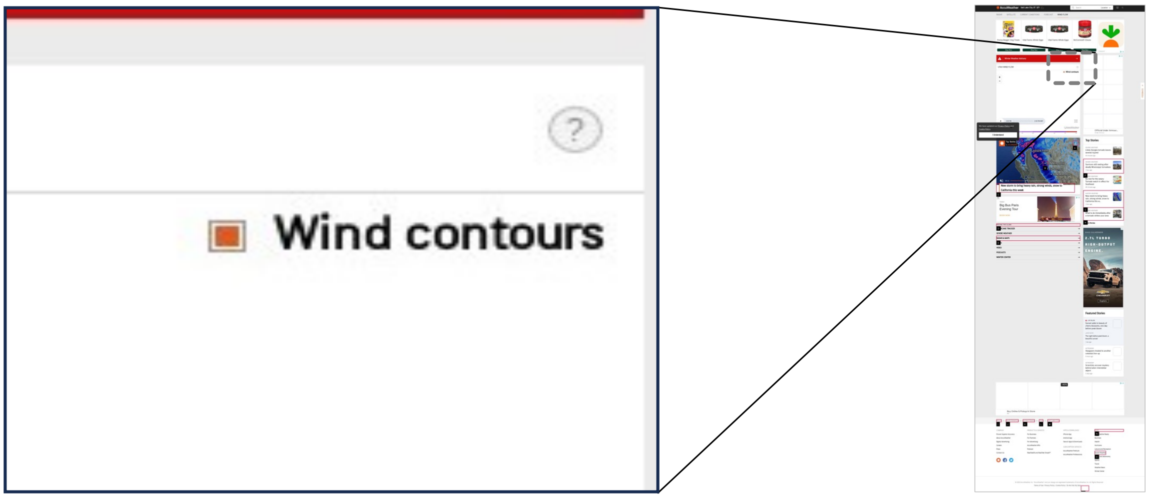
In grounding via image annotation method, we observe significant hallucination errors that can be classified into the following categories:

Making up bounding box & label. In our grounding method, if the correct element is absent from the set of candidate elements, the model is anticipated to generate "None" as the answer. However, as depicted in Figure 6 and Figure 7, the model erroneously claims the element is included within a red bounding box and makes up a wrong index label as the answer.

Failure to link bounding boxes with the correct labels. Another challenge arises in accurately linking bounding boxes to their corresponding index labels. This challenge can be attributed to both LMMs' limitations in understanding relative spatial positions and the complex, dense layout of webpage elements. The model often mistakenly associates the labels of adjacent elements (as illustrated in Figure 8 and Figure 9) or makes up a label entirely (as demonstrated in Figure 10), rather than accurately predicting the intended index label for the targeted element.

Makeup Image Annotation– Example 1

Task: View the wind flow map for Utah, the United States without contours.
Previous Actions:
...
[svg] -> CLICK
[heading] Radar & Maps -> CLICK
[link] WIND FLOW -> CLICK



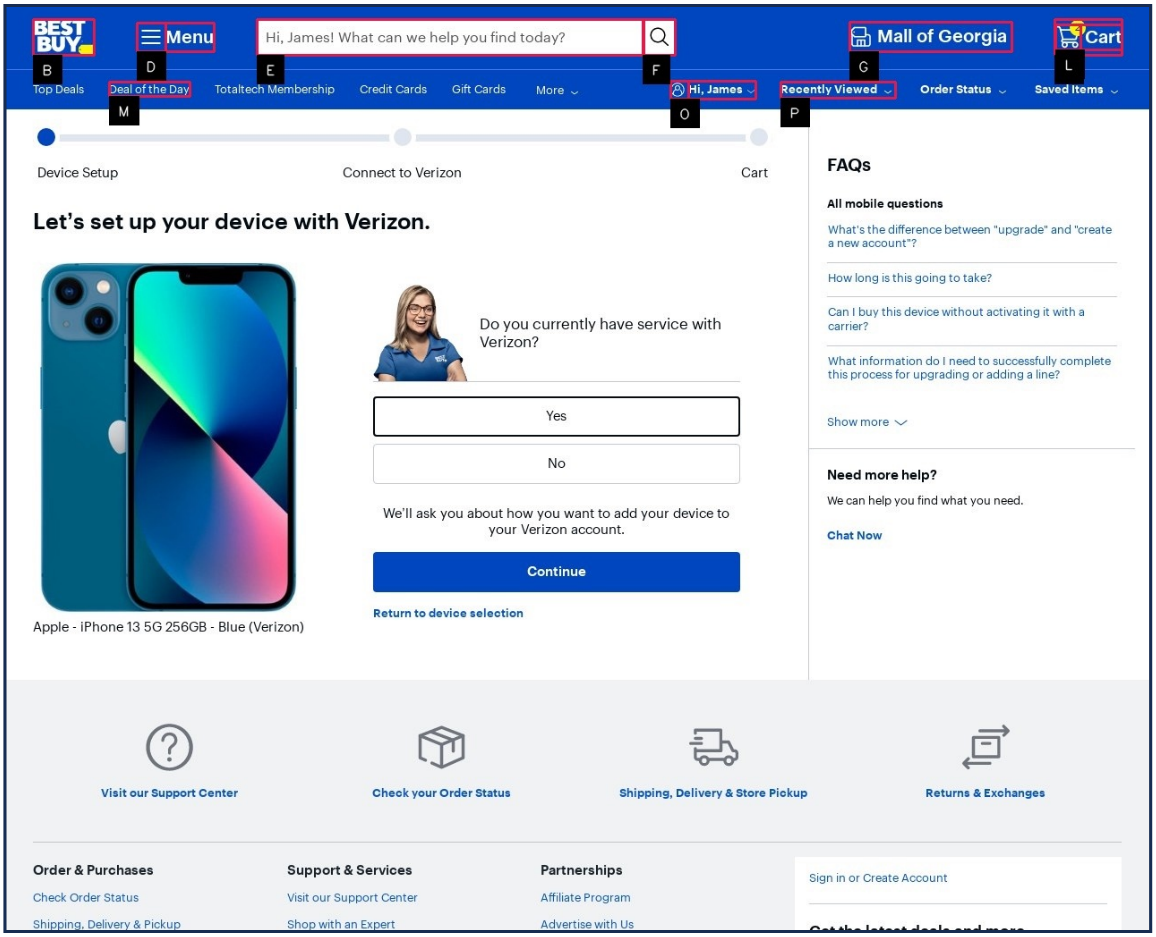
GPT-4V: ...The task at hand is to view this wind flow map without contours. The option to turn off contours might be available on the interface of the current map view, which would adjust the visual presentation of the wind flow data.(Intended Action Based on Current Page and Previous Actions)
To complete the given task, the user would look for a control or setting on the page that allows them to adjust the map view, particularly to turn off or hide the contours. In the current interface, it looks like there might be a button or link that toggles the contours on the map since the task specifies viewing the map without them.
Given that the red highlights and uppercase letters should be ignored, I would look for a map legend, settings icon, or a layer control option that is typically found on interactive maps. Such an element would likely be on or near the map itself.
Since there is a button labeled "Wind contours" on the upper right corner of the wind flow map (it is actually highlighted with the label "B"), this would likely be the control to toggle the contours on the map.
The next step, therefore, would be to:
Move the mouse cursor to the "Wind contours" button.
Click the "Wind contours" button to toggle off the contours and view the wind flow map without them.

ELEMENT: B
ACTION: CLICK
VALUE: None

Figure 6. In this example, the model describes a correct element in action generation. However, the the identified element is absent from the set of candidate elements. Despite this, the model erroneously claims the target element is included within a bounding box and assign it the index label "B".

Makeup Image Annotation – Example 2

Task: As a Verizon user, finance a new blue iPhone 13 with 256gb along with monthly apple care.
Previous Actions:
...
[checkbox] AppleCare+ Theft and Loss for iPhone 13 - Monthly ... -> CLICK
[button] Add to Cart -> CLICK
[button] Let's Get Started -> CLICK
[label] Yes -> CLICK

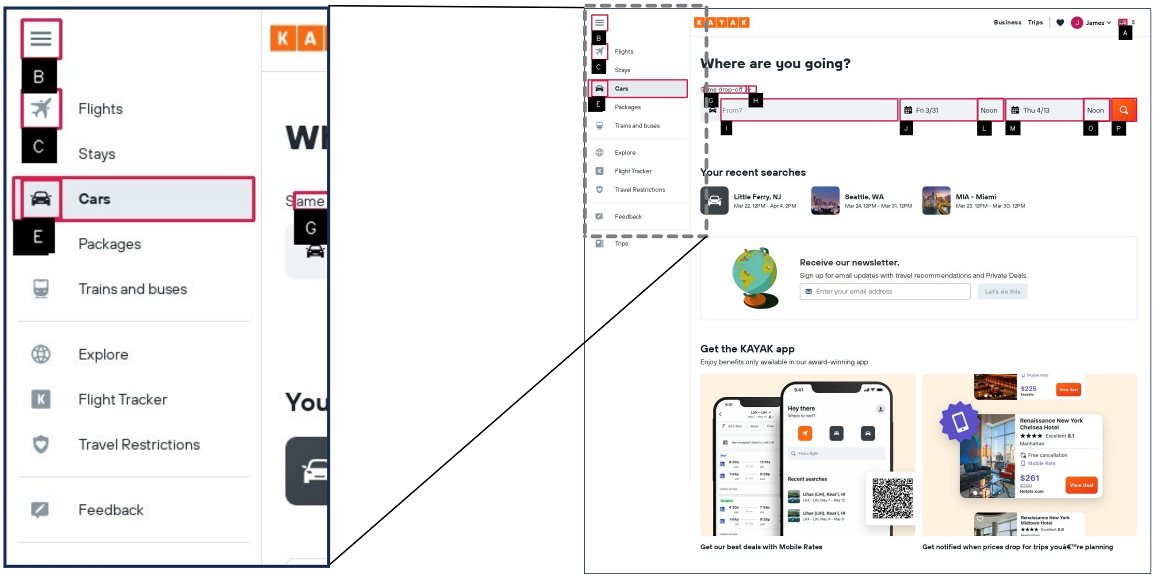


GPT-4V: ...The logical next step is to click the "Continue" button. This action will likely take the user to the next stage of the setup process, where they can continue with the financing options for their new iPhone 13 and add the monthly AppleCare...
...**After reviewing the screenshot provided, the "Continue" button does indeed match one of the highlighted elements with a red bounding box, labeled with the letter H.**
Standardized Summary:
ELEMENT: H
ACTION: CLICK
VALUE: None

Figure 7. In this example, the model predicts the appropriate element. Nevertheless, the identified element is not present in the provided image options. Despite this, the model erroneously claims that it is highlighted by red bounding boxes and assigns it with the label 'H'.

Failure of Linking - Example 1

Task: search the cheapest small car rental deals from Little Ferry, New Jersey, United States on 23th March to same location dropoff on april 4th 2pm.
Previous Actions:
None

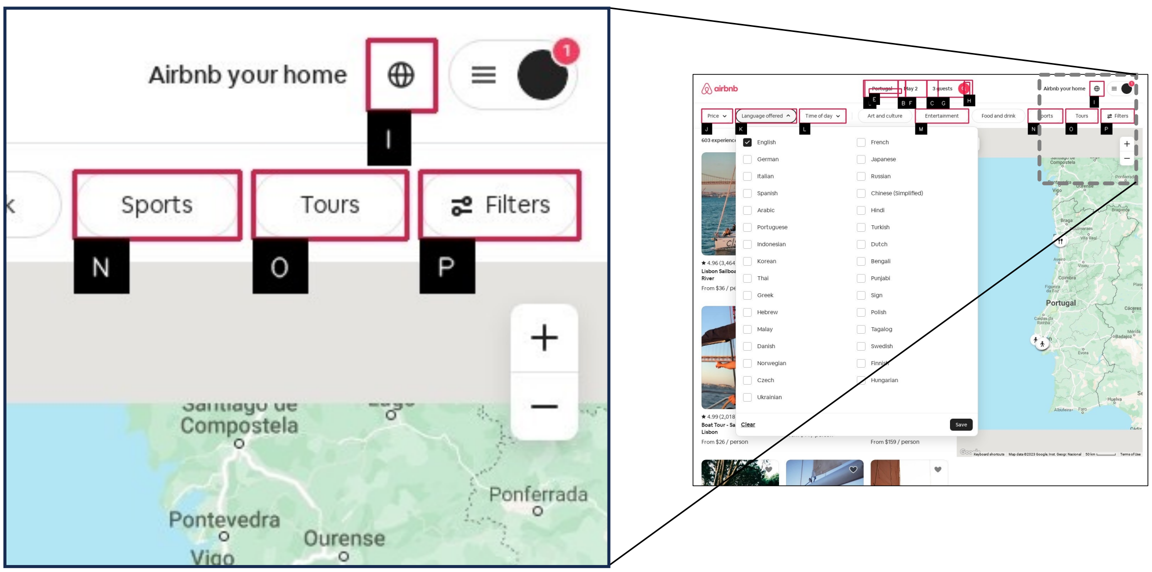


GPT-4V: ...The next step for the user would be to start inputting the required information for the car rental search:
The user should navigate to the "Cars" section of the search interface. Even though the screenshot shows that the "Cars" option is already selected (as indicated by the red bounding box and label 'C'), the user needs to ensure they are in the correct tab to proceed with the car rental search.
Therefore, the next immediate action for the user would be to ensure they are in the "Cars" tab and begin inputting the specific location and date details for the car rental search.
ELEMENT: C
ACTION: CLICK
VALUE: None

Figure 8. In this case, while the model predicts the appropriate element, it incorrectly associates the element with the nearby label 'C' instead of the correct label 'E'.

Failure of Linking - Example 2

Task: Find morning sports experiences in english for one adult and 2 children in portugal on may 2
Previous Actions:
[button] Location Anywhere -> CLICK
[span] Experiences -> CLICK
[textbox] Where -> TYPE: portugal
[div] Portugal -> CLICK
[button] 2, Tuesday, May 2023. Available. Select as start d... -> CLICK
[div] Add guests -> CLICK
[path] -> CLICK
[button] increase value -> CLICK
[span] -> CLICK
[button] Language offered No Filter applied -> CLICK
[span] -> CLICK

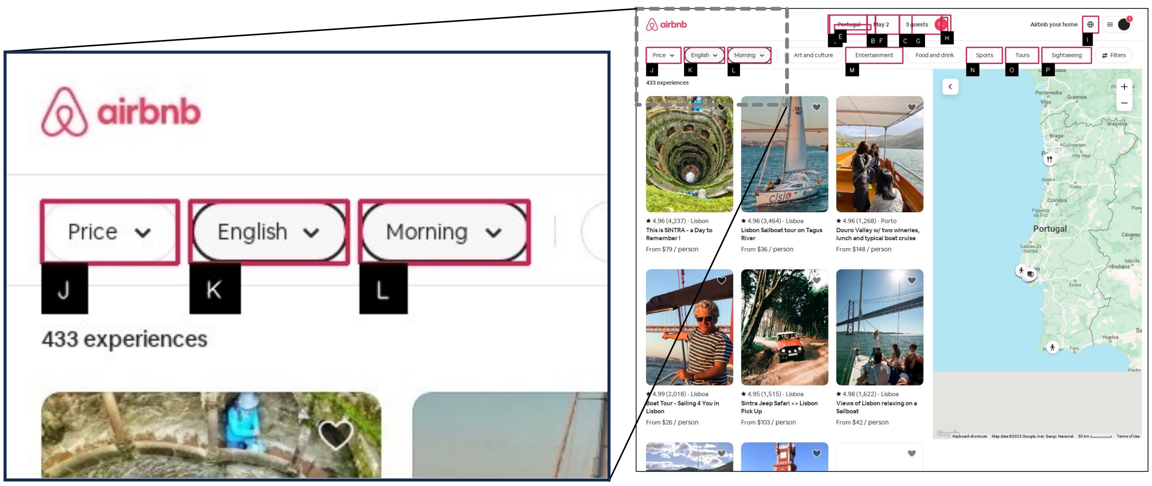


GPT-4V: ...Since the user is looking for sports experiences in the morning, they would need to:
Click on the "Sports" filter (labeled with the letter "O" next to the red box) to refine the search to only display sports-related experiences.
Then, they should click on the "Time of day" filter (labeled with "L" next to the red box) and select "Morning" to ensure that the experiences shown are available in the desired time slot.
These actions would bring the user closer to finding the experiences that match the specified criteria.
ELEMENT: O
ACTION: CLICK
VALUE: None
The first next target element to complete the task is the "Sports" filter on the Airbnb experiences page. The detailed location of this element is within the filters section, specifically labeled with the letter "O" next to the red bounding box in the provided screenshot. The corresponding operation is to click on this "Sports" filter to refine the search results to sports experiences.

Figure 9. In this case, while the model predicts the appropriate element, it incorrectly associates the element with the nearby label 'O' instead of the correct label 'N'.

Failure of Linking - Example 3

Task: Find morning sports experiences in english for one adult and 2 children in portugal on may 2
Previous Actions:
[button] Location Anywhere -> CLICK
[span] Experiences -> CLICK
[textbox] Where -> TYPE: portugal
[div] Portugal -> CLICK
[button] 2, Tuesday, May 2023. Available. Select as start d... -> CLICK
[div] Add guests -> CLICK
[path] -> CLICK
[button] increase value -> CLICK
[span] -> CLICK
[button] Language offered No Filter applied -> CLICK
[span] -> CLICK
[button] Save -> CLICK
[div] Time of day -> CLICK
[span] -> CLICK
[button] Save -> CLICK



GPT-4V: ...
ELEMENT: H
ACTION: CLICK
VALUE: None
Analysis of highlighted elements as choices:
After re-evaluating the highlighted elements, the one that matches the intended action based on the previous analysis is element H. This is because the user needs to verify and potentially adjust the time of day for the experiences. **Element H, which is labeled "Morning" and is enclosed by a red bounding box, aligns with the requirement to find morning activities**. Thus, the next action would be to click on this element to filter the experiences by the time of day, specifically for morning offerings.

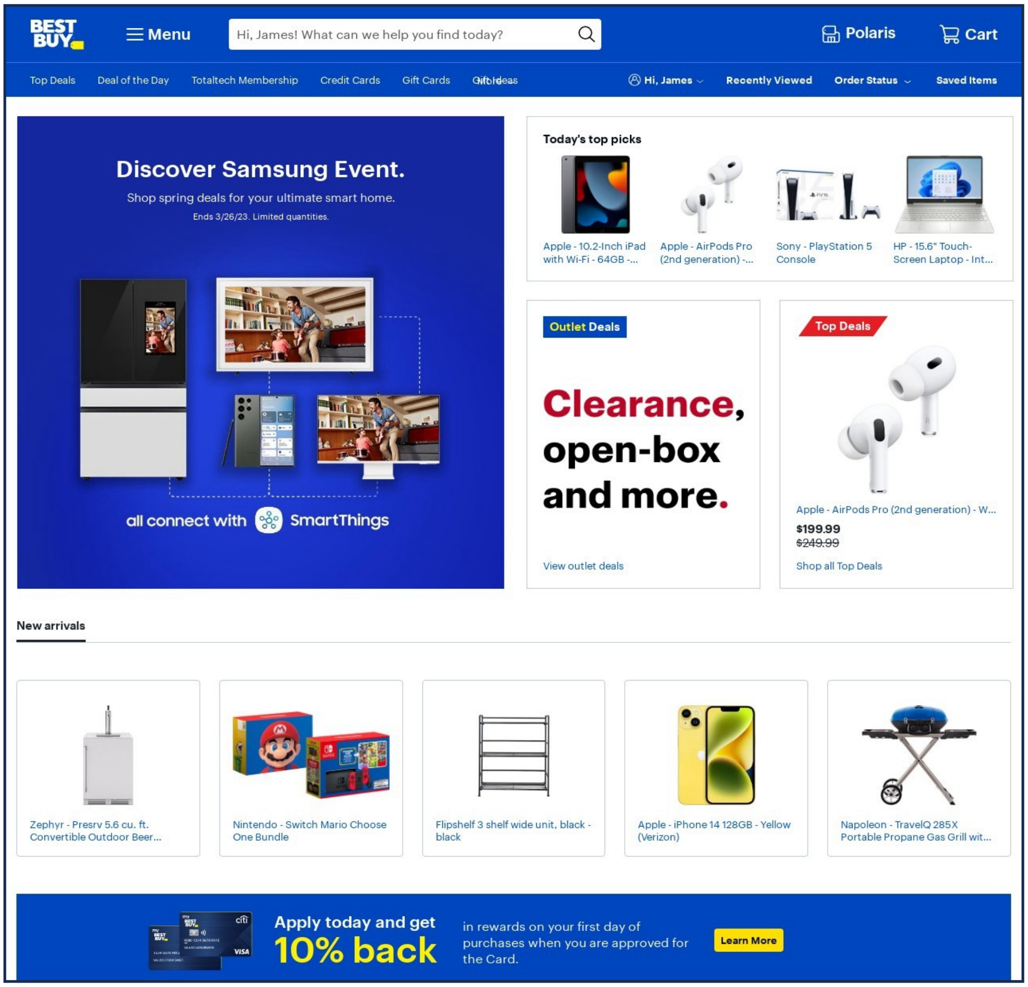
Figure 10. In this case, while the model predicts the appropriate element, it incorrectly associates the element with the label 'C' instead of the correct label 'E'.

D. Strong Capability of Planning

GPT-4V shows remarkable understanding and planning capabilities during our experiments. As depicted in Figure 11, the model is capable of understanding the website and generate a full plans for the given task involving multiple low-level tasks. Specifically, GPT-4V could understand reasonably well about the process and the remaining work of the task by its careful examination of the webpage, as shown in Figure 12.

Strong Capability of Planning

Task: View the speakers that are bluetooth and wireless and filter the results to only show models that are on sale and cost less than $50.
Previous Actions:
None

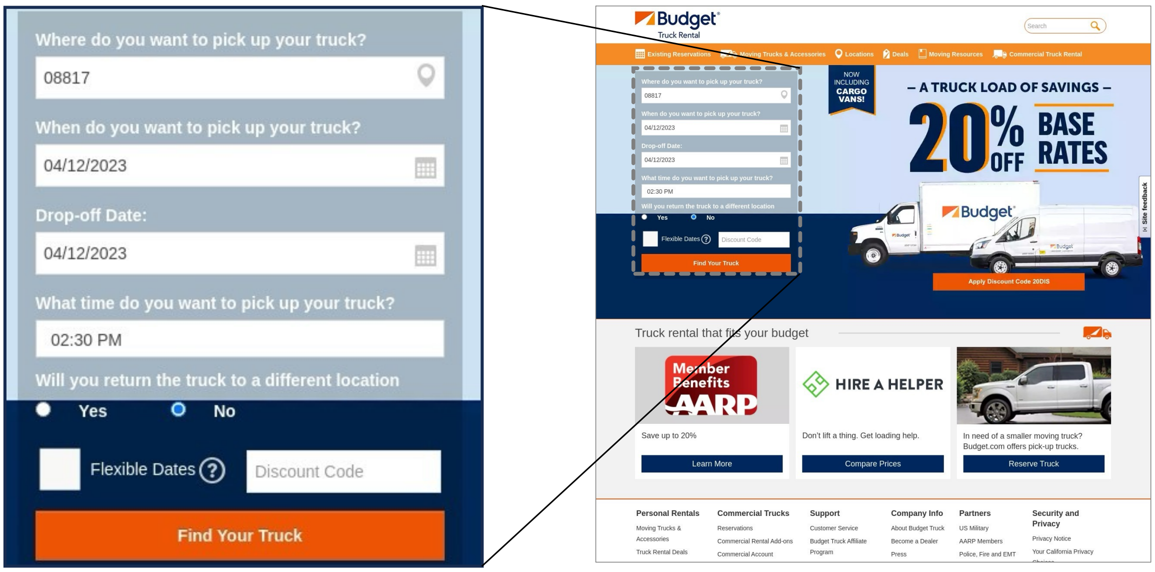


GPT-4V: … To proceed with the task, the user should: **Locate the menu bar to find the category for audio equipment or speakers. This is typically found in a dropdown under a section such as "Products", "Categories", or directly as "Audio". Once the correct category is found, the user should click on it to navigate to the section of the website that lists speakers. After reaching the speakers section, the user will need to use the website's filtering options to select 'Bluetooth' and 'Wireless' features, and then set the price filter to show only the models that are on sale and cost less than $50.** …

Figure 11. In this example, the model generates a comprehensive plan for the task, including subsequent actions on following pages that are not currently visible.

Strong Capability of Understanding

Task: Rent a truck with the lowest rent with two dozen furniture pads for 100 miles at zip 08817 on April 12 at 2:30 pm rented truck is to be returned to the exact location and date, and the pickup and drop off will be at the nearest location.
Previous Actions:
...
[textbox] US City,State or Zip Code -> TYPE: 08817
[textbox] mm/dd/yyyy -> CLICK
[link] 12 -> CLICK
[select] Select Pick-up Time -> SELECT: 02:30 PM
[input] -> CLICK



GPT-4V: ... The current webpage appears to be the truck rental reservation page of the Budget Truck Rental website. This page is designed for customers to input their rental details such as pickup location, dates, and times.
(Screenshot Details and Previous Action Analysis)
The screenshot shows that the user has already entered the pickup location (zip code 08817), selected the pickup date (April 12, 2023), and chosen the pickup time (2:30 PM). The drop-off date is also set to the same day, which aligns with the task requirement of returning the truck to the same location on the same date. The option "Will you return the truck to a different location" is set to "No," which is correct for the task at hand. The last action taken was a click, but it's not clear on what element the click was performed. However, based on the task, it's likely that the user is ready to proceed to the next step of finding a truck.
(Intended Action Based on Current Page and Previous Actions)
The next logical step for the user to take would be to click on the "Find Your Truck" button. This action will likely lead to a page where the user can select the type of truck they wish to rent and add any additional accessories, such as the two dozen furniture pads mentioned in the task. Since the task specifies renting the truck with the lowest rent, the user will need to compare the available options and select the most cost-effective one.
The "Find Your Truck" button is located in the central portion of the webpage, just below the input fields for the rental details. The corresponding operation would be to move the cursor over the "Find Your Truck" button and perform a click action....

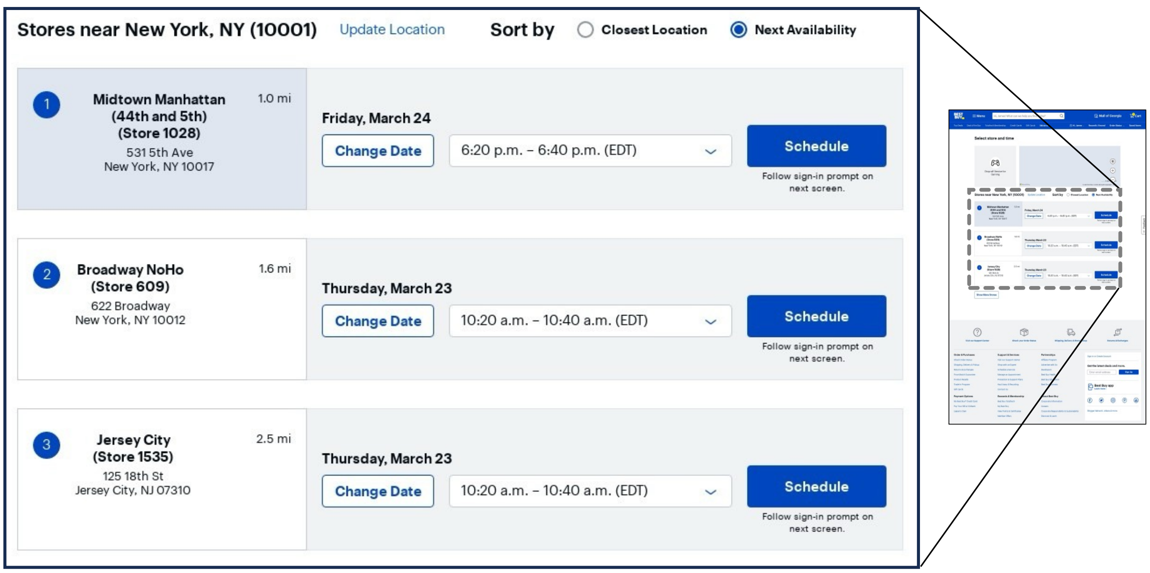
Figure 12. In this case, two critical pieces of information are inadequately captured by the textual history. Firstly, the website automatically set the drop-off date to the same day. Secondly, secondly, the 'No' button was selected (However it was missed in previous actions due to the button's lack of text). Nevertheless, the model discerns these details through a meticulous analysis of the screenshot, enabling it to make the correct decision for the next step.

E. Challenges in Grounding via Textual Choices

Although textual choices achieved the best results among the three grounding approaches, it still suffers from challenges of similar or identical elements which are common in webpages. The model tends to choose the first text choice that seemingly corresponds to its intention. Moreover, this is inevitable, as web pages indeed contain many elements that may even have exact identical HTML information, as the "Schedule" button shown in Figure 13.

Identical Elements

Task: Schedule a repair service for my gaming console closest to the zip code 10001 on March 24, anytime after 6 pm.
Previous Actions:
...
[[link] Schedule a Service -> CLICK
[p] Video Games & Entertainment -> CLICK
[heading] Schedule a repair or tune-up -> CLICK
[button] Update Location -> CLICK
[textbox] Enter Zip Code or City and State -> TYPE: 10001
[button] Find Stores -> CLICK
[button] Change Date -> CLICK
[button] March 24, 2023, is available -> CLICK
[combobox] Select a Time Slot -> SELECT: 6:20 p.m. – 6:40 p.m. (EDT)



GPT-4V: ... The user is currently in the process of selecting a store location and time for the service.(Details and Previous Action Analysis)
The user initiated this process by navigating to the support & services section of the Best Buy menu, then selecting the option to schedule a service. They narrowed down their service type by selecting the "Video Games & Entertainment" category and chose to schedule a repair or tune-up. The user has also updated their location using the zip code 10001 to find stores near that area. They then selected the date of March 24, 2023, and chose the available time slot of 6:20 p.m. – 6:40 p.m. (EDT) for their service. This time slot fits the requirement of scheduling the service anytime after 6 pm on March 24th.From the screenshot, three store options near the zip code 10001 are listed. The closest one is the Midtown Manhattan location, which is 1.0 mile away from the provided zip code. This store already has the selected date and time of Friday, March 24, from 6:20 p.m. to 6:40 p.m. The last action from the previous action history indicates that the user has selected the time slot of 6:20 p.m. – 6:40 p.m. (EDT). This suggests that the user is now ready to finalize and confirm this scheduled service.(Intended Action Based on Current Page and Previous Actions)
Considering the details and the objective to schedule a repair service, the next logical step for the user would be to confirm and schedule the service at the selected store, date, and time. On the current page, the element that seems to be relevant to this action is the "Schedule" button associated with the selected store, date, and time. **Therefore, the next action step is:**
[button] Schedule (associated with the Midtown Manhattan location for the selected date and time) -> CLICK.

Figure 13. In this example, there are three identical 'Schedule' buttons, making it impossible for SEEACT$_{\text{Choice}}$ to distinguish among them. We empirically find that the model tend to choose the first one among the choices.

F. Knowledge and Reasoning Requirement

Some tasks require a certain degree of reasoning and knowledge, which may be challenging for fine-tuned models like MindAct. For instance, the task in Figure 14 necessitates the model to know the specific district of Dublin in Virginia. In the task of Figure 15, the model correctly provided the IATA airport code of airports in Indira Gandhi and Los Cabos.

Knowledge Requirement– Example 1

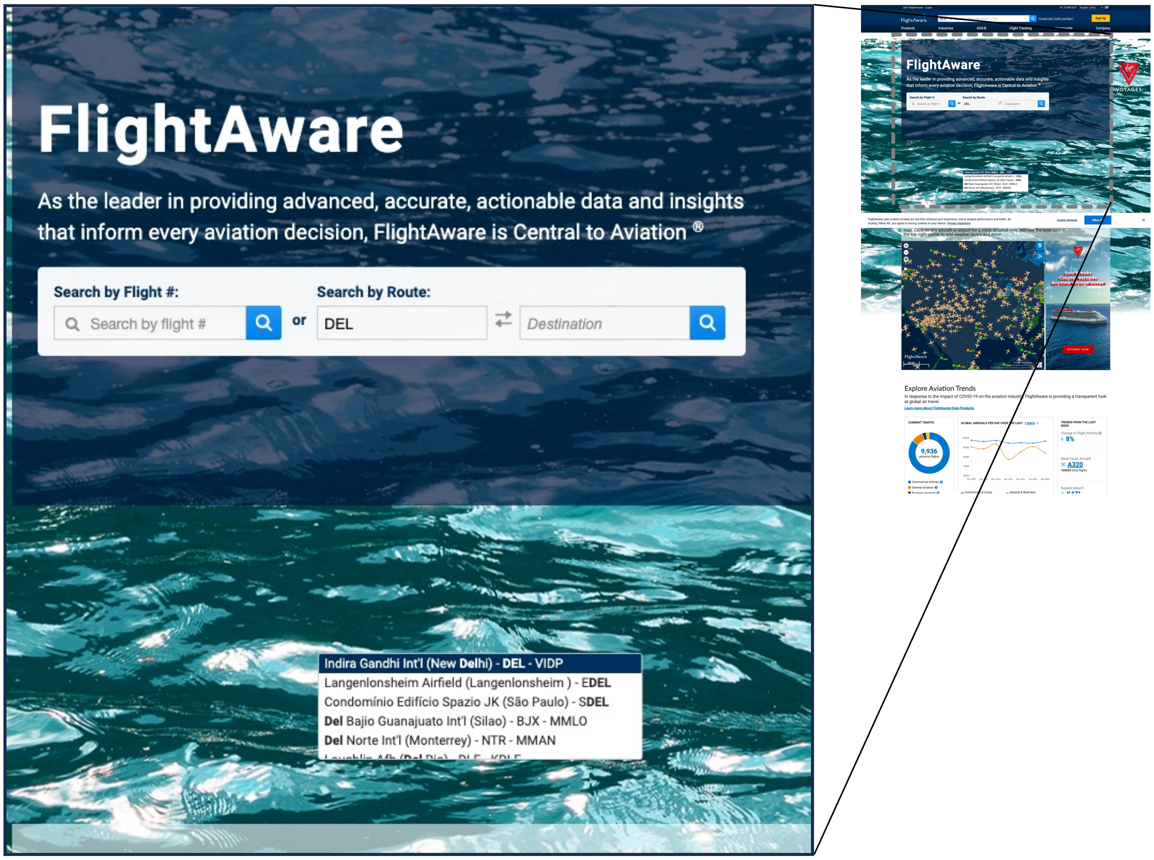
Task: Find a driver training school in Dublin
Previous Actions:
…
[link] All -> CLICK
[link] Attending a Driver Training School -> CLICK
[link] Driver Training Schools -> CLICK
[link] Find a Driver Training School Near You -> CLICK

GPT-4V: … The current page has a list of districts, which implies that the user may need to select a district closest to Dublin, Virginia, to continue their search. **However, since there is no district explicitly named "Dublin," the user must know which district Dublin falls under or look for an option that allows for a more specific search.**
(Intended Action Based on Current Page and Previous Actions)
The first next target element that the user would likely need to operate with is the district name that includes or is closest to Dublin, Virginia. However, the screenshot does not provide enough detail to determine which district Dublin is in.…

Figure 14. In this example, the task necessitates knowledge about which district Dublin is located in.

Knowledge Requirement – Example 2

Task: Find flights going from Indira Gandhi to Los Cabos.
Previous Actions:
[textbox] Origin -> TYPE: DEL



GPT-4V: ... Then, the user needs to identify the IATA code for Los Cabos International Airport, which is "SJD," and enter it into the destination field. Thus, the next target element is the destination field, which is located to the right of the origin field in the "Search by Route" section of the search interface.
Here is the action to be taken:
[textbox] Destination -> TYPE: SJD (after ensuring the origin field correctly contains "DEL").

Figure 15. In this example, the task requires knowledge of the IATA code for Los Cabos International Airport. GPT-4V accurately provides the correct code.

G. Path Variation and Awareness of Error Correction

On webpages, multiple paths often exist to accomplish a given task. For instance, varying the execution order of actions within an interchangeable sequence can result in diverse routes to task completion. Additionally, the agent can navigate to different webpages but still accomplish the give tasks. Figure 16 presents a straightforward example where the model chose a more direct route that differs from the ground truth annotated in the dataset.

When running on live website, the agent's previous action histories is likely to be filled with redundant, unnecessary, erroneous, failed operations generated, or merely exploratory attempts by the model, resulting in a final path that deviates significantly from the ground truth. Despite these circumstances, the model can still accomplish the task amidst numerous incorrect explorations. The process of exploration and correction requires the model to possess a sense of self-correction. As shown in Figure 17, GPT-4V demonstrates this awareness of correcting errors caused by previous steps.

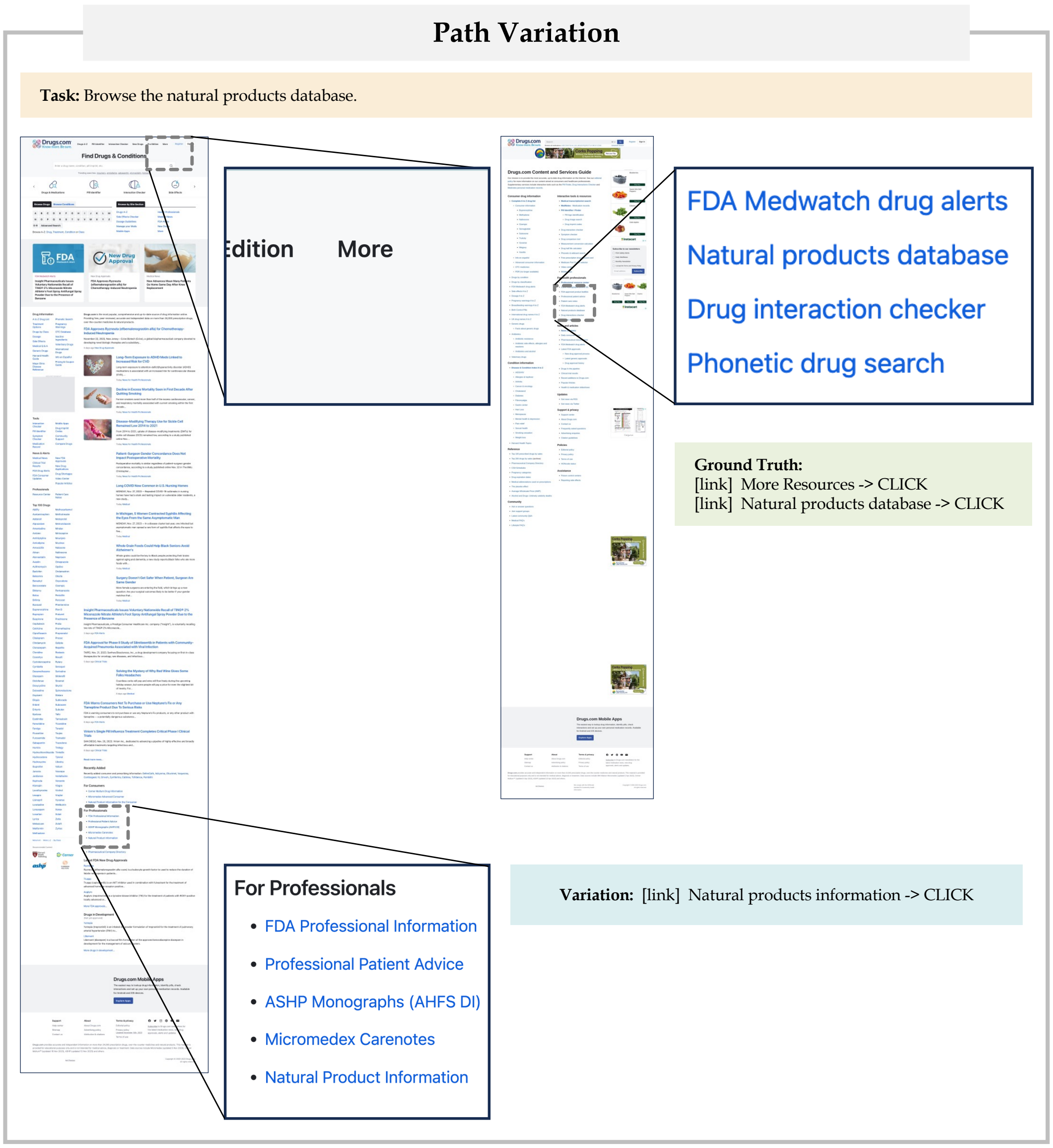
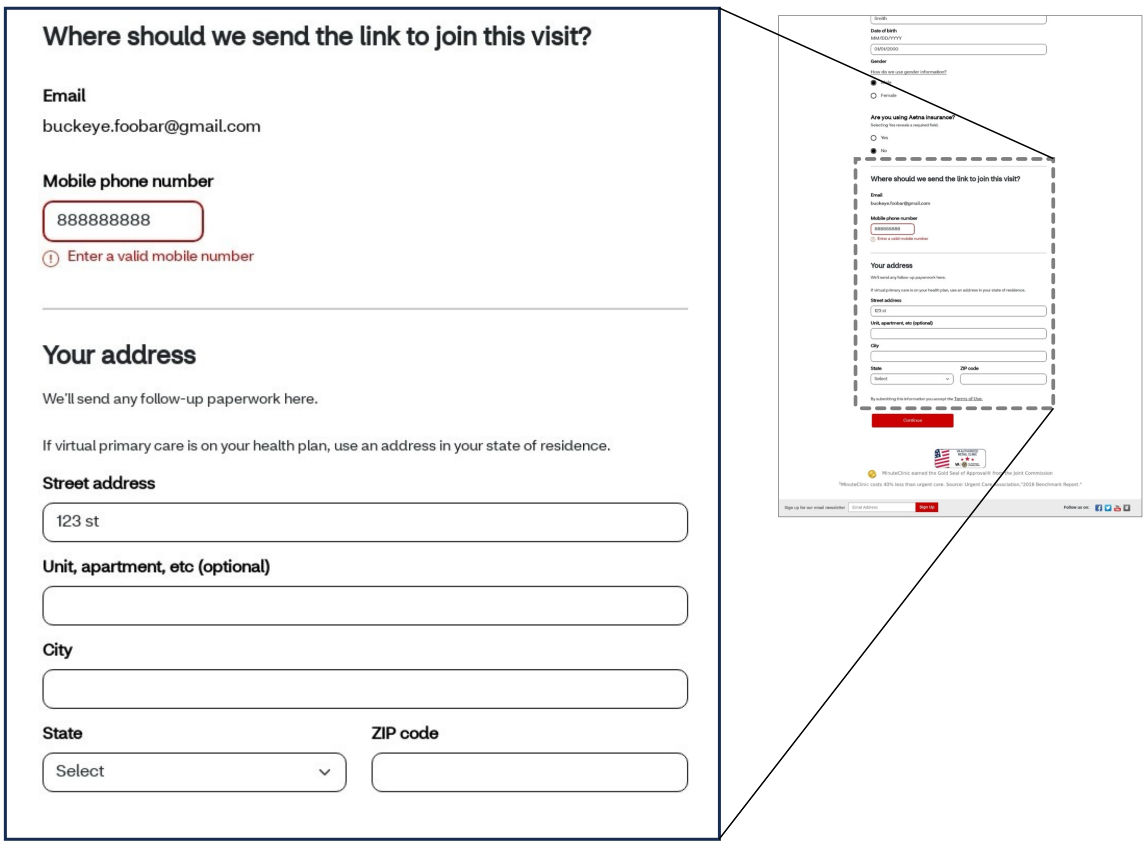


Figure 16. In this example, the ground truth in Mind2Web is to firstly click "More", then click "Natural products database" on the second page. In contrast, the model identifies a more direct approach, achieving the target page through 'Natural products information' on the first page.

Awareness of Error Correction

Task: Signup for virtual healthcare visit.
Previous Actions:
...
[span] James Smith -> CLICK
[button] Select patient and continue -> CLICK
[textbox] Date of birth -> CLICK
[textbox] Date of birth -> TYPE: 01/01/2000
[radio] Male -> CLICK
[radio] No -> CLICK
[textbox] Mobile phone number -> TYPE: 888888888
[textbox] Street address -> TYPE: 123 st



GPT-4V: ... Since the mobile phone number field is highlighted, it suggests there might be a problem with the entry—such as an incorrect format or not enough digits. **Therefore, the next action would be to correct the mobile phone number by clicking on the mobile phone number text box and entering a valid number.** Once that's corrected, the user should continue filling out the rest of the form with the city, state, and ZIP code before proceeding to click the "Continue" button to move forward with the signup process...

Figure 17. In this example, the webpage displays an error message indicating an invalid phone number, a consequence of prior actions. The model identifies this error and prioritizes its immediate rectification, foregoing the subsequent planned steps.