Dynamic Planning for Graphical User Interface Automation with LLM Agents

Anonymous ACL submission

Abstract

The advent of large language models (LLMs) has spurred considerable interest in advancing autonomous agents, empowering them to tackle real-world tasks by perceiving distinct environments, formulating plans, and executing actions. An intriguing application of these agents is within smartphone graphical user interfaces (GUIs). Upon receiving a task goal, the agent generates step-by-step plans and engages in iterative interactions until task completion. However, it remains an open challenge how to generate effective plans to guide the action prediction. Current studies often confine themselves to static plans or lack specific plans entirely. Given that the environment evolves following action execution, the imperative is to adapt plans dynamically based on environmental feedback and action history. To address the challenge, we propose DP-Agent, a novel approach designed to cultivate dynamic planning in agents. DP-Agent involves the dynamic adjustment of planning based on feedback from the environment and interaction history. Experimental results reveal that DP-Agent exhibits superior performance, surpassing the widely adopted GPT-4V baseline by +8.81% (35.58% $\rightarrow$ 44.39%) on the AITW benchmark dataset. Our analysis highlights the efficacy of dynamic planning in not only enhancing action prediction accuracy but also in adapting to previously unfamiliar tasks.

1 Introduction

The pursuit of building autonomous agents that can help humans tackle real-world problems is a long-standing goal of artificial intelligence (Searle, 1972; Wooldridge and Jennings, 1995; Maes, 1994). Recently, large language models (LLMs), such as ChatGPT and GPT-4, have spurred heightened exploration in the realm of autonomous agent (Chowdhery et al., 2022; Wei et al., 2023; Achiam et al., 2023). These agents have shown promising opportunities to address real-world tasks via perceiving distinct environments, formulating plans, and executing actions. Meanwhile, they have demonstrated remarkable capabilities in critical thinking, reasoning, and ultimately, the execution of actions across distinct environments (Huang and Chang, 2023; Yao et al., 2023a; Wang et al., 2023b; Chen et al., 2023).

The early stages of autonomous agent research predominantly concentrated on processing textual information, encompassing environmental descriptions and interaction processes under a text-based framework (Searle, 1969; Wooldridge and Jennings, 1995; Maes, 1998; Hendler, 1999). However, recent advancements signal a paradigm shift from solitary text-based frameworks to more comprehensive multimodal approaches (Wu et al., 2023; Surís et al., 2023; Gupta and Kembhavi, 2022). Multimodal agents demonstrate proficiency in assimilating inputs from various modalities, significantly broadening their applicability across a diverse spectrum of scenarios.

A prevalent scenario is smartphone graphical user interface (GUI) automation, where agents are tasked with controlling smartphones to execute complex instructions through multi-turn interactions (Rawles et al., 2023; Wen et al., 2023). Representative approaches include fine-tuning multi-modal models (Zhang and Zhang, 2023; Hong et al., 2023), or prompting GPT-4V to understand the GUI and execute actions (Yan et al., 2023; Zhang et al., 2023). Nonetheless, the existing body of research concentrates on environment perception, e.g., understanding visual modalities. It remains an open challenge how to generate effective plans to guide the action prediction.

Specifically, one key challenge is the dynamic adjustment of plans based on feedback from the environment and interaction history. The significance of planning in influencing task performance has been well-established (Zhao et al.,

2023; Wang et al., 2024a,b; Yang et al., 2024). However, current studies commonly determine actions with static plans (Zhang and Zhang, 2023) or even without specific plans (Yan et al., 2023; Zhang et al., 2023). For example, CogAgent (Hong et al., 2023) first generates the complete plan upon receiving the task instruction and initial environment and then executes the actions step-by-step accordingly. Given that the environment evolves following action execution, it becomes imperative to dynamically adapt plans based on environmental feedback and action history. This adaptive approach ensures that plans remain effective amid changing circumstances.

To tackle the aforementioned challenge, this paper introduces the Dynamic Planning agent (DP-Agent). DP-Agent dynamically adjusts its plans by leveraging real-time environmental feedback and interaction history throughout the goal attainment process. This adaptability allows the DP-Agent to continuously refine its approach, ensuring persistent optimization until the desired objective is successfully achieved. Experimental results reveal that DP-Agent exhibits superior performance, surpassing the widely adopted GPT-4V baseline by +8.81% (35.58% $\rightarrow$ 44.39%) on the AITW benchmark dataset.

In summary, our work contributes to the field in the following technical aspects:

(i) This work introduces a novel Dynamic Planning agent dubbed DP-Agent. DP-Agent dynamically formulates plans and selects appropriate steps from these plans for action execution. This process is based on environmental feedback and execution history, enhancing the agent's performance and adaptability.

(ii) DP-Agent demonstrates impressive performance on the AITW benchmark dataset (Rawles et al., 2023), notably achieving an +8.81% overall performance improvement compared to the widely adopted GPT-4V baseline. Our work highlights the efficacy of dynamic planning in not only enhancing action prediction accuracy but also in adapting to previously unfamiliar tasks.

(iii) We conducted an extensive analysis to scrutinize the various factors influencing the planning process. Our findings are systematically categorized to elucidate the advantages and disadvantages associated with different factors. This comprehensive examination provides valuable insights into the intricacies of planning dynamics, contributing to a deeper understanding of the field.

2 Related Work

Our work focuses on the use of LLMs, and this section will first review the recent progress of the work on building the mobile control agents and then discuss the planning mechanism of the agents.

2.1 LLMs Agent

LLMs have spurred considerable interest in the realm of language agents, which adeptly adhere to language instructions and execute actions in interactive environments. Notable examples include AutoGPT (Yang et al., 2023a), HuggingGPT (Shen et al., 2023), and MetaGPT (Xi et al., 2023), all of which explored the integration of LLMs as the core of agents aimed at addressing real-life problems.

This work focuses on the development of LLM agents as intelligent assistants for smartphones. These assistants are crafted to assist people in accomplishing their daily tasks and meeting life's requirements, especially enhancing accessibility for individuals with disabilities. Notably, the advent of multi-modal agents such as GPT-4V, showcasing robust image understanding capabilities (Yang et al., 2023b), has prompted previous research to predominantly concentrate on comprehending GUI interactions. For instance, MM-Navigator delved into leveraging optical character recognition (OCR) parsing to enhance GPT-4V's GUI comprehension (Yan et al., 2023), while AppAgent reinforced the understanding of Application GUI elements by introducing the roles of distinct GUI (Zhang et al., 2023). In addition to these, CogAgent fine-tuned the agent's understanding of GUI to enhance performance (Hong et al., 2023).

In contrast to the prior research that concentrates on multimodal perception, our work focuses on the planning mechanism to enhance the agent's proficiency in planning and effectively tackle multi-step tasks on smartphones. Specifically, our approach dynamically updates the plan based on the current environment and execution history, providing a unique perspective that distinguishes our work from previous efforts in this field.

2.2 Planning Mechanisms in LLM Agents

LLMs have shown considerable potential in constructing agents with strong capabilities in following instructions and maintaining coherent chains of thought (CoT) via solving complex

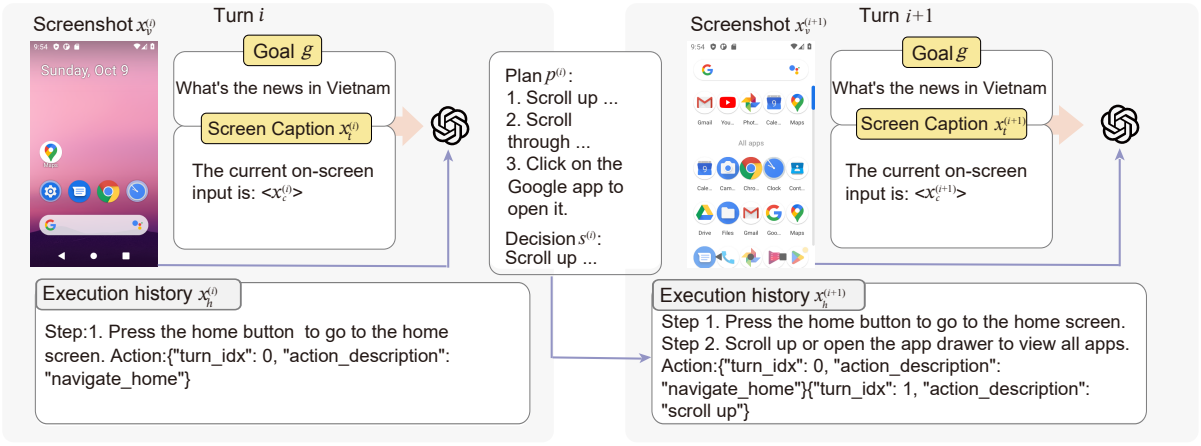

Figure 1: Overview of DP-Agent. In turn, i, the DP-Agent makes a plan based on visual input and textual input, predicts the action to be performed, and then updates the execution history, and then proceeds to the next turn $i+1$.

problems (Wei et al., 2023; Kojima et al., 2023; Zhang et al., 2022). Notably, the CoT prompting technique has enabled LLMs to engage in effective step-by-step problem-solving process (Huang and Chang, 2023; Yao et al., 2023a; Wang et al., 2023b; Chen et al., 2023). To address more complex problems, divide-and-conquer prompting strategies have been proposed, e.g., dividing problems into manageable steps (Zhou et al., 2023; Lee and Kim, 2023) or sequential solutions (Wang et al., 2023a).

The research above mainly focuses on enhancing reasoning abilities. However, the ReAct (Yao et al., 2023b) prompting has inspired researchers to explore more suitable ways for LLMs to complete tasks by leveraging their reasoning abilities. This approach involves LLMs first observing and reasoning before taking action, such as utilizing external tools to identify and rectify errors (Gou et al., 2023; Shinn et al., 2023), or planning before executing (Wang et al., 2023a; Hao et al., 2023).

Inspired by ReAct (Yao et al., 2023b), we design a novel DP-Agent prompting framework called DP-Agent to achieve dynamic plan adjustment during the interaction. In contrast to ReAct, DP-Agent involves actions performed within a simulated environment rather than directly interacting with the physical smartphone environment. Nevertheless, DP-Agent follows a similar logical approach as ReAct by first leveraging the reasoning capabilities of LLMs to plan before executing actions.

3 Method

This section describes our DP-Agent approach, which is grounded in dynamic planning based on environment feedback and execution history.

On a high level, DP-Agent comprises two stages: (i) planning initialization: the agent initiates the planning process by generating an overall plan, considering the ultimate goal, current visual input, and prior execution history. Once the plan is formulated, the agent will select the most plausible step for execution. (ii) dynamic planning adjustment: the executed action is appended to the execution history. This updated history then shapes subsequent planning cycles. In doing so, the agent is equipped with the latest contextual information, thereby enhancing decision-making efficacy in subsequent turns. The framework of DP-Agent is shown in Figure 1.

3.1 Planning Initialization

In pursuit of the task goal g, the agent engages in k turns of interactions until task completion. Specifically, at each turn i ($i = 1, \ldots, k$), the agent f processes the visual input $x_v^{(i)}$ (i.e., the current screenshot) and the textual input $x_t^{(i)}$. It then generates the plan p_i and identifies the optimal step $s^i \in p^i$ to execute:

$$(p^{(i)}, s^{(i)}) = f(x_v^{(i)}, x_t^{(i)}), \quad (1)$$

where the textual input $x_t^{(i)}$ consists of the task goal g, screen caption $x_c^{(i)}$, and execution history $x_h^{(i)}$.

The textual input is further wrapped with prompts (Appendix A.1) before feeding the agent along with the visual input. Concretely, we articulate our task goal at the text's outset by prompting "*Your ultimate goal is:* <g>".

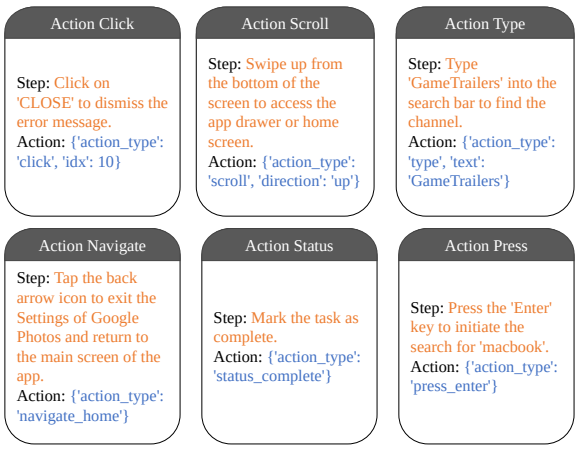

Figure 2: Examples of six types of available actions.

Subsequently, we append the screen caption results under the heading "*The current on-screen input is:* <$x_c^{(i)}$>". Then, we include execution history, structured as "*Here are previous actions:* <$x_h^{(i)}$>".

After feeding the inputs, we request the agent to generate a plan $p^{(i)} = [p_1^{(i)}, p_2^{(i)}, \ldots]$, which consists of a sequence of steps to achieve the ultimate goal. Within those steps, the agent is also required to identify the optimal step $s^{(i)} \in p^{(i)}$.

Action Type	Action Description
Click	Idx
Scroll	Direction (up, down, left and right)
Type	Text
Navigate	Home / Back
Status	Complete
Press	Enter

Table 1: Six types of available actions.

In practice, $s^{(i)}$ is confined to a finite set of available actions in the GUI automation task and will be transformed into the JSON format for execution. Following Rawles et al. (2023), we utilize six distinct types of actions as presented in Table 1. Examples are provided in Figure 2.

3.2 Dynamic Planning Adjustment

After the execution of $s^{(i)}$, the agent becomes anchored in the subsequent interaction turn with an updated visual input $x_v^{(i+1)}$ (e.g., a new screenshot). Simultaneously, we refine the execution history $x_h^{(i+1)}$ by concatenating $x_h^{(i)}$ and $s^{(i)}$:

$$x_h^{(i+1)} = \textsc{Concat}(x_h^{(i)}, s^{(i)}), \qquad (2)$$

where CONCAT denotes the concatenation operation between strings.

Consequently, the execution history is organized with consecutive elements in the format of "*step <turn id>: <action>*". This updated execution history $x_h^{(i+1)}$ is subsequently employed according to the planning initialization process outlined in Section 3.1 for turn $(i+1)$ until the task reaches completion. The task is considered complete when $i = k$ or the agent predicts the "Status" action type with the "Complete" action description.

4 Experiments

In this section, we first describe the dataset, metrics, and baseline settings for our experiments. Then, we present our main results, followed by analysis.

4.1 Dataset

We employ the AITW (Rawles et al., 2023) benchmark dataset for our evaluation on Android devices. AITW is a comprehensive benchmark dataset specifically designed for GUI control, encompassing natural language instructions, corresponding screenshots, and associated actions. The dataset spans tasks across five distinct categories, including Internet search, downloading, and online shopping, and involves interactions with over 350 different applications and websites. In its entirety, the dataset comprises 715,000 episodes, featuring a diverse range of 30,000 unique instructions.

We leverage the provided screen caption results as part of the textual input. Concretely, given a screen, GUI icons were detected using the OCR tool and IconNet (Sunkara et al., 2022). Each GUI icon is associated with a bounding box and OCR-detected text. To align predicted gestures with specific GUI elements, we filter the valid data by selecting instances whose gesture coordinates could fall within the corresponding GUI box. Numeric text tags are added to these GUI boxes for analysis purposes.

Dataset	Episodes	Screens	Instructions
General	9,476	85,413	545
Install	25,760	250,058	688
GoogleApps	625,542	4,903,601	306
Single	26,303	85,668	15,366
WebShopping	28,061	365,253	13,473

Table 2: Dataset statistics.

Table 2 presents the data statistics. Subsequently, each filtered subset is partitioned episode-wise into training, validation, and test sets following an 80/10/10 split. Additionally, considering the

Model	Overall	General	GoogleApps	Install	Single	WebShopping
Fine-tuned Llama 2 (Zhang and Zhang, 2023)	28.40	28.56	30.99	35.18	27.35	19.92
PaLM-2 ZS (Rawles et al., 2023)	30.9	-	-	-	-	-
ChatGPT 5-shot (Zhang and Zhang, 2023)	7.72	5.93	10.47	4.38	9.39	8.42
GPT-4V ZS	35.58	30.40	38.80	**42.67**	35.46	30.56
DP-Agent	**44.39**	**39.08**	**50.63**	41.09	**55.99**	**35.15**

Table 3: Main results (%). Segment 1: fine-tuned Llama 2 baseline; Segment 2: in-context learning LLM baselines, "ZS" stands for "zero-shot" and "5-shot" stands for using 5-shot in-content learning (Section 4.3); Segment 3: GPT-4V as agent model, "DP-Agent" represents our proposed method. The best result is reported in boldface.

constraints of GPT-4V, we limit our data selection to episodes with a length of less than 15. To get more convincing results, we sampled three completely different sets of data samples from the test set, each with 300 episodes sampled from different subsets for analysis and experimentation.

4.2 Metrics

In line with prior research (Zhang and Zhang, 2023; Yan et al., 2023), our primary evaluation metric is the screen-wise action matching score, computed as the ratio of correct actions to the episode length. Specifically, for click actions, correctness is determined if the selected element is within a 14% screen distance from the gold gestures or falls within the same detected bounding box as the user's gestures. Given that the agent responds with the numeric tag of the GUI, we select the top left, top right, bottom left, bottom right, and center of the box as sample points for calculating coordinate distances.

Regarding scroll actions, correctness is assessed if the selected direction aligns with the scroll direction (up, down, left, or right) of the user's gestures. For other actions, correctness is established only if the types of actions match.

4.3 Baseline

We compare our prompting framework with the following baselines (Rawles et al., 2023; Zhang and Zhang, 2023).

• PaLM-2 ZS (Rawles et al., 2023): This setting evaluates the zero-shot performance of PaLM-2 by providing a textual description of the screen and prompting it to predict an action from the supported actions in AITW.

• ChatGPT 5-shot (Zhang and Zhang, 2023): ChatGPT's performance is assessed with a 5-shot prompt format similar to PaLM-2. The experiments are conducted using the ChatGPT API.

• Fine-tuned LlaMa-2 (Zhang and Zhang, 2023): The Llama-2 model is fine-tuned with LoRA, utilizing user instructions and screen descriptions in HTML syntax, which aligns with the format used for in-context learning in LLMs. The model is fine-tuned using 1% randomly sampled training data to facilitate adaptation to the task.

• GPT-4V ZS: Zero-shot prompting with GPT-4V. The model is presented with a screenshot image and a textual description of the screen, tasked with predicting an action from the available actions.

4.4 Implementation Details

We use the GPT-4V (Achiam et al., 2023) interface provided by OpenAI as the backbone of our agent. We set the "max_tokens" as 300 and the "temperature" as 0. The model has a training epoch of 3 and a maximum length of 2560.

We also fine-tune public large models, i.e., Llama (Touvron et al., 2023) and LLaVa (Liu et al., 2023), to verify the general effectiveness of our approach. For the finetuning experimental setup, we use Llama2-7B and LLaVa-7B as our base model, training epochs to be 3, without eval set between epochs. The maximum length of the input sequence is 2560, ensuring that the text can be fully entered. Text input generally includes the goal, screen descriptions in HTML syntax, and execution history. For inputs with a "Plan" experimental group, the step to be selected is spliced at the end of the input, similar to the prompt of GPT-4V requiring the action to be performed.

4.5 Main Results

Table 3 presents the main results.[1] Based on the results, we have the following key findings.

(i) **DP-Agent achieves substantial performance gains over the GPT-4V baseline.** DP-

[1]We run experiments with three random seeds and report the average scores. Details of the three runs are presented in Appendix A.2.

Category	w/ GPT-4V	w/ Human
Total Accuracy	23.57	**52.23**
Click Accuracy	17.83	**27.39**
Scroll Accuracy	0.00	**1.27**
Type Accuracy	2.55	**9.55**
Complete Accuracy	2.55	**7.64**

Table 4: Comparison of GPT-4V generated planning and human-annotated planning in the Install dataset (%). The best average result is reported in boldface.

Agent demonstrates overall improvement of +8.81% (35.58% → 44.39%) compared to the widely adopted GPT-4V baseline. The results show improving the planning mechanism is effective in boosting the GUI agent performance.

(ii) **Perception with the visual modality benefits the task performance.** We observe that even the GPT-4V ZS Baseline exhibits improvements over plain text input LLMs, with an increase from 7.72%→35.58% compared to ChatGPT 5-shot and from 30.9%→35.58% compared to PaLM-2 ZS.

(iii) **Dynamic planning achieves substantially better performance than static planning.** We notice that the impact of introducing planning varies across different tasks. In General, GoogleApps, Single, and Webshopping dataset tasks, the improvement caused by planning is stable and evident. In the Install dataset, there is little improvement in dynamic planning performance. Upon scrutinizing the outcomes produced by the agent, we conclude that planning the Install tasks for GPT-4V remains a formidable challenge. To validate this observation, we choose 20 episodes from the Install dataset and meticulously label them with corresponding plans. Even with a plan in place, the subsequent step involved selecting and executing actions. Examining the results presented in Table 4, we are pleased to note a notable improvement in the overall correctness rate, which rises from 23.57%→52.23%.

4.5.1 Contribution of Dynamic Planning

To investigate the role of dynamic planning in decision-making by GPT-4V, we analyze the correct rate of different actions and the proportion of predicted actions. We combine the results of five datasets in Figure 3. More details are provided in Appendix A.3. Our observations based on these statistics reveal two notable changes:

(i) **Increased Component of Predicted Actions.** In comparison with the GPT-4V Baseline, DP-

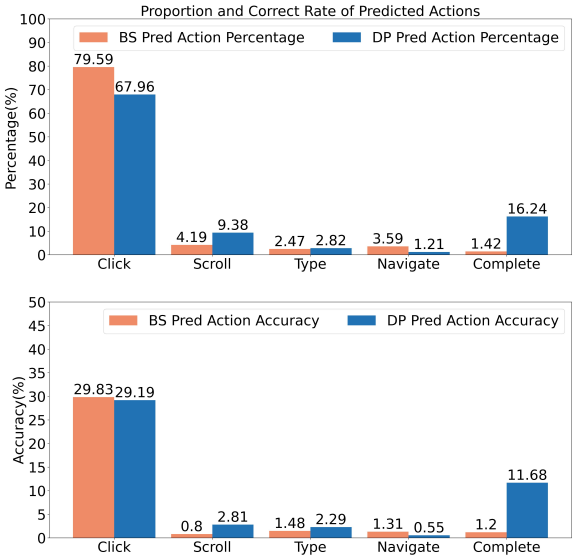

Figure 3: The proportion and correct rate of predicted actions of GPT-4V and DP-Agent. We mainly collected the proportions of "Click", "Scroll", "Type', "Navigate Home" and "Complete" actions. "BS" stands for Baseline, "DP" stands for DP-Agent.

Agent exhibits a significant increase in the proportion of predicted actions for all actions, except for the "Click" action (we will provide a detailed explanation in the subsequent paragraph). Notably, the most prominent increase is observed in the proportion of "Complete" actions. This is encouraging as it indicates that the agent now has a clearer understanding of task completion, which is beneficial for practical applications. Additionally, the rise in the occurrences of other actions, such as "Scroll" and "Type", enhances the agent's ability to tackle more complex tasks.

(ii) **Reduction of Invalid Click Action.** While the accuracy of the click action shows a slight decline, it is essential to note that the total number of click actions has significantly reduced. Consequently, the proportion of correct click predictions within the overall prediction results has increased. Existing work indicates that GPT-4V is more likely to execute the "Click" action (Yan et al., 2023). However, our method, DP-Agent, minimizes invalid and erroneous click actions, showcasing a better comprehension of the implementation progress of the current plan.

4.6 Ablation Study

To study the influencing factors for the planning mechanism in DP-Agent, we randomly sampled 20 episodes from 5 data subsets, with a total of 100

Model	Planning	Updating History	Overall	General	GoogleApps	Install	Single	WebShopping
BS	✗	✗	32.45	28.03	40.32	33.79	26.98	33.13
OP	✓	✗	42.01	31.06	50.81	**40.69**	**57.14**	30.72
PU	✓	✓	**44.50**	**45.45**	**52.42**	37.93	52.38	**34.34**

Table 5: The ablation studies on factors on planning. Planning: Necessitates GPT-4V for the doing plan; Updating History: Involves adding executed steps to the execution history. Each experiment's execution or omission of a particular process is denoted by ✓ (if performed) or ✗ (if not performed). The best average result is in boldface.

episodes as the dataset for the ablation experiment for reducing costs.

We use the following Baseline:

• Baseline (BS): denotes the Baseline, which abstains from adopting a plan.

• Only Planning (OP): It represents the exclusive utilization of planning, excluding execution history updates.

• Planning and Updating History (PU): it specifies the simultaneous execution of planning and updating the execution history.

4.6.1 Impact of Varied Planning Mechanisms

Table 5 presents the results of the ablation study. Based on the results of three experimental sets, we have the following key findings.

(i) **Significant Improvement Through Plan Integration.** Leveraging the plan alone results in a noteworthy enhancement. A comparison between BS, OP, and PU reveals that the inclusion of planning brings about substantial improvements.

(ii) **Enhanced Execution through Step Selection.** The act of selecting steps from the plan contributes to improved execution. A comparison between OP and PU indicates that merely creating a plan for the agent is insufficient. The selection of a specific step for execution within the formulated plan leads to more pronounced improvements.

(iii) **Varied Impact of Historical Information Update.** Intriguingly, when comparing OP with PU, we observe that PU outperforms some certain datasets, and OP excels in others. Despite this, the overall performance of PU remains superior. We speculate that the decline in performance observed in the Single datasets primarily stems from the inherently simpler nature of the tasks, which necessitates less reliance on planning abilities. However, for the Install dataset, we find that the tasks are rather difficult. As a result, the planning process of PU is more likely to be misled by the defective execution history.

Model	Click	Scroll	Type	Navigate Home	Complete
BS	26.67	1.27	2.38	1.11	1.43
OP	25.08	1.11	2.54	**2.06**	8.57
PU	**26.83**	**2.70**	**3.02**	0.79	**9.52**

Table 6: Correct rate of predicted actions in three ablation experiments (%). The best result is in boldface.

4.6.2 Exploring the Proportion and Correct Rate of Predicted Actions

To conduct a more detailed analysis to scrutinize the influencing factors in planning, we dive into the correct rate and the proportion of predicted actions. Given the abundance of datasets in a single experiment, we opt to amalgamate five datasets for an overarching analysis, more details of the correct rate of predicted actions are in Appendix A.4.

Specifically, actions are categorized into state actions and manipulation actions. State actions encompass the "Navigate Home" and "Complete" actions, while manipulation actions consist of the "Click", "Scroll", and "Type" actions.

The rationale behind this classification is as follows: when the agent executes the "Navigate Home" action, it signals unfamiliarity with the current page, suggesting that the current page is deemed unhelpful for task completion. Typically occurring in the initial steps of a task, a high frequency may indicate the agent losing control during task execution. On the other hand, the "Complete" action signifies the agent's determination that the task has been accomplished. All in all, a decrease in the frequency of "Navigate Home" operations coupled with an increase in "Complete" operations indicates an enhanced grasp of the current task status. It's important to note that the discussions on these two actions presume correct prediction results. Finally, all actions, excluding state actions, are categorized as manipulation actions.

Table 6 presents the overall correctness rate of predicted actions for the three sets of ablation exper-

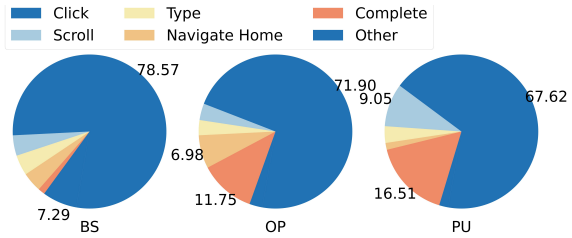

Figure 4: The proportion of predicted actions about planning in total datasets. We mainly collected the correct rate of "Click", "Scroll", "Type', "Navigate" and "Complete" actions.

Model	General	Install	Single	WebShopping
Llama2-7B				
w/ all data	28.56	35.18	27.35	19.92
Llama2-7B				
w/ Plan				
by GPT-4V	**24.67**	**23.46**	**39.48**	**19.48**
by itself	17.81	17.58	15.87	12.46
w/o Plan	13.08	17.12	3.87	8.71
LLaVa-7B				
w/ Plan				
by GPT-4V	27.19	26.77	44.46	20.61
by itself	**30.73**	**29.39**	**45.94**	**21.67**
w/o Plan	17.81	17.98	1.66	10.91

Table 7: Finetuning results of Llama2-7B and LLaVa-7B. Segment 1: "w/ all data" stands for the model is fine-tuned with 1% randomly sampled training data to help adapt to this task. Segments 2 & 3: The training set is 180 episodes in the GoogleApps, and the test set is 180 episodes in other datasets. "by GPT-4V" stands for planning is made by GPT-4V. "itself" stands for planning is made by finetuned model itself. The best average result is in boldface.

iments, while Figure 4 illustrates the distribution of these experiments in predicted actions. We could find that the proportion of "Complete" action in OP and PU is significantly higher than that in BS. This observation underscores that **dynamic planning positively influences the agent's ability to identify task completion.**

Regarding the "Navigate Home" action, OP, focusing solely on planning, exhibits a substantial increase. This issue can be effectively mitigated when a step is chosen for execution within the plan, as evident in both PU. This indicates that selecting a step in the plan enhances the agent's understanding of the current screen state. The analysis of state actions highlights that **incorporating planning and execution steps aid the agent in comprehending the execution process status of the ongoing task.**

4.7 Adaptation to Unfamiliar Tasks

As new applications continually emerge, their interfaces often pose unfamiliarity to agents. Despite the diversity of GUI tasks, there exists a semblance of similarity in screen navigation logic. Even when the interface is unknown, certain screen transition patterns remain consistent. Consequently, employing dynamic planning could be beneficial for adaptation to unfamiliar tasks. To explore whether planning is conducive to agent adaptation to unfamiliar tasks, we fine-tune Llama2-7B and LLaVa-7B. To save computation costs, we randomly sample 180 episodes from GoogleApps as the training set and 180 episodes from other datasets as the test set.

The results are shown in Table 7. Surprisingly, LLMs fine-tuned on the GoogleApps datasets with a plan demonstrated commendable performance across various datasets, sometimes even outperforming models fine-tuned on the entire dataset (Llama2-7B on the Single dataset). Notably, the fine-tuned model exhibited superior planning performance for the LLaVa-7B when compared to the GPT-4V. This finding underscores the significant potential for enhancing the current planning approach, especially given the constrained scale of available fine-tuning data. It also emphasizes the effectiveness of incorporating planning into the model, particularly in its adaptability to unfamiliar tasks. The consequential impact of this advantage is notably significant in the practical realm of smartphone control. Given the perpetual evolution of GUI contents, the planning ability equips the agent with enhanced capabilities to navigate and respond to this dynamic challenge effectively.

5 Conclusion

This study introduces a prompting approach called DP-Agent, designed to facilitate interactions in a multimodal environment. DP-Agent encourages agents to dynamically update planning based on feedback from the environment and execution history. Through the application of DP-Agent, we demonstrate that the DP-Agent surpasses the widely adopted GPT-4V baseline on the AITW benchmark dataset. Meanwhile, our findings indicate that the DP-Agent excels in adapting to unfamiliar tasks, and can choose different actions more correctly.

6 Limitation

The main limitation of this work is that we have focused solely on analyzing the influencing factors in the planning phase. Future research will explore strategies for optimizing the utilization of these factors to enhance the precision of the planning process within an episode.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. 2023. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2023. Critic: Large language models can self-correct with tool-interactive critiquing.

Tanmay Gupta and Aniruddha Kembhavi. 2022. Visual programming: Compositional visual reasoning without training.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model.

James A. Hendler. 1999. Is there an intelligent agent in your future? nature.

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxuan Zhang, Juanzi Li, Bin Xu, Yuxiao Dong, Ming Ding, and Jie Tang. 2023. Cogagent: A visual language model for gui agents.

Jie Huang and Kevin Chen-Chuan Chang. 2023. Towards reasoning in large language models: A survey.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2023. Large language models are zero-shot reasoners.

Soochan Lee and Gunhee Kim. 2023. Recursion of thought: A divide-and-conquer approach to multi-context reasoning with language models.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning.

Pattie Maes. 1994. Agents that reduce work and information overload. *Communications of the ACM*, 37(7):30–40.

Pattie Maes. 1998. *Agents that reduce work and information overload*, page 525–536. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy P Lillicrap. 2023. Androidinthewild: A large-scale dataset for android device control. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

John R. Searle. 1969. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press.

JohnR. Searle. 1972. Speech acts: An essay in the philosophy of language. *Mind,Mind*.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2023. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning.

Srinivas Sunkara, Maria Wang, Lijuan Liu, Gilles Baechler, Yu-Chung Hsiao, Jindong, Chen, Abhanshu Sharma, and James Stout. 2022. Towards better semantic understanding of mobile interfaces.

Dídac Surís, Sachit Menon, and Carl Vondrick. 2023. Vipergpt: Visual inference via python execution for reasoning.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu,

Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Junyang Wang, Haiyang Xu, Jiabo Ye, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. 2024a. Mobile-agent: Autonomous multi-modal mobile device agent with visual perception.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023a. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023b. Self-consistency improves chain of thought reasoning in language models.

Zhenyu Wang, Enze Xie, Aoxue Li, Zhongdao Wang, Xihui Liu, and Zhenguo Li. 2024b. Divide and conquer: Language models can plan and self-correct for compositional text-to-image generation.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models.

Hao Wen, Yuanchun Li, Guohong Liu, Shanhui Zhao, Tao Yu, Toby Jia-Jun Li, Shiqi Jiang, Yunhao Liu, Yaqin Zhang, and Yunxin Liu. 2023. Empowering llm to use smartphone for intelligent task automation.

Michael Wooldridge and Nicholas R. Jennings. 1995. Intelligent agents: Theory and practice. *The Knowledge Engineering Review*, page 115–152.

Chenfei Wu, Shengming Yin, Weizhen Qi, Xiaodong Wang, Zecheng Tang, and Nan Duan. 2023. Visual chatgpt: Talking, drawing and editing with visual foundation models.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, Rui Zheng, Xiaoran Fan, Xiao Wang, Limao Xiong, Yuhao Zhou, Weiran Wang, Changhao Jiang, Yicheng Zou, Xiangyang Liu, Zhangyue Yin, Shihan Dou, Rongxiang Weng, Wensen Cheng, Qi Zhang, Wenjuan Qin, Yongyan Zheng, Xipeng Qiu, Xuanjing Huang, and Tao Gui. 2023. The rise and potential of large language model based agents: A survey.

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. 2023. Gpt-4v in wonderland: Large multimodal models for zero-shot smartphone gui navigation.

Hui Yang, Sifu Yue, and Yunzhong He. 2023a. Auto-gpt for online decision making: Benchmarks and additional opinions.

Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. 2023b. The dawn of lmms: Preliminary explorations with gpt-4v(ision).

Zongxin Yang, Guikun Chen, Xiaodi Li, Wenguan Wang, and Yi Yang. 2024. Doraemongpt: Toward understanding dynamic scenes with large language models.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023a. Tree of thoughts: Deliberate problem solving with large language models.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023b. React: Synergizing reasoning and acting in language models.

Chi Zhang, Zhao Yang, Jiaxuan Liu, Yucheng Han, Xin Chen, Zebiao Huang, Bin Fu, and Gang Yu. 2023. Appagent: Multimodal agents as smartphone users.

Zhuosheng Zhang and Aston Zhang. 2023. You only look at screens: Multimodal chain-of-action agents.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022. Automatic chain of thought prompting in large language models.

Zirui Zhao, Wee Sun Lee, and David Hsu. 2023. Large language models as commonsense knowledge for large-scale task planning.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. 2023. Least-to-most prompting enables complex reasoning in large language models.

A Example Appendix

A.1 Dynamic planning prompting

We use the following prompt for Planning Initialization.

```
Imagine that you are a robot operating a mobile. Like how humans operate the mobile, you can click on the screen, type some text, go home, go back to the last screen, scroll up, down, left and right, or mark the status as complete. Given a goal and a mobiel screen, you need to make a plan to achieve your goals based on the current screen, and choose the steps that should be achieved on the current screen from the plan you have made. Since achieving this goal is a **continuous process**, you will be given the **previous steps and actions** that have been performed, so please pay attention to this information. There may be multiple ways to achieve your goals, but what you need to do is create the plan that best suits your current situation based on the current screen input.

**Your ultimate goal is: check out phone information.**
The current on-screen input is:
Screen:
<p id=0 class=‘‘text’’ alt=‘‘vvaiipaper,’’>vvaiipaper,</p>
<p id=1 class=‘‘text’’ alt=‘‘sieep,’’>sieep,</p>
<p id=2 class=‘‘text’’ alt=‘‘iolL’’>iolL</p>
<p id=3 class=‘‘text’’ alt=‘‘SIZE’’>SIZE</p>
<p id=4 class=‘‘text’’ alt=‘‘Sound’’>Sound</p>
<img id=5 class=ICON\_VOLUME\_STATE alt=‘‘’’></p>\n <p id=6 class=‘‘text’’ alt=‘‘Volume,’’>Volume,</p>
<p id=7 class=‘‘text’’ alt=‘‘vibration,’’>vibration,</p>
<p id=8 class=‘‘text’’ alt=‘‘Do’’>Do</p>
<p id=9 class=‘‘text’’ alt=‘‘Not’’>Not</p>
<p id=10 class=‘‘text’’ alt=‘‘Disturb’’>Disturb</p>\newline <p id=11 class=‘‘text’’ alt=‘‘Storage’’>Storage</p>
<p id=12 class=‘‘text’’ alt=‘‘used’’>used</p>
<p id=13 class=‘‘text’’ alt=‘‘GB free’’>GB free</p>
<p id=14 class=‘‘text’’ alt=‘‘49\%’’>49\%</p>
<p id=15 class=‘‘text’’ alt=‘‘-32.63’’>-32.63</p>
<p id=16 class=‘‘text’’ alt=‘‘Privacy’’>Privacy</p>
<p id=17 class=‘‘text’’ alt=‘‘Permissions,’’>Permissions,</p>
<p id=18 class=‘‘text’’ alt=‘‘account’’>account</p>
<p id=19 class=‘‘text’’ alt=‘‘personal’’>personal</p>
<p id=20 class=‘‘text’’ alt=‘‘data’’>data</p>
<p id=21 class=‘‘text’’ alt=‘‘activity,’’>activity,</p>
<p id=22 class=‘‘text’’ alt=‘‘Location’’>Location</p>
<img id=23 class=ICON\_LOCATION alt=‘‘’’></p>
<p id=24 class=‘‘text’’ alt=‘‘On’’>On</p>
<p id=25 class=‘‘text’’ alt=‘‘have access’’>have access</p>
<p id=26 class=‘‘text’’ alt=‘‘- 4 apps’’>- 4 apps</p>
<p id=27 class=‘‘text’’ alt=‘‘location’’>location</p>
<p id=28 class=‘‘text’’ alt=‘‘to’’>to</p>
<p id=29 class=‘‘text’’ alt=‘‘Security’’>Security</p>
<p id=30 class=‘‘text’’ alt=‘‘lock, fingerprint’’>lock, fingerprint</p>
<p id=31 class=‘‘text’’ alt=‘‘Screen’’>Screen</p>
Here are previous actions: (format: action \u2192 action description)
Previous Actions:
{’’step\_idx’’: 0, ’’action\_description’’: ’’scroll up’’}
{’’step\_idx’’: 1, ’’action\_description’’: ’’click []’’}
{’’step\_idx’’: 2, ’’action\_description’’: ’’scroll up’’}
And the previous steps:
Previous Steps:
Step 1. Swipe up from the bottom of the screen to access the app drawer.
Step 2. Tap on the ’Settings’ icon to open the settings menu.
Step 3. Scroll up to reveal more settings options.

Please formulate an operational guide for future operations for solving the goal. The guide includes:
1. Plan: A **multi-step future** plan **(start from current screen, DON’T include previous steps)**; steps indexed by numbers.
2. Step: Based on the current screen and Previous Steps, provide the **immediate** step that needs to be taken from the Plan.
"**Output Format:** A JSON dictionary strictly following the format: "{’plan’: ’...<Your Plan Here>’, ’step’: ’...<Your Step Here>’} "If the goal has already been implemented, no more planning is required, Provide {’plan’: ’1. Mark the task as complete’, ’step’: ’Mark the task as complet’}.
**Please do not output any content other than the JSON format.**
```

A.2 Three sets of experiments for GPT-4V as core of agent

The detailed results of three distinct sets of experiments featuring GPT-4V as the core agent are presented in Table 8.

Model	Overall	General	GoogleApps	Install	Single	WebShopping
			Test 1			
GPT-4V ZS	34.66	29.69	35.75	43.50	32.95	31.42
DP-Agent	**46.47**	40.10	49.74	**47.18**	**58.96**	36.34
			Test 2			
GPT-4V ZS	36.72	32.34	41.95	43.87	37.84	27.58
DP-Agent	43.62	**40.83**	**54.35**	40.12	50.81	32.00
			Test 3			
GPT-4V ZS	35.36	29.18	38.70	40.65	35.59	32.67
DP-Agent	43.07	36.30	47.79	35.98	58.19	**37.11**

Table 8: Three sets of experiments for GPT-4V as core of agent(%). we sampled three completely different sets of data samples from the test set, each with 300 episodes sampled from different subsets. The best average result is in boldface.

A.3 The proportion and correct rate of predicted actions in all datasets

The proportion of predicted actions is depicted in Table 9, while the correct rate of predicted actions is depicted in Table 10.

Action	General	GoogleApps	Install	Single	Webshopping
Click	**77.6** / 70.57	**72.02** / 55.44	**83.3** / 73.53	**82.08** / 56.07	82.96 / **84.19**
Scroll	1.56 / **5.21**	11.14 / **22.8**	2.52 / **9.8**	3.47 / **5.78**	2.26 / **3.29**
Type	2.6 / 2.6	0.26 / 0.26	3.88 / **3.92**	2.31 / **4.05**	3.29 / 3.29
Navigate Home	**3.39** / 1.3	**3.37** / 1.55	**5.05** / 2.16	0.00 / 0.00	**6.16** / 1.03
Complete	1.56 / **17.45**	1.04 / **17.62**	1.55 / **8.82**	2.31 / **31.79**	0.62 / **5.54**

Table 9: The proportion of predicted actions of GPT-4V and DP-Agent in main results. We mainly collected the proportions of "Click", "Scroll", "Type', "Navigate Home" and "Complete" actions. Actions that are not collected are represented by others. The number on the left of "/" is Baseline, and the number on the right of "/" is DP-Agent. The best average result is in boldface.

Action	General	GoogleApps	Install	Single	Webshopping
Click	**24.74** / 24.48	**31.09** / 29.27	**35.92** / 33.53	**29.48** / 28.90	27.93 / **29.77**
Scroll	0.52 / **2.08**	2.33 / **8.81**	1.17 / **3.14**	0.00 / 0.00	0.00 / 0.00
Type	1.04 / **2.34**	0.26 / 0.26	2.72 / **3.33**	1.73 / **3.47**	1.64 / **2.05**
Navigate Home	**2.08** / 0.52	**1.3** / 1.04	**2.14** / 0.98	0.00 / 0.00	**1.03** / 0.21
Complete	1.3 / **10.68**	0.78 / **10.36**	1.55 / **6.47**	1.73 / **26.59**	0.62 / **4.31**

Table 10: The correct rate of predicted actions of GPT-4V and DP-Agent in main results. We mainly collected the proportions of "Click", "Scroll", "Type', "Navigate Home" and "Complete" actions. Actions that are not collected are represented by others. The number on the left of "/" is Baseline, and the number on the right of "/" is DP-Agent.

A.4 The correct rate of predicted actions in ablation studies

We provide the predicted action accuracy for all datasets of ablation experiments in Table 11.

Model	Action	General	GoogleApps	Install	Single	Webshopping
BS	Click	23.48	33.06	**25.52**	23.81	**26.51**
	Scroll	0.67	2.42	2.07	0.00	0.60
	Type	3.03	0.00	2.76	0.00	**4.22**
	Navigate Home	0.00	1.61	2.07	0.00	1.20
	Complete	0.76	3.23	1.38	1.59	0.60
OP	Click	16.67	**36.29**	24.14	**30.16**	22.29
	Scroll	0.76	1.61	2.07	0.00	0.60
	Type	2.27	**0.81**	**4.14**	**1.59**	3.01
	Navigate Home	**4.52**	**2.42**	**3.45**	0.00	**1.81**
	Complete	9.85	9.68	**6.90**	22.22	**3.01**
PU	Click	**27.27**	35.48	21.38	23.81	25.90
	Scroll	**3.03**	**3.23**	**5.52**	0.00	0.60
	Type	**3.79**	**0.81**	3.45	**1.59**	**4.22**
	Navigate Home	0.00	1.61	1.38	0.00	0.60
	Complete	**11.36**	**11.29**	6.21	**26.98**	**3.01**

Table 11: The correct rate of predicted actions of GPT-4V and DP-Agent in ablation studies. We mainly collected the correct rate of "Click", "Scroll", "Type', "Navigate" and "Complete" actions. The best average result is in boldface.