AndroidLens: Long-latency Evaluation with Nested Sub-targets for Android GUI Agents

Supplementary Material

The appendix provides additional details and supplementary information to further elaborate on the sections above. Sec. A presents the action space defined by our benchmark. Sec. B includes all the everyday scenarios involved as well as the selected representative apps. Sec. C provides a taxonomy of cross-app tasks along with examples. Sec. D analyzes the statistical characteristics of the dataset. Sec. E describes the prompt used for milestone determination during dynamic evaluation. Sec. F details the evaluation experiments, including benchmark evaluation specifics and the Agent's evaluation settings. Sec. G provides examples of tasks from different categories and also showcases some sample model outputs. Sec. H specifically provides examples of various agent responses to real-world emergencies.

A. Action Set

Our trajectory annotation primarily uses the Android Debug Bridge (ADB) to connect to physical devices for GUI navigation. During this process, we capture device states and detailed information about operations, such as the coordinates of click events and the triggering of function keys. The details of our action set are shown in Tab. 4. The "Swipe" action specifies explicit start and end points.

B. All Scenarios and Apps

To best reflect the agent's effectiveness on long-horizon tasks in everyday life, we identified 38 scenarios through discussion, encompassing 62 commercial apps and 12 system apps in both Chinese and English settings. Details are shown in Tab. 5, which also reports the number of long-horizon tasks associated with each app.

C. Cross APP Tasks

As the functionalities of various apps grow increasingly complex, cross-app interaction patterns have also become more diverse. To more comprehensively evaluate an agent's performance in scenarios involving frequent switches between different app interfaces, we incorporate a variety of typical cross-app interaction types into our benchmark, including sharing, clipboard operations, application redirection, authorization, tool call, comparison, information aggregation, and multi-round cross-app interactions. For each interaction type, we provide corresponding examples in the benchmark, as shown in Tab. 6.

- Sharing: After completing an operation in one app, pass the result to another app via the app's built-in sharing feature or by sharing it manually.
- Clipboard Operation: Transfer text or other content between different apps using copy, paste, and similar actions.
- Application Redirection: When obtaining information or tapping an entry point in one app, the system automatically redirects and opens another app.
- Authorization: Use an existing account in one app to complete third-party authorization login or account binding in another app.
- Tool Call: (1) Modify the features or status of other apps through system settings. (2) Invoke built-in phone tools or system apps (such as camera, contacts, etc.) within an app to complete related operations.
- Comparison: Collect similar types of information from multiple apps, compare them, and then make a choice and perform follow-up actions based on the comparison results.
- Information Aggregation: Collect and organize information on one or more platforms, then consolidate it into another app for recording, archiving, or unified management.
- Multi-Round Interactions: Need to switch back and forth between multiple apps multiple times, completing the task step by step, where subsequent steps depend on the results of previous steps.

D. Data Statistics

Overall, AndroidLens comprises 399 Chinese tasks and 172 English tasks, including 273 single-application tasks and 298 cross-application tasks. As described in Sec. 3.3, we divide all tasks into three major categories: Multi-constraint, Multi-goal, and Domain-specific. For each major category, we further define several subcategories, and the distribution of the number of tasks in each subcategory is shown in Fig. 7. Meanwhile, we assign a different number of milestones to each task to characterize its execution difficulty, and their distribution within each subcategory is shown in Fig. 8.

E. Evaluation Prompts

E.1. General Model Evaluation Prompt

To showcase each model's capabilities on GUI tasks as fully as possible, we directly used the official configuration to guide models that provide an official GUI navigation sys-

Table 4. **Definition of Action Space.**

Action	Definition
Click(x, y)	Click the point on the screen with coordinate (x, y).
Swipe(x1, y1, x2, y2)	Swipe from the starting point with coordinate (x1, y1) to the end point with coordinates2 (x2, y2).
LongPress(x, y)	Press the point on the screen with coordinate (x, y) for specified seconds.
Type(text)	Input the specified text into the activated input box.
PressBack()	Press the system button $Back$.
PressHome()	Press the system button $Home$.
PressMenu()	Press the system button $Menu$.
Terminate('success' or 'failure')	Terminate the current task and report its completion status, e.g., $success$, $failure$.
Wait()	Wait specified seconds for the change to happen.

Figure 7. **Distribution of different categories in AndroidLens.**

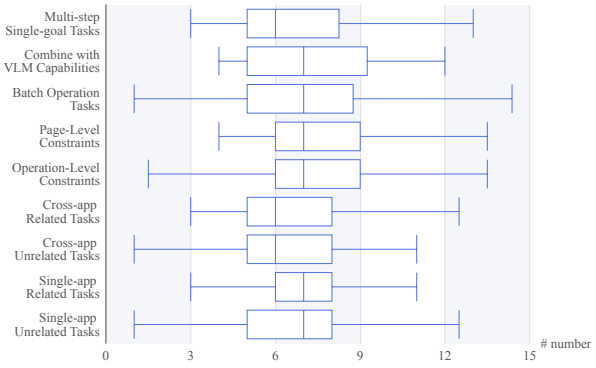


Figure 8. **Distribution of milestone numbers across different categories.**

tem prompt. For general-purpose models that do not offer a dedicated GUI prompt, such as GPT-4o and Claude 3.7 Sonnet, we referred to the prompt design of UI-TARS-1.5-7B and constructed a unified, similar system prompt. The specific content is as follows.

System Prompt

You are a GUI agent. You are given a task and your action history, with screenshots. You need to perform the next action to complete the task.

Output Format

Thought: ...
Action: ...

Action Space
click(start_box='<|box_start|>(x1,y1)<|box_end|>')
long_press(start_box='<|box_start|>(x1,y1)<|box_end|>', time='')
type(content='')
scroll(start_box='<|box_start|>(x1,y1)<|box_end|>', direction='down or up or right or left')
press_back()
press_home()
wait()
finished() # Submit the task regardless of whether it succeeds or fails.

Note
- Use {*language*} in Thought part.
- Summarize your next action (with its target element) in one sentence in Thought part.

User Instruction
{*instruction*}

The placeholders {*language*} and {*instruction*} will be dynamically replaced according to the specific properties of

each task. For actions that involve coordinate information (x1, y1), we adopt the coordinate format preferred by each general model: GPT-4o and Claude-3.7-Sonnet use absolute coordinates, while Gemini-2.5-Flash and Gemini-2.5-Pro use relative coordinates in the range 0–1000.

Since our benchmark also evaluates low-level instructions, in such cases we explicitly append the low-level instruction to the prompt in the following format: " ## The action you need to take now\n {*low_instruction*} " This ensures that the model can accurately reference the corresponding low-level instruction when performing the current action.

E.2. Milestone Evaluation Pipeline

During dynamic evaluation, we measure task progress by determining whether each milestone has been achieved, allowing us to assess task completion at a finer granularity. Specifically, we use screenshots of key components associated with each milestone, textual descriptions, and UI states as criteria to segment the agent's reasoning and action screenshot trajectory, and then determine milestone completion segment by segment.

By default, we segment trajectories in chronological order (e.g., each segment contains 10 actions). This is because, even if some milestones are unordered, such unordered relationships are usually limited to a small subset of milestones, while the overall process is still constrained within a certain time interval.

For ordered milestones, after providing a trajectory segment and its screenshot, we feed only the information for this single milestone into GPT-4o, asking it to determine whether the subtask corresponding to the current milestone has been completed: (1) If it is deemed completed, GPT-4o returns the index of the "last completed step." We then start from the step after that index and select the next trajectory segment to evaluate the next milestone. (2) If it is deemed not completed, we start from the last step of the current trajectory segment, select the next trajectory segment, and continue evaluating the current milestone.

For unordered milestones, we input all milestones in the unordered group into GPT-4o at once. GPT-4o then determines which milestones in the group have been completed within the current trajectory segment and returns the index of the last completed step for each completed milestone. We then choose the latest (largest) index among them as the starting point for the next segmentation. The prompt used in this process can be referred to in the following text:

Milestone Evaluation Prompt

You are an evaluation expert for smartphone operation tasks. Based on the provided sequence of screenshots showing the task being performed on a phone, determine whether each milestone in the specified milestone list has been successfully completed. Note that there is no required execution order among the milestones. If a milestone has been successfully completed, you need to return the index (starting from 0) of the last step in the trajectory at which that milestone is completed.

Guidelines

1. Do not make any assumptions: Judge only based on the provided screenshots. Do not infer or assume any information that is not explicitly shown in the screenshots.
2. Milestone completion criteria: A milestone is considered successfully completed only when:
- There is no correlation between multiple milestones; determining the completion of one milestone does not require consideration of other milestones. - The key component corresponding to the milestone (page, button, icon, text, etc.) clearly appears in the task trajectory.
- The state of that key component exactly matches the expected state required by the milestone.
3. Common failure reasons:
- Not completed: If, throughout the entire operation trajectory, the key component or required state specified by the milestone never appears, then the milestone is considered not executed or not completed.
- Noun/entity mismatch: If the milestone requires "open Taobao," but the screenshots show the "Meituan" page, the milestone fails. Entities that look similar but are not exactly the same (for example, "Xiaomi 15" vs "Xiaomi 16," or "driving route" vs "walking route") are regarded as different entities and should be judged as failures.
- Verb/action mismatch: If the milestone requires "like the post," but the screenshots show "favorite the post," the operation does not match and should be judged as a failure.
- State mismatch: If the milestone requires "follow this user," but the screenshot shows that the user is already in the "Following" state, and then the user performs an action that changes it to "Not following," the final state does not match the requirement and should be judged as a failure.
4. Corrective actions: If an incorrect operation occurs at some intermediate step but is later corrected by subsequent actions, as long as the final state exactly matches the correct state required by the milestone, it is considered successfully completed.

Input format

The milestone list in the input is structured as follows:

[
{
“idx”: 0,
“sub-target”: “Open Google Chrome and search for ‘panda’”,
“screenshot”: ¡image¿,
“bbox”: [0.023, 0.121, 0.976, 0.189],
“text”: “panda”,
“state”: [“selected”]
},
...
]

- “idx”: A unique identifier for each milestone (starting from 0).
- “sub-target”: Describes the target of the current milestone.
- “screenshot”: A reference screenshot of the final result for this milestone.
- “bbox”: The location of the key component in the screenshot, represented by normalized coordinates [x1, y1, x2, y2].
- “text”: The text content of the key component recognized by OCR.
- “state”: The state information of the key component related to the milestone.

Output format

Please return a JSON list, where each item indicates whether the input milestone has been completed, similar to:

[
{
“idx”: 0,
“state”: 0/1,
“last_idx”: 3
}
...
]

- “idx”: The sequence number of the corresponding milestone (provided in the input).
- “state” indicates the completion status of the milestone: 1 for success, 0 for failure.
- “last_idx” indicates the index of the last step in completing the milestone (starting from 0). If the milestone is not complete, -1 is returned.

Please strictly adhere to the JSON format for returning results, without any additional content or explanation.

Notes

- Do not infer or supplement information that is not shown in the screenshots; only judge based on visible content.
- The screenshots provided for each milestone are for reference only. A milestone should not be considered incomplete just because the exact same screen does not appear in the execution trace, since the phone UI may vary due to version differences, settings, or device state.
- You must ensure that every entity and every action described in the milestone is accurately matched, and that the final state is completed.
- Focus on the final state required by the milestone; do not be distracted by irrelevant or minor details in intermediate steps.
- Pay close attention to subtle interface differences, such as whether a tab is highlighted, or whether the text or icon of a button has changed, as these may be key to determining success or failure.

Through this process, we can determine the completion status of each milestone one by one. For a given task, once all milestones are judged to be completed, we consider the task as finished. Compared with directly feeding an entire, very long action trajectory into GPT-4o in a single pass, this segmented dynamic evaluation strategy effectively reduces errors in long-sequence understanding by the judgment model, thereby improving the reliability and accuracy of the evaluation.

F. Experimental Details

To better analyze the causes of the models’ performance limitations, we additionally report the results for Type Matching (TM) in Tab. 7. The results show that many models struggle to accurately grasp the current stage of the task, leading to prediction errors. However, once they are explicitly informed of “what needs to be done in the current step,” their performance improves significantly. This indicates that existing models still have substantial shortcomings in utilizing historical context and in analyzing and reasoning about tasks.

G. Trajectory Examples

As shown in Tab. 8, we further divide the three major categories under AndroidLens and provide corresponding example tasks for reference. It is worth noting that, for multi-step single-goal tasks, we focus on tasks that pursue a single objective within a single application, but require many steps due to the inherent complexity of the goal. Such tasks typically involve multiple page navigations, form filling, and similar operations.

In addition, we selected a price comparison case from the cross-app related tasks to more intuitively demonstrate the task execution process and annotation details. Specifically, as shown in Fig. 9 and Fig. 10, we provide the complete annotation operation trajectory and the corresponding screenshots.

H. Visualization Examples

H.1. Agent's Action Trajectory

To qualitatively analyze how existing models perform on AndroidLens, we present several examples of their actual operation processes during dynamic evaluation on AndroidLens. As shown in Fig. 11, we illustrate how UI TARS handles a case that requires the multimodal large model's own inherent capabilities. In this example, the task is to translate the content of an image into Chinese. Although the instruction is to translate it directly, the model takes a more complicated approach by invoking a third-party translation tool. As shown in Fig. 12, we demonstrate how the model responds to a missing-permission issue and is able to effectively determine and obtain the required permissions. However, this also leads us to reflect on the potential security risks introduced by such automated workflows.

H.2. Unexpected Event Examples

As described in the main text, we incorporated a large amount of data related to unexpected events into AndroidLens to evaluate the model's reflection and self-correction capabilities. Fig. 13 shows the model's performance in several representative, unanticipated scenarios. Encouragingly, the model demonstrates a certain degree of competence in handling sudden events. However, its predictions still contain some localization errors and hallucinations. For example, the agent repeatedly and incorrectly assumes that the close button is located in the top-left or top-right corner.

TASK I want to rent a car for one day tomorrow near X Park Area C in Beijing. Please compare the car rental prices on Fliggy and Meituan, and then stay on the final page of the one with the lowest price.

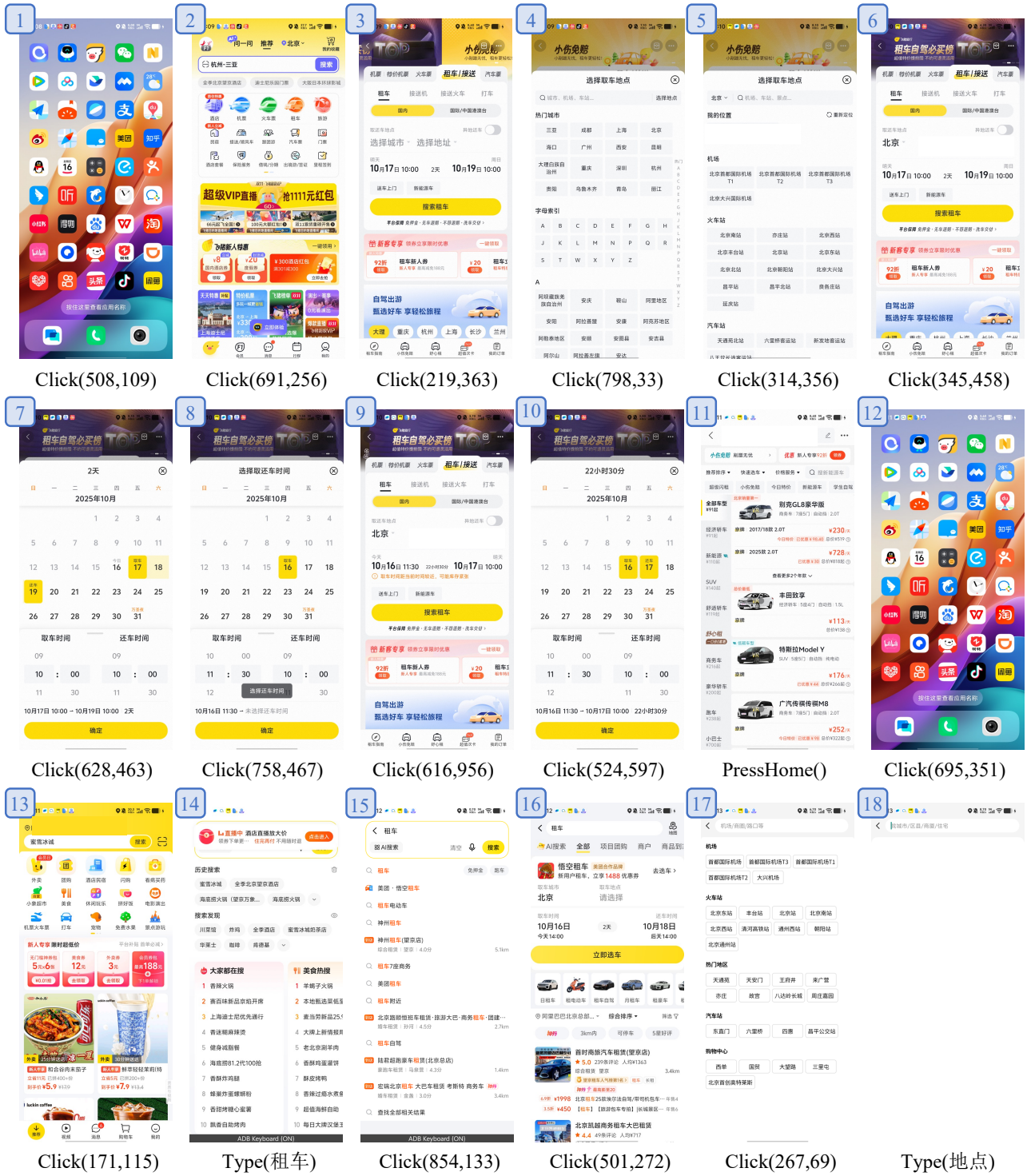


Figure 9. **Example of Annotation Trajectories for Cross-App Related Tasks.**

TASK I want to rent a car for one day tomorrow near X Park Area C in Beijing. Please compare the car rental prices on Fliggy and Meituan, and then stay on the final page of the one with the lowest price.

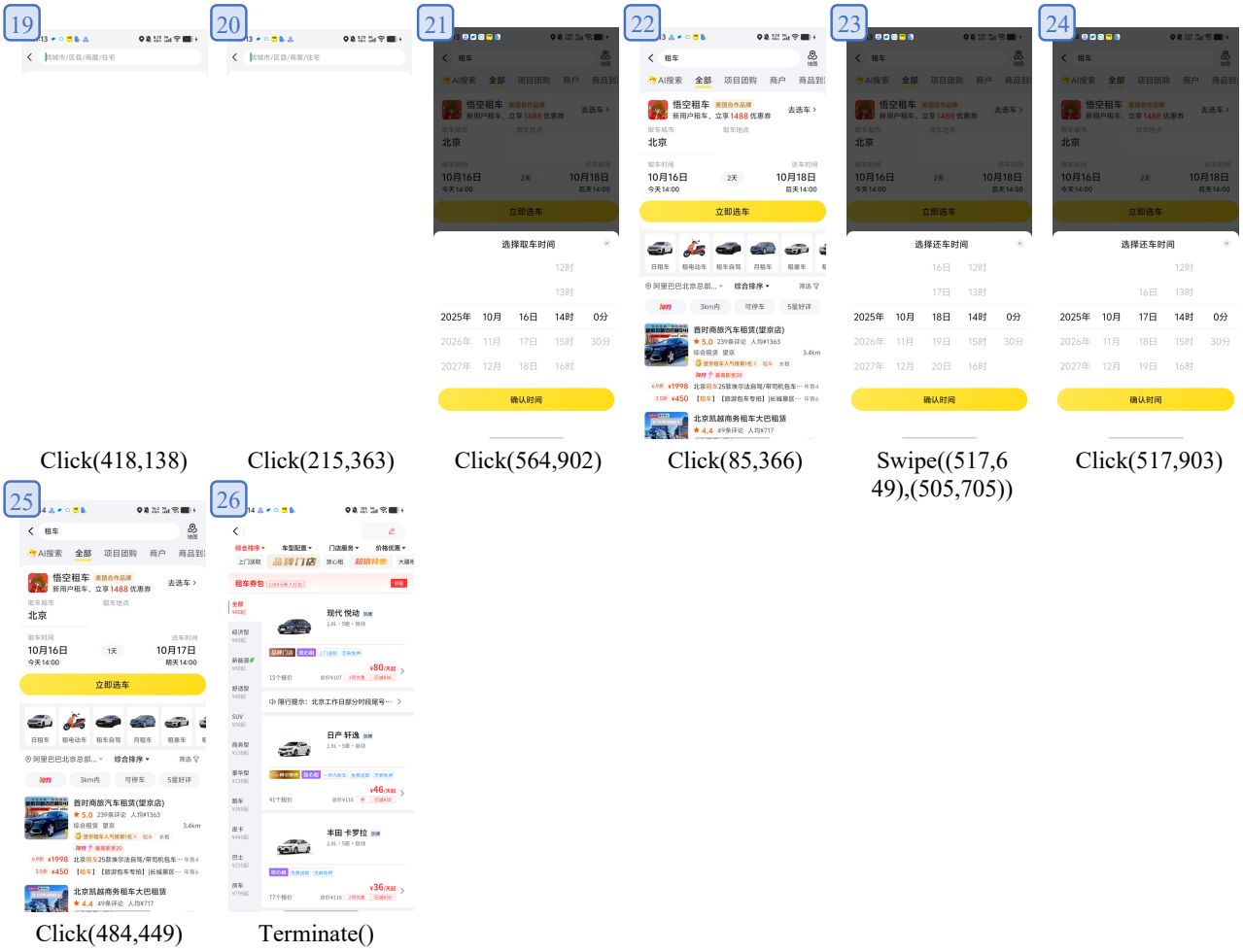


Figure 10. **Example of Annotation Trajectories for Cross-App Related Tasks.**

Table 5. **All Scenarios and Apps.**

Domain	Chinese APP	# Task	English APP	# Task
Commercial Applications				
Online Shopping	Taobao, JD.com	56	eBay, AliExpress	23
Mobile Payments	Alipay	18	PayPal	2
Instant Messaging	WeChat, QQ	95	Facebook, Telegram	22
Lifestyle Sharing	Xiaohongshu	24	Instagram	13
Knowledge Sharing	Zhihu	20	Quora	5
Social Media	Weibo	16	X	4
Video Sharing	Bilibili	21	YouTube	14
Short Video	Douyin	26	TikTok	2
Local Services	Meituan, Fliggy	55	Trip.com, Booking.com	21
Navigation & Transportation	Amap, DiDi	32	Google Maps, Uber	12
Search	Baidu	27	Google Chrome	13
Digital Reading	Tomato Novel	16	WebNovel	5
Online Meetings	Tencent Meeting	10	Zoom	4
News Aggregation	Toutiao	9	Google News	5
Cloud Storage	Alibaba Cloud Drive	9	Google Drive	10
Office Tools	WPS Office	16	Microsoft Office	2
Logistics Tracking	Cainiao	3	17Track	1
Secondhand Marketplace	Xianyu	12	eBay	8
Workplace Collaboration	Feishu, DingDing	18	Teams	7
Music Streaming	QQ Music	16	Spotify	8
Video Streaming	Youku	24	Prime Video	4
Task Management	DIDA	12	Microsoft To Do	-
Local Reviews	Dianping	10	Yelp	5
Job Recruitment	BOSS Zhipin	1	LinkedIn	4
Fitness & Health	KEEP	9	MyFitnessPal	2
App Store	App Store	4	Play Store	-
System Application				
Weather	Weather	17	Weather	7
Calculator	Calculator	13	Calculator	10
Calendar	Calendar	21	Calendar	9
Clock	Clock	5	Clock	5
Messages	Messages	9	Messages	2
Contacts	Contacts	10	Contacts	8
Photos	Photos	15	Photos	15
Mail	Mail	11	Mail	6
Notes	Notes	19	Notes	11
Recorder	Recorder	2	Recorder	1
My Files	My Files	9	My Files	18
Settings	Settings	4	Settings	12

Table 6. **Classification and examples of cross-application interaction tasks.**

Classification	**Example**
Sharing	Help me find a nearby cinema on Meituan and buy two tickets for a popular movie, choosing two adjacent seats as close to the middle as possible. After purchasing, take a screenshot of the ticket information and send it to "GUI Test" on QQ.
Clipboard Operation	Open Google Chrome and search for AI answers to "A healthy lunch suitable for the beginning of autumn". Choose one recommended dish, copy the ingredient list from its recipe into my Notes app, then search YouTube for a video on how to make it.
Application Redirection	Open Trip, search for food, set the location to International Trade CBD, apply the Open now filter, open the nearest place, click its map location, then switch to Google Maps and show driving directions from International Trade CBD to that place.
Authorization	Go to the app store to install Zhihu. After the installation, open the app and complete the login process by authorizing via Weibo.
Tool Call	Open my phone's Settings and go to Emergency SOS; turn on quick emergency access and automatic location sharing, add Dad and Mom from my contacts as emergency contacts, and then enable Earthquake Early Warning.
Comparison	On Yelp, search for coffee shops near the Googleplex, filter to Open Now and within 1 mile, and open the top two listings. Then in Google Maps, get walking directions from the Googleplex to each, pick the one with the shortest walk time, and save that business on Yelp.
Information Aggregation	Go to Quora, search for the query "Chinese food recommend", review at least five posts, identify the three dishes mentioned most often, and add them as separate lines in a new note in the Notes app titled "Chinese food summary".
Multi-Round Interactions	Open Uber Eats, set my delivery address to the Googleplex, then search for Pizza, Poke, Fast Food, and Chinese Food. For each category, pick one restaurant with a delivery ETA under 30 minutes and send the links to my Telegram contact "liuyujue". After you've shared all four, send message: "Which restaurant shall we go to tonight?"

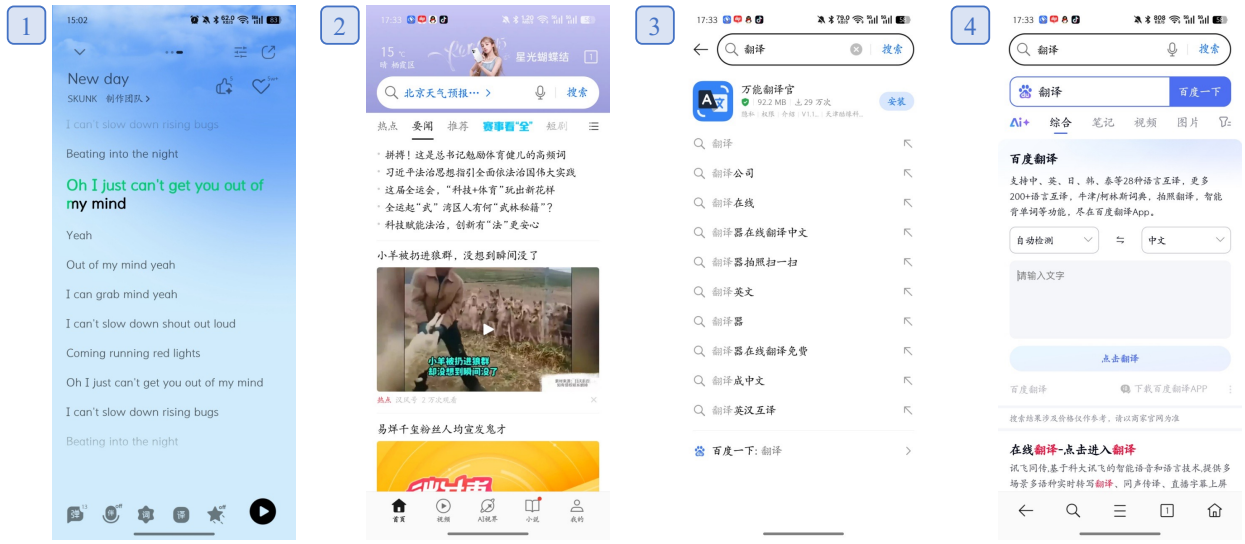
Table 7. **Static Evaluation Results on AndroidLens.** HL denotes high-level instructions, LL denotes low-level instructions, AMS represents Action Matching Score, and TM represents action type matching.

Model	Chinese-LL		Chinese-HL		English-LL		English-HL		Total-LL		Total-HL	
	AMS	TM	AMS	TM	AMS	TM	AMS	TM	AMS	TM	AMS	TM
Agent workflows												
Mobile-Agent-v2	47.18	63.40	32.66	59.76	47.46	64.35	33.43	55.88	47.27	63.72	32.92	58.46
Mobile-Agent-E	74.52	95.36	50.12	81.26	75.22	95.31	49.85	78.12	74.98	95.34	50.03	80.21
GPT-4o+UGround-V1-7B	64.15	94.95	46.08	73.23	66.63	91.68	44.68	71.87	69.73	93.43	45.61	72.77
Agent-as-a-models												
GPT4o	29.98	79.08	23.48	80.53	28.13	78.18	23.78	73.03	29.36	78.78	23.58	78.02
Claude-3.7-sonnet	28.25	83.52	22.28	77.76	32.66	83.58	27.33	78.20	29.73	83.54	23.97	77.91
Gemini-2.5-Flash	32.57	68.83	26.61	60.59	31.46	67.27	23.44	66.49	30.24	68.31	25.55	62.57
Gemini-2.5-Pro	45.23	87.95	39.07	76.55	44.02	86.53	35.85	72.01	44.82	87.48	37.99	75.03
Qwen2.5-VL-7B	64.14	85.89	34.43	53.90	50.95	71.22	26.81	40.34	59.71	80.97	31.88	49.36
OS-Atlas-7B-Pro	49.84	79.37	35.11	71.80	47.28	74.91	37.92	71.04	48.98	77.88	36.05	71.55
Aguvis-7B	66.37	97.66	12.73	36.10	60.04	93.84	10.99	24.42	64.25	96.38	12.15	32.19
UI-Venus-Navi-7B	64.15	83.67	45.92	75.76	55.93	79.07	45.12	75.58	61.40	82.13	45.65	75.70
UI-AGILE-7B	59.30	80.11	38.22	76.46	55.99	81.51	38.10	73.17	58.19	80.58	38.13	75.36
AgentCPM-GUI-8B	71.13	94.49	42.90	78.29	71.67	95.47	43.93	75.28	71.31	94.82	43.25	77.28
UI-TARS-7B-DPO	82.25	97.72	49.61	77.60	79.41	95.79	54.48	76.74	81.30	97.07	51.24	77.31
UI-TARS-1.5-7B	78.92	98.50	55.74	83.34	73.83	97.81	51.27	79.97	77.22	98.27	54.21	82.21

Table 8. **Classification and examples of tasks in AndroidLens.**

Subcategory	Example
Multi-goal tasks.	
Single-app related tasks	First, find out what day of the week it is today, then open Amap and set my vehicle as a small passenger car with the license plate Jing ABDCF0 and fuel type gasoline. After that, go to the Tools tab, find the "Driving Restriction Inquiry" feature, and check whether there are any restrictions for today.
Single-app unrelated tasks	First, open HelloBike in Alipay and check the bike card purchase page. Then go to Ant Insurance to view my insurance policies. Finally, go to Alipay Membership to see all my privileges.
Cross-app related tasks	Remove classmates Zhang San, Zhao Si, and Li Wu from your QQ contacts. Then, on Douyin, mark the first recommended video on the home page as "Not Interested".
Cross-app unrelated tasks	Open Trip.com and find the earliest flight from San Diego to San Francisco for tomorrow, note the actual departure time and estimated arrival time, text Mom: "My flight is expected to arrive at [arrival time] tomorrow. Please pick me up at [airport name] Airport.", and create a calendar event on my phone titled "FlightReminder" set for two hours before the flight's actual departure time.
Multi-constraint tasks.	
Operation-level constraints tasks	On AliExpress, find a Xiaomi smartphone under $200 with 8GB RAM and 256GB storage, in black, with 33W fast charging; pick the best-selling one and add it to my cart.
Page-Level constraints tasks	I need you to open WebNovel, go to Explore, switch to Comics, and select the School category. Filter for Translation titles with 100–500 chapters that are ongoing and updated in the last 30 days, sort by last updated, and then open the first chapter of the second comic in the results.
Domain-specific tasks.	
Batch operation tasks	Please use Zoom to schedule three meetings for me: "Stand-up meeting" tomorrow at 10:00 AM for 30 minutes, set to repeat daily; "Weekly Meeting" next Monday at 2:00 PM for 30 minutes, set to repeat weekly; and "Mid-year summary" the day after tomorrow at 7:00 PM for 1 hour. Use my local time.
Combine with VLM capabilities tasks	Open Zhihu and search for "What books are needed to prepare for IELTS?". Sort the results by number of upvotes and open the third article. Read through it and identify the titles of the three recommended books mentioned there. Then open JD.com, search for these three books one by one, and add the best-selling in-stock item for each book to your shopping cart.
Multi-step single-goal tasks	Help me use the satellite rescue feature in Amap (Gaode Map) to request vehicle assistance. Call a tow truck to tow my car to Building A of the China World Trade Center. The contact phone number is 1504042147 and the license plate number is Jing AABCD1. Then initiate the rescue request.

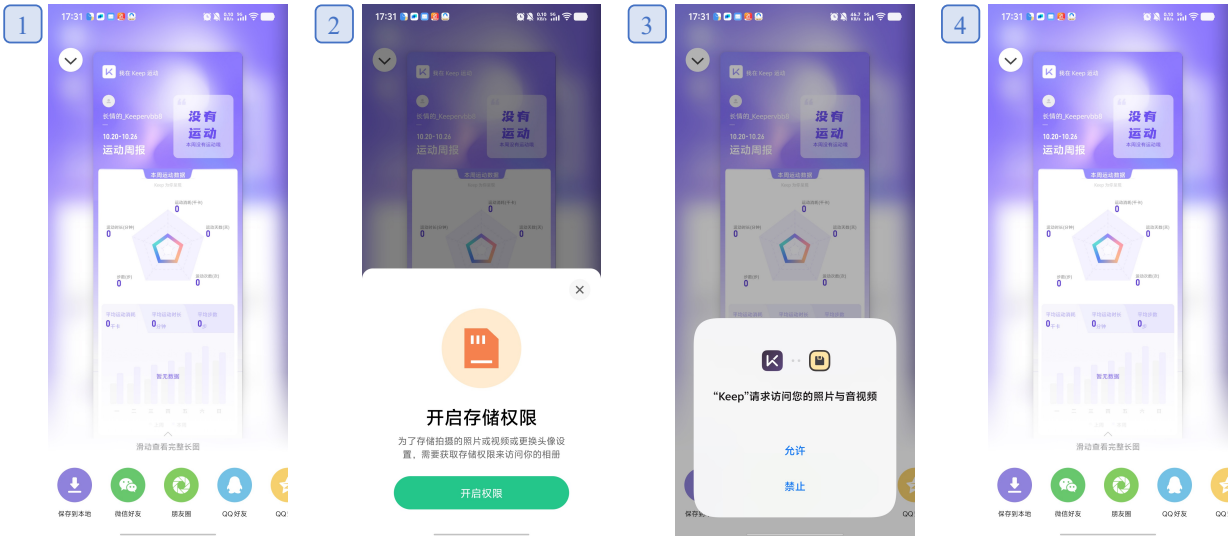
TASK Help me open translate_to_zh.png in GUI_Test/Image, recognize the text in the image and translate it into Chinese. Then use WPS to create a new document named translate_to_zh, write both the original text from the image and its Chinese translation into the document, and save it.



UI TARS: Open(浏览器) | Click(530, 247) | Type(翻译) | Click(511, 627)

Figure 11. **A Visual Example of Agent Execution Requiring the VLM's Own Capabilities.**

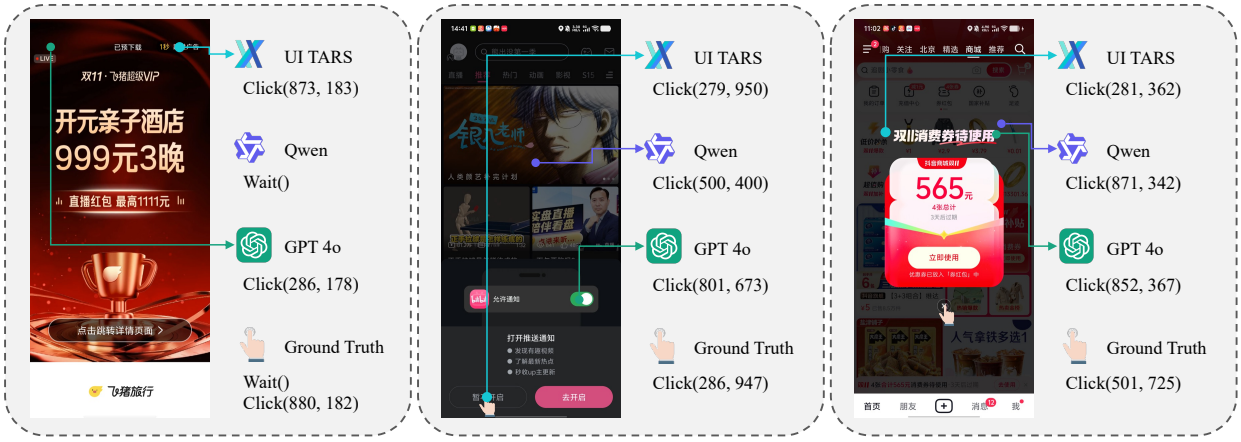
TASK Open Keep and update my weight to 68 kg and my height to 181 cm. Then go to "My Data," view the weekly report in the Trends section, and export it to save locally. After that, exit the app and stop it from running in the background.



Human: Click(119, 915) | Click(544, 911) | Click(517, 838) | Click(119, 915)

UI TARS: Click(121, 910) | Click(530, 903) | Click(510, 823) | Click(123, 917)

Figure 12. **A Visual Example of an Agent Handling Missing Permissions.**



(a) App Launch Advertisements **(b) Notification Permission** **(b) Pop-up Advertisements**

Figure 13. **A Visual Example of an Agent Handling Various Emergencies.**