

METHOD AND APPARATUS FOR AUTOMATED OPERATION OF CLOUD PHONE

TECHNICAL FIELD

5 **[0001]** The present disclosure relates to the field of cloud computing technologies, and in particular, to the field of cloud phone technologies.

BACKGROUND

10 **[0002]** Modern mobile applications and web pages are becoming increasingly feature-rich, and user operation scenarios on smartphones are growing more complex. This has created significant demand for UI (User Interface) automation operations, including automated test case execution in software testing, automatic application operations, and cross-application information integration.

15 **[0003]** Currently, several explorations and practices have emerged in UI automation and intelligent agent interface operations, which can be mainly categorized into the following types of solutions:

20 **[0004]** First, traditional UI automation testing tools: the widely adopted approach involves using professional automation testing frameworks and scripting tools. For mobile platforms, examples include Appium and Airtest; and for web platforms, examples include Selenium and Puppeteer. These tools operate UI elements through underlying interfaces, requiring test engineers to write scripts to locate elements and perform actions such as clicking and inputting.

25 **[0005]** Second, manual rules and OCR (Optical Character Recognition) solutions: in scenarios without direct programmable interfaces, some tools employ image recognition and predefined rules to automatically operate interfaces. For example, computer vision-based tools such as Sikuli determine click positions by capturing screenshots and matching them against pre-provided control images. Additionally, some RPA (Robotic Process Automation) software uses OCR to read text on the screen and then determines the next operation based on preset rules.

5 **[0006]** Third, voice assistants and shortcut commands: another category of related technologies targeting end consumers includes voice assistants on mobile phones and some system-level automation. Users can instruct assistants to perform simple tasks through voice commands, such as sending text messages, setting reminders, and opening applications.

SUMMARY

10 **[0007]** Embodiments of the present disclosure provide a method and an apparatus for an automated operation of a cloud phone, a device, a storage medium and a program product.

[0008] According to a first aspect, an embodiment of the present disclosure provides a method for an automated operation of a cloud phone, including:

[0009] obtaining a task sequence of the cloud phone;

15 **[0010]** for a current task in the task sequence, performing the following operation steps: generating a current operation instruction by using a vision-language model to reason about the current task and current interface information of the cloud phone; after executing the current operation instruction on the cloud phone, loading next interface information; and in response to the current task being the last task of the task sequence, determining that the automated operation of the cloud phone is completed; and

20 **[0011]** in response to the current task not being the last task of the task sequence, taking a next task in the task sequence as a new current task, taking the next interface information as new current interface information, and continuing to perform the operation steps.

25 **[0012]** According to a second aspect, an embodiment of the present disclosure provides a training method for a vision-language model, including:

[0013] obtaining a first training sample and a second training sample of a sample application, where the first training sample includes a previous task of the sample application and a current interface screenshot, and the second training sample includes multiple rounds of dialogue corresponding to a current task of the application; and

30 **[0014]** performing interleaved training on the vision-language model by using the

first training sample and the second training sample until the vision-language model meets a preset condition.

[0015] According to a third aspect, an embodiment of the present disclosure provides an apparatus for an automated operation of a cloud phone, including:

5 **[0016]** an obtaining module configured to obtain a task sequence of the cloud phone;

[0017] an operation module configured to, for a current task in the task sequence, perform the following operation steps: generate a current operation instruction by using a vision-language model to reason about the current task and current interface information of the cloud phone; after executing the current operation instruction on the cloud phone, load next interface information; and in response to the current task being the last task of the task sequence, determine that the automated operation of the cloud phone is completed; and

10 **[0018]** an execution module configured to, in response to the current task not being the last task of the task sequence, take a next task in the task sequence as a new current task, take the next interface information as new current interface information, and continue to perform the operation steps.

15 **[0019]** According to a fourth aspect, an embodiment of the present disclosure provides a training apparatus for a vision-language model, including:

20 **[0020]** an obtaining module configured to obtain a first training sample and a second training sample of a sample application, where the first training sample includes a previous task of the sample application and a current interface screenshot, and the second training sample includes multiple rounds of dialogue corresponding to a current task of the application; and

25 **[0021]** a training module configured to perform interleaved training on the vision-language model by using the first training sample and the second training sample until the vision-language model meets a preset condition.

[0022] According to a fifth aspect, an embodiment of the present disclosure provides an electronic device, including:

[0023] at least one processor; and

30 **[0024]** a memory communicatively connected to the at least one processor;

[0025] where the memory stores instructions executable by the at least one processor, and the instructions, when executed by the at least one processor, enable the at least one processor to perform the method according to the first aspect or the second aspect.

[0026] According to a sixth aspect, an embodiment of the present disclosure provides a non-transitory computer-readable storage medium storing computer instructions, where the computer instructions are used to cause a computer to perform the method according to the first aspect or the second aspect.

[0027] According to a seventh aspect, an embodiment of the present disclosure provides a computer program product, including a computer program which, when executed by a processor, implements the method according to the first aspect or the second aspect.

[0028] The content described in this section is not intended to identify key or important features of the embodiments of the present disclosure, nor is it used to limit the scope of the present disclosure. Other features of the present disclosure will become easily understood through the following description.

BRIEF DESCRIPTION OF THE DRAWINGS

[0029] Other features, objectives and advantages of the present disclosure will become more apparent by reading the detailed description of the non-limiting embodiments made with reference to the following drawings. The drawings are used to better understand the solution and do not constitute a limitation to the present disclosure. In the drawings:

[0030] FIG. 1 is a system architecture diagram of a method for an automated operation of a cloud phone;

[0031] FIG. 2 is a flowchart of an embodiment of a method for an automated operation of a cloud phone according to the present disclosure;

[0032] FIG. 3 is a flowchart of an embodiment of a training method for a vision-language model according to the present disclosure;

[0033] FIG. 4 is a state iteration diagram of a method for an automated operation of a cloud phone;

[0034] FIG. 5 is a structural schematic diagram of an embodiment of an apparatus for an automated operation of a cloud phone according to the present disclosure;

[0035] FIG. 6 is a structural schematic diagram of an embodiment of a training apparatus for a vision-language model according to the present disclosure; and

[0036] FIG. 7 is a block diagram of an electronic device for implementing the method for an automated operation of a cloud phone according to the embodiments of the present disclosure.

DETAILED DESCRIPTION OF EMBODIMENTS

[0037] The following describes exemplary embodiments of the present disclosure with reference to the accompanying drawings, which include various details of the embodiments of the present disclosure to facilitate understanding, and they should be regarded as merely exemplary. Therefore, those of ordinary skill in the art should recognize that various changes and modifications may be made to the embodiments described herein without departing from the scope and spirit of the present disclosure. Similarly, descriptions of well-known functions and structures are omitted in the following for clarity and conciseness.

[0038] It should be noted that the embodiments in the present disclosure and the features in the embodiments may be combined with each other without conflict. The present disclosure will be described in detail below with reference to the accompanying drawings and in conjunction with the embodiments.

[0039] FIG. 1 shows a system architecture diagram of a method for an automated operation of a cloud phone. As shown in FIG. 1, the system architecture may include a three-layer and two-loop structure, i.e., a three-layer structure of an input layer 101, a decision-making layer 102 and an execution layer 103, and a closed-loop control of a perception loop and an action loop.

[0040] The input layer 101 may receive a task description, which is usually a segment of natural language instruction and may be a description of a specific target or a series of operation steps. A user may input through text, or an upper-layer application may provide an instruction in the form of an API (Application Programming Interface).

[0041] The decision-making layer 102 is the brain of the system, including a task parsing and planning module and a multimodal LLM (Large Language Model) controller.

[0042] The task parsing and planning module is responsible for decomposing high-level user instructions into executable task sequences and managing the workflow. In simple cases, instructions may be directly sent to the multimodal LLM Controller for step-by-step execution. However, for complex tasks, especially those containing multiple objectives, the task parsing and planning module may first generate a preliminary step-by-step plan based on rules or LLM. For example, for the instruction "open the shopping app to buy a red dress," the task parsing and planning module may produce the steps: "Step 1: Open the shopping app; Step 2: Search for 'red dress'; Step 3: Filter/select the product and place an order." These steps are then executed one by one by the multimodal LLM Controller. The task parsing and planning module also monitors the execution process and adjusts subsequent plans as needed (for example, retrying or skipping if a step fails).

[0043] The multimodal LLM Controller serves as the core AI (Artificial Intelligence) decision-making unit, equipped with a fine-tuned multimodal large language model. It receives two types of inputs: first, the current natural language subtask description (from user instructions or the task parsing and planning module), and second, the current interface information from the execution layer (including interface screenshots and possible UI structure data). The multimodal LLM Controller may combine the "seen" interface with the understood instruction to infer the next specific operation action (such as "click at coordinates (x,y)" or "input 'hello' in the text box"). It can be considered that the LLM Controller simulates the human cognitive process of viewing the screen - >understanding->operating, and its output is the decision for the next action.

[0044] The execution layer 103 implements actual operations on the cloud phone and interface acquisition, serving as the interface for Agent-environment interaction, including cloud phone instances and device operation interfaces.

[0045] The cloud phone instance is the environment where applications actually run, which may be virtual devices hosted in cloud data centers, with screen output and input control accessible or transmittable via network. The system can launch multiple

instances on demand, with each instance regarded as an independent phone for AI operation.

[0046] The device operation interface may be responsible for converting action instructions from the multimodal LLM Controller into actual input operations on the cloud phone. For example, if the LLM decides to "click somewhere on the screen," the device operation interface invokes ADB (Android Debug Bridge) commands or simulates click events sent to the cloud phone; if "swiping the screen" is required, swipe events are sent; if "inputting text" is needed, ADB's input text commands or simulated keyboard are used. Similarly, the device operation interface is also responsible for collecting interface information from the cloud phone and feeding the interface information to the multimodal LLM Controller, including interface screenshots and possible auxiliary information (such as control structure of the current UI, text of elements, etc.).

[0047] The figure illustrates two closed loops: the perception loop (feedback from cloud phone to LLM) and the action loop (control from LLM to cloud phone), corresponding to interface perception and action execution cycles respectively.

[0048] The entire architecture forms a closed-loop control system: the LLM obtains information from environmental perception and continuously adjusts its decisions; its actions change the environmental state, and the new state is then fed back to the LLM until task completion. This design resembles the Agent-environment loop in reinforcement learning, except that the decision-making core is an LLM trained for specific purposes rather than manually written policies. The architecture introduces an intelligent decision-making layer and visual feedback, endowing the system with adaptive capabilities: if a certain step does not achieve the expected result, the AI can decide to execute other operations or retry based on the new interface content, rather than blindly following a fixed process to the end.

[0049] FIG. 2 shows a flowchart 200 of an embodiment of a method for an automated operation of a cloud phone according to the present disclosure. The method for automated operation of a cloud phone includes the following steps.

[0050] Step 201 includes: obtaining a task sequence of the cloud phone.

[0051] In this embodiment, an execution subject of the method for an automated

operation of a cloud phone may obtain a task sequence of the cloud phone.

[0052] In some embodiments, task description information is received and parsed to generate a task sequence. The task description information is typically a natural language instruction, which may be a specific objective (e.g., "Please post a Weibo with the content 'The weather is nice today' in the Weibo application.") or a description of a series of operation steps. Users may provide instructions through text input, or upper-layer applications may provide instructions in API form.

[0053] During the initialization phase, the task description information is parsed to generate a task sequence. If the task description information contains multiple subtasks (such as "...then...followed by..." structure), it is decomposed into a task sequence. If the objective in the task description information is abstract, a knowledge base or rule base may be queried to supplement specific steps (for example, "Help me book a movie ticket for tonight" requires first determining the cinema application, selecting showtimes, etc., before planning the movie ticket booking). The result of the initialization phase is to determine the preliminary execution plan and goal determination conditions (how to determine task completion). Meanwhile, an idle cloud phone instance is allocated or launched from the resource pool to prepare the application environment to be operated. For example, this may be achieved by pre-installing relevant applications or using snapshots to restore to a clean initial interface.

[0054] Step 202 includes: generating a current operation instruction by using a vision-language model to reason about the current task and current interface information of the cloud phone.

[0055] In this embodiment, for a current task in the task sequence, a current operation instruction may be generated by using a vision-language model to reason about the current task and current interface information of the cloud phone.

[0056] Current interface information may be obtained from the cloud phone through the device operation interface. The current interface information mainly includes interface screenshots, and may also include structured information of UI elements. This information may be organized into a format suitable for vision-language model input and provided to the vision-language model. For example, the interface screenshot is encoded as an image embedding, together with additional text descriptions (such as "The current

interface contains XX button, YY text") provided to the vision-language model.

[0057] In some embodiments, a preset prompt, the current task, and the current interface information are input into the vision-language model, and the current operation instruction may be output.

5 **[0058]** The current task and the current interface information are combined and provided to the vision-language model. For example, the prompt may be "User intent: Click the login button to log in.\n Interface information: [image]". Through a specific prompt template, the vision-language model is guided to follow the user intent while considering visual content. Since the vision-language model has been specially trained, it
10 can effectively associate interface information with tasks and locate the position on the screen corresponding to the target mentioned in the task.

[0059] In some embodiments, the vision-language model may include a vision sub-model and a language sub-model; the current interface information is input into the vision sub-model, and object information on the current interface may be output; the
15 object information on the current interface and the current task are input into the language sub-model, and object information corresponding to the current task may be output; and the current operation instruction may be generated based on the object information corresponding to the current task.

[0060] After receiving the prompt, the vision-language model may first understand
20 the interface through the vision sub-model. This step is similar to OCR and image feature extraction, where the vision sub-model recognizes key information such as text, icons, and layout on the interface. Then, the language sub-model may search for interface elements matching the task requirements in the context and generate operation decisions. Here, the vision-language model may be trained to generate output in an action
25 description format, for example: <action type="tap" target="Login button" coordinates="[100,200]"/>, indicating a click on the "Login" button at coordinates (100,200). Or <action type="input" target="Username input field" text="testuser"/>, indicating input of "testuser" into the username field. This output format may be learned through model fine-tuning, adopting structures similar to JSON (JavaScript Object
30 Notation) or XML (Extensible Markup Language) for machine parsing. The actual output may also be natural language describing the action, but for reliability, structured

output may be agreed upon to reduce ambiguity.

[0061] Step 203 includes: loading next interface information after executing the current operation instruction on the cloud phone.

[0062] In this embodiment, the execution subject may load next interface information after executing the current operation instruction on the cloud phone.

[0063] The device operation interface may parse the current operation instruction output by the vision-language model and execute actual operations on the cloud phone instance through ADB or automation tools. For example, the operations include invoking adb shell input tap 100 200 to implement a click at (100,200), or invoking adb shell input text "testuser" to implement text input. After execution, a brief delay is applied to wait for the interface update.

[0064] In some embodiments, in response to the next interface information not satisfying a preset condition, automatic correction may be performed based on the next interface information. For example, an error type is determined based on the next interface information, and correction is performed using a correction method corresponding to the error type.

[0065] If the interface after executing the operation output by the vision-language model does not exhibit the expected change (such as remaining on the same page or displaying an error prompt), fault tolerance processing may be performed. Specifically, automatic correction strategies are attempted based on the new interface. For example, when a click does not produce any effect, the vision-language model may infer that "it may be necessary to close the popup first," and thus outputs an action to close the dialog box that popped up. This capability stems from the inclusion of examples of abnormal situations in the training of the vision-language model, enabling it to learn some common error recovery patterns (such as "if a 'password error' prompt appears, capture a screenshot for record and terminate"). Secondly, if the vision-language model fails to handle the situation on its own, measures may also be taken when detecting prolonged lack of progress, such as resetting the interface or attempting alternative solutions. The multi-level error handling mechanism can ensure that tasks are completed as much as possible or clear failure feedback is provided under various circumstances, rather than becoming unresponsive or hanging.

[0066] Step 204 includes: determining whether the current task is the last task of the task sequence.

[0067] In this embodiment, the execution subject may determine whether the current task is the last task of the task sequence. If the current task is the last task, Step 205 is executed; and if the current task is not the last task, Step 206 is executed.

[0068] Step 205 includes: determining that the automated operation of the cloud phone is completed.

[0069] In this embodiment, if the current task is the last task of the task sequence, the execution subject may determine that the automated operation of the cloud phone is completed.

[0070] Step 206 includes: taking a next task in the task sequence as a new current task, and taking the next interface information as new current interface information.

[0071] In this embodiment, if the current task is not the last task of the task sequence, the execution subject may take a next task in the task sequence as a new current task, take the next interface information as new current interface information, and return to continue executing Step 202.

[0072] After the action is executed, the next interface information of the cloud phone may be obtained again through the device operation interface. If the previous step is text input or a page jump by clicking, a new interface is loaded. In this case, it is necessary to check whether the task is completed or whether to enter the next task, which may be determined according to preset conditions, for example, the current task is considered completed when a "login successful" prompt is detected. If the task is not completed yet, the process loops back to Step 202, and the new current interface information is submitted to the vision-language model to continue deciding the next step.

[0073] Steps 202-206 are a cyclic execution phase, and this cycle continues until a task completion condition is met or a maximum step limit is reached. The determination of task completion may be based on the user initial requirements. For example, if the user requires "complete the order placement", the task is considered completed when an order confirmation page is detected; if the user requires "query and return certain information", the information is transmitted back through structured output after being read, and then the task is ended. In addition, the operation logs and interface screenshots

during the execution process may be organized into a report and provided to the user or recorded for archiving as test evidence to prove the execution of the task.

[0074] The embodiment of the present disclosure provides a method for an automated operation of a cloud phone, which adopts a one-step decision mode in the "perception-decision-action" cycle, where the vision-language model determines only the action for the current single step at each step. This resembles the step-by-step operations when humans perform tasks, combined with feedback to adjust strategies. This step-by-step closed loop facilitates timely error correction and handling of uncertainty. For example, in human-computer interaction, each step may encounter situations inconsistent with expectations; and by having the model view the interface at every step, subsequent decisions can be ensured to be based on the latest state.

[0075] FIG. 3 shows a flowchart 300 of an embodiment of a training method for a vision-language model according to the present disclosure. The training method for a vision-language model includes the following steps.

[0076] Step 301 includes: obtaining a first training sample and a second training sample of a sample application.

[0077] In this embodiment, an execution subject of the training method for a vision-language model may obtain a first training sample and a second training sample of a sample application.

[0078] The first training sample may include a previous task of the sample application and a current interface screenshot. The second training sample may include multiple rounds of dialogue corresponding to a current task of the application.

[0079] In some embodiments, the current interface screenshot is divided into multiple grid cells, and the multiple grid cells are clustered to remove grid cells corresponding to duplicate background regions.

[0080] Typically, directly inputting the complete current interface screenshot into the vision-language model generates a large number of visual tokens, occupying the attention window. To reduce invalid visual tokens, the current interface screenshot may be preprocessed. First, the current interface screenshot is divided into fixed-size grid cells; then, duplicate background regions are detected through clustering, and only important unique portions of the current interface screenshot are retained. In this way, a

large number of visual tokens are reduced, enabling the vision-language model to focus attention on key UI elements. Without significantly compromising understanding accuracy, inference speed is improved to accommodate real-time interaction.

[0081] In some embodiments, interface elements are selected from the user interface tree of the sample application, and tasks corresponding to the interface elements are automatically generated through scripts; after executing the tasks corresponding to the interface elements in the sample application, a current interface screenshot of the sample application is captured; and the first training sample is generated based on the tasks corresponding to the interface elements and the current interface screenshot.

[0082] Constructing a small-scale but high-quality GUI (Graphical User Interface) instruction dataset enables the vision-language model to undergo further instruction fine-tuning on top of pre-training. Sources of the GUI instruction dataset may include: interface element localization data, application navigation data, and question-answering and assertion data.

[0083] Interface element localization data may include, for example, "click the red login button" or "find the 'Settings' icon on the screen." A large number of screenshots from different applications are collected, with several elements of interest and corresponding instructions annotated manually or with the aid of rules. Some data references open-source screenshot-to-description datasets, processed and organized for training the vision-language model's OCR and localization capabilities. In particular, manual annotation is performed for Chinese application interfaces to ensure the vision-language model learns to recognize common Chinese UI text (Chinese characters on buttons, such as "Confirm," "Cancel") and typical control styles.

[0084] Application navigation data may include, for example, multi-step operation demonstrations such as "Open the Alipay App and enter the balance page." To obtain such data, some synthetic tasks may be automatically generated through scripts. Utilizing the UI trees of real applications, programs are written to randomly select interface elements to form instructions, such as "Please click the XX menu and then open the YY function," and executed through UI automation frameworks, recording intermediate screenshots and actions to generate data with process supervision signals. These synthetic tasks are considerable in number and broad in coverage, helping the vision-

language model learn general cross-interface operation strategies.

[0085] Considering that automated operations of the cloud phone sometimes require returning information, the vision-language model needs to be trained to read answers from the interface. For example, for "Please tell me the product price displayed on the current page," the vision-language model can learn to recognize price text and answer. The above localization data is extended to generate some question-answering format data, enabling the vision-language model not only to output action instructions but also to output text answers when needed. Meanwhile, providing the vision-language model with descriptions of determination conditions enables it to determine whether interface states satisfy conditions and output Boolean-type results. These two types of data endow the vision-language model with simple screen-reading and determination capabilities, satisfying more complex testing requirements.

[0086] Step 302 includes: performing interleaved training on the vision-language model by using the first training sample and the second training sample until the vision-language model meets a preset condition.

[0087] In this embodiment, the execution subject may perform interleaved training on the vision-language model by using the first training sample and the second training sample until the vision-language model meets a preset condition. The preset condition may be that the model effect reaches a preset effect, the number of iterations reaches a preset number, and the like.

[0088] Here, an open-source vision-language model is selected as the underlying multimodal model, which can have the ability to understand and describe image content and support both Chinese and English languages. Without specific fine-tuning, the vision-language model can already show good OCR and common sense question-and-answer capabilities. The vision-language model opens the model weight and may be deployed locally, which is suitable for customized training in a private environment.

[0089] In order to enable the vision-language model to support the automated operation of a cloud phone, targeted secondary training (fine-tuning) may be performed on the vision-language model. During fine-tuning, by designing training data in an interactive dialogue format, the vision-language model can experience multiple alternating perceptions and actions in a single round of dialogue. When constructing

training samples, instead of only providing a picture for the vision-language model to answer where to click, a complete task dialogue is simulated, for example: "User: Please perform login on the App interface. AI: {Image 1 description} I see the login interface. User: OK, please click the login button. AI: <action tap ...> Clicked the login button. {Image 2 description} A login success page is displayed."

[0090] Through such an interleaved format, the vision-language model can learn to process multiple rounds of perception and action flows in a context. In specific implementation, two training modes are mixed: one is action-vision: the action of the previous step and the current screenshot are interleaved as input in each step, so that the vision-language model only predicts the loss of the action part; the other is action-question-and-answer: some originally single-step questions are connected into multiple rounds of dialogue to improve training efficiency. The combined use of these two methods significantly improves the memory and consistency of the vision-language model for long tasks. After interleaved training, the vision-language model is less likely to have context confusion when executing multi-step tasks and can correctly use the information of previous steps.

[0091] In some embodiments, parallel low-rank matrix pairs are added to the vision-language model; and during backpropagation, the weights of the vision-language model are frozen, and the parallel low-rank matrix pairs are updated.

[0092] Efficiency of model fine-tuning can be improved through LoRA (Low-Rank Adaptation) and 8-bit quantization fine-tuning. LoRA is a technique for efficiently fine-tuning large pre-trained models; its core objective is to significantly reduce the number of parameters requiring training and computational resources while maintaining or even approaching the effects of full-parameter fine-tuning. INT8 quantized inference can accelerate model operation and reduce GPU memory occupation. Quantization has minimal impact on performance yet enables a single consumer-grade GPU (Graphics Processing Unit) or even CPU (Central Processing Unit) to run the model, providing a foundation for deploying multiple instances in parallel on the cloud.

[0093] The embodiment of the present disclosure provides a vision-language model training method focused on actual operational requirements in cloud phone environments, with training data and output formats closely aligned with real application

scenarios. The model has small parameters but undergoes more targeted training, achieving higher accuracy and faster speed on many tasks. More Chinese application data is incorporated during training, enabling the model to achieve deeper understanding of details in domestic application interfaces. Furthermore, the adoption of structured instructions for output actions facilitates easier integration of model decisions with programs, reducing parsing ambiguity.

[0094] FIG. 4 shows a state iteration diagram of a method for an automated operation of a cloud phone. As shown in FIG. 4, the state iteration includes the following steps.

[0095] Step 401 includes: initializing the environment (a preparation node). During the initialization phase, task description information is parsed into a task sequence. Meanwhile, an idle cloud phone instance is allocated or launched from the resource pool to prepare the application environment to be operated.

[0096] Step 402 includes: performing knowledge base retrieval (knowledge_check node). The knowledge base stores historical experience, and experience enhancement is performed through knowledge base retrieval.

[0097] Step 403 includes: determining a hit in the knowledge base.

[0098] Step 404 includes: performing experience enhancement (knowledge_enhance node).

[0099] Step 405 includes: performing AI inference and decision-making (model node). Inference is performed using the vision-language model.

[0100] Step 406 includes: performing tool verification and parsing (tool_valid node).

[0101] Step 407 includes: evaluating a verification result (should_tool_exec_continue tool).

[0102] Step 408 includes: terminating the special tool.

[0103] Step 409 includes: executing device operations (tool node).

[0104] Step 410 includes: saving experience (knowledge_save node).

[0105] Step 411 includes: iteration control determination (should_react_continue).

[0106] To more intuitively illustrate the workflow of this solution, a specific application scenario is provided below to demonstrate the entire process of completing a

task by the automated operations of the cloud phone.

[0107] The example task may be that a user wishes to use a certain e-commerce application on a cloud phone to purchase a product, with the specific instruction described as: "Please open a certain e-commerce application, search for 'a certain model phone', add the product with the highest sales volume to the shopping cart, and take a screenshot of the shopping cart page."

[0108] This task involves cross-page, multi-step operations and requires making selections based on interface information (the product with the highest sales volume). The process of the automated operations of the cloud phone is as follows.

[0109] In a first step, a task is issued: the user submits the above natural language instruction through the system interface. The task parsing module reads that this is a composite task containing multiple steps, and thus decomposes the task into several subtasks:

[0110] 1. opening the certain e-commerce application;

[0111] 2. searching for "a certain model phone";

[0112] 3. selecting the product with the highest sales volume from the search results;

[0113] 4. adding this product to the shopping cart;

[0114] 5. opening the shopping cart page and taking a screenshot.

[0115] Meanwhile, the module identifies that the final output of the task requires a screenshot.

[0116] In a second step, an application is launched: the system selects an idle cloud phone instance and executes "open the certain e-commerce application." Priority is given to directly launching the application's Activity (an application component) through ADB. If direct positioning is not possible, the LLM may also be used to identify the desktop icon. Assuming the e-commerce application successfully launches, the cloud phone displays the home page of the e-commerce application. The device operation interface obtains the home page screenshot and passes the screenshot to the multimodal LLM Controller.

[0117] In a third step, product search is performed: the multimodal LLM Controller

now receives subtask 1 "search for 'a certain model phone'." Based on the home page screenshot, the model recognizes that there are a search box on the interface, and other content. The model outputs the action `<action type="tap" target="search box" coordinates="[x1,y1]" />`, and the system executes this click. After clicking, the page transitions to the search input state, and a keyboard appears. The LLM detects the interface change and determines that text input is required, so the next step outputs `<action type="input" target="search box" text="iPhone 13" />`. The system inputs the specified text into the search box, and sends the text. The e-commerce application begins searching, and a search result list page appears.

[0118] In a fourth step, product is selected: the multimodal LLM Controller is faced with a screenshot of the search result list and knows the task requires "select the product with the highest sales volume." Here, the model needs to read the "sales volume" information from the result list for comparison. Typically, e-commerce application list items display sales volume or comment counts. The model has experienced similar comparison tasks during training and therefore possesses certain reasoning capabilities. The model may parse the sales volume numbers of multiple result items and then find the position of the item with the largest number. Subsequently, the model outputs the corresponding click action: `<action type="tap" target="1st product item" coordinates="[x2,y2]" />` (assuming the item with the highest sales volume is the first item in the list). The system executes this click and enters the product detail page of that item.

[0119] In a fifth step, an item is added to shopping cart: The detail page finishes loading, and the LLM receives the screenshot. The subtask is "add the item to shopping cart." The model recognizes that there is an "adding an item to shopping cart" button on the page. The model outputs `<action type="tap" target="a button of adding an item to shopping cart" coordinates="[x3,y3]" />`. After the system clicks, if not logged in, a login prompt may pop up, but assuming already logged in, a "added successfully" prompt pops up after clicking. The LLM may see a prompt floating layer or red dot notification, at which point the add operation may be considered complete.

[0120] In a sixth step: the model knows the next task is "open the shopping cart page and take a screenshot." For many e-commerce applications, there appears an entry point after an item is added to the shopping cart. The LLM outputs `<action type="tap"`

target="shopping cart icon" coordinates="[x4,y4]"/> to enter the shopping cart. The page jumps to the shopping cart list.

[0121] In a seventh step, screenshot is outputted: after the shopping cart page appears, the task planning module detects that the last step requires a screenshot. In this case, instead of having the LLM click, the interface can be directly called to save the image file of the current screen screenshot. Meanwhile, the LLM is notified that the task is complete. The LLM may output "Product added to shopping cart and screenshot taken" in the log. The system returns the screenshot file and execution log to the user as results.

[0122] Throughout this process, the AI operator successfully completed a complex cross-page task, which includes following noteworthy points.

[0123] Decision autonomy: in the fourth step, the AI is instructed to select the product with the "highest sales volume", which is an action requiring understanding interface data and making comparisons. The LLM directly makes the decision in one step through visual and linguistic understanding, demonstrating its powerful semantic understanding and reasoning capabilities.

[0124] Dynamic adaptation: if an unexpected situation occurs at any step, for example, in the fifth step when clicking "add an item to shopping cart," a login interface pops up, the model recognizes the login interface and pauses the task, or continues executing the login steps. Assuming the task stops, the system may notify the user that login is required first. Such fault tolerance processing is not rigid shutdown but rather advancing as much as possible or issuing reasonable termination information.

[0125] Multimodal fusion: the example heavily relies on text and data on the UI. The model effectively fuses OCR results with visual positioning. For example, even if some product titles are very long, the model can infer which product the sales volume number corresponds to based on layout. Furthermore, for buttons with only icons (without text), the model can recognize their meaning through visual features and associate them with instruction requirements.

[0126] Speed: in actual testing, all actions in the above scenario take approximately 10 seconds. Model inference accounts for about 5 calls, each approximately 1 second, and ADB execution and page loading take about 5 seconds. Compared to manual operations that may require 20-30 seconds, the AI has achieved considerable efficiency.

If the model is switched to higher computing power, further acceleration is possible, and the experience is already close to real-time.

[0127] Logs and reports: When the system executes the above task, it records every action output by the model and the interface screenshot at that time. Ultimately, these can generate log fragments as follows:

[0128] 1 [Step1] Click search box (coordinates x1,y1) - Success

[0129] 2 [Step2] Input "a certain model phone" to search box - Success

[0130] 3 [Step3] Click 1st item of the search result (coordinates x2,y2) - Success

[0131] 4 [Step4] Click add an item to shopping cart (coordinates x3,y3) - Success

[0132] 5 [Step5] Click shopping cart icon (coordinates x4,y4) – Success

[0133] 6 [Result] Task completed, screenshot saved to image_123.png.

[0134] The report also includes screenshots of key steps. This allows users or testers to conveniently verify whether the AI correctly executed the expected operations. If there are errors, adjustments to the model or feedback can be provided accordingly.

[0135] Through this example, the automated operations of the cloud phone not only correctly complete complex tasks but also demonstrate clear advantages over existing solutions: zero code, automatic understanding, high adaptability to changes, support for Chinese environments, and good traceability.

[0136] The embodiment of the present disclosure provides a solution of the automated operations of the cloud phone based on multimodal LLM, with the following main technical effects.

[0137] First, significantly lowering automation barriers and improving development efficiency: users do not need programming skills, nor do they need to learn cumbersome test script syntax; they only need to describe requirements in natural language to automatically complete corresponding operations. This extends automation from professional testers to a broader audience. For example, product managers and operations personnel can also use this system to orchestrate mobile phone operation tasks. For enterprises, the writing speed of new test cases is significantly improved, and automation coverage of complex scenarios can be expanded. Meanwhile, since there is no need to

concern themselves with device compatibility and underlying implementation, developers can focus their energy on test logic itself, improving development efficiency by at least several times.

[0138] Second, intelligent robustness and adaptation to interface changes: cloud phone automated operation possesses human-like flexibility. When application interfaces are updated, as long as visual differences do not completely change element semantics, the model can still recognize the correct operation object (for example, slight changes in button position or color changes do not affect, because the model focuses on text and icon features). Even if unexpected pop-ups or error messages appear, reasonable measures may be taken by determining interface content, such as attempting to close pop-ups or screenshotting to record errors. This self-adaptation and fault tolerance capability far exceeds fixed scripts, greatly reducing maintenance costs.

[0139] Third, high universality and cross-application operation capability: not limited to a single application or scenario. Since the model has learned general principles of GUI operations, it can execute tasks across different applications. For example, it can implement cross-application operation workflows such as "read a message content from WeChat and then save it in the Memo application," with just one instruction to complete the execution. Furthermore, it supports any application without requiring special customization for each application. This broad applicability enables it to become a universal mobile phone automatic operation platform, providing underlying support for various purposes such as testing, RPA, and assistance.

[0140] Fourth, outstanding performance in Chinese environments: considering domestic scenarios, optimization for Chinese is performed in both model selection and training. Special enhancement for Chinese in OCR and language understanding improves the accuracy of Chinese interface text recognition, and the model can correctly understand Chinese UI terminology such as "favorite," "like," and "pay." Furthermore, the model itself masters Chinese semantics, performing with ease in understanding user instructions (Chinese descriptions) and generating Chinese output (such as answer content), ensuring barrier-free processing of Chinese information from instructions to interfaces. For complex fonts, mixed traditional and simplified characters, and English abbreviations commonly seen in domestic applications, robustness can also be improved through data coverage training.

[0141] Fifth, resource efficiency and scalable cloud deployment: the adopted model has only about 2.5B parameters and, after optimization, can run on consumer-grade GPUs or even CPUs. Deployment costs and operating costs are significantly reduced, enabling the parallel launch of a large number of instances on the cloud to serve multiple users or multiple test tasks. For example, a server equipped with sufficient GPUs can run dozens of cloud phone automated operation instances in parallel simultaneously, meeting the needs of CI/CD (Continuous Integration/Continuous Delivery) pipelines for concurrent test execution. On the terminal side, the miniaturization of the model also provides a foundation for future embedding in mobile devices and local private environment operation. Furthermore, the direct benefit of low resource occupation is fast response speed. This real-time experience is crucial for enhancing user confidence and expanding application scenarios (such as interactive assistants).

[0142] Sixth, open source and easy integration and expansion: Built based on open source models and open interfaces, with no dependence on specific vendor cloud services. Both the model and framework are achievements of the open source community; innovation is built upon this foundation while preserving openness. This allows users to privately deploy the system to ensure data security or further customize the model according to their own business. The system provides clear API interfaces that can be conveniently integrated with existing test frameworks, monitoring systems, or business process management platforms. For example, testers can trigger a cloud phone operation task through simple HTTP (Hypertext Transfer Protocol) calls and obtain result reports, greatly facilitating continuous integration. Furthermore, due to the modular architecture, it is easy to expand new functions, such as replacing with stronger models or adding plugins for special applications, without affecting overall operation.

[0143] Seventh, improving test quality and user experience: cloud phone automated operation not only improves efficiency but also enhances test quality. It can simulate real user behaviors, including some random or abnormal operations, which helps discover problems in unconventional use of applications. Similarly, in RPA applications, AI's autonomous determination can handle some edge cases, reducing process hang-ups. Meanwhile, because the operation process has detailed records and screenshots, problem traceability is easier, and errors can be quickly located when they occur. For end users, cloud phone automated operation can also be used as a personal mobile phone assistant

to enhance user experience. It can understand vague high-level requirements and complete specific affairs, truly achieving "what you see is what you get" and "what you describe is what you get," representing a leap in human-computer interaction methods.

[0144] With further reference to FIG. 5, as an implementation of the methods shown in the above figures, the present disclosure provides an embodiment of an apparatus for an automated operation of a cloud phone, which corresponds to the method embodiment shown in FIG. 2, and the apparatus can be specifically applied to various electronic devices.

[0145] As shown in FIG. 5, the apparatus 500 for an automated operation of a cloud phone of this embodiment may include: an obtaining module 501, an operation module 502 and an execution module 503. The obtaining module 501 is configured to obtain a task sequence of the cloud phone; the operation module 502 is configured to, for a current task in the task sequence, perform the following operation steps: generate a current operation instruction by using a vision-language model to reason about the current task and current interface information of the cloud phone; after executing the current operation instruction on the cloud phone, load next interface information; and in response to the current task being the last task of the task sequence, determine that the automated operation of the cloud phone is completed; and the execution module 503 is configured to, in response to the current task not being the last task of the task sequence, take a next task in the task sequence as a new current task, take the next interface information as new current interface information, and continue to perform the operation steps.

[0146] In this embodiment, for the apparatus 500 for automated operation of a cloud phone: the specific processing of the obtaining module 501, the operation module 502 and the execution module 503 and the technical effects brought thereby may refer to the relevant descriptions of Steps 201-203 in the corresponding embodiment of FIG. 2 respectively, and will not be repeated here.

[0147] In some optional implementations of this embodiment, the obtaining module 501 is further configured to: receive task description information; and parse the task description information to generate a task sequence.

[0148] In some optional implementations of this embodiment, the operation module

502 is further configured to: input a preset prompt, the current task and the current interface information into the vision-language model, and output a current operation instruction.

5 **[0149]** In some optional implementations of this embodiment, the vision-language model includes a vision sub-model and a language sub-model; and the operation module 502 is further configured to: input the current interface information into the vision sub-model, and output object information on the current interface; input the object information on the current interface and the current task into the language sub-model, and output object information corresponding to the current task; and generate a current operation instruction based on the object information corresponding to the current task.

10 **[0150]** In some optional implementations of this embodiment, the apparatus 500 for an automated operation of a cloud phone further includes a correction module configured to perform automatic correction based on the next interface information in response to the next interface information not meeting a preset condition.

15 **[0151]** In some optional implementations of this embodiment, the correction module is further configured to: determine an error type based on the next interface information, and perform correction by using a correction method corresponding to the error type.

20 **[0152]** With further reference to FIG. 6, as an implementation of the methods shown in the above figures, the present disclosure provides an embodiment of a training apparatus for a vision-language model, which corresponds to the method embodiment shown in FIG. 3, and the apparatus can be specifically applied to various electronic devices.

25 **[0153]** As shown in FIG. 6, the training apparatus 600 for a vision-language model of this embodiment may include: an obtaining module 601 and a training module 602. The obtaining module 601 is configured to obtain a first training sample and a second training sample of a sample application, where the first training sample includes a previous task of the sample application and a current interface screenshot, and the second training sample includes multiple rounds of dialogue corresponding to a current task of the application; and the training module 602 is configured to perform interleaved training on the vision-language model by using the first training sample and the second training sample until the vision-language model meets a preset condition.

[0154] In this embodiment, for the training apparatus 600 for a vision-language model: the specific processing of the obtaining module 601 and the training module 602 and the technical effects brought thereby may refer to the relevant descriptions of Steps 301-302 in the corresponding embodiment of FIG. 3 respectively, and will not be repeated here.

[0155] In some optional implementations of this embodiment, the training apparatus 600 for a vision-language model further includes a preprocessing module configured to: divide the current interface screenshot into multiple grid cells; and cluster the multiple grid cells to remove grid cells corresponding to duplicate background regions.

[0156] In some optional implementations of this embodiment, the training apparatus 600 for a vision-language model further includes a generation module configured to: select interface elements from a user interface tree of the sample application, and automatically generate tasks corresponding to the interface elements through scripts; capture a current interface screenshot of the sample application after executing the tasks corresponding to the interface elements in the sample application; and generate a first training sample based on the tasks corresponding to the interface elements and the current interface screenshot.

[0157] In some optional implementations of this embodiment, the training module 602 is further configured to: add parallel low-rank matrix pairs to the vision-language model; and freeze the weights of the vision-language model and update the parallel low-rank matrix pairs during backpropagation.

[0158] In the technical solution of the present disclosure, the acquisition, storage and application of the involved user personal information all comply with the provisions of relevant laws and regulations and do not violate public order and good customs.

[0159] According to the embodiments of the present disclosure, the present disclosure also provides an electronic device, a readable storage medium and a computer program product.

[0160] FIG. 7 shows a schematic block diagram of an example electronic device 700 that can be used to implement the embodiments of the present disclosure. The electronic device is intended to represent various forms of digital computers, such as laptop computers, desktop computers, workstations, personal digital assistants, servers, blade

servers, mainframe computers, and other suitable computers. The electronic device may also represent various forms of mobile devices, such as personal digital processors, cellular phones, smart phones, wearable devices, and other similar computing devices. The components shown herein, their connections and relationships, and their functions are merely examples, and are not intended to limit the implementation of the present disclosure described and/or claimed herein.

[0161] As shown in FIG. 7, the device 700 includes a computing unit 701, which can perform various appropriate actions and processes according to a computer program stored in a read-only memory (ROM) 702 or a computer program loaded from a storage unit 708 into a random access memory (RAM) 703. Various programs and data required for the operation of the device 700 can also be stored in the RAM 703. The computing unit 701, the ROM 702 and the RAM 703 are connected to each other through a bus 704. An input/output (I/O) interface 705 is also connected to the bus 704.

[0162] Multiple components in the device 700 are connected to the I/O interface 705, including: an input unit 706 such as a keyboard, a mouse and the like; an output unit 707 such as various types of displays, speakers and the like; a storage unit 708 such as a magnetic disk, an optical disk and the like; and a communication unit 709 such as a network card, a modem, a wireless communication transceiver and the like. The communication unit 709 allows the device 700 to exchange information/data with other devices through a computer network such as the Internet and/or various telecommunication networks.

[0163] The computing unit 701 may be various general-purpose and/or special-purpose processing components with processing and computing capabilities. Some examples of the computing unit 701 include, but are not limited to, a central processing unit (CPU), a graphics processing unit (GPU), various dedicated artificial intelligence (AI) computing chips, various computing units running machine learning model algorithms, a digital signal processor (DSP), and any appropriate processor, controller, microcontroller, and the like. The computing unit 701 executes the various methods and processes described above, such as the method for an automated operation of a cloud phone. For example, in some embodiments, the method for the automated operation of the cloud phone may be implemented as a computer software program tangibly embodied in a machine-readable medium, such as the storage unit 708. In some

embodiments, part or all of the computer program may be loaded and/or installed on the device 700 via the ROM 702 and/or the communication unit 709. When the computer program is loaded into the RAM 703 and executed by the computing unit 701, one or more steps of the method for an automated operation of a cloud phone described above may be executed. Alternatively, in other embodiments, the computing unit 701 may be configured to perform the method for the automated operation of the cloud phone by any other appropriate means (for example, by means of firmware).

[0164] Various implementations of the systems and technologies described herein may be implemented in digital electronic circuit systems, integrated circuit systems, field programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), application specific standard products (ASSPs), systems on chips (SOCs), complex programmable logic devices (CPLDs), computer hardware, firmware, software, and/or combinations thereof. These various implementations may include: being implemented in one or more computer programs that can be executed and/or interpreted on a programmable system including at least one programmable processor, which may be a special-purpose or general-purpose programmable processor that can receive data and instructions from a storage system, at least one input device, and at least one output device, and transmit data and instructions to the storage system, the at least one input device, and the at least one output device.

[0165] Program code for implementing the methods of the present disclosure may be written in any combination of one or more programming languages. These program codes may be provided to a processor or controller of a general-purpose computer, a special-purpose computer, or other programmable data processing apparatus, such that the program codes, when executed by the processor or controller, cause the functions/operations specified in the flowcharts and/or block diagrams to be implemented. The program code may be executed entirely on a machine, partially on a machine, as a stand-alone software package partially on a machine and partially on a remote machine, or entirely on a remote machine or server.

[0166] In the context of the present disclosure, a machine-readable medium may be a tangible medium that can contain or store a program for use by or in connection with an instruction execution system, apparatus or device. The machine-readable medium may be a machine-readable signal medium or a machine-readable storage medium. The

machine-readable medium may include, but is not limited to, an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system, apparatus or device, or any suitable combination of the above. More specific examples of the machine-readable storage medium will include an electrical connection based on one or more wires, a portable computer disk, a hard disk, a random access memory (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM or flash memory), an optical fiber, a portable compact disk read-only memory (CD-ROM), an optical storage device, a magnetic storage device, or any suitable combination of the above.

[0167] To provide interaction with a user, the systems and technologies described herein may be implemented on a computer having: a display device (for example, a CRT (cathode ray tube) or LCD (liquid crystal display) monitor) for displaying information to the user; and a keyboard and a pointing device (for example, a mouse or a trackball) through which the user can provide input to the computer. Other types of devices may also be used to provide interaction with the user; for example, the feedback provided to the user may be any form of sensory feedback (for example, visual feedback, auditory feedback, or tactile feedback); and input from the user may be received in any form, including acoustic input, speech input, or tactile input.

[0168] The systems and technologies described herein may be implemented in a computing system that includes a back-end component (e.g., as a data server), a middleware component (e.g., an application server), or a front-end component (e.g., a user computer with a graphical user interface or a web browser, through which a user may interact with implementations of the systems and technologies described herein), or a computing system that includes any combination of such back-end, middleware, or front-end components. The components of the system may be interconnected by any form or medium of digital data communication (e.g., a communication network). Examples of communication networks include a local area network (LAN), a wide area network (WAN), and the Internet.

[0169] A computer system may include a client and a server. The client and server are generally remote from each other and typically interact through a communication network. The relationship of client and server arises by virtue of computer programs running on the respective computers and having a client-server relationship to each other.

The server may be a cloud server, a server of a distributed system, or a server integrated with blockchain technology.

[0170] It should be understood that the steps of reordering, adding or deleting may be performed using the various forms shown above. For example, the steps described in the present disclosure may be performed in parallel or sequentially or in a different order, so long as the desired results of the technical solution disclosed in the present disclosure can be realized, and no limitation is imposed herein.

[0171] The foregoing detailed description is not intended to limit the scope of the present disclosure. It will be appreciated by those skilled in the art that various modifications, combinations, sub-combinations, and substitutions may be made depending on design requirements and other factors. Any modifications and equivalents, that fall within the spirit and principles of the disclosure are intended to be included within the scope of protection of the disclosure.

WHAT IS CLAIMED IS:

1. A method for an automated operation of a cloud phone, comprising:

obtaining a task sequence of the cloud phone;

for a current task in the task sequence, performing operation steps including:
5 generating a current operation instruction by using a vision-language model to reason about the current task and current interface information of the cloud phone; after executing the current operation instruction on the cloud phone, loading next interface information; and in response to the current task being the last task of the task sequence, determining that the automated operation of the cloud phone is completed; and

10 in response to the current task not being the last task of the task sequence, taking a next task in the task sequence as a new current task, taking the next interface information as new current interface information, and continuing to perform the operation steps.

2. The method according to claim 1, wherein the obtaining the task sequence comprises:

15 receiving task description information; and

parsing the task description information to generate the task sequence.

3. The method according to claim 1, wherein the generating the current operation instruction by using the vision-language model to reason about the current task and the
20 current interface information of the cloud phone comprises:

inputting a preset prompt, the current task, and the current interface information into the vision-language model, and outputting the current operation instruction.

4. The method according to claim 1, wherein the vision-language model comprises a vision
25 sub-model and a language sub-model; and

the generating the current operation instruction by using the vision-language model to reason about the current task and the current interface information of the cloud phone comprises:

inputting the current interface information into the vision sub-model, and outputting object information on the current interface;

inputting the object information on the current interface and the current task into the language sub-model, and outputting object information corresponding to the current task; and

generating the current operation instruction based on the object information corresponding to the current task.

5. The method according to claim 1, wherein the method further comprises:

in response to the next interface information not satisfying a preset condition, performing automatic correction based on the next interface information.

6. The method according to claim 5, wherein the performing automatic correction based on the next interface information comprises:

determining an error type based on the next interface information, and performing correction using a correction method corresponding to the error type.

7. A vision-language model training method, comprising:

obtaining a first training sample and a second training sample of a sample application, wherein the first training sample comprises a previous task of the sample application and a current interface screenshot, and the second training sample comprises multi-turn dialogue corresponding to a current task of the application; and

performing interleaved training on the vision-language model using the first training

sample and the second training sample until the vision-language model satisfies a preset condition.

8. The method according to claim 7, wherein the method further comprises:

5 dividing the current interface screenshot into a plurality of grid cells;
 performing clustering on the plurality of grid cells, and removing grid cells
corresponding to duplicate background regions.

9. The method according to claim 7, wherein the method further comprises:

10 selecting interface elements from a user interface tree of the sample application, and
automatically generating tasks corresponding to the interface elements through scripts;
 after executing the tasks corresponding to the interface elements in the sample
application, capturing a current interface screenshot of the sample application; and
 generating the first training sample based on the tasks corresponding to the interface
15 elements and the current interface screenshot.

10. The method according to claim 7, wherein the performing interleaved training on the
vision-language model using the first training sample and the second training sample
comprises:

20 adding parallel low-rank matrix pairs to the vision-language model; and
 during backpropagation, freezing weights of the vision-language model, and updating
the parallel low-rank matrix pairs.

11. An apparatus for an automated operation of a cloud phone, comprising:

25 an obtaining module configured to obtain a task sequence of the cloud phone;

an operation module configured to, for a current task in the task sequence, perform operation steps including: generating a current operation instruction by using a vision-language model to reason about the current task and current interface information of the cloud phone; after executing the current operation instruction on the cloud phone, loading
5 next interface information; and in response to the current task being the last task of the task sequence, determining that the automated operation of the cloud phone is completed; and

an execution module configured to, in response to the current task not being the last task of the task sequence, take a next task in the task sequence as a new current task, take
10 the next interface information as new current interface information, and continue to perform the operation steps.

12. A vision-language model training apparatus, comprising:

an obtaining module configured to obtain a first training sample and a second training
15 sample of a sample application, wherein the first training sample comprises a previous task of the sample application and a current interface screenshot, and the second training sample comprises multi-turn dialogue corresponding to a current task of the application; and

a training module configured to perform interleaved training on the vision-language
20 model using the first training sample and the second training sample until the vision-language model satisfies a preset condition.

13. An electronic device, comprising:

at least one processor; and

25 a memory communicatively connected to the at least one processor; wherein the memory stores instructions executable by the at least one processor, and the

instructions, when executed by the at least one processor, enable the at least one processor to perform the method according to any one of claims 1-6 or 7-10.

5 14. A non-transitory computer-readable storage medium storing computer instructions, wherein the computer instructions are used to cause a computer to perform the method according to any one of claims 1-6 or 7-10.

10 15. A computer program product comprising a computer program, wherein the computer program, when executed by a processor, implements the method according to any one of claims 1-6 or 7-10.