TOWARDS AUTONOMOUS AGENTS: ADAPTIVE PLANNING, REASONING, AND ACTING IN LANGUAGE MODELS

Anonymous authors

Paper under double-blind review

ABSTRACT

We propose a novel in-context learning algorithm for building autonomous decision-making language agents. The language agent continuously attempts to solve the same task by reasoning, acting, observing and then self-correcting each time the task fails. Our selected language agent demonstrates the ability to solve tasks in a text-based game environment. Our results show that the gemma-2-9b-it language model, using our proposed method, can successfully complete two of six tasks that failed in the first attempt. This highlights the effectiveness of our approach in enhancing the problem-solving capabilities of a single language model through self-correction, paving the way for more advanced autonomous agents. The code is publicly available at https://anonymous.4open.science/r/AutonomousLLMAgentwithAdaptingPlanning-D613/.

023

005 006

007

008 009 010

011

013

014

015

016

017

018

019

021

1 INTRODUCTION

025 026

Large language models (LLMs) are large statistical models that predict the next word, phrase, sentence, or paragraph based on a given input (Demszky et al., 2023). The quality of the output from a language model can be heavily influenced by the input prompt it receives (Arvidsson & Axell, 2023). One of the capabilities of LLMs is in-context learning, where they learn a new task from a small set of exemplars provided in the prompt during inference (Minaee et al., 2024). Prompt engineering is the process of designing and refining input prompts to elicit desired responses from LLMs (Ekin, 2023).

034 In Chain-of-Thought (CoT) (Wei et al., 2022), given a prompt with exemplars that include an input part and an output part, a chain of thought consists of a series of intermediate natural language rea-035 soning steps added between the input and output parts in each exemplar to produce the final output. 036 However, the CoT prompting doesn't have the ability to update its knowledge from the external 037 world. ReAct (Yao et al., 2023) prompting addresses the problem by providing the language model 038 with a prior language description to guide its reasoning about solving diverse language reasoning and decision making tasks and adapting this reasoning by acting on and receiving the feedback from 040 the external world. In Reflexion (Shinn et al., 2023), they proposed autonomous decision-making 041 LLM agents by adding a reflection step to the CoT or ReAct prompt to adjust the reasoning, facilitat-042 ing language agents' learning from prior failings through verbal reinforcement. VOYAGER (Wang 043 et al., 2023) is a LLM based agent designed to explore an open-ended world and attain diverse 044 skills through the integration of automatic curriculum, skill library management, and an iterative 045 prompting mechanism incorporating environmental feedback, execution errors, and self-verification to enhance program performance. In Motif (Klissarov et al., 2024), an agent is trained through re-046 inforcement learning to maximize rewards from a parameterized model, which is trained based on 047 preferences selected by a language model over pairs of actions, aimed at achieving a specific goal in 048 a given environment. 049

In this study, we propose Self-Adaptive Language Agent (SALA), which is an adaptive decision making language agent that self-adjusts the reasoning process of ReAct with a correction mechanism
 from Reflexion. Here, we let the language model agent adapt its own policy by correcting its previous failure using its internal knowledge. The proposed SALA differs from Reflexion (Shinn et al., 2023), which uses two LLMs: one for action generation and another for reflection whereas SALA

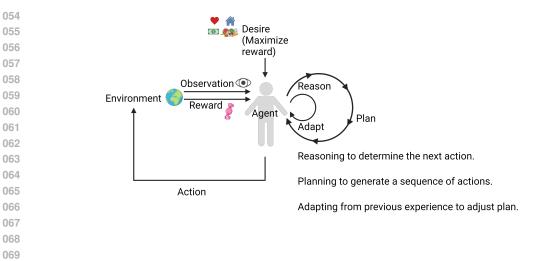


Figure 1: An architecture towards autonomous agent. Created with BioRender.com.

employs a single LLM that can self-adapt its reasoning and acting behavior making it an autonomous language agent. Our experimental results show that in twelve different decision making tasks from the ALFWorld (Shridhar et al., 2020) environment, the proposed SALA achieves a success rate of approximately 83%, which is higher than the ReAct-based agent, which achieved a success rate of 67%. In addition, the SALA could solve two tasks that couldn't be completed in the first trail, demonstrating the effectiveness of our approach.

2 Methods

081 082

070

071

073

074

075

076

077

079

Let a text game be denoted as a function f that maps the state $s \in \mathbb{V}$ and the action $a \in \mathbb{V}$ to an observation $o \in \mathbb{V}$, where \mathbb{V} is a set of vocabulary. Let π_{θ} be an LLM agent over a pre-trained set of parameters θ . Let s_0 be the initial state of the environment f, we aim to produce a sequence of actions $(a_0, a_1, a_2, ...)$, where $a_i \in \mathbb{V}$ for $i \in \mathbb{Z}$, to change the state to a terminal state that indicates the game is cleared.

In ReAct prompting (Yao et al., 2023), they propose to use an LLM to produce the thought $(t_0, t_1, t_2, ...)$, where $t_i \in \mathbb{V}$ for $i \in \mathbb{Z}$, by $t_i \sim \pi_{\theta}(t_i|s_i)$, where $s_i = \{s_{i-1}, t_{i-1}, a_{i-1}, o_i\}$ for $i \in \mathbb{Z}^+$, $a_i \sim \pi_{\theta}(a_i|s_i, t_i)$ for $i \in \mathbb{Z}$, and $o_{i+1} = f(s_i, a_i)$ for $i \in \mathbb{Z}$.

091 The limitation of ReAct prompting (Yao et al., 2023) is that complex tasks with a large action space 092 require more demonstrations to learn effectively. The LLM may produce incorrect actions that do not 093 lead to task completion. Reflexion (Shinn et al., 2023) addresses this problem by using an additional 094 LLM to iteratively provide reflection text that will be added to the ReAct prompt for improvement. More specifically, for each trial $ep \in \mathbb{Z}^+$, if $ep \ge N$, where N is a maximum number for each trail, a self-reflection r^{ep} is generated and a new state $s_0^{ep+1} = \{s_N^{ep+1}, r^{ep}\}$ is formed to be used 096 as the initial state in the next trial ep + 1. However, the method in Reflexion (Shinn et al., 2023) 098 necessitates two LLMs, where one LLM is used to generate the thought or action, and another LLM 099 is used to generate the reflection. We will modify this by using a single LLM to generate thought, action, and self-adaptation, which is the correction from the previous failed trial. 100

The architecture of the main idea of our work is shown in Figure 1. A desire is provided to an agent to motivate it to solve a specific task in a given environment. The agent can perform an action to interact with the environment, causing the state of the environment to change. The agent then receives an observation that describes the status of the environment and a reward signal. The action may be proposed from three different processes: the reasoning process determines the next action based on the current progress; the planning process proposes a series of actions that can be used to solve a specific task; and the adaptation process summarizes previous progress to provide a better plan towards maximizing the reward.

108 We present a novel algorithm in Algorithm 1. Initially, we have the initial state s_0 which provides 109 instructions, presents exemplars, and describes the environment and the goal for a specific task. 110 π_{θ} is an LLM agent with a set of parameters θ . $\tau = \{s_0, t_0, a_0, o_1, \ldots\}$ is a sequence of the 111 concatenation of state, thought, action, and observation, where s_k , t_k , a_k , and o_k are sequences of tokens representing the k-th state, thought, action, and observation for $k \in \mathbb{Z}$, respectively. The 112 return $R(\tau)$ is a string indicating whether the task is completed or not. ep is a variable indicating 113 the number of trials. The environment is reinitialized at each trial. 114

115 At the first time step k = 0, the thought is sampled from

116 117

121

132

142

143

$$_{0}^{1} \sim \pi_{\theta}(t_{0}^{1}|s_{0}^{1}),$$
 (1)

where the subscript 0 indicates the first time step, and the superscript 1 indicates the first trail. t_0^1 118 represents the first thought in the first trial, and s_0^1 represents the first state in the first trial, with both 119 being sequences of tokens. The action is then sampled from 120

$${}^{1}_{0} \sim \pi_{\theta}(a^{1}_{0}|s^{1}_{0}, t^{1}_{0}),$$
 (2)

122 where a_0^1 represents the first action in the first trial, and the action is a sequence of tokens. The 123 second observation in the first trial, o_1^1 , is a sequence of tokens obtained by executing the action a_0^1 124 in the environment f at state s_0^1 as 125

$$o_1^1 = f(s_0^1, a_0^1). (3)$$

126 A new state s_1^1 is formed by concatenating the thought t_0^1 , action a_0^1 , and observation a_1^1 after state 127 s_0^1 as 128

$$s_1^1 = \{s_0^1, t_0^1, a_0^1, o_1^1\}.$$
(4)

129 If a maximum time step is reached, the task fails and the return $R(\tau)$ is concatenated with "New 130 plan: ". They are concatenated after the current state of the environment s_{50}^1 to form the initial state 131 in the next trial s_0^2 as

$$s_0^2 = \{s_{50}^1, R(\tau)\},\tag{5}$$

133 where $\tau = \{s_0, t_0, a_0, o_1, t_1, a_1, o_2, t_2, \dots, o_{50}\}$. In the next trial, the first thought in the second trial, 134 t_0^2 , is sampled from the LLM by 135

$$t_0^2 \sim \pi_\theta(t_0^2|s_0^2),$$
 (6)

136 We call the initial thought t_0^{ep} at the *ep*-th trial for ep > 1 as the adaptation from the (ep - 1)-th 137 trail and t_0^{ep} indicates the correction of the (ep - 1)-th failed trail to improve the next trail. In the 138 next step, we propose to replace the initial state in the second trial with the initial state in the first 139 trail to reduce the context length. We call this step compression. By performing compression, the 140 first action in the second trail will only be conditioned on the initial state in the first trail s_0^1 and the 141 adaptation from the first trail t_0^2 as

$$a_0^2 \sim \pi_\theta(a_0^2 | s_0^2, t_0^2).$$
 (7)

144 Algorithm 1 Self-Adaptive Language Agent 145 Initialize the world state s_0 as a text of exemplars and task, where each token $\in Vocab$. 146 Let π_{θ} be a LLM agent over a pre-trained set of parameters θ . 147 Let a trajectory $\tau = \{s_0, t_0, a_0, o_1, \dots\}$ be a sequence of state, thought, action, and observation. 148 Let $R(\tau)$ be the return for trajectory τ . Let ep = 1. 149 While $R(\tau) \neq "OK"$ do 150 Let k = 0. 151 While $k < 50 || R(\tau) = "OK"$ do 152 Generate thought $t_k^{ep} \sim \pi_{\theta}(t_k^{ep}|s_k^{ep})$. 153 Compression step: If k = 0, then $s_0^{ep} = s_0$. Generate action $a_{kp}^{ep} \sim \pi_{\theta}(a_{kp}^{ep}|s_k^{ep}, t_k^{ep})$. Get observation $o_{k+1}^{ep} = f(s_k^{ep}, a_k^{ep})$. Let $s_{k+1}^{ep} = \{s_k^{ep}, t_k^{ep}, a_k^{ep}, o_{k+1}^{ep}\}$. k := k + 1154 157 159 Concatenate $R(\tau)$ with "New plan: ". $s_0^{ep+1} = \{s_k^{ep}, R(\tau)\} \\ ep := ep + 1$ 161

¹⁶² 3 THE ALFWORLD ENVIRONMENT

163 164

There are six types of tasks in the ALFWorld environment (Shridhar et al., 2020): Pick and Place, Examine in Light, Clean and Place, Heat and Place, Cool and Place, and Pick Two and Place. For each task, a description of available receptacles is given in the first part of the instruction as follows: You are in the middle of a room. Looking quickly around you, you see a {recep1 id}, a {recep2 id}, ..., and a {recepN id}, where recepN refers to the Nth receptacles like drawers or cabinet and id $\in \mathbb{Z}^+$. An example is shown in the text in Figure 2 with a green background.

170 After the description of the available receptacles, the goal instructions are provided based on the six 171 different task types. For a Pick and Place task, the instruction will be either "put a obj in recap" or 172 "put some obj on recap". For an Examine in Light task, it will be either "look at obj under the lamp" 173 or "examine the obj with the lamp". For a Clean and Place task, it will be either "put a clean obj in 174 recap" or "clean some obj and put it in recap". For a Heat and Place task, it will be either "put a hot obj in recap" or "heat some obj and put it in recap". For a Cool and Place task, it will be either "put 175 a cool obj in recap" or "cool some obj and put it in recap". For a Pick Two and Place task, it will 176 be either "put two obj in recap" or "find two obj and put them in recap". Inside the curly brackets, 177 {obj}, {recep} and {lamp} refer to object, receptacle, and lamp classes, respectively. An example 178 is shown in the text in Figure 2 with a red background. 179

After the goal instruction of the task, the agent or the user can interact with the game environment using the following nine different text actions: go to recap id, open recap id, clean obj id with recap id, take obj id from recap id, close recap id, close recap id, heat obj id with recap id, put obj id in/on recap id, and use recap id. Given the action, the game environment will return text observations accordingly. An example is shown in the text in Figure 2 with a magenta background.

- 185
- 186
- 187 188

189

4 EXPERIMENTAL DESIGNS AND RESULTS

4.1 EXPERIMENT ON REACT PROMPTING IN THE ALFWORLD ENVIRONMENT

190 In ReAct (Yao et al., 2023), they randomly annotate the trajectories for each task type. Each trajectory includes sparse thoughts that decompose the goal, track subgoal completion, determine the 191 next subgoal, and use commonsense reasoning to find an object and determine what to do with it. 192 An exemplar used in ReAct (Yao et al., 2023) is shown in Figure 3. The instruction text, "Interact 193 with a household to solve a task. Here are two examples.", is added above the exemplars to indicate 194 that the general goal is to complete a household task, and the texts below it include two exemplars. 195 Following the examples, the text 'Here is the task.' is concatenated, followed by the description of 196 available receptacles and the description of the goal instruction from the ALFWorld environment 197 (Shridhar et al., 2020). Finally, the greater-than symbol ('>') is added to indicate that the language 198 model should generate text after it, starting with actions or thoughts. 199

ReAct (Yao et al., 2023) evaluated their method on 134 tasks in the ALFWorld environment (Shrid-200 har et al., 2020), achieving a success rate of 70.9% using PaLM-540B (Chowdhery et al., 2023) and 201 78.4% using the GPT-3 text-davinci-002 model (Brown et al., 2020), where each output token is se-202 lected with the highest probability. However, PaLM-540B (Chowdhery et al., 2023) is not publicly 203 available, and the text-davinci-002 model (Brown et al., 2020) was shut down by OpenAI on January 204 4th, 2024. Due to the difficulty in reproducing the results from (Yao et al., 2023), we tested various 205 open-source LLMs and selected the one with the highest success rate to develop our method. The 206 results in Table 1 show that the gemma-2-9b-it model outperforms other models with a success rate of 62% in a 12-hour run. We will use gemma-2-9b-it for the subsequent experiments. 207

208 We observed that gemma-2-9b-it model outperforms other models (gemma-2-9b, Mistral-7B-v0.3, 209 Mistral-7B-Instruct-v0.3, Llama-2-7b-hf, Phi-3-medium-128k-instruct, deepseek-llm-7b-base, and 210 zephyr-7b-alpha) with a success rate of 40% in solving various tasks in the ALFWorld environment 211 (Shridhar et al., 2020) using ReAct prompting (Yao et al., 2023). We identified three common issues 212 with these LLMs. First, the LLMs may attempt to retrieve an object from a location where the object 213 does not exist, repeatedly performing the same action until reaching the maximum number of steps, 50. Second, the LLMs may select an incorrect item. For instance, when the task is to "put a clean 214 cloth in countertop", the gemma-2-9b model may pick up a "handtowel" instead. Subsequently, the 215 LLM cannot complete the task by cleaning the "handtowel", resulting in repeated attempts to clean

216		
217	You are in the middle of a room. Looking quickly around you,	
	you see a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a	(1) A description of available receptacles
218	cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 3, a	(1) A description of available receptacies
219	countertop 2, a countertop 1, a drawer 3, a drawer 2, a	
220	drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a shelf	
221	3, a shelf 2, a shelf 1, a sinkbasin 1, a stoveburner 4, a	
222	stoveburner 3, a stoveburner 2, a stoveburner 1, and a	
223	toaster 1.	
224		
	Your task is to: cool some pan and put it in countertop.	(2) A description of the goal instruction
225		
226	<mark>> go to stoveburner 1</mark>	
227	On the stoveburner 1, you see a pot 1.	
228	<mark>> go to stoveburner 2</mark>	
229	On the stoveburner 2, you see a pan 1.	
	> take pan 1 from stoveburner 2	
230	You pick up the pan 1 from the stoveburner 2.	
231	> go to fridge 1	(3) A sequence of actions and observations.
232	The fridge 1 is closed	
233	> cool pan 1 with fridge 1	
234	You cool the pan 1 using the fridge 1.	
235	<mark>> go to countertop 1</mark>	
236	On the countertop 1, you see a mug 1.	
237	<mark>> put pan 1 in/on countertop 1</mark>	
	You put the pan 1 in/on the countertop 1.	
238		

Figure 2: A trajectory in the ALFWorld environment (Shridhar et al., 2020). The text in the black box is composed of the description of available receptacles, the description of the goal instruction, and a sequence of actions and observations. (1) The description of available receptacles is listed in the first part of the text with a green background. (2) The description of the goal instruction with a red background shows that the task is a Cool and Place task, and the goal is to cool some pan and put it in countertop. (3) The sequence of actions and observations with a magenta background shows the actions performed and the corresponding observations from the environment. The actions are in boldface after the greater than symbol, and the observations are in regular text below each action.

Table 1: Success rate of different open-source language models with ReAct (Yao et al., 2023) applied

Success

rate (%)

40

62

28

25

8

41

14

15

Number of

success tasks

25

37

16

14

4

12

15

8

number of

tasks

62

59

57

54

50

29

103

52

to the ALFWorld environment (Shridhar et al., 2020) in a 12-hour run for each model

Name of the language model

gemma-2-9b

gemma-2-9b-it

Mistral-7B-v0.3

Mistral-7B-Instruct-v0.3

Llama-2-7b-hf

Phi-3-medium-128k-instruct

deepseek-llm-7b-base

zephyr-7b-alpha

247 248

249

239

240

241

242

243

244

245

246

250 251

252

253

Developer

Google

Google

Mistral AI

Mistral AI

LLaMA

Microsoft

DeepSeek

Hugging Face H4

254

256

257 258

- 7	

261

262

263 264



and place the "handtowel" in the "countertop" until the maximum step limit is reached. Third, some
LLMs will misinterpret the order of sub-goals. For example, when the goal is to "examine the cd
with the desklamp", the gemma-2-9b-it model erroneously attempts to retrieve a "desklamp" first.
After failing to obtain the "desklamp", the LLM searches for the "cd" but only revisits previously
searched locations instead of exploring new ones.

5

270	You are in the middle of a room. Looking quickly around you,	
271	you see a cabinet 13, a cabinet 12, a cabinet 11, a cabinet	
272	10, a cabinet 9, a cabinet 8, a cabinet 7, a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a	
273	coffeemachine 1, a countertop 1, a diningtable 1, a drawer 1,	
274	a fridge 1, a garbagecan 1, a microwave 1, a shelf 3, a shelf 2,	
275	a shelf 1, a sinkbasin 1, a stoveburner 4, a stoveburner 3, a	
	stoveburner 2, a stoveburner 1, and a toaster 1.	
276	Your task is to: put a hot apple in fridge.	
277		
278	> think: To solve the task, I need to find and take an apple,	Decompose the goal
279	then heat it with microwave, then put it in fridge. OK.	
280	> think: First I need to find an apple. An apple is more likely	
281	to appear in fridge (1), diningtable (1), coffeetable (1),	Reason via commonsense
282	drawer (1), cabinet (1-13), garbagecan (1). I can check one by one, starting with fridge 1.	
	OK.	
283	> go to fridge 1	
284	The fridge 1 is closed.	
285	> open fridge 1 You open the fridge 1. The fridge 1 is open. In it, you see a	
286	cup 1, and a egg 1.	
287	> go to diningtable 1	
288	On the diningtable 1, you see a apple 1, a bread 1, a fork 3, a fork 2, a fork 1, a mug 2, a peppershaker 3, a plate 2, a pot	
289	1, a soapbottle 3, a spatula 1, a spoon 3, a spoon 2, a spoon	
	1, a winebottle 3, and a winebottle 2.	
290	> think: Now I find an apple (1). Next, I need to take it.	Track subgoal completion and
291	OK. > take apple 1 from diningtable 1	determine the next subgoal
292	You pick up the apple 1 from the diningtable 1.	
293	> think: Now I take an apple (1). Next, I need to go to a	
294	microwave (1) and heat it. OK.	
295	> go to microwave 1	
296	The microwave 1 is closed.	
	> heat apple 1 with microwave 1	
297	You heat the apple 1 using the microwave 1. > think: Now I heat an apple (1). Next, I need to put it in/on	
298	fridge 1.	
299	ок.	
300	> go to fridge 1	
301	The fridge 1 is open. In it, you see a cup 1, and a egg 1. > put apple 1 in/on fridge 1	
302	You put the apple 1 in/on the fridge 1.	

Figure 3: An exemplar in ReAct (Yao et al., 2023) for a Heat and Place task in the ALFWorld environment (Shridhar et al., 2020). The text in the black box is composed of the description from the ALFWorld environment (Shridhar et al., 2020) and the thoughts annotated by ReAct (Yao et al., 2023). The thought that decomposes the goal is shown with a magenta background. The thought that uses commonsense reasoning to find an object and determine what to do with it is shown with a red background. The thoughts that track subgoal completion are shown with a cyan background. The thoughts that determine the next subgoal are shown with a green background.

311 312

313

4.2 EXPERIMENTAL RESULT OF SALA

In Reflexion (Shinn et al., 2023), a status text indicating whether a task is successful or not, along
with a reflection text that guides the next trial toward success, are concatenated after the ReAct
exemplars. Reflexion (Shinn et al., 2023) used two exemplars, as shown in Figure 4 in Appendix
A.1, to guide the LLM in generating the reflection.

Unlike Reflexion (Shinn et al., 2023), which uses one LLM to generate thoughts and actions and another LLM to generate reflections, SALA uses a single LLM to generate thoughts, actions, and self-adaptations by concatenating the two Reflexion exemplars (Shinn et al., 2023) after the two Re-Act exemplars (Yao et al., 2023) for each task. This approach allows the adaptation to be generated after the LLM receives the string "STATUS: FAIL". We experiment with this single-LLM setup using the exemplars from ReAct (Yao et al., 2023) and Reflexion (Shinn et al., 2023) as depicted in Figure 5 in Appendix A.1.

0	0	л
J	4	4
з	2	5

Table 2: Number of steps for the ReAct-based (Yao et al., 2023) agent and the proposed SALA to complete tasks in the ALFWorld environment (Shridhar et al., 2020)

e mon	is in the right (volta environment (Sinfanar et al., 2020)							
	Task number	1	2	3	4	5	6	7
	ReAct (Yao et al., 2023)	fail	fail	14	10	12	19	fail
	SALA	13	12	fail	10	17	87	fail
	Task number	8	9	10	11	12	13	14
	ReAct (Yao et al., 2023)	23	fail	10	19	fail	16	15
	SALA	19	fail	10	17	67	fail	10

332 333

330 331

334

Initially, the input to the LLM is a prompt with the instruction, exemplars, and the description of the environment and the task to be completed. The output of the LLM will be the input prompt concatenated with a series of thoughts, actions, and observations. We extract the first thought or action from the output by extracting the line immediately following the input prompt content. The line immediately after the greater than symbol (>) is used as input to the environment. If the input to the environment contains the string "think:", we overwrite the output with "OK.", following the process described in (Yao et al., 2023; Shinn et al., 2023).

342 If the task is completed, the environment will return a variable with a value of 1, and the process is 343 finished. Otherwise, we concatenate the extracted thought or action with the observation and append 344 this new text to the input prompt to form the next input prompt. After 50 steps, if the task remains 345 unfinished, we reinitialize the ALFWorld environment (Shridhar et al., 2020) and append "STATUS: 346 FAIL" and "New plan:" after the last observation. This updated text becomes the next input prompt to the LLM. The output of the LLM will include the adaptation text following "New plan:", which 347 is extracted and concatenated above the initial prompt, s_0^{ep} . This new prompt is then used as the next 348 input to the LLM to obtain corrected thoughts and actions. We set a maximum of 9 trails; if $ep \ge 10$, 349 we terminate the trial and proceed to the next task. The results of this experiment, conducted for 14 350 different tasks using gemma-2-9b-it, are shown in Table 2. 351

352 Among the fourteen tasks, ten were completed successfully, while four were not finished after reach-353 ing 10 adaptation steps. Of the ten successful tasks, eight were completed without reaching the adaptation step. Failures in tasks 7 and 9 for both methods were due to issues within the ALF-354 World environment (Shridhar et al., 2020). Although both agents successfully finished the tasks for 355 tasks 7 and 9, the environment failed to indicate the task were completed. Consequently, tasks 7 356 and 9 are excluded from the evaluation of different methods. Excluding these tasks, the proposed 357 SALA achieves a success rate of approximately 83%, outperforming the ReAct-based agent, which 358 achieved a success rate of 67%. 359

In one of the failed tasks (task 3) from SALA, the only adaptation obtained was: "I was stuck in 360 a loop in which I continually looked for a lettuce in the fridge. I should have looked for a lettuce 361 in the fridge, then taken it, then put it in the countertop. I will try to execute a different action if 362 I am stuck in a loop again." for every trial. However, the SALA agent did not attempt different 363 actions in subsequent trials and continued to examine the fridge. The actual issue in task 3 is that the 364 fridge does not contain lettuce, but the SALA agent incorrectly assumes it is present and persistently attempts to retrieve it. A similar issue is also found in task number 13 which has the adaptation: "I 366 was stuck in a loop in which I continually examined fridge 1 instead of using the fridge to cool the 367 lettuce. I should have taken the lettuce from the fridge and then put it on the countertop. I will try to 368 execute a different action if I am stuck in a loop again." The SALA agent did not attempt different 369 actions in subsequent trials and continued to examine the fridge.

370 In task 6, the task could not be completed in the first trial but was successfully completed in the 371 second trial with 38 steps. The goal of this task is to "find two pillow and put them in sofa". In the 372 first trial, the agent failed to complete the task because it attempted to pick up 'pillow 2' from 'sofa 373 1'; however, 'sofa 1' only contained 'pillow 1'. Afterward, the agent continued trying to put 'pillow 374 2' on 'sofa 1,' but it failed because it did not have the pillow. Subsequently, the LLM did not output 375 any text until the end of the trial. After this, the following adaptation was generated and appended to the input prompt for the second trial: "I was stuck in a loop in which I continually looked for the 376 second pillow in sofa 1. I should have looked for the second pillow in armchair 1, sidetable 1, and 377 cabinet 1-4. I will try to execute a different action if I am stuck in a loop again.". In the second trial,

the agent found that 'pillow 2' was on 'armchair 1,' picked it up, placed it on the sofa, and completed the task. The trajectories of the two trials in task number 6 are shown in Figure 6 in Appendix A.1.

In task 12, the task could not be completed in the first trial but was successfully completed in the 381 second trial within 18 steps. The goal of this task is to "put a cool tomato in microwave". In the first 382 trial, the agent failed to complete the task because it attempted to take a 'tomato' from 'fridge 1'; 383 however, 'fridge 1' doesn't contained any tomato. Afterward, the agent continued trying to take a 384 'tomato' from 'fridge 1' until the end of the trial. After this, the following adaptation was generated 385 and appended to the input prompt for the second trial: "I was stuck in a loop in which I continually 386 looked for a tomato in the fridge. I should have looked for a tomato in a different environment. I 387 will try to look for a tomato in a different environment in the next trial.". In the second trial, the 388 agent found that 'tomato 1' was on 'countertop 1,' picked it up, cooled it with the fridge, placed it in the microwave, and completed the task. The trajectories of the two trials in task number 12 are 389 shown in Figure 7 in Appendix A.1. 390

391 392

393

5 CONCLUSION

394 We present a novel in-context learning algorithm designed for a single language model to complete 395 tasks in a text-based game by correcting its previous failures. This approach, by introducing self-396 adaptation reduces the number of models used in previous work (Shinn et al., 2023) from two LLMs 397 to one LLM, gaining in autonomy. Our findings indicate that the gemma-2-9b-it model achieves 398 the highest success rate of 62% for completing tasks in the ALFWorld environment (Shridhar et al., 399 2020) using ReAct prompting (Yao et al., 2023), compared to other selected open-source language models. In the twelve selected tasks from the ALFWorld environment (Shridhar et al., 2020), the 400 proposed SALA, utilizing the gemma-2-9b-it model, achieved a success rate of 83%, outperforming 401 the ReAct-based agent (Yao et al., 2023) with the same model, which attained a 67% success rate. 402 In addition, we show that using SALA with gemma-2-9b-it, two of the six tasks that could not be 403 completed in one trial can be completed in the second attempt by appending the adaptation from the 404 previous trial. Future work will involve further experimentation with different in-context learning 405 algorithms to complete decision-making tasks using a single language model without Reflexion 406 exemplars for reducing the number of tokens in the input prompt.

407 408 409

417

422

423

424

425

428

References

- Simon Arvidsson and Johan Axell. Prompt engineering guidelines for llms in requirements engineering. Bachelor's thesis, University of Gothenburg and Chalmers University of Technology, 2023.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal,
 Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are
 few-shot learners. Advances in neural information processing systems, 33:1877–1901, 2020.
- Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm:
 Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240): 1–113, 2023.
 - Dorottya Demszky, Diyi Yang, David S Yeager, Christopher J Bryan, Margarett Clapper, Susannah Chandhok, Johannes C Eichstaedt, Cameron Hecht, Jeremy Jamieson, Meghann Johnson, et al. Using large language models in psychology. *Nature Reviews Psychology*, 2(11):688–701, 2023.
- Sabit Ekin. Prompt engineering for chatgpt: a quick guide to techniques, tips, and best practices.
 Authorea Preprints, 2023.
- Martin Klissarov, Pierluca D'Oro, Shagun Sodhani, Roberta Raileanu, Pierre-Luc Bacon, Pascal Vincent, Amy Zhang, and Mikael Henaff. Motif: Intrinsic motivation from artificial intelligence feedback. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=tmBKIecDE9.

- Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. Large language models: A survey. *arXiv preprint arXiv:2402.06196*, 2024.
- Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), Advances in Neural Information Processing Systems, volume 36, pp. 8634–8652. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/1b44b878bb782e6954cd888628510e90-Paper-Conference.pdf.
- Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew
 Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020.
- Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*, 2023. URL https://openreview.net/forum?id=P8E4Br72j3.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), Advances in Neural Information Processing Systems, volume 35, pp. 24824–24837. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/ file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf.
 - Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.
- 459 460 461

464

456

457

458

A APPENDIX

A.1 FIGURES

465 In Figure 6, the text in each black box represents one trajectory in the ALFWorld environment (Shridhar et al., 2020), as detailed in the experiment in Section 4.2. In each black box, the text at 466 the top includes the description of available receptacles and the goal instruction, as mentioned in 467 Fig. 2. Below the goal instruction, the text following the greater-than symbol ('>') represents the 468 thoughts or actions generated by the gemma-2-9b-it LLM. Text beginning with '> think:' indicates 469 a thought, while all other text beginning with without 'think' represents actions. The text beneath 470 each thought or action is the observation from the ALFWorld environment (Shridhar et al., 2020). 471 The trajectory of the first trial in task number 6 is shown in the left black box. The LLM agent failed 472 to complete the task because it couldn't find the second pillow, incorrectly believing it had finished 473 the task, and then continued generating empty strings. The adaptation the LLM agent generated is 474 highlighted with a cyan background. In the second trial, shown in the right black box, the adaptation 475 is appended above the description of the available receptacles. The agent successfully found the 476 second pillow in the second trial and completed the task.

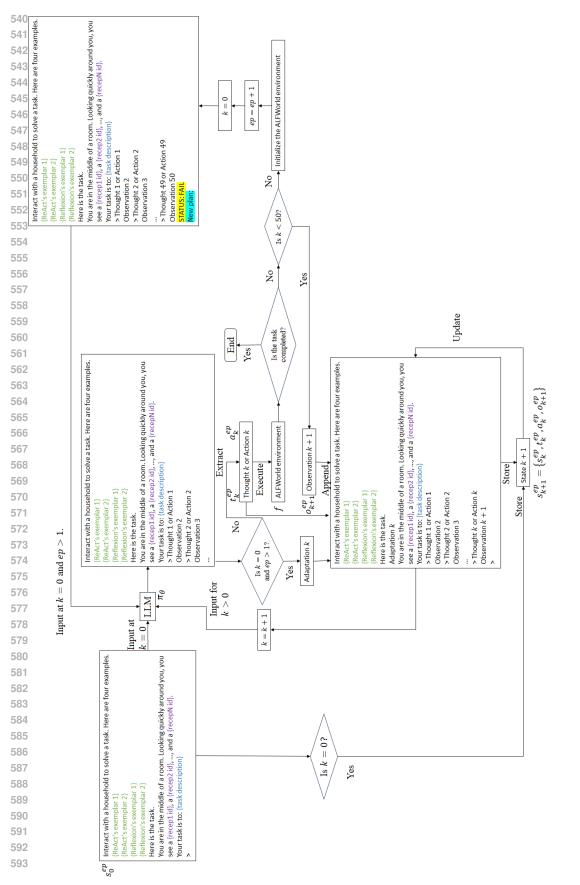
477 In Figure 7, the text in each black box represents one trajectory in the ALFWorld environment 478 (Shridhar et al., 2020), as detailed in the experiment in Section 4.2. In each black box, the text at 479 the top includes the description of available receptacles and the goal instruction, as mentioned in 480 Fig. 2. Below the goal instruction, the text following the greater-than symbol ('>') represents the 481 thoughts or actions generated by the gemma-2-9b-it LLM. Text beginning with '> think:' indicates 482 a thought, while all other text beginning with without 'think' represents actions. The text beneath 483 each thought or action is the observation from the ALFWorld environment (Shridhar et al., 2020). The trajectory of the first trial in task number 6 is shown in the left black box. The LLM agent 484 failed to complete the task because it couldn't find a tomato, and then continued checking the same 485 receptacle. The adaptation the LLM agent generated is highlighted with a cyan background. In the

400		
486	You are in the middle of a room. Looking quickly around you,	You are in the middle of a room. Looking quickly around you,
487	you see a bed 1, a desk 2, a desk 1, a drawer 6, a drawer 5, a	you see a cabinet 10, a cabinet 9, a cabinet 8, a cabinet 7, a
488	drawer 4, a drawer 3, a drawer 2, a drawer 1, a garbagecan 1, a laundryhamper 1, a safe 1, a shelf 6, a shelf 5, a shelf 4,	cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 2, a countertop
489	a shelf 3, a shelf 2, and a shelf 1.	1, a diningtable 2, a diningtable 1, a drawer 3, a drawer 2, a
490	Your task is to: look at bowl under the desklamp.	drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a
	> think: To solve the task, I need to find and take a bowl,	sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner
491	then find and use a desklamp.	2, a stoveburner 1, and a toaster 1.
492	OK.	Your task is to: heat some mug and put it in coffeemachine.
493	> think: First I need to find a bowl. A bowl is more likely to appear in drawer (1-6), desk (1-2), shelf (1-6), garbagecan	> think: To solve the task, I need to find and take a mug, then heat it with stoveburner, then put it in coffeemachine.
494	(1), laundryhamper (1). I can check one by one, starting	Ok.
	with drawer 1.	>look
495	ОК.	You are in the middle of a room. Looking quickly around you,
496	> go to drawer 1	you see nothing.
497	The drawer 1 is closed. > open drawer 1	> think: First I need to find a mug. A mug is more likely to appear in countertop (1-2), diningtable (1-2), drawer (1-3),
498	You open the drawer 1. The drawer 1 is open. In it, you see	fridge (1), sinkbasin (1), stoveburner (1-4), toaster (1). I can
	a cellphone 2.	check one by one, starting with countertop 1.
499	> go to drawer 2	Ok.
500	On the drawer 2, you see a keychain 1.	> look
501	> go to drawer 3	You are in the middle of a room. Looking quickly around you,
502	Nothing happens. > go to drawer 4	you see nothing. > go to countertop 1
	The drawer 4 is closed.	On the countertop 1, you see a butterknife 1, a cup 1, a fork
503	> open drawer 4	2, and a mug 1.
504	You open the drawer 4. The drawer 4 is open. In it, you see	> think: Now I find a mug (1). Next, I need to take it.
505	nothing.	Ok.
506	> go to drawer 5 On the drawer 5, you see nothing.	> take mug 1 from countertop 1 You pick up the mug 1 from the countertop 1.
	> go to drawer 6	> think: Now I take a mug (1). Next, I need to heat it with
507	The drawer 6 is closed.	stoveburner (1-4).
508	> open drawer 6	Ok.
509	You open the drawer 6. The drawer 6 is open. In it, you see	> go to stoveburner 1
510	a book 2, and a pen 2. > go to desk 1	On the stoveburner 1, you see a pan 2. > examine stoveburner 1
511	On the desk 1, you see a alarmclock 1, a desklamp 1, a mug	On the stoveburner 1, you see a pan 2.
	1, and a pencil 1.	> examine stoveburner 1
512	> think: Now I find a desklamp (1). Next, I need to use it.	On the stoveburner 1, you see a pan 2.
513	ОК.	> examine stoveburner 1
514	> use desklamp 1	On the stoveburner 1, you see a pan 2. > examine stoveburner 1
515	You turn on the desklamp 1.	On the stoveburner 1, you see a pan 2.
	Nothing happens.	STATUS: FAIL
516	> go to desk 2	New plan: I was stuck in a loop in which I continually
517	On the desk 2, you see a bowl 2, a bowl 1, a cd 2, a cd 1,	examined stoveburner 1 instead of heating mug 1 with
518	and a creditcard 3.	stoveburner 1. I should have taken mug 1 from countertop 1,
519	> think: Now I find a bowl (2). Next, I need to take it. OK.	then heated it with stoveburner 1, then put it in coffeemachine 1. It did not help to execute two identical
520	> take bowl 2 from desk 2	actions in a row. I will try to execute a different action if I am
	You pick up the bowl 2 from the desk 2.	stuck in a loop again.
521	>	
522	Nothing happens.	
523	> Nothing happens.	
524	STATUS: FAIL	
	New plan: In this environment, my plan was to find a bowl	
525	then find and use a desklamp. However, the task says to	
526	look at bowl under the desklamp. I should have looked for	
527	the desklamp then looked for the bowl. I noticed that the desklamp was found on desk 1. In the next trial, I will go to	
528	desk 1, find the lamp, then look for the bowl under the	
	decklown	

desklamp.

531 Figure 4: Two exemplars in Reflexion (Shinn et al., 2023) for the ALFWorld environment (Shridhar et al., 2020). The text in each black box comprises one exemplar from Reflexion (Shinn et al., 2023) 532 designed to guide an LLM in generating the correct action to complete a task in the ALFWorld 533 environment (Shridhar et al., 2020). In each black box, the text preceding the yellow-background 534 text represents a ReAct trajectory, as shown in Fig. 3. The text next to "STATUS:" indicates whether 535 the task is completed. If the task is completed, the yellow-background text will read "STATUS: OK". 536 If the task is not completed, it will read "STATUS: FAIL". The reflection text is highlighted with a 537 cyan background. 538

539



Here is the task. You are in the middle of a room. Looking quickly around you, you see a armchair 1, a cabinet 4, a cabinet 2, a cabinet 1, a drawer 1, a dra			
	i94		
 and the state of a state	595		
Procession Procession Procession P	596	armchair 1, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a drawer 5,	pillow in sofa 1. I should have looked for the second pillow in armcha
 b. J. J. Start of J. J. Start of J. J. Start of J. Start	97	garbagecan 1, a safe 1, a shelf 12, a shelf 11, a shelf 10, a shelf 9, a	 sidetable 1, and cabinet 1-4. I will try to execute a different action I am stuck in a loop again.
 Anten is and the last is and is a finite and is finite and is a finite and is a finite and is a finit	98	shelf 1, a sidetable 1, and a sofa 1.	You are in the middle of a room. Looking quickly around you, you see
0.	99	> think: To solve the task, I need to find two pillow, then put them in	a drawer 4, a drawer 3, a drawer 2, a drawer 1, a dresser 1, a
 appent to only (1), matched (1), local theorem by even to the section of the section of		ок.	shelf 8, a shelf 7, a shelf 6, a shelf 5, a shelf 4, a shelf 3, a shelf 2, a
 Set of intervalue. Set of intervalue.<td>00</td><td>appear in sofa (1), armchair (1), sidetable (1). I can check one by one,</td><td>Your task is to: find two pillow and put them in sofa. > think: To solve the task, I need to find two pillow, then put them in</td>	00	appear in sofa (1), armchair (1), sidetable (1). I can check one by one,	Your task is to: find two pillow and put them in sofa. > think: To solve the task, I need to find two pillow, then put them in
<pre>// / prove pr</pre>	01	> go to sofa 1	ОК.
<pre></pre>	02	> go to armchair 1	appear in sofa (1), armchair (1), sidetable (1), cabinet (1-4), dresser (
14 0. 0. 0. 0. 0. 15 0. 0. 0. 0. 0. 16 0. 0. 0. 0. 0. 0. 17 0.	03	1.	OK.
See Set with a plane 1. So we are s	04	ок.	On the sofa 1, you see nothing.
1. 1. 1. 1. 1	15	You pick up the pillow 1 from the armchair 1.	On the armchair 1, you see a pillow 2, a pillow 1, and a remotecontro
1 i = b to 5h 1 i = pe to 5h 1 i = pe		1.	
 Jack Ney Lands Lands		> go to sofa 1	> take pillow 1 from armchair 1
 a hather is not just the fing plane on the Nation Street of both with the page of the street of both the str)7	> put pillow 1 in/on sofa 1	> think: Now I take the first pillow (1). Next, I need to put it in/on so 1.
99 istriction with a large status 1, we status 1,	80	> think: Now I put the first pillow in sofa. Next, I need to find the	
0 _sps to S1 11 _spt to S1 12 _spt to S1 13 _spt to S1 14 _spt to S1 15 _spt to S1 16 _spt to S1 17 _spt to S1 18 _spt to S1 19 _spt to S1 10 _spt to S1 10 _spt to S1 11 _spt to S1 12 _spt to S1 13 _spt to S1 14 _spt to S1 15 _spt to S1 16 _spt to S1 17 _spt to S1 16 _spt to S1 17 _spt to S1 18 _spt to S1 19 _spt to S1 10 _spt to S1 10 _spt to S1 11 _spt to S1 12 _spt to S1 12 _spt to S1 13 _spt to S1 14 _spt to S1 15 _spt to S1 16 _spt to S1)9	starting with sofa 1.	> put pillow 1 in/on sofa 1
11 One begin space 1: one begin 1: one begin space 1: one begin space 1: one begi	0	OK. > go to sofa 1	You put the pillow 1 in/on the sofa 1. > think: Now I put the first pillow in sofa. Next, I need to find the
2		> go to sidetable 1	pillow is more likely to appear in sidetable (1), cabinet (1-4), dresser
0 0 33 0 44 0 55 1 56 1 57 1 58 0 69 0 60 0 70 0 71 0 72 0 73 0 74 0 75 0 76 0 77 0 78 0 79 0 70 0 70 0 70 0 70 0 71 0 72 0 73 0 74 0 75 0 76 0 77 0 78 0 79 0 70 0 70 0 70 0 70 0 70 0 70		newspaper 1.	 shelf (1-12). I can check one by one, starting with sidetable 1. OK.
33 On the soft if, you we a pillow 1. Set to define 1 44 One, have a pillow 2 from soft 1 Set to define 1 55 One have base to pillow (1). Heat, 1 need to path in (non, 1), it, you are a the soft 1. Set to define 1 56 One have base to pillow (1). Heat, 1 need to path in (non, 1), it, you are a the soft 1. Set to define 1 77 One have base to pillow (1). Heat, 1 need to path in (non, 1), it, you are a base 3. Set to define 1 78 One have base to pillow (1). Heat, 1 need to path in (non, 1), it, you are a base 3. Set to define 1 78 One have base to pillow (1). Heat, 1 need to path in (non, 1), it, you are a base 3. Set to define 1 78 One have base 1. Set to define 1 Set to define 1 78 One have base 1. Set to define 1 Set to define 1 78 Set to define 1 Set to define 1 Set to define 1 78 Set to define 1 Set to define 1 Set to define 1 78 Set to define 1 Set to define 1 Set to define 1 78 Set to define 1 Set to define 1 Set to define 1 78 Set to define 1 Set to define 1 Set to define 1 78 Set	12	ок.	On the sidetable 1, you see a creditcard 1, a keychain 1, and a
14 Of. - Segme data 5 - Segme data - Segme data 6 - Segme data - Segme data 7 - Segme data - Segme data 8 - Segme data - Segme data 9 - Segme data - Segme data	3	On the sofa 1, you see a pillow 1.	> go to cabinet 1
15 Noting bagent. 1. And A. Kowing bagent. Instant B. Kowing bagent. 1. And A. Kowing bagent. Instant B. Kowing bagent. 1. And A. Kowing bagent. Instant B. Kowing bagent. 1. And A. Kowing bagent. Instant B. Kowing bagent. 1. And A. Kowing bagent. Instant B. Kowing bagent. 1. And A. Kowing bagent. Instant B. Kowing bagent. 1. And A. Kowing bagent. Instant B. Kowing bagent. 1. And A. Kowing bagent. Instant B. Kowing bagent. 1. And A. Kowing bagent. Instant B. Kowing bagent. 1. And A. Kowing bagent. Instant B. Kowing bagent. 1. And A. Kowing bagent. Instant B. Kowing bagent. 1. And A. Kowing bagent. Instant B. Kowing bagent. 1. And A. Kowing bagent. Instant B. Kowing bagent. 1. And A. Kowing bagent. Instant B. Kowing bagent. 1. And A. Kowing bagent. Instant B. Kowing bagent. 1. And A. Kowing bagent. Instant B. Kowing bagent. 1. And A. Kowing bagent. Instant B. Kowing bagent. 1. And I. Kowing bagent. Instant B. Kowing bagent. 1. And I. Kowing bagent. Instant B. Kowing bagent. 1. And I. Kowing bagent. Instant	4	ОК.	> open cabinet 1
of 1. Meding happen. 0.	5	Nothing happens.	tissuebox 1.
	-	sofa 1.	Nothing happens.
1/1 > blaik. Kowi just the second pillow in ofe. Vou open the sublet 3. In outburt 1. to use a bas 3. 1/2 Anthong happens. You open the sublet 3. In outburt 1. to use a bas 3. 1/2 Anthong happens. You open the sublet 3. In outburt 1. to use a bas 3. 1/2 Anthong happens. You open the sublet 3. In outburt 1. a watch 2. a wa	-	> put pillow 2 in/on sofa 1	The cabinet 3 is closed.
88 > > both the happens. > both the happens. 90 > > both the happens. > both the happens. 90 > > both the happens. > both the happens. 91 > both the happens. > both the happens. > both the happens. 92 > > both the happens. > both the happens. 92 > > both the happens. > both the happens. 92 > > both the happens. > both the happens. 92 > > both the happens. > both the happens. 92 > > both the happens. > both the happens. 92 > > both the happens. > both the happens. 92 > > both the happens. > both the happens. 92 > > both the happens. > both the happens. 92 > > both the happens. > both the happens. 92 > > both the happens. > both the happens. 92 > > both the happens. > both the happens. 92 > > both the happens. > both the happens. 92 > > both the happens. > both the happens. 92 > > both the happens. > both the happens.	17	> think: Now I put the second pillow in sofa.	You open the cabinet 3. The cabinet 3 is open. In it, you see a box 3,
99 > 90 Nothing happens. 90 Nothing happens. 91 Nothing happens. 92 Nothing hap	8	> Nothing happens.	
20 Nothing happens. > point of all in the intervention of the interventint interven	9	>	> go to dresser 1
1 Nothing happens. > got baff 2 22 Nothing happens. > got baff 3 33 Nothing happens. > got baff 4 34 Nothing happens. On the sheff 2, you see nothing. 34 Nothing happens. On the sheff 2, you see nothing. 35 Nothing happens. On the sheff 2, you see nothing. 36 Nothing happens. > got baff 3 36 Nothing happens. > got baff 4 37 Nothing happens. > got baff 7 36 Nothing happens. > got baff 7 36 Nothing happens. > got baff 7 37 Nothing happens. > got baff 7 38 Nothing happens. > got baff 7 36 Nothing happens. > got baff 7 37 Nothing happens. > got baff 7 38 Nothing happens. > got baff 7 39 Nothing happens. > got baff 7 30 Nothing happens. > got baff 7 31 Nothing happens. > got baff 7 32 Nothing happens. > got baff 7 30 Nothing happens. > got baff 7 31 Nothing happens. > got baff 7 32 Nothing happens. > got baff	20	> Nothing happens.	> go to shelf 1
volting happers. volting happers. 22 volting happers. 23 volting happers. 24 volting happers. 25 volting happers. 26 volting happers. 27 volting happers. 28 volting happers. 29 volting happers. 20 volting happers. 21 volting happers. 22 volting happers. 23 volting happers. 24 volting happers. 25 volting happers. 26 volting happers. 27 volting happers. 28 volting happers. 29 volting happers. 20 volting happers. 21 volting happers. 22 volting happers. 23 volting happers. 24 volting happers. 25 volting happers. 26 volting happers. 27 volting happers. 28 volting happers. 29 volting happers. 29 volting happers. 29 volting happers. 20 volting happers. 21 volting happers. 22	01	> Nothing happens.	> go to shelf 2
Nothing happens. > 24 Nothing happens. 25 Nothing happens. 26 > 27 Nothing happens. 28 Nothing happens. 29 Nothing happens. 20 Nothing happens. 27 Nothing happens. 28 Nothing happens. 29 Nothing happens. 20 Nothing happens.		> Nothing happens.	> go to shelf 3
Solution Nothing happens. Solution Solution 24 Nothing happens. Solution Solution 25 Nothing happens. Solution Solution 26 Nothing happens. Solution Solution 27 Nothing happens. Solution Solution 28 Nothing happens. Solution Solution 29 Nothing happens. Solution Solution 29 Nothing happens. Solution Solution 29 Nothing happens. Solution Solution 20 Nothing happens. Solution Solution 21 Nothing happens. Solution Solution 22 Nothing happens. Solution Solution 23 Nothing happens. Solution Solution 24 Nothing happens. Solution Solution 25 Nothing happ		Nothing happens.	> go to shelf 4
24 Nothing happens. > go to shelf 6 55 Nothing happens. > go to shelf 6 26 Nothing happens. > go to shelf 7 26 Nothing happens. > go to shelf 7 27 Nothing happens. > go to shelf 3 28 Nothing happens. > go to shelf 3 29 Nothing happens. > go to shelf 3 20 Nothing happens. > go to shelf 3 21 Nothing happens. > go to shelf 3 22 Nothing happens. > go to shelf 3 23 Nothing happens. > go to shelf 3 24 Nothing happens. > go to shelf 3 25 Nothing happens. > go to shelf 3 26 Nothing happens. > go to shelf 3 27 Nothing happens. > go to shelf 3 28 Nothing happens. > g	23	Nothing happens.	> go to shelf 5
25 Nothing happens. > pert to shelf 7 26 Nothing happens. > pert to shelf 7 27 Nothing happens. 0 28 Nothing happens. 0 29 Nothing happens. 0 29 Nothing happens. 0 29 Nothing happens. 0 29 Nothing happens. 0 20 Nothing happens. 0 21 Nothing happens. 0 22 Nothing happens. 0 23 Nothing happens. 0 24 Nothing happens. 0 25 Nothing happens. 0 26 Nothing happens. 0 26 Nothing happens. 0 26 Nothing happens. 0 26 Nothi	24	> Nothing happens. >	> go to shelf 6
66 > pot os heff 8 77 > Nothing happens. 88 > Nothing happens. 99 Nothing happens. 90 <	5	> Nothing happens.	> go to shelf 7
Nothing happens. > pot a sheff 9 Nothing happens. > pot a sheff 9 Nothing happens. > pot a sheff 10, you see nothing. Ye to sheff 11, you see nothing. > pot a sheff 11, you see nothing. Ye to sheff 11, you see nothing. > pot a sheff 11, you see nothing. Ye to sheff 11, you see nothing. > pot a sheff 11, you see a cellphone 1. Ye to sheff 11, you see a cellphone 1. > pot a sheff 12, you see a cellphone 1. Ye to sheff 11, you see a cellphone 1. > pot a sheff 12, you see a cellphone 1. Ye to sheff 11, you see a cellphone 1. > pot a sheff 12, you see a cellphone 1. Ye to sheff 11, you see a cellphone 1. > pot a sheff 12, you see a cellphone 1. Ye to sheff 11, you see a cellphone 1. > pot a sheff 12, you see a cellphone 1. Ye to sheff 12, you see a cellphone 1. > pot a sheff 12, you see a pellow 1. Ye to sheff 11, you see a pellow 1. > think: it doet of that he second pillow 1, should have looked for the second pillow 1, should have looked for the second pillow 2, how that he second pillow 2, how that he second pillow 2, how the net arrchair 1. Ye to sheff 11, you see a pillow 1. > think: Kow i that he second pillow 2, how the net her thing. Ye to pillow 2 how the second pillow 2, how the s	26	Nothing happens.	> go to shelf 8
1 1 1 1 1 1 28 Nothing happens. 2 2 3 2 29 Nothing happens. 2 2 3 0 20 Nothing happens. 2 2 0 1 20 Nothing happens. 2 0 1 0 1 20 Nothing happens. 2 0 1 0 1 21 Nothing happens. 2 0 1 0 1 0 22 Nothing happens. 2 0 1 0 1 0 1 23 Nothing happens. 2 0 1 0 1 1 1 24 Nothing happens. 2 0 0 1 1 1 24 Nothing happens. 2 0 0 1 1 1 25 Nothing happens. 0 0 1 1 1 1 25 Nothing happens. 0 0 1 1 1 1 26 Nothing happens. 0 0 1 1 1 1 26 Nothing happens. 0 <td></td> <td>Nothing happens.</td> <td>> go to shelf 9</td>		Nothing happens.	> go to shelf 9
8 Nothing happens. > pot is shell 11, you see nothing. 9 Nothing happens. > pot is shell 12, you see a cellphone 1. 1 Nothing happens. > pot is shell 12, you see a cellphone 1. 1 Nothing happens. > pot is shell 12, you see a cellphone 1. 2 Nothing happens. > pot is shell 12, you see a cellphone 1. 1 Nothing happens. > pot is shell 12, you see a cellphone 1. 2 Nothing happens. > pot is shell 12, you see a cellphone 1. 3 Nothing happens. > pot is shell 12, you see a cellphone 1. 2 Nothing happens. > pot is shell 12, you see a cellphone 1. 3 Nothing happens. > pot is shell 12, you see a cellphone 1. 3 Nothing happens. > pot is shell 12, you see a pillow 2, and a remotecontrol 1. 4 > pot is shell 12, you see a pillow 2, form armchair 1. > think: Now I find the second pillow (2). Next, I need to put k in/or sofa 1. 5 Nothing happens. > pot is for a 1. > think: Now I find the second pillow 2//on shell 12. 6 Nothing happens. > pot pillow 2 hron the armchair 1. > think: Now I find the second pillow 2//on shell 1. 7 Nothing happens. > pot pillow 2 hron		>	On the shelf 10, you see nothing.
0 > 0 Nothing happens. 1 > 2 Nothing happens. 3 On the soft 1, you see a pillow 1. 3 Nothing happens. 4 > 5 > 5 > 6 Nothing happens. 6 > 7 Nothing happens. 6 > 7 Nothing happens. 6 > 7 Nothing happens. 7 Nothing happens. 9 > 7 Nothing happens. 9 > 6 Nothing happens. 7 Nothing happens. 7 Nothing happens. 7 Nothing happens. 9 > 7 Nothing happens. 9 > 7 Nothing happens. 9 > 9 > 9 > 10 Nothing happens. 9 > 10 Nothing happens. 9 > 9 > 10 Nothing happens. 9 > 10 <	8	>	> go to shelf 11 On the shelf 11, you see nothing.
0 > On the soft 1, yous see a pillow 1. 1 > > 2 Nothing happens. > 3 Nothing happens. > 3 Nothing happens. > 4 > > 5 Nothing happens. > 6 Nothing happens. > 7 Nothing happens. > 6 Nothing happens. > 7 Nothing happens. > 8 > > 7 Nothing happens. > 8 > > 7 Nothing happens. > 6 Nothing happens. > 7 Nothing happens. > 8 > > 8 > > 9 Nothing happens. >	9	>	On the shelf 12, you see a cellphone 1.
1 > second pillow in armchair 1, sidetable 1, and cabinet 1-4. 2 Nothing happens. > 3 Nothing happens. > 4 > > take pillow 2 from the armchair 1 5 Nothing happens. > 6 Nothing happens. > 7 Nothing happens. > 6 Nothing happens. > 7 Nothing happens. > 8 > > pot a michair 1 7 Nothing happens. > 8 > > pot a michair 1 9 > > 7 Nothing happens. > 9 > > 9 > > 10 Nothing happens. > 2 Nothing happens. > 3 Nothing happens. > 3 Nothing happens. > 3 Nothing happens. > 3 Nothing happens	0	>	On the sofa 1, you see a pillow 1.
2 Nothing happens. > po to arrchait 1 3 Or the smchait 1, you see a pillow 2, and a remotecontrol 1. 4 Nothing happens. > hink: Now 1 kas the second pillow (2). Next, i need to take it. 4 Nothing happens. > kink: Now 1 kas the second pillow (2). Next, i need to take it. 5 Nothing happens. > kink: Now 1 kas the second pillow (2). Next, i need to put it in/or 5 Nothing happens. > kink: Now 1 kas the second pillow (2). Next, i need to put it in/or 5 Nothing happens. > kink: Now 1 kas the second pillow (2). Next, i need to put it in/or 5 Nothing happens. > kink: Now 1 kas the second pillow (2). Next, i need to put it in/or 5 Nothing happens. > kink: Now 1 kas the second pillow (2). Next, i need to put it in/or 6 Nothing happens. > pot northait 1. 7 Nothing happens. > pot northait 1. 8 Nothing happens. > pot northait 1. 9 Nothing happens. > pot pillow 2 in/on ofta 1. 7 Nothing happens. > pot pillow 2 in/on ofta 1. 8 Nothing happens. > pot pillow 2 in/on the sofa 1. 9 Nothing happens. > pot pillow 2 in/on the sofa 1. 9 Nothing happens. > pot pillow 2 in/on the sofa 1. 9 Nothing happens.	1	>	second pillow in armchair 1, sidetable 1, and cabinet 1-4.
 > bink: Nov I find the second pillow (2). Next, I need to take it. OK. > bink: Nov I find the second pillow (2). Next, I need to take it. OK. > bink: Nov I find the second pillow 2 from the second pillow 2. Next, I need to put it in/on to a so a . > bink: Nov I find the second pillow 2. Next, I need to put it in/on to a so a . > bink: Nov I find the second pillow 2. Next, I need to put it in/on to a so a . > bink: Nov I find the second pillow 2. Next, I need to put it in/on to a so a . > bink: Nov I find the second pillow 2. Next, I need to put it in/on to a so a . > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second pillow 1. > bink: Nov I find the second		>	> go to armchair 1
 > take pillow 2 from armchair 1 > take pillow 2 from armchair 1 > think: Now 1 take the second pillow 2 (2). Next, 1 need to put it in/or so a 1. > think: Now 1 take the second pillow 2 (2). Next, 1 need to put it in/or so a 1. > think: Now 1 take the second pillow 2 in/or not a pillow 2. > Nothing happens. > Nothing happ		>	> think: Now I find the second pillow (2). Next, I need to take it.
44 > > 55 Nothing happens. > 66 Nothing happens. > 7 Nothing happens. > 7 Nothing happens. > 88 Nothing happens. > 9 Nothing happens. > > STATUS: ON STATUS: ON 1 Nothing happens. > > STATUS: FAU STATUS: FAU 3 armchair I, sidehab, and cohard II, will try to execute a different action if I an stuck in a loop again.	-	>	> take pillow 2 from armchair 1
55 > OK. 66 Nothing happens. > oto sofa 1, you see a pillow 1. 77 Nothing happens. > up tillow 2 in/on sofa 1 78 > > oto sofa 1, you see a pillow 1. 79 Nothing happens. > oto sofa 1, you see a pillow 1. 88 > > oto put pillow 2 in/on sofa 1. 99 > > oto sofa 1, you see a pillow 1. 70 Nothing happens. > oto sofa 1. 88 > > oto sofa 1. 99 > > oto sofa 1. 90 > > oto sofa 1. 90 > > oto sofa 1. 91 Nothing happens. > oto sofa 1. 92 > > oto sofa 1. 93 > > oto sofa 1. 94 > > oto sofa 1. 95 > > oto sofa 1. 96 > > oto sofa 1. 97 > > oto sofa 1. 98 > > oto sofa 1. 99 > > oto sofa 1. 90 > > oto sofa 1. 91 Nothing happens. > oto sofa 1. 92 > > oto sofa 1. 93 > > oto sofa 1. 93	4	>	> think: Now I take the second pillow (2). Next, I need to put it in/on
on the sofe 1, you see a pillow 1. on the sofe 1, you see a pillow 1. on the sofe 1, you see a pillow 1. op tillow 2 in/on sofe 1 You put the pillow 2 in/on the sofe 1. Status Nothing happens. > > > Nothing happens. > > STATUS: NU Nothing happens. > > > Nothing happens. > > STATUS: NU Nothing happens. > > STATUS: NU New plan; I's astuck in a loop in which I continually looked for the second pillow in armchair I, sidelab 1, and chanke look for the second pillow in armchair I, sidelab 1, and chanke lift, will try to execute a different at the second pillow in armchair I, sidelab 1, and chanke lift. At the init I an stuck in a loop again.	5	>	OK.
You put the pillow 2 in/on the sofs 1. You put the pillow 2 in/on the sofs 1. You put the pillow 2 in/on the sofs 1. You put the pillow 2 in/on the sofs 1. You put the pillow 2 in/on the sofs 1. You put the pillow 2 in/on the sofs 1. You put the pillow 2 in/on the sofs 1. You put the pillow 2 in/on the sofs 1. You put the pillow 2 in/on the sofs 1. You put the pillow 2 in/on the sofs 1. You put the pillow 2 in/on the sofs 1. You put the pillow 2 in/on the sofs 1. You put the pillow 2 in/on the sofs 1. You put the pillow 2 in/on the sofs 1. You put the pillow 2 in/on the sofs 1. You put the pillow 2 in/on the sofs 1. You put the pillow 2 in/on the sofs 1. You put the pillow 2 in/on the sofs 1. You put the pillow in sofs 1. You put the	6	>	On the sofa 1, you see a pillow 1.
37 A Nothing happens. 38 Nothing happens. 99 Nothing happens. 100 Nothing happens. 11 Nothing happens. 12 New plant i loas tuck in a loop in which I continually looked for the second pillow in ascend pilow in sofa 1.1 should have looked for the second pillow in a annchair 1,3 delibel 1.4 will try to execute a different action if I am stuck in a loop again.		>	You put the pillow 2 in/on the sofa 1.
Nothing happens. 99 Nothing happens. 9 Nothing happens. 9 11 Nothing happens. 22 New plan: I was stuck in a loop in which I continually looked for the second pillow in a stuck in a loop again. 13 14		>	
99 > 100 Nothing happens. 201 Nothing happens. 202 New plant: I was stuck in a loop in which I continually looked for the second pillow in a second pillow in sofa 1.1 should have looked for the second pillow in a second pillow in a stuck in a loop again. 13 armchair 1, sideble 1, and calibre 1.41, will try to execute a different action if I am stuck in a loop again.		>	
11 Nothing happens. •STATUS: FAIL •STATUS: FAIL 12 New plan. I was stuck in a loop in which I continually looked for the second pillow in a mechanic J, sidetable 1, and calibre I -14, will try to execute a different action #1 am stuck in a loop again. 13 action #1 am stuck in a loop again.	9	>	
•STATUS: FAIL •STATUS: FAIL 12 New plan: 1 was stuck in a loop in which I continually looked for the second pillow in sets 1.1 should have looked for the second pillow in armchar i, sidetable 1, and calibre 1.4.1 will try to execute a different action if I am stuck in a loop again. 13 action if I am stuck in a loop again.	0	>	
•STATUS: FAIL •STATUS: FAIL 12 New plan: 1 was stuck in a loop in which I continually looked for the second pillow in sets 1.1 should have looked for the second pillow in armchar i, sidetable 1, and calibre 1.4.1 will try to execute a different action if I am stuck in a loop again. 13 action if I am stuck in a loop again.	11	> Nothing happens.	
articular juncidad sobie 1, and able the decision of the second purchases 13 action if I am stuck in a loop again. 14	-	New plan: I was stuck in a loop in which I continually looked for the	
4	_	armchair 1, sidetable 1, and cabinet 1-4. I will try to execute a different	
	-	action if i am stuck in a loop again.	

Figure 6: Two trajectories for task 6 generated by SALA. The left trajectory represents a failure, where the agent incorrectly assumes task completion and ceases actions. The right trajectory includes an adaptation (highlighted in cyan), leading to successful task completion.

648	second trial shown in the right block here, the adoptation is appended above the description of the
649	second trial, shown in the right black box, the adaptation is appended above the description of the available receptacles. The agent successfully found a tomato in the second trial and completed the
650	task.
651	LASK.
652	
653	
654	
655	
656 657	
658	
659	
660	
661	
662	
663	
664	
665	
666	
667	
668	
669	
670	
671	
672	
673	
674	
675	
676	
677	
678	
679	
680	
681	
682	
683	
684	
685	
686	
687	
688	
689	
690	
691	
692	
693	
694	
695	
696	
697	
698	
699	
700	
701	

702		
703	Here is the task.	Here is the task. I was stuck in a loop in which I continually looked for a tomato in the
704	You are in the middle of a room. Looking quickly around you, you see a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 3, a countertop 2, a countertop 1.	
705	drawer 3, a drawer 2, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a shelf 3, a shelf 2, a shelf 1, a sinkbasin 1, a stoveburner	You are in the middle of a room. Looking quickly around you, you see a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1,
706	4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1. Your task is to: put a cool tomato in microwave.	a coffeemachine 1, a countertop 3, a countertop 2, a countertop 1, a drawer 3, a drawer 2, a drawer 1, a fridge 1, a garbagecan 1, a
	> think: To solve the task, I need to find and take a tomato, then cool it with fridge, then put it in microwave.	t microwave 1, a shelf 3, a shelf 2, a shelf 1, a sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1.
707	OK. > think: First I need to find a tomato. A tomato is more likely to appear	
708	in fridge (1), countertop (1-3), shelf (1-3), drawer (1-3). I can check one by one, starting with fridge 1.	ОК.
709	OK. > go to fridge 1 The fridge 1 is closed.	> think: First I need to find a tomato. A tomato is more likely to appear in fridge (1), countertop (1-3), shelf (1-3), drawer (1-3). I can check one by one, starting with fridge 1.
710	> open fridge 1 You open the fridge 1. The fridge 1 is open. In it, you see a apple 1, a	OK. > go to fridge 1
711	bowl 3, a bowl 2, a cup 1, a mug 1, and a pan 1. > think: Now I find a tomato. Next, I need to take it.	The fridge 1 is closed. > open fridge 1
712	OK. > take tomato from fridge 1	You open the fridge 1. The fridge 1 is open. In it, you see a apple 1, a bowl 3, a bowl 2, a cup 1, a mug 1, and a pan 1.
713	Nothing happens. > think: I need to find a tomato in fridge 1.	> think: Now I find a tomato. Next, I need to take it. OK.
714	OK. > go to fridge 1	> take tomato from fridge 1 Nothing happens.
715	Nothing happens. > think: I need to find a tomato in fridge 1. OK.	 > think: Now I need to find a tomato. OK. > go to countertop 1
716	> go to fridge 1 Nothing happens.	On the countertop 1, you see a peppershaker 2, and a tomato 1. > take tomato 1 from countertop 1
717	> think: I need to find a tomato in fridge 1. OK.	You pick up the tomato 1 from the countertop 1. > think: Now I take a tomato (1). Next, I need to cool it with fridge (1).
718	> go to fridge 1 Nothing happens.	OK. > go to fridge 1
719	> think: I need to find a tomato in fridge 1. OK.	The fridge 1 is open. In it, you see a apple 1, a bowl 3, a bowl 2, a cup 1, a mug 1, and a pan 1.
720	> go to fridge 1 Nothing happens.	> cool tomato 1 with fridge 1 You cool the tomato 1 using the fridge 1.
721	> think: I need to find a tomato in fridge 1. OK. > go to fridge 1	> think: Now I cool a tomato (1). Next, I need to put it in/on microwave (1). OK.
722	> go to mage 1 Nothing happens. > think: I need to find a tomato in fridge 1.	> go to microwave 1 The microwave 1 is closed.
	OK. > go to fridge 1	> put tomato 1 in/on microwave 1 Nothing happens.
723	Nothing happens. > think: I need to find a tomato in fridge 1.	> think: Now I need to open the microwave. OK.
724	OK. > go to fridge 1	> open microwave 1 You open the microwave 1. The microwave 1 is open. In it, you see
725	Nothing happens. > think: I need to find a tomato in fridge 1.	nothing. > put tomato 1 in/on microwave 1
726	OK. > go to fridge 1 Nothing happens.	You put the tomato 1 in/on the microwave 1. STATUS: OK
727	> think: I need to find a tomato in fridge 1. OK.	
728	> go to fridge 1 Nothing happens.	
729	> think: I need to find a tomato in fridge 1. OK.	
730	> go to fridge 1 Nothing happens.	
731	> think: I need to find a tomato in fridge 1. OK. > go to fridge 1	
732	> go to mage 1 Nothing happens. > think: I need to find a tomato in fridge 1.	
733	OK. > go to fridge 1	
734	Nothing happens. > think: I need to find a tomato in fridge 1.	
735	OK. > go to fridge 1	
736	Nothing happens. > think: I need to find a tomato in fridge 1. OK.	
737	or. > go to fridge 1 Nothing happens.	
738	> think: I need to find a tomato in fridge 1. OK.	
739	> go to fridge 1 Nothing happens.	
740	> think: I need to find a tomato in fridge 1. OK.	
741	> go to fridge 1 Nothing happens.	
741	> think: I need to find a tomato in fridge 1. OK. > go to fridge 1	
	> go to mage 1 Nothing happens. > think: I need to find a tomato in fridge 1.	
743	OK. > go to fridge 1	
744	Nothing happens. > think: I need to find a tomato in fridge 1.	
745	OK. > go to fridge 1	
746	Nothing happens. > think: I need to find a tomato in fridge 1. OK.	
747	UK. > go to fridge 1 Nothing happens.	
748	> think: I need to find a tomato in fridge 1. OK.	
749	> go to fridge 1 Nothing happens.	
750	>STATUS: FAIL New plan: I was stuck in a loop in which I continually looked for a	
751	tomato in the fridge. I should have looked for a tomato in a different environment. I will try to look for a tomato in a different environment	
752	in the next trial.	
753		

Figure 7: Two trajectories for task 12 generated by SALA. The left trajectory shows a failure, where the agent repeats the same action with no progress until the maximum step is reached. The right trajectory includes an adaptation (in cyan), leading to successful task completion.