Federated Self-Evolving Embodied AI Agents

Leming Shen
The Hong Kong Polytechnic University
Hong Kong SAR, China
leming.shen@connect.polyu.hk

Yuanqing Zheng
The Hong Kong Polytechnic University
Hong Kong SAR, China
csyqzheng@comp.polyu.edu.hk

***Abstract*—The integration of Large Language Models (LLMs) fosters Embodied AI (EAI) agents to perceive and interact with the physical world through natural language instructions. However, existing EAI agents typically operate within fixed agentic workflows and predefined design spaces, struggling to handle rapidly evolving real-world EAI scenarios (*e.g.*, diverse task pipelines, dynamic environments). Unlike prior systems that treat LLMs mainly as *text-generators* following rigid workflows, we fully exploit their *reasoning capabilities* to allow agents to determine their own workflows on the fly. To this end, we propose `FSEAI` that encourages EAI agents to self-explore and self-evolve via federated collaboration across heterogeneous environments. Inspired by human learning processes, we distill EAI tasks into three atomic operations (`observe`, `reason`, `act`), and empower agents to explore workflows by dynamically and adaptively selecting an appropriate next operation based on the current state. Our evaluations show that, with federated collaboration, our `FSEAI` agents can achieve up to a 42.6% higher task success rate and a 41.6K token cost reduction than state-of-the-art (SOTA) baselines, while maintaining adaptability to unforeseen EAI scenarios. This highlights the potential of reasoning-driven adaptive agentic workflow towards cognitive EAI.**

***Index Terms*—Embodied AI, LLM, Federated Learning**

I. Introduction

Embodied AI (EAI) has emerged as a prominent paradigm where AI agents are physically situated in and interact with the real world through sensors and actuators [1]. Armed with diverse sensors [2], control policies [3], and AI models [4], EAI agents can automate perception, reasoning, and robotic control with minimal human intervention. Furthermore, recent advances in LLMs empower EAI agents with enhanced semantic understanding, high-level reasoning, and flexible task planning abilities, thereby facilitating more natural and generalizable interactions with complex real-world environments.

Fig. 1 (left) shows the general workflow of EAI agents. 1) *Environment Comprehension*. The agent encodes multimodal sensor data into LLM-interpretable embeddings or textual representations that capture environmental semantics, enabling the LLM to infer spatial relations. 2) *High-Level Task Decomposition*: Based on the user-specified EAI task and current environment semantics, the agent decomposes the task into multiple manageable subtasks. 3) *Low-Level Action Execution*: For each subtask, the agent invokes various actuators (*e.g.*, robot arms) to execute different actions (*e.g.*, grabbing objects). As such, EAI agents can understand complex instructions and reason over multimodal inputs by combining LLM-driven reasoning with sensory-grounded interaction.

This work is supported by Hong Kong GRF under Grant No. 15206123 and 15211924. Yuanqing Zheng is the corresponding author.

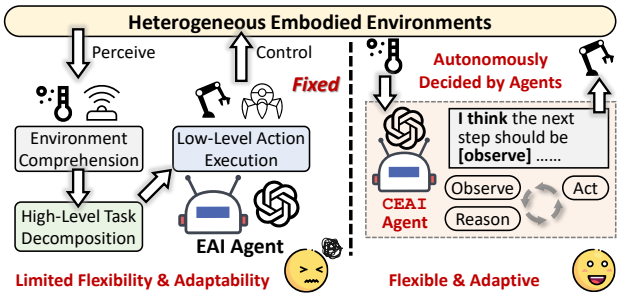


Fig. 1. Previous works handcrafted fixed agentic workflows to perform EAI tasks. `FSEAI` leverages LLMs' reasoning capabilities to autonomously explore and select the next atomic operation to handle diverse EAI tasks.

Nevertheless, most existing EAI agentic systems rely on predefined, fixed agentic workflows, often struggling to handle diverse EAI tasks in dynamic real-world scenarios. In practice, EAI tasks span various embodiments, heterogeneous operation environments, and dynamic runtime constraints. As a result, even well-engineered EAI agentic systems may struggle to accommodate such variability, suffering from inflexibility, limited adaptability, and high maintenance costs (§ II-B).

The root cause is that fixed workflows are often *constrained by the specific assumptions and development experiences of workflow designers*. Developers must explicitly enumerate workflow steps by handcrafting detailed prompts and tool usage rules to regulate agent actions for each. As a result, LLMs solely function as *text-generators* following predefined instruction flows. Yet, a key strength of LLMs remains underexplored: their reasoning capabilities, *i.e.*, the ability to make decisions in response to dynamic context information. This gap motivates a key research question: *Can we harness LLMs' reasoning capabilities to autonomously construct adaptive workflows to accommodate dynamic and ever-changing EAI environments?* If successful, we could shift the burden of workflow design from developers to EAI agents themselves, fostering more flexible, adaptive, and scalable EAI systems.

To this end, we propose `FSEAI`, a federated EAI paradigm that leverages LLMs' reasoning capabilities to autonomously and adaptively construct distinct workflows via federated collaboration [5]. The core insight is derived from human learning processes, which involve iterative knowledge acquisition, reasoning, practice, and reflection on accumulated experiences. Likewise, a `FSEAI` agent explores its workflow and evolves through iterations of self-exploration and reflection. Building

upon this vision, we distill EAI tasks into three atomic operations (`observe`, `reason`, `act`), which serve as meta building blocks of EAI agentic workflows (Fig. 1 right). The `FSEAI` agent autonomously selects the next appropriate meta operation based on its reasoning about the accumulated context, thereby flexibly adapting to diverse EAI scenarios. In addition, our `FSEAI` agent evolves through self-reflection, rather than blindly adhering to rigid workflows. Finally, we exploit federated learning [6] to collaboratively and continuously fine-tune multiple EAI agents for robust performance.

We validate and test `FSEAI` on an EAI simulator, *i.e.*, VirtualHome [7], which emulates 7 distinct household environments containing a large number of interactive objects. We compare `FSEAI` with 3 baseline EAI agents across 4 representative EAI tasks: environment Q&A, grabbing objects, placing objects, and hybrid tasks. Evaluation results demonstrate that `FSEAI` outperforms the baselines in terms of both task success rate (42.6% ↑), total token cost (41.6K ↓), and robust *generalizability* across tasks. These results validate the effectiveness of our reasoning-driven adaptive workflow construction paradigm. In summary, we make the following contributions:

- `FSEAI` exploits LLMs' reasoning abilities to autonomously construct agentic workflows via self-exploration and federated tuning, addressing the limitations of fixed workflows in heterogeneous and evolving EAI environments.
- We implement and evaluate `FSEAI` against 3 recent EAI agents, demonstrating its superior accuracy and generalizability. We also share actionable insights (§ IV) for designing future cognitive EAI agents in dynamic environments.

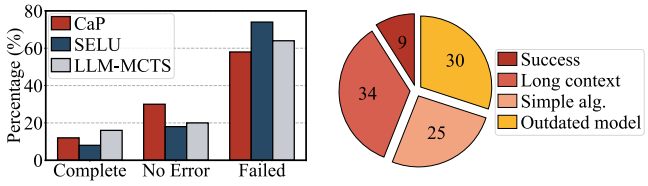
II. Background & Motivation

A. LLM & Embodied AI Agents

LLMs exhibit remarkable language generation and reasoning abilities [8], [9], [10]. By equipping LLMs with external tools (*e.g.*, search engines, databases), we can build agents and design agentic workflows guiding them to automatically solve complex tasks such as robotic manipulation. To pursue *general intelligence*, existing studies integrate LLMs into EAI agents as a central "brain" to enable more advanced environment comprehension, task reasoning, and robotic control. For example, CaP [3] leverages LLMs to generate programs to comprehend EAI environments and perform robotic manipulation. Moreover, SELU [11] further enhances EAI agents' reasoning capabilities through reinforcement learning.

B. Preliminary Study

While promising, most existing EAI agents rely on fixed agentic workflows that are often bound by developers' experiences, thus struggling to handle heterogeneous EAI tasks and exhibiting *inflexibility*, *limited adaptability*, and *high maintenance costs*. To better understand these limitations, we conduct a preliminary study by reproducing three recent EAI agents: Cap [3], SELU [11], and LLM-MCTS [12]. We evaluate their performance in a household simulator, VirtualHome [7], using a dataset [11] spanning across diverse EAI tasks. To simulate dynamic real-world home scenarios, we randomly make some objects keep moving during baseline execution.



(a) Existing EAI agents often fail to handle dynamic environments

(b) Though feasible, most EAI tasks can not be successfully completed

Fig. 2. Preliminary & feasibility study.

As shown in Fig. 2(a), only 34.6% of the EAI tasks can be executed without errors, while only 12% completely satisfy task requirements. **It is highly challenging to design a one-size-fits-all workflow to adapt to dynamic, unforeseen EAI environments and tasks.** Further inspection reveals two heterogeneous issues of the three EAI agents:

- *No handling of dynamic environments.* Built upon a predefined *sense-plan-act* workflow, they assume static household layouts and reliable localization. Thus, when the environment changes or contains moving obstacles, fixed workflows break down: the sensing module overlooks such variations, causing the planning module to generate unexecutable decisions (*e.g.*, moving while ignoring dynamic obstacles).
- *Poor scalability to long-horizon tasks.* When given a long-horizon task such as "tidy up the apartment and prepare it for guests", the agent must coordinate dozens of subtasks, including searching for objects across rooms, deciding what constitutes "tidy", handling dependencies (*e.g.*, clearing tables before placing items), and revisiting rooms as the environment changes. Predefined workflows typically rely on bounded planning horizons or static subgoal templates, causing planning complexity to grow rapidly and leading to brittle behavior, repeated actions, or premature termination.

To remedy the above as well as other unpredictable failures, developers must repeatedly revise and refine the workflows by explicitly listing detailed regulations, such as tool usage rules, edge-case handling advice, and consistency checks. Maintaining such handcrafted rules for unexpected cases with diverse requirements becomes highly time-consuming and error-prone.

C. Motivation

To overcome the above limitations, we propose `FSEAI`, a federated EAI agentic paradigm. Rather than treating embodied LLMs merely as *text-generators*, `FSEAI` fully exploits their *reasoning capabilities* to dynamically and autonomously optimize and adjust workflows in response to ever-changing environments. Inspired by human learning processes, we analyze over 30 EAI agentic systems and find that nearly all EAI workflows are composed of three atomic agentic operations:

- **`observe`**: An agent observes its current surroundings by capturing and analyzing various multimodal sensor data.
- **`reason`**: An agent analyzes the current and past state based on the available information and makes inferences before selecting the next operation. This operation relies on the agent's reasoning capabilities for decision-making.
- **`act`**: An agent executes actions (*e.g.*, moving forward, grabbing objects) in the environment to change its perspective for further observation or to accomplish the EAI task.

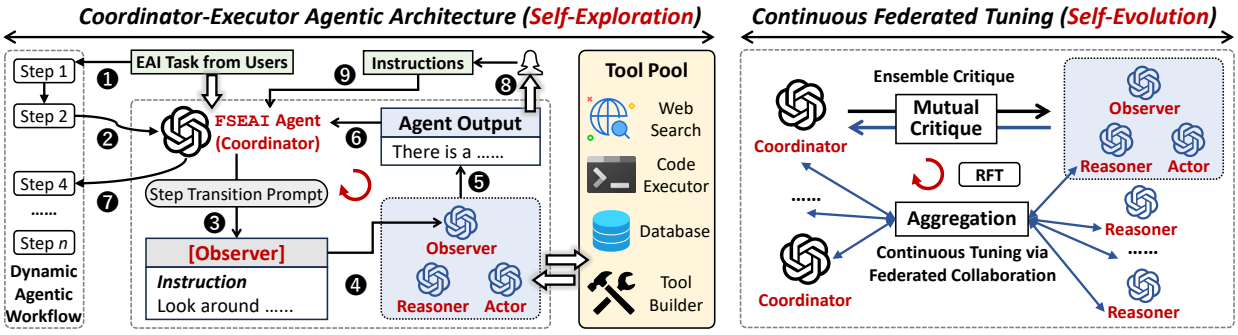


Fig. 3. The overall architecture and workflow of FSEAI (with detailed operations in step 3)

By enabling EAI agents to autonomously select among these operations in response to changing states, agentic workflows can be organically constructed. Similar to next token prediction in LLMs, FSEAI instead predicts the next atomic operation that drives the workflow forward.

III. Design of FSEAI

A. System Overview

FSEAI exploits LLMs' reasoning abilities to adaptively construct agentic workflows, meanwhile facilitating collaboration among federated EAI agents for continuous enhancement. Fig. 3 illustrates the overview of our proposed FSEAI, consisting of two technical modules: a hierarchical *coordinator-executor agentic architecture* for self-exploration (§ III-B) and *continuous federated tuning* for self-evolution (§ III-C).

B. Self-Exploration

This module exploits LLMs' reasoning capabilities to dynamically construct workflows for diverse EAI tasks.

Challenge. To realize self-exploration, an intuitive method is to construct a single EAI agent and guide it to iteratively select appropriate atomic operations. However, we find it inefficient and impractical to rely on a single agent to perform all the atomic operations due to the limited long-context reasoning abilities of current LLMs, even if they have large context windows [13]. We conduct a feasibility study where a single EAI agent is instructed to complete 100 different EAI tasks in VirtualHome. As shown in Fig. 2(a), only 9% of the tasks can be finished, while 35% fail and repeatedly execute the same action (*e.g.*, keep moving forward). This phenomenon exacerbates when embodied LLMs directly take multiple sensor data as inputs. This is because the agent accumulates increasingly long contexts, while LLMs inherently pay more attention to the earliest and most recent tokens in the context window, causing the agent to forget previously executed steps.

Solution. We design a *coordinator-executor agentic architecture* (Fig. 3 left), where a *coordinator* agent maintains a global view and autonomously orchestrates three task-specific atomic agents: *observer*, *reasoner*, and *actor*. Given an EAI task (❶), the *coordinator* iteratively determines the next appropriate operation. At each step, the *coordinator* will analyze the output from the previous operation (❷) using a step transition prompt, and generate the next atomic operation with detailed instructions (❸). The corresponding atomic agent is activated (❹) to execute the operation based on the instruction and generate an output (❺). The output is further sent to the *coordinator* (❻). Based on the currently constructed workflow, the *coordinator* decides if extra operations are required (❼) or concludes that the user's EAI task is completed with expected results (❽). In addition, FSEAI provides an interface for users to offer specific instructions for workflow refinement (❾). Such modular division of cognitive tasks effectively reduces reasoning complexity over long contexts and fosters interpretable and traceable decision-making. Note that the agents communicate with each other and with external tools via tailored HTTP and MCP interfaces, respectively. We design tailored prompts to regulate each agent's behavior and outputs.

Coordinator Prompt (P.1) includes five parts:

- *Role & objective.* We assign the agent a *coordinator* role and provide a task description that guides it to proactively orchestrate multiple agents for EAI tasks.
- *Respond rules* include detailed descriptions of the three atomic operations and their selection rules. We task the *coordinator* with iteratively selecting an appropriate operation and activating the corresponding atomic agent for execution. We also introduce a `quit` operation to terminate the EAI task if it is completed. Importantly, beyond sticking to the general loop of "*observe* → *reason* → *act*", we encourage the *coordinator* to repeat any stage as needed, *e.g.*, "*observe* → *reason* → *observe*". As such, the *coordinator* can dynamically explore workflows for various EAI tasks.
- *Current state.* Atomic agents' activities are recorded during operation execution, serving as the current state and thereby fostering the *coordinator* to determine the next operation with enhanced contextual coherence.
- *Output specifications.* To ensure that atomic agents can accurately interpret the operation, we stipulate that the *coordinator* only outputs three items: the operation name, a brief instruction, and relevant context information.
- *One-shot example* for ambiguity reduction.

Atomic Agent Prompt consists of four parts:

- *Role & objective.* We also assign distinct roles to the atomic agents, along with descriptions of their jobs.
- *Operation execution rules.* We encourage each atomic agent to proactively and repeatedly invoke relevant tools to obtain sufficient information or conduct thorough tests.
- *Output specifications.* We stipulate that each atomic agent can only respond with a `function_call` with parameters to invoke a tool, or a `message` to quit the current operation.

• *Instruction & context information* from the *coordinator*.

We instruct atomic agents to invoke multiple tools until all necessary information is gathered. Once a "finished" `message` is generated, the atomic agent's activity is summarized and returned to the *coordinator* for further decision-making.

Remarks. Through our *coordinator-actor agentic architecture*, the *coordinator* can autonomously explore appropriate atomic operations using its reasoning abilities. As such, EAI workflows can be dynamically constructed to cope with dynamic environments with enhanced flexibility and adaptability.

C. Continuous Federated Tuning Paradigm

In the feasibility study (Fig. 2(b)), we also find that 55% of the tasks adopt overly simple EAI algorithms or outdated AI models, motivating us to continuously fine-tune embodied LLMs to enhance their reasoning capabilities in response to evolving EAI environments and contexts. Specifically, we focus on two aspects: 1) The *coordinator* should gain enhanced cognitive reasoning to autonomously determine the next appropriate atomic operation at each step. 2) Each atomic agent should execute its operation properly by leveraging its domain-specific reasoning capabilities to analyze EAI environments and perform precise manipulation. To achieve these goals, we adopt *modified* reinforcement fine-tuning (RFT) [14] to enhance agents' reasoning abilities in the EAI domain without user intervention. Different from supervised fine-tuning, RFT can enhance LLMs' reasoning capabilities in specific domains by tuning on smaller amounts of data [14].

RFT Procedure. Rather than forcing LLMs to generate precise answers, RFT relies on a grader that scores generated responses, followed by policy gradients to refine LLM parameters. RFT mainly consists of three stages: 1) *Grader design.* A grader is created to score the quality of LLM outputs. Generally, graders can take various forms, such as format checkers, Python code, or even LLMs [14]. 2) *Tuning dataset construction.* We first prepare a question set, Q, with an expected answer set, A_e, and input Q into the LLM, $\mathcal{L}$, to obtain a generated answer set, A_g. Accordingly, we prompt the grader, $\mathcal{G}$, to score each answer. The tuning dataset, T, is thus obtained as a set of questions, expected answers, generated answers, and scores, s_i. The entire process is:

$$\begin{aligned} &a_i^g = \mathcal{L}(q_i), \quad \forall q_i \in Q \\ &s_i = \mathcal{G}(q_i, a_i^g, a_i^e), \quad \forall a_i^g \in A_g,\ \forall a_i^e \in A_e \\ &T = \{(q_i, a_i^g, a_i^e, s_i), \cdots\} \end{aligned} \tag{1}$$

3) *Fine-tuning.* With the obtained training dataset, policy-gradient updates are adopted for LLM tuning:

$$\hat{g} = \frac{1}{|T|} \sum_{s_i \in T} \nabla_\theta \log s_i \big|_\theta \hat{A}, \quad \theta' = \theta + \eta \cdot \hat{g} \tag{2}$$

where $\hat{g}$ is policy gradient, $|T|$ is dataset cardinality, θ are LLM parameters, $\hat{A}$ are advantage estimates, and η is learning rate. After several epochs, RFT enhances the reasoning capabilities of LLMs by encouraging them to generate high-scoring outputs more frequently while suppressing low-scoring ones.

Challenge. `FSEAI` still faces two main challenges: 1) As agents continuously evolve, it becomes necessary to design multiple evolving graders to assess both the *coordinator* and atomic agents. Manually providing feedback from a global view demands substantial domain expertise and is highly time-consuming. 2) A single `FSEAI` agentic system typically operates within confined exploration spaces with limited perception ranges. As a result, a single agent may struggle to generalize to unseen and heterogeneous environments.

Solution. To address the first challenge, we propose an *ensemble mutual-actor-critic strategy* (Fig. 3 right) to enhance both the *coordinator* and atomic agents in a mutual-critique manner. Our insight is that both the *coordinator* and atomic agents can function as graders to criticize each other's behaviors from distinct perspectives. As such, they can progressively evolve through mutual reflection. To overcome the second challenge, we further integrate RFT with federated scoring, enabling multiple distributed embodied LLMs to learn from one another's perspectives and decisions.

Ensemble Mutual-Actor-Critic Strategy. We augment the actor-critic algorithm [15] with an ensemble updating strategy to refine both the *coordinator* and atomic agents continuously. In particular, to update atomic agents, we treat the *coordinator* as the critic (*i.e.*, grader) to evaluate atomic agent behaviors. In turn, to refine the *coordinator*, we regard atomic agents as three separate graders, each criticizing the *coordinator* from a distinct angle regarding their specific tasks. Thus, both the *coordinator* and atomic agents can be mutually updated as self-exploration progresses. Specifically, we prompt the *coordinator* to assign a score reflecting whether each atomic agent's response is relevant to the current context and the overarching EAI task. Tuning atomic agents is expressed as:

$$\hat{g} = \frac{1}{|D_a = \{s_1^a, s_2^a, \cdots\}|} \sum \nabla_\theta \log s_i^a \big|_\theta \hat{A}, \quad \theta' = \theta + \eta \cdot \hat{g} \tag{3}$$

where D_a contains the scores graded by the *coordinator*. To fine-tune the *coordinator*, we first design three tailored grader prompts for atomic agents to rate the current atomic operation and instruction generated by the *coordinator*. The grading prompt considers both the correctness and necessity of the *coordinator*'s decision. An ensemble score is then calculated via weighted aggregation, which can be expressed as:

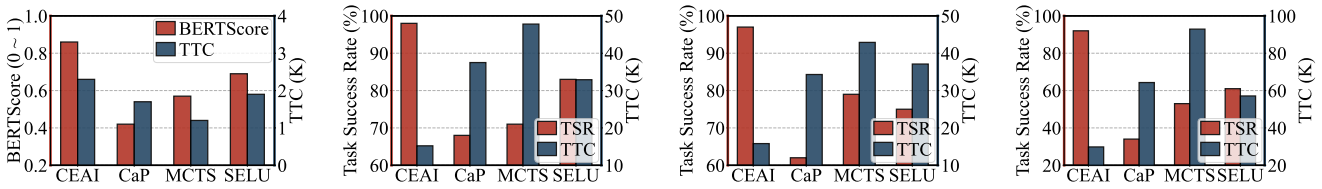
$$\begin{aligned} &s_i^c = \textstyle\sum_{o \in \mathcal{O}} w_o \cdot \rho_o, \quad s_i^c \in [0, 5] \\ &\rho_o = \begin{cases} s_i^o & \text{if the } \textit{coordinator}\text{'s operation = } o \\ 5 - s_i^o & \text{otherwise} \end{cases} \end{aligned} \tag{4}$$

where o is the current operation, w_o is its assigned weight, and s_i^o is the score rated by each atomic agent. If o does not match the atomic agent's operation, we compute the score as $5 - s_i^o$, as in this case, a higher score means that the atomic agent considers its own operation to be more appropriate. The *coordinator* is then updated using the ensemble score:

$$\hat{g} = \frac{1}{|D_c = \{s_1^o, s_2^o, \cdots\}|} \sum \nabla_\theta \log s_i^o \big|_\theta \hat{A}, \quad \theta' = \theta + \eta \cdot \hat{g} \tag{5}$$

Such an ensemble strategy allows bidirectional critique between the *coordinator* and atomic agents during self-exploration, thereby fostering reflection on accumulated context and enabling continuous, iterative self-evolution.

Federated Scoring. In the large environment, there are N `FSEAI` agentic systems operating simultaneously. After each interaction, we record and aggregate the ensemble scores in a federated manner to foster information sharing among multiple



(a) TSR on environment Q&A tasks (b) TSR on object grabbing tasks (c) TSR on object placing tasks (d) TSR on hybrid tasks

Fig. 4. The overall performance of FSEAI and the baselines across four distinct types of EAI tasks in the VirtualHome environment.

EAI agents. The aggregation process is expressed as:

$$\hat{s}^a = \sum_{j=1}^{N} s_j^a / N, \quad \hat{s}^o = \sum_{j=1}^{N} s_j^o / N \tag{6}$$

The aggregated scores $\hat{s}^a$ and $\hat{s}^o$ are further used for RFT.

Remarks. Our tuning paradigm fosters mutual critique between the *coordinator* and atomic agents for continual reasoning enhancement. Such bidirectional supervision and collaborative progress mitigate LLM hallucinations and facilitate a more robust self-evolution paradigm for adaptive agentic systems to handle ever-changing EAI environments.

IV. Evaluation

A. Experiment Setup

Implementation. We fully implement FSEAI using OpenAI's o4-mini for balanced reasoning and response latency. We equip the FSEAI agent with a web search engine, a database, a Python executor within a sandbox, and a tool builder that allows the agent to dynamically create new tools. We carefully design and implement all graders and adopt RFT following OpenAI's official guidelines [14]. We test FSEAI and the baselines in VirtualHome [7], which emulates 7 distinct household environments with numerous interactive items. Following [11], we create a test dataset containing 400 EAI tasks spanning across four types: environment Q&A, grabbing objects, placing objects, and hybrid tasks. We randomly split the augmented dataset into two parts: 20% for tuning and 80% for evaluation (RFT requires less tuning data). *The tuning and testing data have completely different EAI tasks.*

Baselines. We compare FSEAI with three SOTA baselines that rely on developer-predefined agentic workflows:

- Code-as-Policies (*CaP*) [3] formulates EAI task planning and robotic manipulation as a coding problem.
- *LLM-MCTS* [12] builds a world model and adopts Monte Carlo Tree Search to scale up planning.
- *SELU* [11] leverages reinforcement learning to enhance MLLMs' reasoning capabilities in EAI tasks.

Metrics. For environment Q&A, we compute **BERTScore** between the generated responses and the reference answers to assess token-level similarity. For the other three tasks, we measure the **Task Success Rate** (TSR), *i.e.*, the ratio of tasks that can be successfully executed by the agent without execution errors. We also record the **Total Token Cost** (TTC) of the embodied LLM throughout the entire process.

B. Overall Performance

Fig. 4 illustrates the overall performance of FSEAI and the baselines across four types of EAI tasks. We find that:

FSEAI can effectively handle diverse EAI tasks in heterogeneous and dynamic environments. On average, FSEAI achieves up to a 30% increase in BERTScore, a 42.6% rise in TSR, and a reduction of 41.6K in TTC. This indicates that FSEAI has better generalizability than the baselines, benefiting from our federated EAI agentic paradigm, where the FSEAI agent is encouraged to dynamically determine the next appropriate steps and share knowledge with others.

In environment Q&A tasks, baseline EAI agents consume 30.4% fewer tokens than FSEAI. After conducting a closer examination of the agentic workflows constructed by FSEAI, we find that FSEAI not only describes the surroundings at the agent's initial position but also proactively controls the agent to move around and gather more environmental information. Consequently, FSEAI achieves a substantially higher BERTScore than the baselines, indicating a certain degree of self-cognitive capability in aligning agent control and environmental information with users' EAI tasks.

In simple robotic manipulation tasks (*i.e.*, grabbing and placing objects), FSEAI outperforms the baselines in terms of both TSR and TTC. Specifically, FSEAI achieves a near-100% TSR and around 15K TTC on average. In contrast, the baseline agents incur approximately two to three times higher TTCs. Delving deeper into their intermediate outputs, we find that in most tasks, the baselines repeatedly execute the predefined "observe-plan-act" loop without making substantial progress towards the task goal, especially in dynamic environments. For example, an obstacle is moving around when CaP is navigating toward the target item. As a result, CaP collides with the obstacle and repeatedly attempts to move forward, ultimately causing the VirtualHome simulator to crash. On the contrary, FSEAI promptly adjusted its workflow by dynamically adding additional steps (*i.e.*, observe, reason, and move away) to effectively handle the environmental changes.

The performance gap expands as EAI tasks are more complex. As shown in Fig. 4(d), when handling hybrid tasks, the baselines cost dramatically more tokens (exceeding 60K) to finish a single task, with LLM-MCTC consuming nearly 100K tokens. The underlying reason is that hybrid EAI tasks demand more sophisticated reasoning capabilities and necessitate EAI agents to interact with environments multiple times. Due to limited reasoning abilities and a lack of self-awareness, the baselines tend to explore the environment blindly and repeatedly execute redundant actions, thereby accumulating excessive token costs, which in turn exacerbate long-context reasoning overhead and ultimately result in a vicious cycle.

Remarks. The superior performance and generalizability of FSEAI stems from its *coordinator-executor agentic archi-*

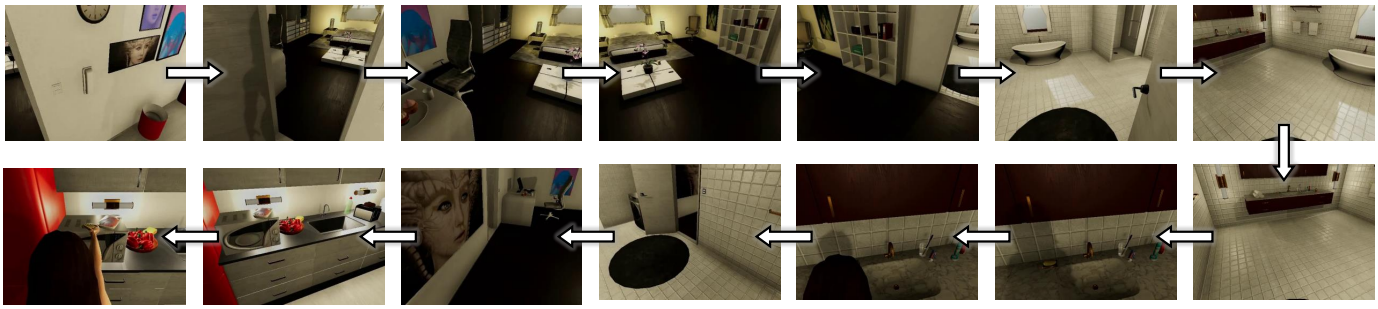


Fig. 5. Trajectories of the case study (first-person perspective of the agent in VirtualHome).

tecture for efficient labor division and workflow exploration, and its tuning paradigm that continuously enhances decision-making capabilities of EAI agents. More importantly, FSEAI exhibits flexibility and adaptability in handling diverse EAI tasks, unforeseen cases, and heterogeneous EAI environments.

C. Case Study

To better illustrate how FSEAI autonomously and dynamically constructs its workflow, we present a frame-by-frame, first-person perspective of the agent when tasked with "Put the soap on the microwave". Initially, the agent is located near the kitchen door that connects to the study room. The agent's trajectories during self-exploration are shown in Fig. 5.

1) observe I need to capture images to observe the room.
2) reason I cannot find a soap.
3) reason It seems that I'm currently in a kitchen and there is a study room nearby. Maybe I can first enter the room to gather additional observations.
4) act Turn left and move forward.
5) observe I need to capture images to observe the room.
6) reason The left side is obstructed by walls; therefore, turning right may provide a broader perceptual view.
7) act Turn right.
8) observe I should capture images for further observation.
9) reason It seems that there is a bathroom on the right. The soap is likely to be located in that room.
10) reason Based on my current location, I need to first turn right and then move forward to enter the bathroom.
11) act Turn right and move forward.
12) observe Capture images for further observation.
13) reason In front of me is a tub, and on the left is a washbasin. The soap is likely to be located on the basin.
14) reason I need to turn left and move forward.
15) act Turn left and move forward.
16) observe Capture images for further observation.
17) reason Bingo! The soap is right on the basin.
18) reason I can grab the soap and return to the kitchen.
19) act Grab the soap.
20) The agent executes similar operations: navigate back to the kitchen, find a microwave, and put the soap on it.

Notably, our FSEAI agent even automates the entire process of "observation via image capturing $\rightarrow$ image comprehension" by proactively generating a Python script to reduce reasoning overhead, since the agent finds that such combined steps are executed repeatedly. Moreover, the agent leverages an external multimodal LLM to interpret the captured images. Based on these findings, we can see that FSEAI's behavior closely resembles humans, exhibiting remarkable adaptability and consciousness of FSEAI in handling EAI environments.

V. Conclusion

We present FSEAI that leverages LLMs' reasoning capabilities to dynamically construct appropriate workflows for heterogeneous EAI tasks. With two crafted technical modules, our evaluations demonstrate the effectiveness, generalizability, flexibility, and adaptability of FSEAI in handling diverse and evolving EAI scenarios. Looking ahead, we envision FSEAI as a stepping stone toward more autonomous agentic systems capable of not only adapting to dynamic environments but also evolving through lifelong learning and reflection.

References

[1] L. Shen and Y. Zheng, "Poster: Towards federated embodied ai with feai," in *ACM MobiSys*, 2025, pp. 599–600.

[2] Q. Yang and Y. Zheng, "Neural enhanced underwater sos detection," in *IEEE INFOCOM*, 2024, pp. 971–980.

[3] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng, "Code as policies: Language model programs for embodied control," *arXiv preprint arXiv:2209.07753*, 2022.

[4] F. Wang, Y. Zhu, and B. Li, "Unraveling elevated data leakage in split learning for fine-tuning stable diffusion models," in *Asia CCS*, 2025.

[5] N. Su and B. Li, "Asynchronous federated unlearning," in *IEEE INFOCOM*, 2023, pp. 1–10.

[6] L. Shen, Q. Yang, K. Cui, Y. Zheng, X.-Y. Wei, J. Liu, and J. Han, "Fedconv: A learning-on-model paradigm for heterogeneous federated clients," in *ACM MobiSys*, 2024.

[7] X. Puig, K. Ra, M. Boben, J. Li, T. Wang, S. Fidler, and A. Torralba, "Virtualhome: Simulating household activities via programs," in *IEEE CVPR*, 2018, pp. 8494–8502.

[8] L. Shen, Q. Yang, Y. Zheng, and M. Li, "Autoiot: Llm-driven automated natural language programming for aiot applications," in *MobiCom*, 2025.

[9] H. Xu, L. Han, Q. Yang, M. Li, and M. Srivastava, "Penetrative ai: Making llms comprehend the physical world," in *HotMobile*, 2024.

[10] L. Shen, Q. Yang, X. Huang, Z. Ma, and Y. Zheng, "Gpiot: Tailoring small language models for iot program synthesis and development," in *SenSys*, 2025, pp. 199–212.

[11] B. Li, H. Jiang, Z. Ding, X. Xu, H. Li, D. Zhao, and Z. Lu, "Selu: Self-learning embodied mllms in unknown environments," *arXiv preprint arXiv:2410.03303*, 2024.

[12] Z. Zhao, W. S. Lee, and D. Hsu, "Large language models as commonsense knowledge for large-scale task planning," *NeurIPS*, vol. 36, pp. 31 967–31 987, 2023.

[13] X. Zhang, Z. Huang, E. O. Taga, C. Joe-Wong, S. Oymak, and J. Chen, "Efficient contextual llm cascades through budget-constrained policy learning," *NeurIPS*, vol. 37, pp. 91 691–91 722, 2024.

[14] "Reinforcement fine-tuning," 2024. [Online]. Available: https://platform.openai.com/docs/guides/reinforcement-fine-tuning

[15] V. Konda and J. Tsitsiklis, "Actor-critic algorithms," *NeurIPS*, 1999.