Semantic Skill Grounding for Embodied Instruction-Following in Cross-Domain Environments

Sangwoo Shin[1*], **Seunghyun Kim**[1*], **Youngsoo Jang**[2], **Moontae Lee**[2,3], **Honguk Woo**[1†]
Department of Computer Science and Engineering, Sungkyunkwan University[1]
LG AI Research[2] University of Illinois Chicago[3]
jsw7460@skku.edu, kimsh571@skku.edu
youngsoo.jang@lgresearch.ai, moontae.lee@lgresearch.ai, hwoo@skku.edu

Abstract

In embodied instruction-following (EIF), the integration of pretrained language models (LMs) as task planners emerges as a significant branch, where tasks are planned at the skill level by prompting LMs with pretrained skills and user instructions. However, grounding these pretrained skills in different domains remains challenging due to their intricate entanglement with the domain-specific knowledge. To address this challenge, we present a semantic skill grounding (SemGro) framework that leverages the hierarchical nature of semantic skills. SemGro recognizes the broad spectrum of these skills, ranging from short-horizon low-semantic skills that are universally applicable across domains to long-horizon rich-semantic skills that are highly specialized and tailored for particular domains. The framework employs an iterative skill decomposition approach, starting from the higher levels of semantic skill hierarchy and then moving downwards, so as to ground each planned skill to an executable level within the target domain. To do so, we use the reasoning capabilities of LMs for composing and decomposing semantic skills, as well as their multimodal extension for assessing the skill feasibility in the target domain. Our experiments in the VirtualHome benchmark show the efficacy of SemGro in 300 cross-domain EIF scenarios.

1 Introduction

In the area of embodied instruction-following (EIF), the quest to develop systems capable of understanding user instructions and achieving tasks in a physical environment has been a pivotal aspiration for general artificial intelligence. Recent advancements in language models (LMs) have demonstrated their extensive and impressive application in EIF, where LMs are employed as task planners, drawing on their commonsense knowledge of the physical world (Yang et al., 2023; Padmakumar et al., 2023; Pantazopoulos et al., 2023; Yun et al., 2023; Logeswaran et al., 2022).

The process of LM-based task planning in EIF unfolds via a two-stage pipeline: (i) Interpretable skills (i.e., semantic skills) are acquired, and (ii) given a user instruction, the LM is prompted with the instruction and the skill semantics to identify appropriate skills to perform. To ensure that such planned skill can be executed within the domain context the agent is placed in, several works have focused on incorporating environmental observations into task planning (Singh et al., 2023; Song et al., 2023; Xiang et al., 2023; Lin et al., 2023b,a; Shinn et al., 2023; Carta et al., 2023). Yet, EIF in cross-domain environments has not been fully explored. In cross-domain settings, the complexity goes beyond logical task planning by LMs; it necessitates the adaptation of pretrained skills to align with the specific characteristics and limitations of the target physical environment. This adaptation challenge stems from the inherent domain specificity of semantic skill acquisition, calling for a strategy that disentangles these learned behaviors from their original domain contexts and reapplies them in new domains.

In response to this challenge, we present a semantic skill grounding framework SemGro. We observe a trade-off between semantic depth and domain-agnostic executability within the skill hierarchy. High-level skills, such as “prepare a meal”, embody a richer semantic content and long-horizon behavior compared to low-level skills, such as “grab a cup”. However, these high-level skills tend to be effective only within specific domains, as their execution heavily depends on a multitude of environmental factors, unlike their low-level counterparts which can be universally applicable across domains. SemGro capitalizes on this insight by employing an iterative skill grounding scheme, designed to progressively decompose pretrained skills

[*]Equally contributed to this work
[†]Honguk Woo is the corresponding author

to fit in the cross-domain EIF environment as well as to fully utilize the reasoning strengths of LMs.

To be specific, the iterative mechanism of SemGro involves two procedures: (i) task planning with skills that align with a given instruction (in Section 3.2) and (ii) assessing the executability of planned skills (in Section 3.3). In SemGro, a task planner employs in-context learning of LMs to interpret a given instruction and generate a series of semantic skills for the instruction through iterative skill grounding and decomposition. Concurrently, a skill critic employs the multi-modal extension of LMs to evaluate the applicability and execution potential of these skills within target domains. When the critic deems a planned skill as unsuitable for execution, the planner responds by generating an alternative, detailed instruction. This instruction is intended to be carried out through a sequence of lower-level skills that align with the previously planned skill. Through the collaborative interaction between the task planner and the skill critic, SemGro navigates the semantic hierarchy of skills to establish the best trade-off between the logical relevance of skills to a given instruction and the skill feasibility in cross-domain environments.

Using the VirtualHome benchmark (Puig et al., 2018) to evaluate EIF agents (Singh et al., 2023; Hazra et al., 2023; Chen et al., 2023; Xiang et al., 2023), we demonstrate the advantages of SemGro for cross-domain EIF scenarios. For instance, SemGro outperforms the state-of-the-art task planning framework LLM-Planner (Song et al., 2023) by 23.97% in task completion performance.

Our contributions are three-fold: **(i)** We present a novel framework SemGro to address the issue of cross-domain skill grounding in EIF. **(ii)** We devise an iterative skill grounding algorithm, leveraging the reasoning capabilities of LMs and their multi-modal extension. **(iii)** We conduct an intensive evaluation of SemGro in 300 cross-domain scenarios of the VirtualHome benchmark, complemented by ablation studies to explore the efficacy of SemGro.

2 Preliminaries

2.1 Language Models for Task Planning

For complex embodied tasks, prior works have leveraged the notion of semantic skills, which represent interpretable patterns of expert behavior. These skills have a wide spectrum, ranging from short-horizon basic low-level skills (e.g., "stir the drink") to long-horizon high-level skills

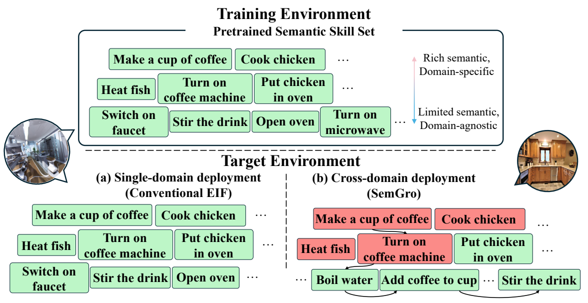
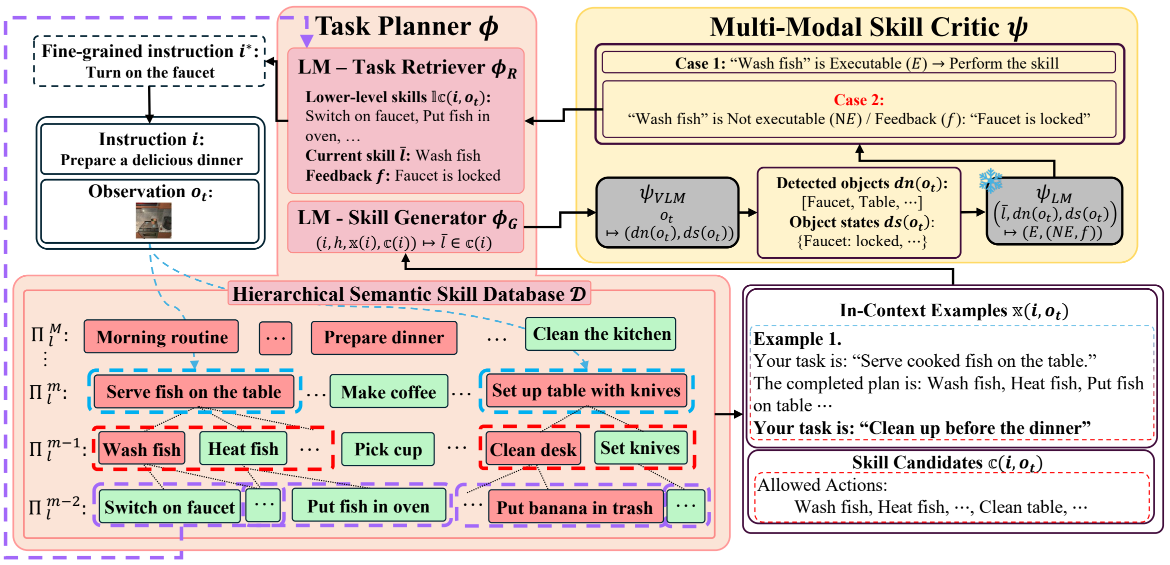


Figure 1: Cross-domain EIF. (a) Conventional approaches focus on **single-domain** deployment, where the pretrained skills are effective within only that particular domain (marked in the green boxes). (b) Our SemGro caters to **cross-domain** deployment, where the pretrained skills may be inexecutable (marked by red boxes) in the target environment. For this, SemGro explores the semantic hierarchy of skills.

(e.g., "prepare a meal"). These diverse skills $\Pi_l = \{\pi_l \,|\, l : \text{skill semantic}\}$ can be efficiently established within a training environment, through deterministic algorithms, reinforcement learning (RL), or imitation learning. The integration of semantic skills and the reasoning capabilities of LMs enables the zero-shot adaptation of EIF agents, without additional environment interactions or data collection efforts (Song et al., 2023; Singh et al., 2023; Brohan et al., 2023c). In these approaches, an LM is prompted with the semantics of pretrained skills $L_\pi = \{l : \pi_l \in \Pi\}$ along with a specific instruction i from the user. Subsequently, the LM produces a series of semantic skills. The execution of these skills facilitates the decomposition of the complex task associated with instruction i into smaller, achievable skill-level subtasks.

2.2 Cross-Domain Skill Grounding

While several works have explored the use of LMs as task planners, they primarily focus on single-domain agent deployment, wherein skills are both acquired and used within the same environment setting. In contrast, the cross-domain setting presents unique challenges for embodied task planning, as the feasibility of skill execution in the environment differs, necessitating the adaptation of pretrained skills to the specific domain contexts of the target environment. Consider the skill "Make a cup of coffee", as illustrated in Figure 1, learned in a high-tech kitchen environment equipped with voice-activated appliances, a digital inventory system, and automated utensil handlers. In this environment, the skill entails complex interactions with



Task Planner ϕ
Fine-grained instruction i^*: Turn on the faucet
Instruction i: Prepare a delicious dinner
Observation o_t:
LM – Task Retriever ϕ_R
Lower-level skills $\mathbb{lc}(i, o_t)$: Switch on faucet, Put fish in oven, …
Current skill $\bar{l}$: Wash fish
Feedback f: Faucet is locked
LM - Skill Generator ϕ_G
$(i, h, \mathbb{x}(i), \mathbb{c}(i)) \mapsto \bar{l} \in \mathbb{c}(i)$
Multi-Modal Skill Critic ψ
Case 1: "Wash fish" is Executable (E) → Perform the skill
Case 2: "Wash fish" is Not executable (NE) / Feedback (f): "Faucet is locked"
ψ_{VLM} $o_t \mapsto (dn(o_t), ds(o_t))$
Detected objects $dn(o_t)$: [Faucet, Table, ⋯]
Object states $ds(o_t)$: {Faucet: locked, ⋯}
ψ_{LM} $(\bar{l}, dn(o_t), ds(o_t)) \mapsto (E, (NE, f))$
Hierarchical Semantic Skill Database $\mathcal{D}$
Π_l^M:
Morning routine
Prepare dinner
Clean the kitchen
Π_l^m:
Serve fish on the table
Make coffee
Set up table with knives
Π_l^{m-1}:
Wash fish
Heat fish
Pick cup
Clean desk
Set knives
Π_l^{m-2}:
Switch on faucet
Put fish in oven
Put banana in trash
In-Context Examples $\mathbb{x}(i, o_t)$
Example 1.
Your task is: "Serve cooked fish on the table."
The completed plan is: Wash fish, Heat fish, Put fish on table ⋯
Your task is: "Clean up before the dinner"
Skill Candidates $\mathbb{c}(i, o_t)$
Allowed Actions:
Wash fish, Heat fish, ⋯, Clean table, ⋯

Figure 2: Overview of SemGro. Given a user instruction i and detected object names from visual observations o_t, SemGro selects k entries (marked in blue dotted entries) from the hierarchical skill database $\mathcal{D}$. Within $\mathcal{D}$, skills are labeled based on their executability in the target domain: non-executable skills in red boxes and executable ones in green. From these selections, in-context examples $\mathbf{x}(i, o_t)$ and skill candidates $\mathbf{c}(i, o_t)$ are obtained from the constituent low-level skills (marked in red dotted entries in $\mathcal{D}$) of these entries. The skill generator ϕ_G is then prompted with $\mathbf{x}(i, o_t)$, $\mathbf{c}(i, o_t)$, i, and skill execution history h to generate the skill $\bar{l} \in \mathbf{c}(i, o_t)$. The LM-part ψ_{LM} of the critic ψ assesses the feasibility of the planned skill $\bar{l}$ based on the detected objects $dn(o_t)$ and their states $ds(o_t)$ as identified by the VLM-part ψ_{VLM}. If $\bar{l}$ is deemed executable (Case 1), the agent proceeds to interact with the environment by performing the skill. Otherwise (Case 2), the task retriever ϕ_R generates a more fine-grained instruction i^*, derived from lower-level skills $\mathbf{lc}(i, o_t)$ (marked in purple dotted entries in $\mathcal{D}$) and feedback f explaining the inexecutability. The i^* is treated as a new instruction, prompting the framework to iterate the process.

technology to efficiently prepare a cup of coffee. However, when a robot is deployed in a traditional kitchen devoid of such advanced features, it encounters a significantly different domain context. The skill "Make a cup of coffee", initially tailored for the environment with high-tech gadgets, now faces significant execution challenges due to the unavailability of similar technological supports.

To address this gap, we introduce **cross-domain skill grounding**, a process aimed at adapting pretrained skills $\pi_l \in \Pi$ for diverse domains. This process involves not only reinterpreting the original skill but also deconstructing it into a series of achievable lower-level subtasks, such as "Boil water", "Add coffee to cup", "Pour water", and "Stir the drink". Through cross-domain skill grounding, we ensure that embodied agents can perform a vast set of skills within domains that differ significantly from those in which the skills were initially learned.

3 Approaches

We utilize the intrinsic semantic hierarchy of skills, a structure derived from their compositional nature; high-level semantic skills are established by chaining short-horizon low-level skills. High-level skills, characterized by rich semantics, closely align with the abstract nature of user instructions. This alignment in abstraction levels facilitates the task planning process, providing the LM with skills that are directly beneficial for interpreting and accomplishing the instructions. However, the execution of these long-horizon high-level skills is contingent upon the precise alignment of various environmental and domain-specific conditions. This precludes their adaptability in cross-domain settings. In contrast, low-level skills exhibit a robust capacity for execution that is not as tightly bound to specific domain contexts, offering greater adaptability across diverse domains. Nonetheless, the sheer volume of low-level skills and their limited semantic depths pose additional challenges for task planning.

The core of SemGro lies in the hierarchical task planning scheme via iterative skill grounding. The framework initiates task planning with high-level rich-semantic skills, and then progressively decomposes each of these planned high-level skills into a series of low-level skills executable within the domain of the target environment. This can be seen as finding an optimal intermediate semantic representation for skills pertinent to a given instruction, exploring a balance between their logical relevance and actual feasibility in the target environment. Such a balance ensures that the planned skills not only match the conceptual demands of the given instruction but are also effective within the limitations and resources of the target environment.

3.1 Overall Framework

As illustrated in Figure 2, SemGro is structured with (a) a task planner and (b) a multi-modal skill critic, which collaboratively ground skills for user instructions in cross-domains. At each skill-step, the task planner produces an appropriate semantic skill by prompting an LM with information retrieved from a skill database. The skill critic then evaluates each skill from the skill planner by prompting a multi-modal LM with visual observations from the environment. When evaluated as unsuitable, the task planner decomposes the skill into smaller, more general subtasks using another retrieval from the database. Through this iterative process, the framework systematically assembles a sequence of semantic skills, executable within the domain context of the target environment.

3.2 LM-based Task Planner

For LM-based task planners, the development of effective prompts is essential to fully harness the reasoning capabilities of LMs. We consider two aspects: (i) selecting skill candidates that facilitate the skill-level task decomposition and (ii) incorporating in-context examples applied to similar instructions. To capture these aspects, we construct a hierarchical semantic skill database, leveraging the reasoning prowess of LMs. Integrated with a kNN retriever, this database not only assists in identifying suitable high-level skill candidates for task planning but also facilitating the retrieval of pertinent in-context examples that can further aid the planning process. The task planner, enriched with this selection of skill candidates and in-context examples, ensures the application of suitable skill semantics in cross-domains.

Hierarchical semantic skill database. For the hierarchical semantic skill database $\mathcal{D}$, we use a structured approach to semantic skill training, similar to (Zhang et al., 2023). The skill acquisition initiates with a set of short-horizon basic low-level skills $\Pi_l^1 = \{\pi_l^{(1,1)}, \cdots, \pi_l^{(1,N_1)}\}$ in the training environment, which can be acquired via RL or imitation learning. Under the LM guidance, the agent produces skill chains from Π_l^1, achieving high-level long-horizon skills Π_l^2. This bootstrapping process iteratively expands to form a comprehensive set of skill levels $\Pi_l = \{\Pi_l^1, \cdots, \Pi_l^M\}$. This collection, encompassing skills from low-level to high-level, is used to establish the skill database $\mathcal{D}$:

$$\begin{cases} \Pi_l^m = \{\pi_l^{(m,1)}, \cdots, \pi_l^{(m,N_m)}\} \\ \mathcal{D} := \{e^{(m,n)} : 1 \leq m \leq M, 1 \leq n \leq N_m\} \\ e^{(m,n)} := (l^{(m,n)}, dn(e^{(m,n)}), p(e^{(m,n)})) \\ l^{(m,n)} := \text{Skill semantic of } \pi_l^{(m,n)} \\ p(e^{(m,n)}) := (e^{(m-1,n_{j_1})}, \cdots). \end{cases} \tag{1}$$

The entry $e^{(m,n)}$ of $\mathcal{D}$ includes the skill semantic $l^{(m,n)}$ of $\pi_l^{(m,n)}$, detected object names $dn(e^{(m,n)})$ collected during training of $\pi_l^{(m,n)}$, and a one-step low-level semantic skill planning $p(e^{(m,n)}) = (e^{(m-1,n_{j_1})}, \cdots)$, in that the skill $\pi_l^{(m,n)}$ is acquired by chaining of $(\pi_l^{(m-1,n_{j_1})}, \cdots)$. To train the skill critic (Section 3.3), we also establish a dataset:

$$\mathcal{D}_O = \{o_t^{(TR)}, dn(o_t^{(TR)}), ds(o_t^{(TR)}) : t\} \tag{2}$$

of visual observations $o_t^{(TR)}$ in the training environment, detected object names $dn(o_t^{(TR)})$ and object physical states $ds(o_t^{(TR)})$ at each timestep t during the establishment of Π_l. To obtain such object names (e.g., "microwave") and physical states (e.g., "microwave: closed"), we use an open-vocabulary detector tailored for the training environment.

Skill generator. For a given user instruction i and target environment observation o_t, the LM-based skill generator ϕ_G integrates skill candidates $\mathbf{c}(i, o_t)$, in-context examples $\mathbf{x}(i, o_t)$, and the skill execution history h to generate the skill semantic $\bar{l}$ which is most likely to be executed at the present:

$$\phi_G : (i, h, \mathbf{x}(i, o_t), \mathbf{c}(i, o_t)) \mapsto \bar{l} \in \mathbf{c}(i, o_t). \tag{3}$$

To derive these in-context examples and skill candidates, we use a kNN retriever that selects entries $\mathbf{e}(i, o_t) = \{e^{(m_1,n_1)}, \cdots, e^{(m_k,n_k)}\}$ in $\mathcal{D}$. The retrieval process calculates similarity by combining the instruction i and the visual context of o_t into a single query. It evaluates $\langle i, l^{(m,n)}\rangle + \langle dn(e^{(m,n)}), o_t\rangle$ to determine the k entries with the highest similarity scores, where $\langle \cdot, \cdot \rangle$ denotes the cosine similarity. Here, $l^{(m,n)}$ is the skill semantic and $dn(e^{(m,n)})$ is the detected object names associated with entry $e^{(m,n)}$. These retrieved entries are then used to define one-step lower-level plans as in-context examples $\mathbf{x}(i, o_t)$ and one-step lower-level skill semantics $\mathbf{c}(i, o_t)$ as the planning candidates for the skill generator ϕ_G:

$$\begin{cases} \mathbf{e}(i, o_t) = \{(e^{(m_1,n_1)}, \cdots, e^{(m_k,n_k)})\} \\ \mathbf{x}(i, o_t) = \{(l^{(m_1,n_1)}, p_l(e^{(m_1,n_1)}), \cdots\} \\ \mathbf{c}(i, o_t) = \{l^{(m_1-1,n_{j_1})}, \cdots\}. \end{cases} \tag{4}$$

Here, $p_l(e^{(m,n)})$ is the semantics in $p(e^{(m,n)})$.

Task retriever. If the planned skill semantic $\bar{l}$ from the skill generator ϕ_G is identified as inexecutable by the skill critic, the LM-based task retriever ϕ_R steps in to generate a detailed, fine-grained language instruction i^*. This new instruction i^* is generated in response to the feedback f regarding the rationale for this inexecutability and lower-level skill semantics $\mathbf{lc}(i, o_t)$ that are constituent of the original skill semantic candidates $\mathbf{c}(i, o_t)$ in Eq (4):

$$\begin{cases} \phi_R : (\bar{l}, f, \mathbf{lc}(i, o_t)) \mapsto i^* \\ \mathbf{lc}(i, o_t) = \bigcup_{j=1}^{k} \{p_l(e') : e' \in p(e^{(m_j,n_j)})\}. \end{cases} \tag{5}$$

This instruction i^* guides the framework to perform smaller, more manageable subtasks by triggering the reiteration of skill generation in Eq (3).

3.3 Multi-Modal Skill Critic

For observation o_t and skill semantic $\bar{l}$ derived from the skill generator ϕ_G (in Eq (3)), the skill critic ψ assesses the execution feasibility of skill $\pi_{\bar{l}}$:

$$\psi : (o_t, \bar{l}) \mapsto c \in \{\text{E}, (\text{NE}, f)\}. \tag{6}$$

Here, E specifies that the skill is executable, while NE denotes inexecutability with language feedback f. The critic ψ includes the vision-language model (VLM)-part ψ_{VLM} and LM-part ψ_{LM}. Here, ψ_{VLM} identifies object names $dn(o_t)$ and states $ds(o_t)$ from o_t, and ψ_{LM} utilizes this information to determine the executability of the skill $\pi_{\bar{l}}$:

$$\begin{cases} \psi = \psi_{\text{LM}} \circ \psi_{\text{VLM}} : (o_t, \bar{l}) \mapsto \{\text{E}, (\text{NE}, f)\} \\ \psi_{\text{VLM}} : o_t \mapsto (dn(o_t), ds(o_t)) \\ \psi_{\text{LM}} : (\bar{l}, dn(o_t), ds(o_t)) \mapsto \{\text{E}, (\text{NE}, f)\}. \end{cases} \tag{7}$$

We use the dataset $\mathcal{D}_O$ (in Eq (2)) to fine-tune the physically grounded VLM (Gao et al., 2023). Our ablation study (Section 4.3) specifies the benefits of this LM's multi-modal extension with VLMs in cross-domains.

Algorithm 1 Iterative Skill Grounding for EIF

```
1: User instruction $i$, skill database $\mathcal{D}$
2: Task planner $\phi = (\phi_G, \phi_R)$
3: Skill critic $\psi = \psi_{\text{LM}} \circ \psi_{\text{VLM}}$
4: Instruction stack $I \leftarrow [i]$
5: Execution history $H \leftarrow []$
6: $t \leftarrow 0$, $o_t \leftarrow env$.reset()
7: repeat
8:    $i \leftarrow I$.pop(), $h \leftarrow H$.pop()
9:    $\mathbf{e}(i, o_t) \leftarrow k$ entries via kNN retriever
10:   $\mathbf{x}(i, o_t) \leftarrow$ in-context examples (Eq (4))
11:   $\mathbf{c}(i, o_t) \leftarrow$ skill candidate (Eq (4))
12:   $\bar{l} \leftarrow \phi_G(i, h, \mathbf{x}(i, o_t), \mathbf{c}(i, o_t))$ (Eq (3))
13:   $c \leftarrow \psi(o_t, \bar{l})$ (Eq (6))
14:   if $c = \text{E}$ then
15:      $I \leftarrow I$.append($i$), $H \leftarrow H$.append($\bar{l}$)
16:      $t \leftarrow t + 1$
17:      $o_t \leftarrow env$.step($\bar{l}$)
18:   else if $c = (\text{NE}, f)$ then
19:      $\mathbf{lc}(i, o_t) \leftarrow$ lower skills (Eq (5))
20:      $i^* \leftarrow \phi_R(\bar{l}, f, \mathbf{lc}(i, o_t))$(Eq (5))
21:      $I \leftarrow I$.append($i$)
22:      $I \leftarrow I$.append($i^*$)
23:   end if
24: until $env$.done()
```

The iterative skill grounding scheme of SemGro is illustrated in Algorithm 1.

4 Experiments

Environment. We use VirtualHome, a realistic household activity simulator and benchmark. For EIF agents, we define 3949 low-level skills (i.e., Π_1 in Section 3.2) based on the combination of behaviors (e.g., "grasp") and objects (e.g., "apple"). For cross-domain evaluations, we use 15 distinct environment settings, categorized by one of domain shift types: **OL** involves changes in the placements and locations of objects; **PA** presents changes in the physical attributes of objects; **RS** is characterized by different room structures and visual attributes. These cross-domain contexts necessitate the refinement of certain pretrained high-level skills into low-level skills for successful execution. Consider a high-level skill "cook salmon" intended to fulfill the user instruction "prepare delicious food for me". If this skill was trained in a setup where salmon is always prepared on a cutting board, it might not

Instruction	Cross Domain	LLM-Planner SR	LLM-Planner CGC	LLM-Planner Plan	SayCan SR	SayCan CGC	SayCan Plan	ProgPrompt SR	ProgPrompt CGC	ProgPrompt Plan	SemGro SR	SemGro CGC	SemGro Plan
Abstract Noun	OL	20.51%	28.20%	43.59%	30.77%	34.19%	38.75%	17.95%	29.49%	34.90%	53.85%	67.31%	70.51%
	PA	20.69%	35.63%	31.03%	17.24%	25.86%	27.24%	27.58%	37.93%	46.38%	52.17%	68.84%	71.74%
	RS	16.22%	22.97%	29.73%	18.93%	32.48%	39.74%	13.51%	25.68%	29.28%	56.52%	73.91%	85.51%
Abstract Verb	OL	28.57%	38.10%	45.24%	26.19%	40.48%	46.03%	28.57%	39.29%	39.29%	53.57%	67.86%	70.04%
	PA	25.93%	44.44%	55.86%	19.23%	26.28%	33.07%	40.74%	54.32%	54.94%	61.54%	75.64%	80.45%
	RS	30.95%	44.84%	39.81%	23.81%	37.70%	35.58%	35.71%	51.19%	40.61%	57.14%	71.43%	80.16%
Structured	OL	30.77%	35.90%	39.89%	38.89%	41.20%	46.30%	23.08%	30.77%	38.46%	51.28%	64.10%	69.23%
	PA	25.93%	46.30%	48.89%	26.09%	39.13%	50.94%	15.38%	26.92%	44.87%	46.15%	60.26%	61.86%
	RS	38.46%	42.31%	49.72%	25.64%	46.58%	45.44%	21.05%	30.26%	38.16%	43.59%	61.11%	70.51%
Long Horizon	OL	25.81%	40.32%	48.92%	41.94%	46.24%	46.59%	35.48%	43.55%	53.58%	54.84%	64.52%	68.81%
	PA	29.63%	58.64%	55.80%	40.74%	70.37%	58.09%	44.44%	64.81%	57.22%	62.96%	74.07%	61.11%
	RS	25.93%	39.51%	38.27%	35.48%	46.77%	41.22%	30.95%	48.81%	38.22%	51.61%	60.21%	68.64%

Table 1: Cross-domain performance in VirtualHome. The success rate (**SR**), completed goal conditions (**CGC**), and planning accuracy (**Plan**) are measured with 3 different seeds for each cross-domain scenario.

work in **OL** cross-domain scenarios where salmon is instead found inside a refrigerator. In this case, it becomes necessary to decompose the skill into a sequence of lower-level executable skills such as "go to the refrigerator", "grasp the refrigerator handle", among others, ensuring the skill be adapted to the new domain context.

Instructions. Given 5 evaluation tasks, we use 4 different instruction types per task: **Abstract Nouns** and **Abstract Verbs** focus on instruction queries involving abstract nouns and verbs, respectively; **Structured** language queries include precise verbs; **Long-Horizon** queries involve multiple reasoning steps. These instruction types are used to test various linguistic aspects of task interpretation and execution, similar to (Brohan et al., 2023c). By combining 15 environment settings, 5 tasks, and 4 types of instructions, we establish a comprehensive set of 300 distinct unseen cross-domain scenarios.

Evaluation metrics. We use several evaluation metrics, consistent with previous works (Singh et al., 2023; Song et al., 2023). **SR** measures the percentage of successfully completed tasks, where a task is considered to be completed if its all goal conditions are completed; **CGC** measures the percentage of completed goal conditions; **Plan** measures the percentage of the planned skill sequence that continuously aligns with the ground-truth sequence from the beginning.

Baselines. We compare our SemGro with three baselines, each leveraging an LM as a task planner but with distinct approaches to filter executable skill candidates, including **SayCan** (Brohan et al., 2023c) which employs an offline RL dataset collected from the training environment to learn an affordance function; **LLM-Planner** (Song et al., 2023) which leverages templatized semantic skills; **ProgPrompt** (Singh et al., 2023) which uses human-crafted programmatic assertion syntax to verify the pre-conditions of skill execution and respond to the failed cases with recovery.

4.1 SemGro Implementation

For the kNN retriever, we configure k to be 10 and utilize the sentence-transformer (Reimers and Gurevych, 2019) to compute the similarity between the instruction and skill semantics in the database. The task planner ϕ and the LM-part ψ_{LM} of the skill critic are both implemented using GPT-3.5 API with the temperature hyperparameter set to 0. For fair comparisons, the same GPT-3.5 configuration is used across all baseline task planners. More details, including the baseline implementation, are in the Appendix.

4.2 Performance

Overall performance. Table 1 compares the cross-domain performance, achieved by our SemGro and the baselines (LLM-Planner, SayCan, ProgPrompt). SemGro outperforms the baselines in all cross-domain scenarios, achieving average performance gains of 25.02%, 26.83%, and 27.65% over the most competitive baseline SayCan in **SR** and **CGC**, and LLM-Planner in **Plan**, respectively. The exceptional performance of SemGro highlights its superiority in hierarchical task planning via iterative skill grounding across unseen domains.

Interestingly, for long-horizon instruction scenarios, there are several instances where **SR** and **CGC** metrics exceed **Plan**. In the **PA** cross-domain context, SemGro achieves **Plan** of 61.11%, while **SR** and **CGC** reach 62.96% and 74.07%, respectively. This phenomenon, reminiscent of solving math problems in an NLP context (Lightman et al., 2023; Fu et al., 2023; Imani et al., 2023), suggests that tasks requiring multiple reasoning steps can be accomplished through diverse skill sequences.

Executable skill identification. We investigate the efficacy of SemGro in generating executable skills. To this end, we introduce the **Exec** metric, which quantifies the percentage of the planned semantic skills that are executable in a given domain, regardless of their impact on task completion. In Table 2, SemGro surpasses the most competitive baseline ProgPrompt by 17.99% in **Exec** at average. This specifies our framework's proficiency in discerning the varying conditions of cross-domain environments for skill grounding. This ability is attributed to the skill critic which uses the domain-agnostic representation capabilities of the adopted VLM.

Cross Domain	LLM-Planner	SayCan	ProgPrompt	SemGro
OL	40.28%	60.22%	68.29%	85.05%
PA	46.46%	68.18%	72.55%	97.40%
RS	43.48%	69.41%	67.74%	79.12%

Table 2: Executable skill identification

Iterations for cross-domain skill grounding. We investigate the number of iterations used for skill grounding in the target domain across varying degrees of domain shift. To quantify the degree of shift, we calculate the number of high-level skills that must be decomposed into different series of low-level skills for execution in the target domain, compared to the training environment. The degree of shift is categorized into 4 levels. 'None' indicates no domain shift, while 'Small,' 'Medium,' and 'Large' denote increasing degrees of shift. The detailed criterion for quantifying domain shift degrees can be found in Appendix B. To accurately measure the iterations induced by domain shift, distinctions are made between skills rendered inexecutable due to the immediate observational state (Obs. & Dom. in the row name) and those affected by domain disparities (Dom.). Table 3 reveals a positive correlation between the domain shift degrees and the number of iterations. This specifies the benefits of SemGro in handling cross-domain settings with varying degrees of domain shift, progressively searching the appropriate abstraction level of skills.

4.3 Ablation Studies

Skill hierarchy for planning. To verify the effectiveness of the iterative skill grounding scheme, we conduct task planning with the fixed abstraction level of skills, bypassing the executability check of the skill critic. In Table 4, SG-L and SG-H employ the lowest-level skills Π_l^1 and highest-level skills Π_l^M for task planning, respectively, while SG-M employs middle-tier skills $\{\Pi_l^m : 1 < m < M\}$. As shown, SG-H exhibits the most accurate task planning (high **Plan**), while its planned skills are rarely executable (low **Exec**) due to domain shift. Conversely, SG-L, equipped with potentially executable skills, shows unreliable task planning due to the abstraction level mismatch with the instruction. Although SG-M employs skills with moderate semantic content and executability by adhering to a fixed abstraction level, it falls short in performance. SemGro adaptively identifies skill semantics at the intermediate level, through iterative downward decomposition within the semantic hierarchy of skills. This strategy not only facilitates task planning but also achieves robust executions in new domains.

	None	Small	Medium	Large
Obs. & Dom.	3.73	5.53	5.80	6.53
Dom.	0.87	1.64	1.96	2.64

Table 3: The number of iterations w.r.t. domain shift

Cross Domain		SG-L	SG-M	SG-H	SemGro
OL	Plan	46.30%	65.93%	89.26%	82.96%
	Exec	93.05%	75.64%	56.15%	92.27%
PA	Plan	48.45%	69.78%	80.56%	78.44%
	Exec	91.80%	78.76%	57.32%	91.02%
RS	Plan	35.56%	48.15%	87.04%	85.19%
	Exec	89.72%	73.93%	44.92%	87.03%

Table 4: Skill hierarchy for planning

In-context example retrieval. In Table 5, we conduct an ablation study on the kNN retriever by varying the number k of retrieved entries in Eq (4). Furthermore, we test a random-selection retriever that selects 10 examples at random (Random in the column name) as well as another variant that calculates the similarity based solely on instructions without observation information ($\mathbf{k} = \mathbf{10} \setminus O$ in the column name). For the random-selection retriever, skill candidates are determined using the kNN retriever to precisely assess the impact of in-context examples. As observed, selecting an appropriate number of examples ($k = 10$) leads to performance gain in **Plan** by 30.89% and 7.87% compared to the random retriever and $\mathbf{k} = \mathbf{10} \setminus O$, respectively. Furthermore, we observe an increase in planning performance when k ranges from 5 to 10, but beyond this range, larger values do not yield further enhancements. This implies that including skill candidates unrelated to the task may deteriorate the task planning.

Cross Domain	Random ($\mathrm{k}=10$)	$\mathrm{k}=5$	$\mathrm{k}=10$	$\mathrm{k}=15$	$\mathrm{k}=10\backslash O$
OL	29.63%	48.51%	69.63%	47.04%	59.25%
PA	65.56%	61.25%	81.94%	79.86%	80.56%
RS	34.07%	58.15%	70.37%	64.81%	58.52%

Table 5: In-context example retrieval

The dual-model structure for skill critic. As explained in Section 3.3, we structure the skill critic ψ with two models ψ_{VLM} and ψ_{LM}. To validate this dual-model structure, we conduct an ablation study using a single VLM for the skill critic. Specifically, we use InstructBLIP (Dai et al., 2023), a state-of-the-art general-purpose VLM, alongside its fine-tuned counterpart PG-InstructBLIP (Gao et al., 2023), which is optimized for object-centric physical understanding. We fine-tune these VLMs on the VirtualHome dataset. We provide the dataset collection process in the Appendix. In Table 6, the dual-model structure for ψ improves the performance in **Exec** by 9.62% over the singular VLM. This specifies that while VLMs are adept at assessing physical attributes, their capacity to discern the executability of semantic skills in diverse physical environments can be more effectively augmented by the inclusion of LMs.

Cross Domain	ψ_{VLM}-IB	ψ_{VLM}-PGIB	$\psi_{\mathrm{LM}} \circ \psi_{\mathrm{VLM}}$
OL	79.67%	77.36%	89.06%
PA	87.32%	91.53%	98.46%
RS	70.59%	71.15%	81.39%

Table 6: The dual-model structure for skill critic

LMs for task planner and skill critic. To implement the task planner $\phi = (\phi_G, \phi_R)$ and the LM-part ψ_{LM} of the skill critic, we test a range of LMs from open-source LLaMA-2-70B to proprietary capable LMs such as PaLM, GPT-3.5, and GPT-4. In Table 7, while LLaMa-2-70B, with significantly fewer parameters, does not achieve comparable performance, SemGro demonstrates robust performance consistently with capable LM backbones.

Metric	LLaMa-2-70B	PaLM	GPT-3.5	GPT-4
SR	25.93%	46.15%	46.43%	48.15%
CGC	43.20%	64.10%	62.25%	60.49%
Plan	50.43%	67.73%	66.09%	67.90%
Exec	71.21%	91.81%	89.76%	85.63%

Table 7: LMs for task planner and skill critic

5 Related Works

Language models for embodied control. The use of pretrained LMs for embodied control tasks has been increasingly prevalent. A significant focus of this research is on bridging the gap between LMs' extensive knowledge and their lack of inherent understanding of embodied experiences. Several works tackle the challenge by directly incorporating environmental contexts to LMs through textual prompts (Xiang et al., 2023; Lin et al., 2023b,a; Shinn et al., 2023; Carta et al., 2023), based on the premise that observations can be described in text form. For example, (Lin et al., 2023b) utilizes a text-based table to display objects' location and their relation. Expanding beyond the mere textual environment, other works have pivoted towards visual tasks and EIF (Wang et al., 2023; Song et al., 2023; Driess et al., 2023; Wu et al., 2023; Singh et al., 2023). Particularly, (Brohan et al., 2023c) introduces a method to learn skill affordance from visual observations using an offline RL dataset collected from the specific environment. This approach integrates skill logits from an LM with identified skill affordance to select the most valuable skill for execution. Meanwhile, (Wu et al., 2023; Song et al., 2023) use detected object classes in the observation to refine the selection of actionable skills.

Our SemGro distinguishes itself in EIF by being the first to harness the capabilities of LMs for grounding pretrained skills in cross-domains.

Language conditioned policy. A number of works have concentrated on the generalization of language-conditioned policies or skills for unseen language instructions (Guhur et al., 2022; Garg et al., 2022), visual attributes (Brohan et al., 2023b,a; Stone et al., 2023; Shridhar et al., 2022, 2021), or environment dynamics (Shin et al., 2023), with some extending to multi-modal instructions (Li et al., 2023b; Shah et al., 2023; Shin et al., 2024; Jiang et al., 2022; Jang et al., 2022). For instance, (Brohan et al., 2023a) adapts a pretrained foundation model to robot trajectory datasets in an end-to-end manner, by representing robot actions as text tokens. (Li et al., 2023b; Jiang et al., 2022) offer a unified framework capable of achieving interleaved vision-language instructions, employing a set of pretrained models to encode them.

While prior works emphasize the robust policy structure, SemGro leverages an LM-based iterative reasoning mechanism, aiming for balanced skill grounding across diverse domain shifts. Moreover, SemGro addresses realistic EIF tasks with the VirtualHome benchmark, unlike the prior works that often focus on straightforward manipulation tasks.

6 Conclusion

In this paper, we presented a semantic skill grounding framework SemGro, which supports the adaptation of a broad range of skills in cross-domain environments. SemGro harnesses the reasoning capabilities of LMs not only to compose semantic skills for establishing a hierarchical skill database but also to decompose semantic skills into executable low-level skills. This novel approach uniquely explores the trade-off between semantic richness and domain-agnostic executability of semantic skills. We hope our work provides valuable insights into the development and application of real-world EIF agents.

7 Limitations

Despite the strong performance of SemGro, we identify several limitations for improvement. **(i) Single-step visual observation for skill feasibility:** Currently, the skill feasibility is assessed based on a single observation. This approach can be improved by integrating a VLM into our framework that is capable of generating imaginary observations, at the expense of increased resource usage by the embodied agent. By employing Monte Carlo Tree Search, the feasibility of skills could then be computed across multiple planning paths. Addressing this enhancement remains for our future work.

(ii) Unidirectional decomposition strategy: The strategy for semantic skill decomposition is unidirectional, proceeding only in a downward direction. This may often restrict the framework as it cannot revert to initiating planning from a high-level skill at an appropriate juncture. Introducing bidirectionality in task planning could improve the versatility of SemGro. In scenarios with substantial domain shifts, such bidirectionality would allow for completely novel planning that has not been trained to achieve complex, unseen tasks. Enhancing our framework to incorporate this bidirectional planning presents a challenge but crucial direction for future development.

8 Ethics Statement

This work introduces a technology that can impact the application of robotics in household tasks. This technology signals a major shift in the human experience, as household tasks increasingly become automated by robotic agents. The deployment of robotic agents in household tasks presents both significant benefits and challenges. On the positive side, this technology can save time and effort for users, leading to increased convenience and efficiency in managing household chores. However, it also raises important ethical considerations related to labor displacement. As robotic agents take over tasks traditionally performed by humans, there is a potential impact on the livelihoods of current service providers who may need to seek alternative sources of income. This transition necessitates careful consideration and planning to mitigate adverse effects on workers and to ensure that the benefits of technological advancement are equitably distributed. By acknowledging and addressing these societal impacts, our research aims to contribute positively to the field of robotics while fostering an inclusive approach that considers the broader implications for society.

9 Acknowledgements

We would like to thank anonymous reviewers for their valuable comments and suggestions. This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2022-0-01045, 2022-0-00043, 2019-0-00421, 2020-0-01821), and by the National Research Foundation of Korea (NRF) grant funded by MSIT (RS-2023-00213118).

References

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil J. Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael S. Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong T. Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. 2023a. RT-2: vision-language-action models transfer web knowledge to robotic control. *CoRR*, abs/2307.15818.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J. Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael S. Ryoo, Grecia Salazar, Pannag R. Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong T. Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. 2023b. RT-1: robotics transformer for real-world control at scale. In *Robotics: Science and Systems XIX, Daegu, Republic of Korea, July 10-14, 2023*.

Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, et al. 2023c. Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on Robot Learning*, pages 287–318. PMLR.

Thomas Carta, Clément Romac, Thomas Wolf, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. 2023. Grounding large language models in interactive environments with online reinforcement learning. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 3676–3713. PMLR.

Siwei Chen, Anxing Xiao, and David Hsu. 2023. Llm-state: Expandable state representation for long-horizon task planning in the open world. *CoRR*, abs/2311.17406.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven C. H. Hoi. 2023. Instructblip: Towards general-purpose vision-language models with instruction tuning. *CoRR*, abs/2305.06500.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. 2023. Palm-e: An embodied multimodal language model. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 8469–8488. PMLR.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. 2023. Specializing smaller language models towards multi-step reasoning. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 10421–10430. PMLR.

Jensen Gao, Bidipta Sarkar, Fei Xia, Ted Xiao, Jiajun Wu, Brian Ichter, Anirudha Majumdar, and Dorsa Sadigh. 2023. Physically grounded vision-language models for robotic manipulation. *CoRR*, abs/2309.02561.

Divyansh Garg, Skanda Vaidyanath, Kuno Kim, Jiaming Song, and Stefano Ermon. 2022. LISA: learning interpretable skill abstractions from language. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Pierre-Louis Guhur, Shizhe Chen, Ricardo Garcia Pinel, Makarand Tapaswi, Ivan Laptev, and Cordelia Schmid. 2022. Instruction-driven history-aware policies for robotic manipulations. In *Conference on Robot Learning, CoRL 2022, 14-18 December 2022, Auckland, New Zealand*, volume 205 of *Proceedings of Machine Learning Research*, pages 175–187. PMLR.

Rishi Hazra, Pedro Zuidberg Dos Martires, and Luc De Raedt. 2023. Saycanpay: Heuristic planning with large language models using learnable domain knowledge. *CoRR*, abs/2308.12682.

Shima Imani, Liang Du, and Harsh Shrivastava. 2023. Mathprompter: Mathematical reasoning using large language models. In *Proceedings of the The 61st Annual Meeting of the Association for Computational Linguistics: Industry Track, ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 37–42. Association for Computational Linguistics.

Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. 2022. BC-Z: zero-shot task generalization with robotic imitation learning. *CoRR*, abs/2202.02005.

Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. 2022. VIMA: general robot manipulation with multimodal prompts. *CoRR*, abs/2210.03094.

Dongxu Li, Junnan Li, Hung Le, Guangsen Wang, Silvio Savarese, and Steven C.H. Hoi. 2023a. LAVIS: A one-stop library for language-vision intelligence. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 31–41, Toronto, Canada. Association for Computational Linguistics.

Jiachen Li, Qiaozi Gao, Michael Johnston, Xiaofeng Gao, Xuehai He, Suhaila Shakiah, Hangjie Shi, Reza Ghanadan, and William Yang Wang. 2023b. Mastering robot manipulation with multimodal prompts through pretraining and multi-task finetuning. *CoRR*, abs/2310.09676.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *CoRR*, abs/2305.20050.

Bill Yuchen Lin, Yicheng Fu, Karina Yang, Prithviraj Ammanabrolu, Faeze Brahman, Shiyu Huang, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2023a. Swiftsage: A generative agent with fast and slow thinking for complex interactive tasks. *CoRR*, abs/2305.17390.

Bill Yuchen Lin, Chengsong Huang, Qian Liu, Wenda Gu, Sam Sommerer, and Xiang Ren. 2023b. On grounded planning for embodied tasks with language models. In *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 13192–13200. AAAI Press.

Lajanugen Logeswaran, Yao Fu, Moontae Lee, and Honglak Lee. 2022. Few-shot subgoal planning with language models. *arXiv preprint arXiv:2205.14288*.

Aishwarya Padmakumar, Mert Inan, Spandana Gella, Patrick L Lange, and Dilek Hakkani-Tur. 2023. Multimodal embodied plan prediction augmented with synthetic embodied dialogue. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6114–6131.

Georgios Pantazopoulos, Malvina Nikandrou, Amit Parekh, Bhathiya Hemanthage, Arash Eshghi, Ioannis Konstas, Verena Rieser, Oliver Lemon, and Alessandro Suglia. 2023. Multitask multimodal prompted training for interactive embodied task completion. *arXiv preprint arXiv:2311.04067*.

Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. 2018. Virtualhome: Simulating household activities via programs. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 8494–8502. Computer Vision Foundation / IEEE Computer Society.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics.

Rutav Shah, Roberto Martín-Martín, and Yuke Zhu. 2023. MUTEX: learning unified policies from multimodal task specifications. *CoRR*, abs/2309.14320.

Sangwoo Shin, Daehee Lee, Minjong Yoo, Woo Kyung Kim, and Honguk Woo. 2023. One-shot imitation in a non-stationary environment via multi-modal skill. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 31562–31578. PMLR.

Sangwoo Shin, Minjong Yoo, Jeongwoo Lee, and Honguk Woo. 2024. Semtra: A semantic skill translator for cross-domain zero-shot policy adaptation.

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

Mohit Shridhar, Lucas Manuelli, and Dieter Fox. 2021. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning, 8-11 November 2021, London, UK*, volume 164 of *Proceedings of Machine Learning Research*, pages 894–906. PMLR.

Mohit Shridhar, Lucas Manuelli, and Dieter Fox. 2022. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning, CoRL 2022, 14-18 December 2022, Auckland, New Zealand*, volume 205 of *Proceedings of Machine Learning Research*, pages 785–799. PMLR.

Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. 2023. Progprompt: Generating situated robot task plans using large language models. In *IEEE International Conference on Robotics and Automation, ICRA 2023, London, UK, May 29 - June 2, 2023*, pages 11523–11530. IEEE.

Chan Hee Song, Brian M. Sadler, Jiaman Wu, Wei-Lun Chao, Clayton Washington, and Yu Su. 2023. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *IEEE/CVF*

International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023, pages 2986–2997. IEEE.

Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Brianna Zitkovich, Fei Xia, Chelsea Finn, and Karol Hausman. 2023. Open-world object manipulation using pre-trained vision-language models. *CoRR*, abs/2303.00905.

Zihao Wang, Shaofei Cai, Anji Liu, Xiaojian Ma, and Yitao Liang. 2023. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. *CoRR*, abs/2302.01560.

Zhenyu Wu, Ziwei Wang, Xiuwei Xu, Jiwen Lu, and Haibin Yan. 2023. Embodied task planning with large language models. *CoRR*, abs/2307.01848.

Jiannan Xiang, Tianhua Tao, Yi Gu, Tianmin Shu, Zirui Wang, Zichao Yang, and Zhiting Hu. 2023. Language models meet world models: Embodied experiences enhance language models. *CoRR*, abs/2305.10626.

Cheng-Fu Yang, Yen-Chun Chen, Jianwei Yang, Xiyang Dai, Lu Yuan, Yu-Chiang Frank Wang, and Kai-Wei Chang. 2023. Lacma: Language-aligning contrastive learning with meta-actions for embodied instruction following. *arXiv preprint arXiv:2310.12344*.

Tian Yun, Zilai Zeng, Kunal Handa, Ashish V Thapliyal, Bo Pang, Ellie Pavlick, and Chen Sun. 2023. Emergence of abstract state representations in embodied sequence modeling. *arXiv preprint arXiv:2311.02171*.

Jesse Zhang, Jiahui Zhang, Karl Pertsch, Ziyi Liu, Xiang Ren, Minsuk Chang, Shao-Hua Sun, and Joseph J. Lim. 2023. Bootstrap your own skills: Learning to solve new tasks with large language model guidance. *CoRR*, abs/2310.10021.

A Implementation Details

In this section, we present the implementation details of each baseline method and our framework SemGro.

A.1 LLM-Planner

The LLM-Planner (Song et al., 2023) employs templatized semantic skills, k-nearest neighbors (kNN) retrievers, and a Language Model(LM) planner to generate plans for embodied tasks. The semantic skills are defined through templates which then are matched with the objects visible in the environment; this allows for instant skill construction options based on the specific situation observed. The LLM-Planner initially retrieves in-context examples from the training data, utilizing k-nearest neighbors (kNN) retrievers, which are then prompted in the LM planner. Subsequently, the planner merges these skill templates with currently visible objects to determine the skill that is both achievable and capable of completing the task.

We use the open-source code from the llm-planner official github. Due to the lack of existing training datasets in VirtualHome, we exploit data from the primitive level of our skill database (in SemGro) as the training data for the retriever to generate in-context examples. We use GPT-3.5 turbo for the planner and a BERT-base-uncased model from the Huggingface Transformers library for the kNN retriever.

A.2 SayCan

SayCan (Brohan et al., 2023c) employs an LM planner, pre-defined semantic skills, and an affordance value function to generate a feasible skill plan for user instructions. The LM planner is responsible for finding the suitable skill from the skill set to achieve the given task using the LM log probability between the human instruction and the pre-defined set of skills. Then, the affordance score of each skill in the skill set is calculated using the pre-trained affordance value function. Saycan combines two scores and uses them to choose the appropriate feasible skill to achieve the task.

We implement SayCan in a similar way as LLM-Planner. First, in-context examples are generated using kNN retrieval from our framework. The retrieved in-context examples are then prompted to the LM planner. We found that the skill generation is dependent on the in-context prompts. To facilitate the LM Planner in producing skills that are achievable in the current observations, we provide additional task examples that are attainable within the present context. Then, the planner chooses a suitable skill from the skill candidates to accomplish the task. We use GPT-3.5 turbo for the planner.

We use the formulation of skill and value functions similar to LLM-Planner. In SayCan, each skill is formulated as actions that can perform a short-horizon task, such as picking up a particular object. Similarly, we use the atomic actions provided in VirtualHome, such as grab, walk, switchon, and also combine them with individual objects to formulate the set of low-level skills. Subsequently, the value function emerges as another critical element in SayCan, evaluating the feasibility of each skill in light of the present observation. Due to the challenges associated with training low-level policies in VirtualHome, we adopt the approach used by LLM-Planner, providing SayCan with Oracle object data to establish the Oracle value function. This provides SayCan with preemptive access to comprehensive knowledge about all objects and their affordance in the current observation; thus, this shrinks the list of potential skills for the LLM to consider. Consequently, this simplifies the decision-making process for the planner, facilitating the selection of a suitable skill to complete the task effectively.

A.3 ProgPrompt

ProgPrompt (Singh et al., 2023) employs a human-designed programming assertion syntax to check the pre-conditions for executing skills and addresses failures by initiating predefined recovery skills. In detail, ProgPrompt uses a plan structure consisting of comments, actions, and assertions in the style of programming language. Comments are used to cluster high-level skills together and actions are expressed as imported function calls. Assertions are used to check conditions right before actions start and execute the recovery skills if needed.

We use the code from the ProgPrompt official github. We use the same plan templates as employed by ProgPrompt, featuring a Pythonic style where the task name is designated as a function, available actions are included through headers, accessible objects are specified in variables, and each action is delineated as a line of code within the function. Similar to SayCan implementation, we also use knn retrieval to generate in-context examples for the LM Planner. We use GPT-3.5 turbo for the planner.

Utilizing assertions to verify the pre-conditions of each action is a pivotal concept in ProgPrompt, enabling the correction of incorrect behaviors to accomplish tasks. However, generating pre-conditions for every action involves substantial effort and resources; furthermore, given the assumption that an innumerable set of skills could be integrated, manually annotating pre-conditions for each skill becomes practically infeasible. Similar to SayCan implementation, we endow ProgPrompt with Oracle pre-conditions for each action. By implementing Oracle assertions before each skill, the system automatically corrects behaviors, thus allowing the agent to perform only feasible skills to complete the task efficiently.

A.4 SemGro

Our SemGro framework comprises two primary components: the task planner ϕ, which is composed of the skill generator ϕ_G and task retriever ϕ_R, and the multi-modal skill critic ψ. We employ the kNN retriever to extract k in-context examples from the hierarchical skill database using the instruction i and current visual observations o_t. These k in-context examples' low-level skill plans serve as candidates for the skill generator ϕ_G.

The execution feasibility of the planned skill is then assessed by the multi-modal skill critic ψ based on the current observation. Within the critic, the Vision-Language Model (VLM)-part utilizes visual observations from the environment to infer detected objects and their states. Leveraging this inferred information, the LM-part of the critic assesses the execution feasibility of the planned skill. If the skill is assessed as executable, the skill is performed as it is; otherwise, we prompt the feedback from the critic along with the one-step further lower-level skill candidates to generate alternative, more fine-grained instructions. Utilizing the revised instructions, the database is queried for skills at a lower abstraction level to reinitiate the skill grounding process. An example of detailed process of our framework is in Table 10.

This iterative feedback mechanism ensures that the agent can adaptively refine its actions to better align with the dynamic conditions of its target physical environment, thereby enhancing the overall effectiveness and precision of skill execution. In the development of our skill generator, we utilize GPT-3.5 Turbo. For the multi-modal critic, we employ InstructBLIP as the VLM-part, and GPT-3.5 as the LM-part. For the k-Nearest Neighbors (kNN) retriever, we integrate the BERT-base-uncased model from the Huggingface transformers library.

B Experiment Details

In this section, we describe the experiment settings in detail. We also specify the cross-domain scenarios and environment details used in the experiments, as well as the data collection process for our hierarchical skill database and the training procedure of our multi-modal critic.

Cross-domain scenarios. We used 5 EIF tasks, 4 types of language instructions, and 15 environmental scenes to test across 300 cross-domain scenarios. The variation in instructions, akin to the approach used in SayCan, is crafted to assess the extent to which our framework can deconstruct abstract instructions and ground skills in the target domain. We utilize 4 instruction types such as Abstract Noun, Abstract Verb, Structured, and Long Horizon, comprising a total of 137 instructions.

The **Abstract Noun** and **Abstract Verb** categories are created to examine whether SemGro can execute the appropriate sequences when given abstractions are in nouns and verbs. **Structured Language**, paralleling Abstract Verbs, aims to evaluate SemGro's performance with well-structured languages. **Long Horizon**, necessitating multi-step reasoning, is designed to evaluate SemGro's capability to process and reason through temporally extended instructions.

VirtualHome simulation. We use VirtualHome (Puig et al., 2018), an interactive simulation platform for complex household tasks. The skills provided in the environment include manipulation actions such as picking up objects and toggling appliances on or off. We impose restrictions on these actions within the environment to align with our settings; the skills performed are implemented to be feasible only with the objects currently visible. For example, "grab apple" is executable when the apple is closed enough and is visible to the agent. The action space of the VirtualHome experiments is provided in Table 8.

Table 8: VirtualHome Actions

Actions
find <obj>, grab <obj>, walk <obj>, sit <obj>
put <obj> <obj>, open <obj>, close <obj>
switchon <obj>, switchoff <obj>

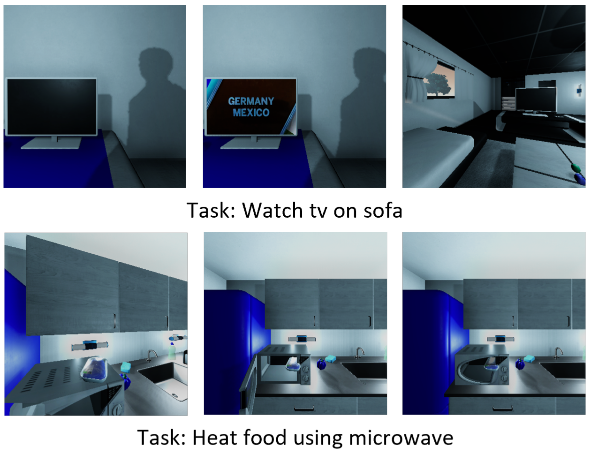


Figure 3: Example tasks in VirtualHome

We use a single training environment and 15 distinct environment settings for cross-domain evaluation, with the training environment being environment ID 0 in VirtualHome. We use three different domain-shift types, OL, PA, and RS as explained in the main manuscript. For each type, we use 5 distinct environment settings. The OL scenario involves alterations in the locations of objects and their relationships compared to the training environment. The PA scenario reflects changes in the physical attributes of objects from the training environment. RS scenario encompasses shifts in the visual context, object locations, and their physical attributes. Figure 3 presents visual examples of tasks, and Table 9 details the cross-domain scenario examples employed in our experiments.

Dataset collection process. The skill database $\mathcal{D}$ comprises 4 distinct abstraction levels, each containing 3949, 183, 142, and 90 skills, from the first to the fourth level, respectively. We adopt a bottom-up approach to construct the skill database by progressively chaining low-level skills to form complex high-level skills. Specifically, we employ the skill bootstrapping method in (Zhang et al., 2023), which leverages an LM to guide the development of high-level skills from pretrained low-level skills.

Upon initiation in the environment, the agent selects a skill from the available low-level skills. The LM is then queried to identify the most suitable next skill to execute, based on the provided prompt. Following this, the agent performs the selected skill. This cycle repeats until reaching a predetermined maximum number of steps or encountering a skill execution failure. Successful skill executions are summarized by the LM to derive a high-level abstract skill name, enabling the accumulation of a comprehensive skill set across different abstraction levels. For both skill selection and summarization, the text-bison-001 model from PaLM (Chowdhery et al., 2023) is utilized.

Furthermore, we use an open-vocabulary detection model owl-ViT (Stone et al., 2023) to obtain the object names $dn(o^{(TR)})$ (e.g., “microwave”) and InstructBLIP (Dai et al., 2023) to obtain the physical state of each object $ds(o^{(TR)})$ (e.g., “microwave: closed”).

VLM fine-tuning. We fine-tune the PG-InstructBLIP to implement the VLM-part ψ_{VLM} of the skill critic, using the dataset D_O in Eq (2). This process enhances the VLM’s understanding of physical objects within the VirtualHome environment by adopting a Visual Question Answering (VQA) training approach. An object detector is employed to isolate the portions of images that feature objects, and these cropped images are then used to train the VLM in recognizing the objects’ states. Specifically, we formulate questions about the objects’ conditions (e.g., “Is this object open or closed? Answer ’unknown’ if uncertain. Short answer:”) and train the VLM to accurately answer these queries.

We use the FlanT5-XL (Chung et al., 2022) variant of InstructBLIP (Dai et al., 2023), implemented in vision-language multi-modal open-source project LAVIS (Li et al., 2023a). The model is trained for a total of 12 GPU hours on a system equipped with an Intel(R) Core(TM) i9-10980XE processor and an NVIDIA RTX A6000 GPU.

Quantification of domain disparities in Table 3. To quantify domain disparities, we calculate the number of high-level skills that must be decomposed into different series of low-level skills for execution in the target domain, compared to the training environment.

For instance, the high-level skill "cook salmon", initially learned in the training environment through a skill-chaining process involving low-level skills such as "grasp salmon" and "place salmon in the microwave", may require adaptation in the target domain. This adaptation might involve a different series of low-level skills "go to refrigerator", and "grasp refrigerator handle", among others, for successful execution. Each instance where a high-level skill’s execution strategy deviates from its training environment counterpart contributes to the domain shift quantification. The term ‘None’ refers to scenarios without domain shift, meaning the model is evaluated within the training environment itself. The categories of "Small," "Medium," and "Large" domain shifts correspond to the increasing percentages (20%, 30%, and 50%, respectively) of high-level skills that require adaptation through decomposition into low-level skills in the target domain.

C Trajectory Examples and Prompts

From Table 10 to 12, we describe the trajectory collected by the VirtualHome agent. Specifically, across each skill step, the inputs and outputs of the task planner and skill critic (e.g., detected object names and their states, retrieved skill semantics and skill candidates, generated skills, inputs and outputs of the critic, and fine-grained instructions) are shown in the Table. Following these tables, we also document the prompts used for task planning and skill chaining.

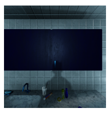
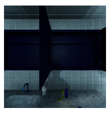
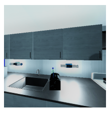
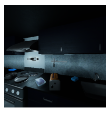
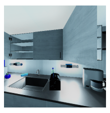
ID	Visual Observation	Cross-Domain (Domain-Shift Type)	Task	Instruction Information	
				Instruction Type	Instruction Example
1		Object Location (OL)	Put 3 fruits on the kitchentable	Abstract Noun	Grab and put various fruits on the kitchentable.
2		Room Structure (RS)	Watch tv while sitting on sofa	Abstract Verb	Chill on sofa with tv.
3		Physical Attribute (PA)	Put 3 fruits in the kitchencabinet	Structured	Grab fruits and put them in the kitchencabinet.
4		Object Location (OL)	Put 2 fruits in the fridge	Long Horizon	The kids messed up the kitchen. Put food in the fridge.
5		Room Structure (RS)	Grab book and sit on sofa	Abstract Noun	Relax on bunk with book.
6		Physcial Attribute (PA)	Find the food and heat it using microwave	Abstract Verb	Heat food.
7		Object Location (OL)	Put 3 bathroom objects in the bathroom cabinet	Structured	Grab bathroom objects and put them in cabinet.
8		Room Structure (RS)	Watch tv while sitting on sofa	Long Horizon	I finished household chores. Relex on sofa and watch tv.
9		Physical Attribute (PA)	Put 2 fruits in the fridge	Abstract Noun	Put food in refrigerator.

Table 9: VirtualHome cross-domain scenario examples

Skill Step 0	
Instruction:	I need to clean up the kitchen. store fruits in the kitchencabinet
Visible Object Names:	bananas, kitchencabinet, fridge, faucet, salmon, peach, kitchencounterdrawer, bellpepper, washingsponge, microwave
Visible Objects States:	garbagecan is CLOSED, fridge is OPEN, faucet is CLOSED, kitchencabinet is CLOSED, microwave is CLOSED
Retrieved Skill Semantics:	store fruit in kitchen, put fruit and food in fridge and kitchencabinet, put away fruit in kitchencabinet, put away fruit in kitchencabinet and fridge, put fruit away, fridge food and fruit
Retrieved Skill Candidates:	put peach and plum in fridge, put plum and peach in kitchencabinet, put fruit in kitchencabinet, put plum and cupcake in fridge, put apple in kitchencabinet and peach in fridge
Generated Sematnic Skill:	put fruit in kitchencabinet
Critic Input:	detected objects: bananas, kitchencabinet, fridge, faucet, salmon, peach, kitchencounterdrawer, bellpepper, washingsponge, microwave detected states: garbage can is CLOSED, fridge is OPEN, faucet is CLOSED, kitchencabinet is CLOSED, microwave is CLOSED low level skills: put peach and plum in fridge, put plum and peach in kitchencabinet, put fruit in kitchencabinet, put plum and cupcake in fridge, put apple in kitchencabinet and peach in fridge cur skill: put fruit in kitchencabinet
Critic Output:	executable: False feedback: kitchencabinet is closed, you need to open the kitchencabinet
Fine-grained Instruction:	Open kitchencabinet and put fruit in it
Skill Step 1	
Instruction:	Open kitchencabinet and put fruit in it
Visible Object Names:	bananas, kitchencabinet, fridge, faucet, salmon, peach, kitchencounterdrawer, bellpepper, washingsponge, microwave
Visible Objects States:	garbagecan is CLOSED, fridge is OPEN, faucet is CLOSED, kitchencabinet is CLOSED, microwave is CLOSED
Retrieved Skill Semantics:	put fruit in kitchencabinet, store bananas and apple in kitchencabinet, place apple and peach in kitchencabinet, put cereal and bananas in kitchencabinet, put bananas in kitchencabinet and apple on coffeetable
Retrieved Skill Candidates:	put peach in kitchencabinet, open kitchencabinet and put bananas in kitchencabinet, open kitchencabinet and put pie in kitchencabinet, open kitchencabinet and put bananas in kitchencabinet
Generated Semantic Skill:	open kitchencabinet and put bananas in kitchencabinet
Critic Input:	detected objects: bananas, kitchencabinet, fridge, faucet, salmon, peach, kitchencounterdrawer, bellpepper, washingsponge, microwave detected states: garbagecan is CLOSED, fridge is OPEN, faucet is CLOSED, kitchencabinet is CLOSED, microwave is CLOSED low level skills: put peach in kitchencabinet, open kitchencabinet and put bananas in kitchencabinet, open kitchencabinet and put bananas in kitchencabinet cur skill: open kitchencabinet and put bananas in kitchencabinet
Critic Output:	executable: True
Executed Skill:	skill: open kitchencabinet and put bananas in kitchencabinet, skill step: walk bananas, grab bananas, walk kitchencabinet, find kitchencabinet, open kitchencabinet, put bananas kitchencabinet

Table 10: Sampled trajectory within the VirtualHome environment. (Continued on next page)

Skill Step 2	
Instruction:	I need to clean up the kitchen. store fruits in the kitchencabinet
Visible Object Names:	washingsponge, coffeemaker, sink, kitchencabinet, kitchencounter, kitchencounterdrawer, toaster, stove, fryingpan, coffeepot
Visible Objects States:	coffeemaker is CLOSED, kitchencabinet is CLOSED, kitchencounter is CLOSED, toaster is CLOSED, stove is CLOSED, coffeepot is CLOSED
Retrieved Skill Semantic:	put fruit in kitchencabinet, put away fruit in kitchencabinet, store fruit in kitchen, put away fruit in kitchencabinet and fridge, put fruits in fridge, fridge, put fruit away, put away fruit, store fruit in kitchen cabinet and fridge
Retrieved Skill Candidates:	put apple and peach in kitchencabinet, place apple in kitchencabinet, put fruit in kitchencabinet, put peach and plum in fridge, put plum and peach in kitchencabinet, put plum and cupcake in fridge
Generated Semantic Skill:	place apple in kitchencabinet
Critic Input:	detected objects: washingsponge, coffeemaker, sink, kitchencabinet, apple, kitchencounter, kitchencounterdrawer, toaster, stove, fryingpan, coffeepot detected states: coffeemaker is CLOSED, kitchencabinet is OPEN, kitchencounter is CLOSED, toaster is CLOSED, stove is CLOSED, coffeepot is CLOSED, lower level skills: put apple and peach in kitchencabinet, place apple in kitchencabinet, put fruit in kitchencabinet, put peach and plum in fridge, put plum and peach in kitchencabinet, put plum and cupcake in fridge cur skill: place apple in kitchencabinet
Critic Output:	executable: True
Executed Skill:	skill: place apple in kitchencabinet skill steps: walk apple, grab apple, walk kitchencabinet, put apple kitchencabinet, close kitchencabinet
Skill Step 3	
Instruction:	I need to clean up the kitchen. store fruits in the kitchencabinet
Visible Objects:	washingsponge, coffeemaker, sink, kitchencabinet, kitchencounter, kitchencounterdrawer, toaster, stove, fryingpan, coffeepot, peach
Visible Objects States:	coffeemaker is CLOSED, kitchencabinet is CLOSED, kitchencounter is CLOSED, toaster is CLOSED, stove is CLOSED, coffeepot is CLOSED
Retrieved Skill Semantic:	put fruit in kitchencabinet, put fruit in fridge and kitchencabinet, put fruit and food in fridge and kitchencabinet, put away fruit in kitchencabinet
Retrieved Skill Candidates:	put apple and peach in kitchencabinet, put peach and plum in fridge, put plum and peach in kitchencabinet, store apple in kitchen cabinet and fridge, put apple in kitchencabinet and peach in fridge
Generated Semantic Skill:	put apple and peach in kitchencabinet

Table 11: Sampled trajectory within the VirtualHome environment. (Continued on next page)

Critic Input:	detected objects: washingsponge, coffeemaker, sink, kitchencabinet, kitchencounter, kitchencounterdrawer, toaster, stove, fryingpan, coffeepot, peach detected states: coffeemaker is CLOSED, kitchencabinet is CLOSED, kitchencounter is CLOSED, toaster is CLOSED, stove is CLOSED, coffeepot is CLOSED, low level skills: put apple and peach in kitchencabinet, put peach and plum in fridge, put plum and peach in kitchencabinet, store apple in kitchen cabinet and fridge, put apple in kitchencabinet and peach in fridge, cur skill: put apple and peach in kitchencabinet
Critic Output:	executable: False feedback: kitchencabinet is closed, you need to open the kitchencabinet
Fine-grained Instruction:	Open the kitchencabinet and put apple and peach in kitchencabinet
Skill Step 4	
Instruction:	I need to clean up the kitchen. store fruits in the kitchencabinet
Visible Objects:	washingsponge, coffeemaker, sink, kitchencabinet, kitchencounter, kitchencounterdrawer, toaster, stove, fryingpan, coffeepot, peach
Visible Objects States:	coffeemaker is CLOSED, kitchencabinet is CLOSED, kitchencounter is CLOSED, toaster is CLOSED, stove is CLOSED, coffeepot is CLOSED
Retrieved Skill Semantic:	put apple and peach in kitchencabinet, put fruit in kitchencabinet, put plum and peach in kitchencabinet, put peach and plum in kitchencabinet,put cupcake and peach in kitchencabinet
Retrieved Skill Candidates:	put apple in kitchencabinet, open kitchencabinet and put peach in kitchencabinet, put peach in kitchencabinet, put peach in kitchencabinet, open kitchencabinet and put pie in kitchencabinet
Generated Semantic Skill:	open kitchencabinet and put peach in kitchencabinet
Critic Input:	detected objects: washingsponge, coffeemaker, sink, kitchencabinet, kitchencounter, kitchencounterdrawer, toaster, stove, fryingpan, coffeepot, peach detected states: coffeemaker is CLOSED, kitchencabinet is CLOSED, kitchencounter is CLOSED, toaster is CLOSED, stove is CLOSED, coffeepot is CLOSED cur skill: open kitchencabinet and put peach in kitchencabinet
Critic Output:	executable: True
Executed Skill:	skill: open kitchencabinet and put peach in kitchencabinet skill steps: walk peach, grab peach, walk kitchencabinet, find kitchencabinet, open kitchencabinet, put peach kitchencabinet, close kitchencabinet

Table 12: Sampled trajectory within the VirtualHome environment.

Prompt examples utilized by the skill generator in SemGro.

```
Examples of common household tasks and their skill sequence to achieve the tasks using skills from
    Skill Library:

Skill Library: find kitchencabinet, put bananas kitchencabinet, open kitchencabinet, walk
    kitchencabinet, put bananas kitchentable, find peach, close kitchencabinet, walk bananas, walk
    cereal, put peach kitchentable, put bananas coffeetable, find coffeetable, find kitchentable,
    grab bananas
Task: open kitchencabinet and put bananas in kitchencabinet
skill steps: 1. walk bananas 2. grab bananas 3. walk kitchencabinet 4. find kitchencabinet 5. open
    kitchencabinet 6. put bananas kitchencabinet 7. close kitchencabinet

Skill Library: find bananas, put bananas kitchentable, walk peach, open kitchencabinet, walk
    coffeetable, put apple kitchencabinet, walk kitchencabinet, close kitchencabinet, walk cereal,
    find kitchencabinet, put peach kitchentable, put peach kitchencabinet, grab peach
Task: open kitchencabinet and put peach in kitchencabinet
skill steps: 1. walk peach 2. grab peach 3. walk kitchencabinet 4. find kitchencabinet 5. open
    kitchencabinet 6. put peach kitchencabinet 7. close kitchencabinet

Skill Library: walk peach, walk bananas, walk kitchentable, open kitchencabinet, put apple
    kitchencabinet, find apple, put peach kitchencabinet, grab peach, walk kitchencabinet, walk
    apple, find kitchencabinet, find peach, grab apple, grab cereal, close kitchencabinet
Task: put peach in kitchencabinet
skill steps: 1. walk peach 2. find peach 3. grab peach 4. walk kitchencabinet 5. find kitchencabinet
    6. open kitchencabinet 7. put peach kitchencabinet 8. close kitchencabinet

Skill Library: put cereal kitchencabinet, find bananas, walk bananas, put bananas kitchentable, walk
    cereal, walk coffeetable, put bananas kitchencabinet, walk kitchencabinet, find kitchentable,
    grab peach, find apple, find coffeetable, walk kitchentable, grab bananas, put peach
    kitchencabinet
Task: find bananas and put bananas on kitchentable
skill steps: 1. walk bananas 2. find bananas 3. grab bananas 4. walk kitchentable 5. find
    kitchentable 6. put bananas kitchentable

Skill Library: close kitchencabinet, find apple, grab apple, walk apple, walk kitchentable, open
    kitchencabinet, walk coffeetable, put cereal kitchencabinet, find kitchencabinet, put apple
    kitchencabinet, walk cereal, find coffeetable, walk bananas, walk kitchencabinet
Task: put apple in kitchencabinet
skill steps: 1. walk apple 2. find apple 3. grab apple 4. walk kitchencabinet 5. find kitchencabinet
    6. open kitchencabinet 7. put apple kitchencabinet 8. close kitchencabinet

Skill Library: put apple kitchencabinet, walk cereal, put bananas kitchencabinet, find coffeetable,
    find kitchentable, walk bananas, grab bananas, open kitchencabinet, put peach kitchentable, put
    bananas coffeetable, grab peach, walk coffeetable, find peach, put bananas kitchentable, find
    bananas, walk kitchentable
Task: find bananas and put bananas on coffeetable
skill steps: 1. walk bananas 2. find bananas 3. grab bananas 4. walk coffeetable 5. find coffeetable
    6. put bananas coffeetable

Skill Library: find bananas, walk coffeetable, grab cereal, walk bananas, find coffeetable, open
    kitchencabinet, grab bananas, find kitchencabinet, close kitchencabinet, put bananas coffeetable,
     walk kitchencabinet, put cereal kitchencabinet, put peach kitchentable, walk cereal
Task: open kitchencabinet and put cereal in kitchencabinet
skill steps: 1. walk cereal 2. grab cereal 3. walk kitchencabinet 4. find kitchencabinet 5. open
    kitchencabinet 6. put cereal kitchencabinet 7. close kitchencabinet

Choose the next skill from Skill library to successfully accomplish the task.
Skill library: walk bananas, grab bananas, walk kitchencabinet, find kitchencabinet, open
    kitchencabinet, put bananas kitchencabinet, close kitchencabinet, walk bananas, find bananas,
    grab bananas, walk kitchencabinet, find kitchencabinet, open kitchencabinet, put bananas
    kitchencabinet, close kitchencabinet, walk peach, grab peach, walk kitchencabinet, find
    kitchencabinet, open kitchencabinet, put peach kitchencabinet, close kitchencabinet, walk apple,
     grab apple, walk kitchencabinet, find kitchencabinet, open kitchencabinet, put apple
    kitchencabinet,
skill steps: 1.
```

Prompt examples utilized for the skill chaining.

```
Examples of common household tasks and their descriptions:
The Task are composed of skill steps chosen from Skill Library,
Skill Library: wash the frying pan, put the candle on the living room shelf,throw away the lime, open
     window, listen to radio, wash clothes, wash the plate, get my pants and shirt and place on the
    bed, call a friend, wash mug, wash the rug in washing machine,
cut apple, make toast, read book under table lamp, brush teeth, open the curtains, put clothes in
    closet, throw away apple, put all the cutlery in the sink, put papers in the folder, put my
    slippers near the bed, take out the trash, microwave chicken, heat cupcakte

Skill Steps: 1. wash the plate 2. wash mug
Task: wash the plate and mug

Skill Steps: 1. brush teeth and wash clothes
Task: brush teeth and wash clothes

Skill Steps: 1. brush teeth 2. read book under table lamp
Task: read book under table lamp after brushing teeth

Skill Steps: 1. wash the frying pan 2. wash the plate 3. plate mug
Task: wash the kitchenware

Skill Steps: 1. wash clothes 2. wash the rug in washing machine
Task: wash clothes and rug using washing machine

Skill Steps: 1. throw away apple 2. throw away the lime
Task: throw away apple and lime

Skill Steps: 1. cut apple 2. make toast
Task: cut apple and make toast

Skill Steps: 1. make tea and bring it to the coffee table 2. set up coffee table for fruits
Task: set up tea and fruits on coffeetable

Skill Steps 1. put apple in fridge and bring cellphone to sofa 2. put hairproduct in bathroom cabinet
     and bring paper to sofa
Task: clean up and bring cellphone and paper to sofa

Skill Steps: 1. reheat food 2. prepare chicken for dinner
Task: cook chichen for dinner

Skill Steps: 1. put book back and sit on the sofa 2. watch tv and turn off microwave
Task: organize books and watch tv

Skill Steps: 1. put things back in their place 2. put away mug and lime
Task: clean bedroom

Skill Steps: 1. reheat cupcake and put it on coffeetable 2. reheat chicken and put it on kitchentable
Task: heat cupcake and chicken and put them on coffeetable

Skill Steps: 1. put away mug and lime 2. eat bread on the sofa and put book back in the bookshelf
Task: clean livingroom

Skill Steps: 1. open the curtains 2. put my slippers near the bed
Task: open the curtains and put my slippers near the bed

Skill Steps: 1. open window 2. call a friend
Task: open window and call a friend

Skill Steps: 1. microwave chicken 2. heat some cupcake
Task: microwave the chicken and heat some cupcake

Skill Steps: 1. microwave chicken 2. heat some cupcake
Task: microwave the chicken and heat some cupcake

Predict the next skill for creating plausible task by correctly choosing from Skill Library:
```

Prompt examples utilized for the skill chaining.

```
Instructions: give a high-level description for the following steps describing common household tasks
     .

Skill Steps: 1. wash the plate 2. wash mug
Summary: wash the plate and mug

Skill Steps: 1. brush teeth and wash clothes
Summary: brush teeth and wash clothes

Skill Steps: 1. brush teeth 2. read book under table lamp
Summary: read book under table lamp after brushing teeth

Skill Steps: 1. wash the frying pan 2. wash the plate 3. plate mug
Summary: wash the kitchenware

Skill Steps: 1. wash clothes 2. wash the rug in washing machine
Summary: wash clothes and rug using washing machine

Skill Steps: 1. make tea and bring it to the coffee table 2. set up coffee table for fruits
Summary: set up tea and fruits on coffeetable

Skill Steps 1. put apple in fridge and bring cellphone to sofa 2. put hairproduct in bathroom cabinet
     and bring paper to sofa
Summary: clean up and bring cellphone and paper to sofa

Skill Steps: 1. reheat food 2. prepare chicken for dinner
Summary: cook chichen for dinner

Skill Steps: 1. put book back and sit on the sofa 2. watch tv and turn off microwave
Summary: organize books and watch tv

Skill Steps: 1. put things back in their place 2. put away mug and lime
Summary: clean bedroom

Skill Steps: 1. reheat cupcake and put it on coffeetable 2. reheat chicken and put it on kitchentable
Summary: heat cupcake and chicken and put them on coffeetable

Skill Steps: 1. put away mug and lime 2. eat bread on the sofa and put book back in the bookshelf
Summary: clean livingroom

Skill Steps: 1. throw away apple 2. throw away the lime
Summary: throw away apple and lime

Skill Steps: 1. cut apple 2. make toast
Summary: cut apple and make toast

Skill Steps: 1. open the curtains 2. put my slippers near the bed
Summary: open the curtains and put my slippers near the bed

Skill Steps: 1. open window 2. call a friend
Summary: open window and call a friend

Skill Steps: 1. microwave chicken 2. put chicken on coffeetable
Summary: heat chicken and put it on coffeetable


Summarize the skill steps and give a plausible name for the household task executed.
```