Multi-level Compositional Reasoning for Interactive Instruction Following

Suvaansh Bhambri*, **Byeonghwi Kim***, **Jonghyun Choi**

Yonsei University
suvaanshbhambri@gmail.com   {byeonghwikim,jc}@yonsei.ac.kr

Abstract

Robotic agents performing domestic chores by natural language directives are required to master the complex job of navigating environment and interacting with objects in the environments. The tasks given to the agents are often composite thus are challenging as completing them require to reason about multiple subtasks, *e.g.*, bring a cup of coffee. To address the challenge, we propose to divide and conquer it by breaking the task into multiple subgoals and attend to them individually for better navigation and interaction. We call it *Multilevel Compositional Reasoning Agent (MCR-Agent)*. Specifically, we learn a three-level action policy. At the highest level, we infer a sequence of human-interpretable subgoals to be executed based on language instructions by a high-level *policy composition controller*. At the middle level, we discriminatively control the agent's navigation by a *master policy* by alternating between a navigation policy and various independent interaction policies. Finally, at the lowest level, we infer manipulation actions with the corresponding object masks using the appropriate *interaction policy*. Our approach not only generates human interpretable subgoals but also achieves 2.03% absolute gain to comparable state of the arts in the efficiency metric (PLWSR in unseen set) without using rule-based planning or a semantic spatial memory.

1 Introduction

For the long-awaited dream of building a robot to assist humans in daily life, we now witness rapid advances in various embodied AI tasks such as visual navigation (Anderson et al. 2018b; Chen et al. 2019; Krantz et al. 2020), object interaction (Zhu et al. 2017; Misra, Langford, and Artzi 2017), and interactive reasoning (Das et al. 2018a; Gordon et al. 2018). Towards building an ideal robotic assistant, the agent should be capable of all of these tasks to address more complex problems. A typical approach for combining these abilities is to build a unified model (Shridhar et al. 2020; Singh et al. 2021) to jointly perform different sub-tasks. However, the reasoning for navigation can differ significantly from the one for object interaction; the former needs to detect navigable space and explore to reach a target location while the latter requires detecting objects and analysing their distances and states (Singh et al. 2021).

Meanwhile, the human cognition process learns to divide a task into sub-objectives such as navigation or interaction, which enables humans to facilitate complex reasoning in various circumstances (Hayes-Roth and Hayes-Roth 1979). Inspired by this, we propose a multi-level compositional reasoning agent (MCR-Agent) that disentangles the task into high-level subgoals; then learns and infers a low-level policy for each sub-task. Specifically, we propose a multi-level agent comprised of (1) a policy composition controller (PCC) that specifies a sub-policy sequence, (2) a master policy (MP) that specialises in navigation, and (3) a set of interaction policies (IP) that execute interactions. This disentanglement enables easier analysis of subtasks with shorter horizons (See Sec. 4.3 for empirical evidence).

In addition, to interact with multiple objects in a long sequence, the agent should be able to keep track of the current target object at each time instance. Inspired by (Yang et al. 2018; Wortsman et al. 2019), we additionally propose an object encoding module (OEM) that provides target object information which is used as a navigational subgoal monitor, *i.e.*, stopping criterion for the navigation policy.

In our empirical evaluations with a long horizon instruction following task with the condition of not requiring additional depth supervision and perfect egomotion assumption, usually not available for real world deployment, we observe that MCR-Agent outperforms most prior arts in literature by large margins. We summarize our contributions as follows:

- We propose a multi-level hierarchical framework, MCR-Agent, that decomposes a compositional task into semantic subgoals and effectively addresses them with corresponding submodules.

- We propose an object encoding module (OEM) that encodes object information from natural language instructions for effective navigation.

- By extensive quantitative analyses on a challenging interactive instruction following benchmark (Shridhar et al. 2020), we show that MCR-Agent yields competitive performance with higher efficiency than prior arts that do not assume perfect egomotion and extra depth supervision.

2 Related Work

There are numerous task setups and benchmarks proposed for developing an agent to complete complicated tasks given
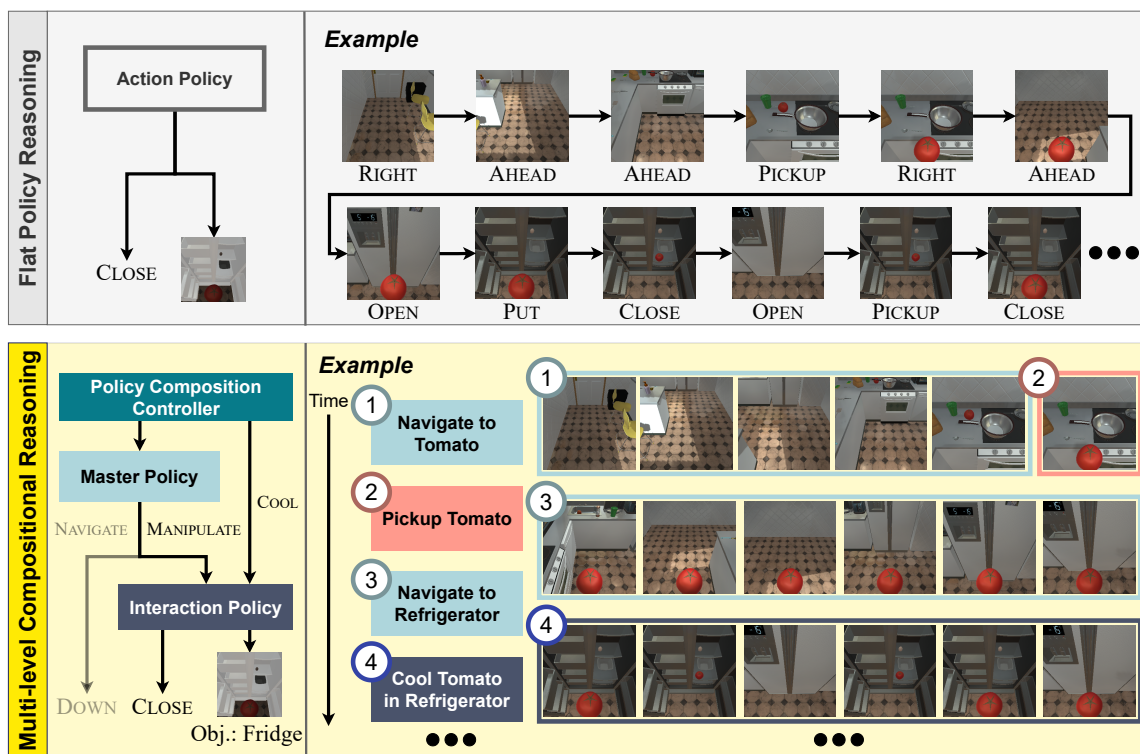

Figure 1: The proposed 'Multi-level compositional reasoning' contrasted to 'Flat policy reasoning'. The flat policy reasoning has been employed in prior arts (Shridhar et al. 2020; Singh et al. 2021; Pashevich et al. 2021; Nguyen et al. 2021), training an agent to directly learn the low-level actions. On the contrary, our multi-level policy decomposes a long-horizon task into multiple subtasks and leverages the high-level abstract planning, which enables an agent to better address long-horizon planning.

natural language directives, such as agents trained to navigate (Li et al. 2020; Xu et al. 2021) or solve household tasks (Shridhar et al. 2020). However, the vast majority of approaches for these tasks employ flat reasoning (Singh et al. 2021; Nguyen et al. 2021), in which the agent decides on the low-level actions accessible while moving through the environment (Gupta et al. 2017; Zhu et al. 2020). When the prior arts seek to define subtasks, some define them with two layers of hierarchy (Zhang and Chai 2021; Corona et al. 2020; Blukis et al. 2021; Andreas, Klein, and Levine 2017; Gordon et al. 2018; Yu et al. 2019; Das et al. 2018b). However, these strategies require a good amount of data due to the semantic gap between abstract natural language instructions and concrete executions (Zhou, Yin, and Neubig 2021). Natural language is subjective and even a seemingly simple command can contain several unstated meanings. Because of this semantic gap, most approaches (Landi et al. 2019; Krantz et al. 2020; Singh et al. 2021; Pashevich et al. 2021) require either a large amount of labeled data or trial-and-error learning to map language to low-level actions. In contrast, we propose to use deeper hierarchical knowledge for better control of embodied agents. Thanks to the modular structure, our agent reasons and accomplishes tasks along longer paths, spanning numerous subgoals.

The described task requires not only navigation but also interaction. (Shridhar et al. 2020) proposes a CNN-LSTM-based baseline agent with progress tracking (Ma et al. 2019). (Singh et al. 2021) offers a modular strategy for factoris-

ing action prediction and mask generation while (Nottingham et al. 2021) offers a system that encodes language and visual state, and performs action prediction using independently trained modules. (Zhang and Chai 2021) propose a transformer-based hierarchical agent whereas (Suglia et al. 2021) presents a transformer-based agent that uses object landmarks for navigation. (Pashevich et al. 2021) also presents a transformer-based agent that uses a multimodal transformer for exploiting the multiple input modalities.

Recent work proposes to construct semantic maps and leverage the relative localization for improved navigation where (Blukis et al. 2021) uses a 3D map to encode spatial semantic representation, (Min et al. 2022) suggests a SLAM-based approach that keeps observed information in a 2D top-down map while (Liu et al. 2022) presents a planning-based approach that keeps a semantic spatial graph to encode visual inputs and the agent's poses.

Finally, a modular policy with two levels of hierarchy has been proposed by (Corona et al. 2020) which does not perform well on a long-horizon task. In contrast, our policy operates at three hierarchical levels, exploiting the fact that navigation and interaction are semantically diverse activities that require independent processing.

3 Model

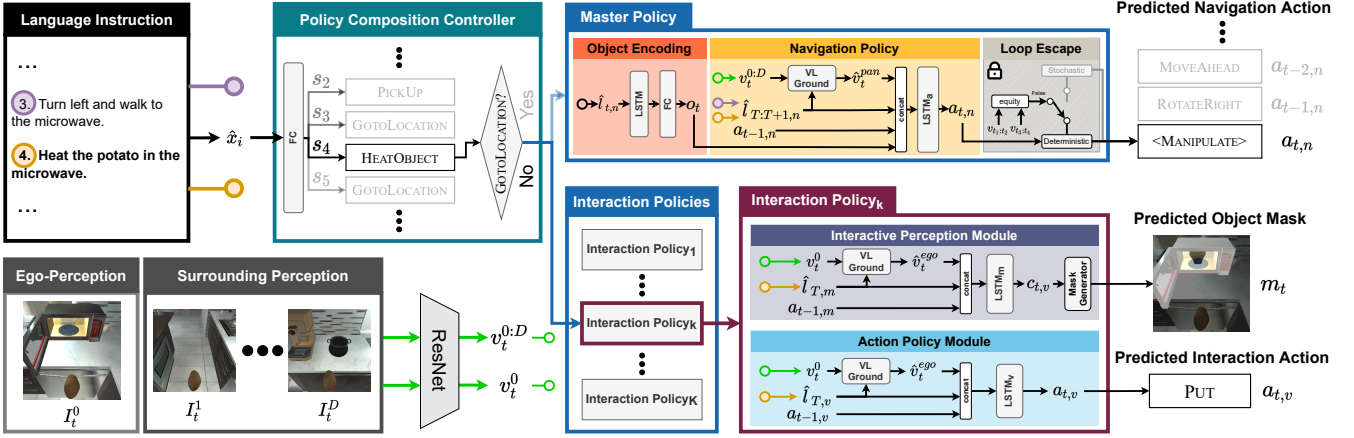Observing that the visual information for navigation considerably varies over time while interacting with objects is

Figure 2: **Model Architecture.** I_t^d denotes an RGB frame from an explorable direction, $d \in [0, D]$, at the time step, t, where $d = 0$ indicates the egocentric direction. We encode I_t^d using a pretrained ResNet and acquire a visual feature, v_t^d. $\hat{x}_i$ denotes each step-by-step instruction. $\hat{l}_{T,v}$, $\hat{l}_{T,m}$ denotes the encoded instruction for the 'interactive perception module' and 'action prediction module' respectively. $\hat{l}_{T:T+1,n}$ denotes the encoded 'subtask' instruction described in Sec. 3.2. T refers to the index of the current subgoal. In our master policy, OEM outputs object encoding, o_t, using $\hat{l}_{T:T+1,n}$. 'VL-Ground' uses dynamic filters to capture the correspondence between visual and language features and outputs attended visual features, $\hat{v}_t^{pan}$ and $\hat{v}_t^{ego}$.

largely stationary, we argue that the agent benefits from learning different policy modules for these two different tasks as follows. The navigation needs to reason about the temporal history and global environment information. The interaction with objects requires focusing on local visual cues for precise object localization. In addition, there is a sample imbalance between navigation and interaction actions as navigation actions are far more frequent than interaction actions. This would bias a learned model towards more frequent actions, *i.e.*, navigation.

Based on these observations, we design an architecture with three levels of compositional learning; (1) a high-level policy composition controller (PCC) that uses language instructions to generate a sequence of sub-objectives, (2) a master policy that specialises in navigation and determines when and where the agent is required to perform interaction tasks, and (3) interaction policies (IP) that are a collection of subgoal policies that specialise in precise interaction tasks.

Specifically, MCR-Agent first analyzes each language instruction and uses the information to determine the basic high-level policy sequence required to perform the task. Following the predicted sequence, the control of the agent is shifted between (1) the master policy and (2) different interaction policies for object interaction. Moreover, all interaction policies are compositional and independent, which allows formulating an instance-specific high-level action sequence. In particular, we learn multiple interaction policies, each of which specialises in a different sub-objective and can be integrated in a precise order to complete long-horizon tasks. We illustrate the model overview in Fig. 2.

3.1 Policy Composition Controller

The nature of the long-horizon instruction following is highly complex. To address this, we argue that it is beneficial to first generate a high-level subgoal sequence, and then tackle each subgoal individually. Specifically, the tra-

jectories are first divided into meaningful subgoals based on the given language instruction (called 'step-by-step' instruction) (Shridhar et al. 2020). For inferring the subgoals, we propose a policy composition controller (PCC), shown as dark cyan box in Fig. 2, that predicts a subgoal $\mathcal{S} = \{s_i\}$ (where s_i belongs to a set of predefined subgoals) for each 'step-by-step' instruction. The PCC's predictions correlate to semantic subgoals, subjecting the agent's logic to observation. It gives the intuition on what the agent is attempting to accomplish at any particular instance. This enables us to track the progress of task completion by the agent.

Specifically, we first encode the language instructions with a Bi-LSTM, followed by a self-attention module. Each encoded step-by-step language instruction $\hat{x}_i$ is used as input for the PCC to generate the subgoal sequences. The agent completes these subgoals in the specified order to accomplish the goal task. Formally, for each language encoding $\hat{x}_i$, the PCC predicts the subgoal action as:

$$s_i = \arg\max_k(FC_1(\hat{x}_i)), \quad \text{where } k \in [1, N_{subgoals}], \quad (1)$$

where FC_1 denotes a single layer perceptron, and $N_{subgoals}$ denotes the number of subgoals. We train the PCC module using imitation learning, with the associated subgoal labels. On the validation split used in (Shridhar et al. 2020), the controller achieves 98.5% accuracy. We provide further details on these subgoals in the supplementary for space's sake.

3.2 Master Policy

As we discussed in Sec. 3, the reasoning required for navigation is significantly varied from the interaction. To this end, we propose to use a dedicated module for navigation, which we call 'master policy' (illustrated by the upper-right blue box in Fig. 2). It not only performs navigation but simultaneously also marks the locations for object interaction along the way. In other words, it generates the navigational action sequence based on the multi-modal inputs.

Specifically, let $\mathscr{A}_n$ denote the set of primitive navigation actions {MOVEAHEAD, ROTATERIGHT, ROTATELEFT, LOOKUP, LOOKDOWN}. The master policy learns to navigate in the environment by learning a distribution over $\mathscr{A}_n \cup$ <MANIPULATE>, where <MANIPULATE> is the abstract token we introduce for the agent to signify when to move control to the next level of the hierarchy, *i.e.*, the interaction policies, for completing manipulation subgoals. It comprises of two modules: (1) an *object encoding module* that provides information about the object the agent needs to locate for interaction, and (2) a navigation policy that outputs the navigation action sequence based on the multi-modal input for traversing the environment. For the instruction to be used in the master policy, we additionally propose a new way for combining subtask language instructions.

Subtask Language Encoding. The language instructions for a given task can be divided into two types; (1) navigation and (2) interaction. We observed that for completing a given compositional task, the agent needs to navigate to the necessary locations and then interact with relevant objects. An embodied task would consist of multiple combinations of such pairs with varying locations and interaction subgoals.

We further propose a method for encoding the combination of instructions for navigation. In particular, we regard the subtask instruction as a combination of (1) navigation to discover the relevant object and (2) corresponding interactions. For instance, in the subtask, *"Turn around and walk to the garbage bin by the TV. Pick up the blue credit card on the TV stand."*, the agent needs to interact with the credit card, which is crucial information for the agent and also serves as a navigational criterion, *i.e.*, the agent should stop if it encounters the credit card in close vicinity. We observe that this information is often missing in a navigation command but present in the next interaction instruction. We encode these language instruction combinations in a similar manner as PCC. Here, $\hat{l}_{T:T+1,n}$ refers to the encoded feature of the combined subtask instruction of the navigation subgoal T and the corresponding interaction subgoal $T+1$.

Object Encoding Module (OEM) (box in orange). Locating the required objects in an essential part of navigation. Trying to interact with incorrect objects can lead to catastrophic failure. To find the correct object, we propose an object encoding module that takes as input the subtask language instruction $l_{T:T+1,n}$ and gives the target object that the agent must locate for interaction. This guides the agent's navigation by acting as a *navigation subgoal monitor* that indicates the end of the navigation subgoal and shifts control to the next interaction policy. The object encoder is composed of a Bi-LSTM with a two-layer perceptron which outputs the object class (Eq. 2). During navigation, the subgoal monitor uses a pretrained object detector (He et al. 2017) that validates if the relevant object is present in the current view or not. If the agent spots the item, it switches to the appropriate interaction policy; otherwise, it continues to navigate.

Navigation Policy (box in yellow). The second component of the master policy is the navigation policy that generates the sequence of navigable low-level actions using the processed multi-modal data as input. The architecture is based on the action prediction module of (Singh et al. 2021). It uses visual features, subtask instruction features, object encoding and the embedding of the preceding time step action as inputs. The goal of the navigation policy is locating the correct object for interaction. Therefore, it utilises the subtask combination instruction $l_{T:T+1}$ as input which provides low-level information relevant for navigation as well as the information about the object that the agent needs to interact with. This aids the agent in arriving at the correct location. To capture the relationship between the visual observation and language instructions, we dynamically generate filters based on the attended language features and convolve visual features with the filters, denoted by "VL-Ground" in Fig. 2. To summarise, the LSTM hidden state $h_{t,n}$ of the master policy decoder, LSTM$_n$, is updated with four different features concatenated together as:

$$o_t = \underset{k'}{\mathrm{argmax}}(\mathrm{FC}_o(\hat{l}_{T:T+1,n})) \quad k' \in [1, N_{objects}]$$
$$h_{t,n} = \mathrm{LSTM}_n([\hat{v}_t^{pan}; \hat{l}_{T:T+1,n}; a_{t-1,n}; o_t])$$
$$a_{t,n} = \underset{k}{\mathrm{argmax}}(\mathrm{FC}_n([\hat{v}_t^{pan}; \hat{l}_{T:T+1,n}; a_{t-1,n}; o_t; h_{t,n}]))$$
$$\text{where } k \in [1, |\mathscr{A}_n| + 1] \tag{2}$$

where $\hat{v}_t^{pan}$ denotes the attended visual features for surrounding views (See supp.) at time step t; $\hat{l}_{T:T+1,n}$ the attended subtask language features for the navigation subgoal T and the corresponding interaction subgoal $T+1$; $a_{t-1,n}$ the action given by master policy in the previous time step; and o_t the object encoding given by the OEM.

Loop Escape (box in gray). In addition, we use subgoal progress monitor and overall progress monitor similar to (Shridhar et al. 2020) to train the navigation policy and also utilize a heuristic loop escape module for escaping the deadlock conditions. We provide details in the supplementary.

3.3 Interaction Policy

To abstract a visual observation to a consequent action, the agent requires a global scene-level comprehension of the visual observation whereas, for the localisation task, the agent needs to focus on both global as well local object-specific information. Following (Singh et al. 2021), we exploit separate streams for action prediction and object localization due to the contrasting nature of the two tasks, illustrated as 'Interaction Policy$_k$' in Fig. 2. Each interaction policy consists of an action policy module which is responsible for predicting the sequence of actions corresponding to the interaction subgoal, and an interaction perception module which generates the pixel-level segmentation mask for objects that the agent needs to interact with at a particular time step.

The task requires the execution of varied subgoals with different levels of complexity. For instance, a HEAT subgoal might require interaction with either a stove or a microwave whereas for a PICKUP subgoal, there is a variety of receptacles but the action sequence is simpler. To focus on individual sub-objectives, we train an interaction policy, for each

Model	Language	Model			Validation				Test			
	Goal-Only	Rule-based Planning	Semantic Memory	Subtask Division	*Seen*		*Unseen*		*Seen*		*Unseen*	
					SR	PLWSR	SR	PLWSR	SR	PLWSR	SR	PLWSR
Seq2Seq (Shridhar et al. 2020)	✗	✗	✗	-	3.70	2.10	0.00	0.00	3.98	2.02	0.39	0.80
MOCA (Singh et al. 2021)	✗	✗	✗	-	25.85	18.95	5.36	3.19	26.81	19.52	7.65	4.21
EmBERT (Suglia et al. 2021)	✗	✗	✗	-	37.44	**28.81**	5.73	3.09	31.77	23.41	7.52	3.58
E.T. (Pashevich et al. 2021)	✗	✗	✗	-	**46.59**	-	7.32	-	**38.42**	**27.78**	8.57	4.10
LWIT (Nguyen et al. 2021)	✗	✗	✗	-	33.70	28.40	9.70	7.30	30.92	25.90	9.42	5.60
HiTUT (Zhang and Chai 2021)	✗	✗	✗	Subgoal	25.24	12.20	12.44	6.85	21.27	11.10	13.87	5.86
M-Track (Song et al. 2022)	✗	✗	✗	Binary	26.70	-	17.29	-	24.79	13.88	16.29	7.66
MCR-Agent (Ours)	✗	✗	✗	Subgoal	34.39	23.04	**20.08**	**10.84**	30.13	21.19	**17.04**	**9.69**
LAV (Nottingham et al. 2021)	✓	✓	✗	Subgoal	12.70	5.9	-	-	13.35	6.31	6.38	3.12
HLSM (Blukis et al. 2021)	✓	✗	✓	Primitive	29.63	-	18.28	-	29.94	8.74	20.27	5.55
MAT (Ishikawa and Sugiura 2022)	✓	✗	✓	Primitive	30.98	-	17.66	-	33.01	-	21.84	-
FILM (Min et al. 2022)	✗	✓	✓	Primitive	38.51	15.06	27.67	11.23	27.67	11.23	26.49	10.55
EPA (Liu et al. 2022)	✓	✓	✓	-	-	-	-	-	39.96	2.56	36.07	2.92

Table 1: **Task and Goal-Condition Success Rates.** ✓ in "Goal-Only" column under "Language" indicates that the corresponding approach uses only goal statements. "Rule-based Planning" indicates if a model exploits rule-based planning such as shortest path algorithms. "Semantic Memory" denotes if the approach requires external memory for storing semantic information (*e.g.*, object positions, classes, *etc.*) using data structures (*e.g.*, grid maps, graphs, *etc.*). "Subtask Division" represents if an agent breaks a task into subtasks (Primitive/Subgoal/Binary) or not (-). A subtask can be a "Primitive" interaction action, a set of "Subgoal" actions, or a "Binary" indicator for navigation/interaction. Our MCR-Agent achieves the highest unseen SR and PLWSR in both validation and test folds compared to prior works without rule-based planning or semantic memories We indicate the highest values in bold among them.

subgoal where $k \in [1, N_{subgoals}]$. We observed that each interaction has its own properties and the navigation information history is irrelevant to the task, which allows us to keep an isolated hidden state for each interaction subgoal. We provide further details about the architecture and the training process for interaction policies in the supplementary.

4 Experiments

Dataset To evaluate our approach in challenging scenarios, we focus on the problem of interactive instruction following in the ALFRED benchmark (Shridhar et al. 2020), which poses numerous challenges including long-term planning, partial observability, and irreversible state changes. To complete a task successfully, an agent needs to navigate through very long horizons. Along the trajectory, the agent can interact with 118 objects in novel environments, which requires a thorough comprehension of both visual observations and their relation with the natural language directives. It provides expert trajectories for the agents performing household tasks in simulated environments on AI2-THOR (Kolve et al. 2017). The dataset is divided into three splits; 'train', 'validation', and 'test' set. To evaluate the generalisation ability of an embodied agent to novel environments, the benchmark further divides 'validation' and 'test' trajectories into *seen* and *unseen* splits. *Unseen* comprises a set of rooms that are held out during training and scenes that are exposed to the agent during training are termed as *seen*. For each task, ALFRED provides a goal statement with multiple (4+) step-by-step instructions describing each subtask (Supp. Sec. 2.4).

Metrics. We use the widely used evaluation metrics in literature (Shridhar et al. 2020; Padmakumar et al. 2022), success rate (SR) is the ratio of the successfully completed episodes to the total episodes. The path length weighted success rate (PLWSR) penalizes the success rate by the length of the trajectory traversed by the agent, which indicates the efficiency of the embodied agent. The goal-condition success rate (Goal-Cond.) is the ratio of the satisfied conditions among the total goal conditions for tasks, which takes into account the partial task completion ability of the agent.

4.1 Comparison with State of the Arts

First, we conduct a quantitative analysis of task success rates (SR) and path length weighted success rates (PLWSR) (Anderson et al. 2018a) by comparing our approach with prior arts on the interactive instruction following task (Shridhar et al. 2020) and summarize the results in Table 1. We indicate the highest value for each metric in bold font among methods that are comparable to ours that do not use rule-based planning or semantic memories for a fair comparison. We also present recent methods that use expensive external supervision or well-designed planners for reference.

We observe that in unseen environments, MCR-Agent outperforms most prior-arts in terms of PLWSR for both test and validation folds. This demonstrates the ability of our agent to accomplish tasks in novel environments with higher efficiency. For seen environments in the test fold, MCR-Agent shows comparable performance with LWIT and EmBERT in terms of SR and PLWSR but these works exhibit relatively stronger bias towards seen environments, which is evidenced by the significant drop in their unseen SR (*i.e.*, 69.5% and 76.3% relative drop, respectively). Similarly, E.T. decently performs in seen environments but shows a drastic drop (77.7% relative) of SR in unseen environments. Note that E.T. utilises extra synthetic training data.

4.2 Bias Towards Seen Environment

It is previously observed that embodied agents relying on low-level visual features for perception generally exhibit bias towards seen environments (Zhang, Tan, and Bansal 2020). Unfortunately, MCR-Agent also exhibits similar bias
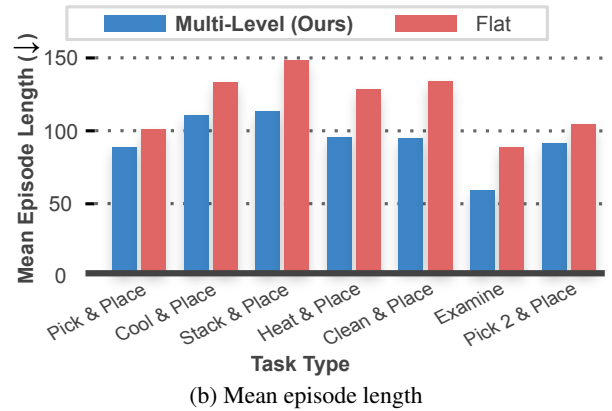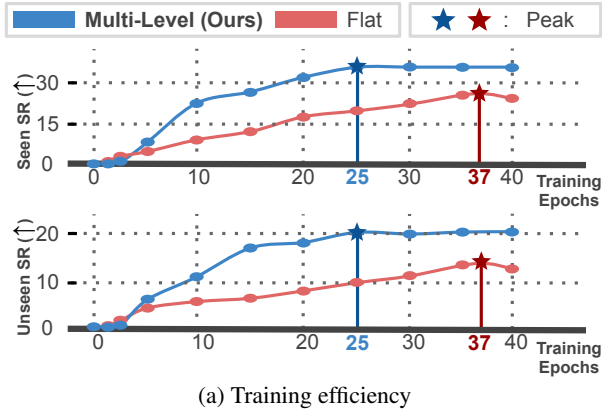
(a) Training efficiency

(b) Mean episode length

Figure 3: **Multi-level policy learns faster and more effective action sequences.** Plot (a) shows the learning curves (success rates vs. epochs) of the hierarchical and flat policy agents for unseen and seen environments. Plot (b) presents the average length of an episode traversed by a hierarchical or flat policy for the seven task types (Shridhar et al. 2020). The flat policy denotes the NIH ablated agent, #(c) in Table 2.

towards seen environments but the degree is significantly lower than other works (E.T., LAV, MOCA).

To mitigate this bias, recent works such as HLSM, MAT, FILM, and EPA utilize spatial semantic representations based on the depth followed by additional depth supervision and semantic segmentation data by assuming perfect egomotion that enables retrieval of accurate camera poses for estimation of the environment layout. These assumptions limit the approaches' deployment capabilities since perfect egomotion may not be accessible in a real-world scenario and such spatial representations may lead to an exponential increase in memory requirements when deployed in larger environments. Note that our approach outperforms all these works and shows comparable performance with FILM in terms of PLWSR score without requiring additional memory and perfect egomotion for generating spatial maps.

Furthermore, HLSM and MAT redefine the agent's action space to adopt a pretrained grid-based navigation system on 3D semantic maps for effective navigation. Similarly, FILM and EPA are equipped with rule-based algorithms for obstacle-free path planning. These agents incorporate heuristics for performance gains whereas MCR-Agent utilises purely learning-based algorithms. While the heuristics may help task completion (improved SR), they adversely affect the efficiency and generalisation of the agents as evidenced by the drop in unseen PLWSR for HLSM and EPA.

4.3 Multi-level Policy *vs*. Flat Policy

We compare the learning efficiency for the hierarchical and flat policy agents for seen and unseen environments. The performance of our hierarchical agent and the flat agent are compared quantitatively in relation to the number of iterations (expressed in terms of epochs), and the results are presented in Fig. 3. As shown, the multi-level hierarchical policy gives a major improvement over the flat policy. Higher success rates in unseen scenarios evidence its ability to perform in novel environments. As depicted in Table 2 (#(a) *vs*. #(c)), for seen and unseen task SR, the hierarchical agent outperforms the flat policy by 8.04% and 7.65%, respectively. In both seen and unseen 'Goal-Cond.', the hierarchical approach outperforms, with improvements

of 10.51% and 9.38%, respectively. The greater performance of the hierarchical approach on both overall task success rate and goal condition suggests its comprehension of both short-term subtasks and long-horizon whole tasks.

The multi-level hierarchical agent converges significantly faster than the flat agent (25th *vs*. 37th epoch), as shown in Fig. 3a, demonstrating the computational efficiency of our approach. Our policies are trained in two stages. We train interaction policies first, which collectively takes two epochs to converge. Details on convergence of interaction policies are provided in the supplementary. We include them in computation and begin the hierarchical agent's curve from the 3rd epoch, which is effectively the 1st epoch for the master policy.

Fig. 3b represents the average lengths of a successful trajectory traversed by the hierarchical and flat policy agents, for different task types, contrasting the efficiency of each agent. The hierarchical agent consists of the master policy that is dedicated solely to navigation, giving it a significant advantage over the flat agent that learns everything using the same network parameters. It was observed that due to the wide action space, the flat agent occasionally executes irrelevant interactions along the trajectory, which is not the case with MCR-Agent. The dedicated action sets for the master policy and interaction policies improve MCR-Agent by allowing the agent to avoid any unnecessary interactions while traversing to discover the desired object. The interaction policies also perform significantly better because they only master certain short-horizon tasks, which speeds up and simplifies the learning process. We also provide the subgoal performance for each module in the supplementary.

4.4 Interpretable Subgoals

The interpretability of embodied agents recently gain attention in the literature for the transparency of their reasoning process (Patel et al. 2021; Dwivedi et al. 2022). Despite recent advances in the domain, many approaches still provide little to no transparency about the agent's actions due to their primitive action space that cannot fully represent the intention of the agents. To demystify the agent's behavior, MCR-Agent generates semantically meaningful sub-

	Components			Validation-Seen		Validation-Unseen	
#	MIP	NIH	OEM	Task	Goal-Cond.	Task	Goal-Cond.
a)	✓	✓	✓	$34.39_{(0.2)}$	$41.96_{(0.5)}$	$20.08_{(0.3)}$	$38.42_{(0.2)}$
b)	✓	✓	✗	$28.61_{(0.4)}$	$32.96_{(0.3)}$	$13.31_{(0.9)}$	$29.13_{(0.3)}$
c)	✓	✗	✓	$26.35_{(0.9)}$	$31.45_{(0.9)}$	$12.43_{(0.7)}$	$29.04_{(0.9)}$
d)	✗	✓	✓	$31.08_{(0.9)}$	$39.26_{(0.8)}$	$15.82_{(0.5)}$	$30.81_{(0.3)}$
e)	✗	✗	✓	$20.59_{(1.4)}$	$25.13_{(2.1)}$	$7.45_{(1.1)}$	$14.05_{(1.0)}$
f)	✗	✓	✗	$23.54_{(1.8)}$	$31.61_{(1.5)}$	$10.30_{(0.7)}$	$25.42_{(1.1)}$

Table 2: **Ablation study for components of MCR-Agent.** We report the task success rate for each ablation. ✓ and ✗ denote that the corresponding component is present/absent in MCR-Agent. MIP (Modular Interaction Policy) denotes the subgoal modules for interaction policies. NIH (Navigation Interaction Hierarchy) denotes the third level of hierarchy between navigation and interaction policies. OEM (Object Encoding Module) denotes the object encoding module. We report averages of 5 runs with random seeds with standard deviations depicted in sub-script parentheses (*e.g.*, (0.2)).

goals that subject the agent's logic to observation ('What is the agent attempting to accomplish right now?'). This makes it easier for humans to monitor the progress of task.

The generated subgoals are far more interpretable than low-level action sequences. For instance, a low-level PUT action might be associated with any of the subgoals such as HEAT, COOL, or PUT. The high-level semantics are reasoned about by the hierarchical agent, and the agent's intent is considerably clearer. For instance, if the agent is performing a COOL subgoal, then it is more likely to interact with the refrigerator. If it is a HEAT subgoal, then it is more likely to interact with a microwave or stove. The subgoal information provided by the PCC provides extra useful information to the multi-level agent as well as the observer. In contrast, the flat policy agent considers it as a single atomic action regardless of the object or receptacle involved.

4.5   Ablation Study

We conduct a series of ablation analyses on the proposed components of MCR-Agent and report the results in Table 2 to evaluate the significance of each module. In the supplementary, we further provide ablation studies for model input, design components, task types, and subgoal types.

Without object encoding module (OEM).   We ablate the navigation subgoal monitor and train the navigation policy without object information. The agent can complete some objectives, but it lacks object information, which functions as a stopping criterion, preventing proper navigation. Hence, it is unable to completely comprehend the relationship between the step-by-step instruction and the visual trajectory. This limits the agent's capacity to explore and connect various interaction policies required for task completion, leading to a significant performance drop (Table 2 #(a) *vs.* #(b)).

Without navigation interaction hierarchy (NIH).   Next, we demonstrate the importance of hierarchy between navigation and interaction policies *i.e.* the second level of hierarchy in our framework. For this, we utilize the same network for learning navigation and interaction action prediction. For interaction mask generation, we preserve the interaction perception module. To ablate the benefits of the subtask lan-

guage encoding, we use the concatenation of all step-by-step instructions as language input and conduct action and mask prediction while leaving the other modules unaltered. The ablated model's task success rates drop significantly (Table 2 #(a) *vs.* #(c)), showing that it is unable to effectively utilise the available inputs.

Without modular interaction policy (MIP).   In modular networks, the decision-making process is separated into numerous modules. Each module is designed to perform a certain function and is put together in a structure that is unique to each trajectory instance. Because of their compositional nature, such networks with the help of specialised modules often perform better in new environments than their flat counterparts (Hu et al. 2019; Blukis et al. 2019). We present a quantitative comparison of interaction policies' modular structure (Table 2 #(a) *vs.* #(d)). For this experiment, we train a single policy module to learn all interaction tasks. The decoupled pipeline for action and mask prediction, as well as the rest of the settings, are preserved. The modular agent outperforms the non-modular agent by 3.31% and 4.26% in seen and unseen task SR, respectively. It also performs significantly well in both seen and unseen 'Goal-Cond.' criteria, with gains of 2.70% and 7.61%, respectively. The greater performance of the modular policy in both task and goal-condition metrics highlights the benefits of modular structure in long-horizon planning tasks. Next, we provide the individual performance of the two major components of our framework which brings the most empirical gain, OEM and NIH in the absence of other components.

Object encoding module only.   In this ablation, we evaluate the effect of the object encoding module (OEM) in the absence of the hierarchical and modular structure. This makes the agent flat and thus analogous to (Singh et al. 2021) except for OEM. The agent (Table 2 #(e)) demonstrates significantly higher performance than (Singh et al. 2021), highlighting the relevance of target object information for navigation and the effectiveness of the proposed OEM.

Navigation interaction hierarchy only.   While ablating the modular structure and object encoding module, we observe degradation in performance (Table 2 #(f)) which implies that the multi-level hierarchical architecture needs to include other proposed components for optimal performance. The overall performance improves when these components are combined (#(a) *vs.* #(f)) indicating that the proposed components are complementary to each other.

5   Conclusion

We address the problem of interactive instruction following. To effectively tackle the long horizon task, we propose a multi-level compositional approach to learn agents that navigate and manipulate objects in a divide-and-conquer manner for the diverse nature of the entailing task. To improve navigation performance, we propose an object encoding module to explicitly encode target object information during internal state updates. Our approach yields competitive performance with higher efficiency than prior arts in novel environments without extra supervision and well-designed planners.

Acknowledgments

References

Anderson, P.; Chang, A.; Chaplot, D. S.; Dosovitskiy, A.; Gupta, S.; Koltun, V.; Kosecka, J.; Malik, J.; Mottaghi, R.; Savva, M.; et al. 2018a. On evaluation of embodied navigation agents. *arXiv:1807.06757*.

Anderson, P.; Wu, Q.; Teney, D.; Bruce, J.; Johnson, M.; Sünderhauf, N.; Reid, I.; Gould, S.; and van den Hengel, A. 2018b. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*.

Andreas, J.; Klein, D.; and Levine, S. 2017. Modular Multitask Reinforcement Learning with Policy Sketches. In *ICML*.

Blukis, V.; Paxton, C.; Fox, D.; Garg, A.; and Artzi, Y. 2021. A Persistent Spatial Semantic Representation for High-level Natural Language Instruction Execution. *arXiv:2107.05612*.

Blukis, V.; Terme, Y.; Niklasson, E.; Knepper, R. A.; and Artzi, Y. 2019. Learning to map natural language instructions to physical quadcopter control using simulated flight. *arXiv:1910.09664*.

Chen, H.; Suhr, A.; Misra, D.; Snavely, N.; and Artzi, Y. 2019. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *CVPR*.

Corona, R.; Fried, D.; Devin, C.; Klein, D.; and Darrell, T. 2020. Modular Networks for Compositional Instruction Following. *arXiv:2010.12764*.

Das, A.; Datta, S.; Gkioxari, G.; Lee, S.; Parikh, D.; and Batra, D. 2018a. Embodied Question Answering. In *CVPR*.

Das, A.; Gkioxari, G.; Lee, S.; Parikh, D.; and Batra, D. 2018b. Neural Modular Control for Embodied Question Answering. In *CoRL*.

Dwivedi, K.; Roig, G.; Kembhavi, A.; and Mottaghi, R. 2022. What Do Navigation Agents Learn about Their Environment? In *CVPR*.

Gordon, D.; Kembhavi, A.; Rastegari, M.; Redmon, J.; Fox, D.; and Farhadi, A. 2018. Iqa: Visual question answering in interactive environments. In *CVPR*.

Gupta, S.; Davidson, J.; Levine, S.; Sukthankar, R.; and Malik, J. 2017. Cognitive Mapping and Planning for Visual Navigation. In *CVPR*.

Hayes-Roth, B.; and Hayes-Roth, F. 1979. A Cognitive Model of Planning. In *Cognitive Science*.

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask r-cnn. In *ICCV*.

Hu, R.; Fried, D.; Rohrbach, A.; Klein, D.; Darrell, T.; and Saenko, K. 2019. Are you looking? grounding to multiple modalities in vision-and-language navigation. *arXiv:1906.00347*.

Ishikawa, S.; and Sugiura, K. 2022. Moment-based Adversarial Training for Embodied Language Comprehension. In *ICPR*.

Kolve, E.; Mottaghi, R.; Han, W.; VanderBilt, E.; Weihs, L.; Herrasti, A.; Gordon, D.; Zhu, Y.; Gupta, A.; and Farhadi, A. 2017. AI2-THOR: An Interactive 3D Environment for Visual AI. *arXiv:1712.05474*.

Krantz, J.; Wijmans, E.; Majumdar, A.; Batra, D.; and Lee, S. 2020. Beyond the Nav-Graph: Vision-and-Language Navigation in Continuous Environments. In *ECCV*.

Landi, F.; Baraldi, L.; Corsini, M.; and Cucchiara, R. 2019. Embodied Vision-and-Language Navigation with Dynamic Convolutional Filters. In *BMVC*.

Li, Y.; He, J.; Zhou, X.; Zhang, Y.; and Baldridge, J. 2020. Mapping Natural Language Instructions to Mobile UI Action Sequences. In *ACL*.

Liu, X.; Palacios, H.; Muise; and Christian. 2022. A Planning based Neural-Symbolic Approach for Embodied Instruction Following. In *Embodied AI Workshop @ CVPR*.

Ma, C.-Y.; Lu, J.; Wu, Z.; AlRegib, G.; Kira, Z.; Socher, R.; and Xiong, C. 2019. Self-Monitoring Navigation Agent via Auxiliary Progress Estimation. In *ICLR*.

Min, S. Y.; Chaplot, D. S.; Ravikumar, P.; Bisk, Y.; and Salakhutdinov, R. 2022. FILM: Following Instructions in Language with Modular Methods. In *ICLR*.

Misra, D.; Langford, J.; and Artzi, Y. 2017. Mapping instructions and visual observations to actions with reinforcement learning. In *EMNLP*.

Nguyen, V.-Q.; Suganuma, M.; Okatani; and Takayuki. 2021. Look Wide and Interpret Twice: Improving Performance on Interactive Instruction-following Tasks. *arXiv:2106.00596*.

Nottingham, K.; Liang, L.; Shin, D.; Fowlkes, C. C.; Fox, R.; and Singh, S. 2021. Modular framework for visuomotor language grounding. In *Embodied AI Workshop @ CVPR*.

Padmakumar, A.; Thomason, J.; Shrivastava, A.; Lange, P.; Narayan-Chen, A.; Gella, S.; Piramuthu, R.; Tur, G.; and Hakkani-Tur, D. 2022. TEACh: Task-driven Embodied Agents that Chat. In *AAAI*.

Pashevich, A.; Schmid, C.; Sun; and Chen. 2021. Episodic Transformer for Vision-and-Language Navigation. *arXiv:2105.06453*.

Patel, S.; Wani, S.; Jain, U.; Schwing, A.; Lazebnik, S.; Savva, M.; and Chang, A. X. 2021. Interpretation of Emergent Communication in Heterogeneous Collaborative Embodied Agents. In *ICCV*.

Shridhar, M.; Thomason, J.; Gordon, D.; Bisk, Y.; Han, W.; Mottaghi, R.; Zettlemoyer, L.; and Fox, D. 2020. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *CVPR*.

Singh, K. P.; Bhambri, S.; Kim, B.; Mottaghi, R.; and Choi, J. 2021. Factorizing Perception and Policy for Interactive Instruction Following. In *ICCV*.

Song, C. H.; Kil, J.; Pan, T.-Y.; Sadler, B. M.; Chao, W.-L.; and Su, Y. 2022. One Step at a Time: Long-Horizon Vision-and-Language Navigation with Milestones. In *CVPR*.

Suglia, A.; Gao, Q.; Thomason, J.; Thattai, G.; and Sukhatme, G. 2021. Embodied BERT: A Transformer Model for Embodied, Language-guided Visual Task Completion. *arXiv:2108.04927*.

Wortsman, M.; Ehsani, K.; Rastegari, M.; Farhadi, A.; and Mottaghi, R. 2019. Learning to learn how to learn: Self-adaptive visual navigation using meta-learning. In *CVPR*.

Xu, N.; Masling, S.; Du, M.; Campagna, G.; Heck, L.; Landay, J.; and Lam, M. 2021. Grounding Open-Domain Instructions to Automate Web Support Tasks. In *NAACL*.

Yang, W.; Wang, X.; Farhadi, A.; Gupta, A.; and Mottaghi, R. 2018. Visual semantic navigation using scene priors. *arXiv:1810.06543*.

Yu, L.; Chen, X.; Gkioxari, G.; Bansal, M.; Berg, T. L.; and Batra, D. 2019. Multi-Target Embodied Question Answering. In *CVPR*.

Zhang, Y.; and Chai, J. 2021. Hierarchical Task Learning from Language Instructions with Unified Transformers and Self-Monitoring. *arXiv:2106.03427*.

Zhang, Y.; Tan, H.; and Bansal, M. 2020. Diagnosing the environment bias in vision-and-language navigation. *arXiv:2005.03086*.

Zhou, S.; Yin, P.; and Neubig, G. 2021. Hierarchical Control of Situated Agents through Natural Language. *arXiv preprint arXiv:2109.08214*.

Zhu, F.; Zhu, Y.; Chang, X.; and Liang, X. 2020. Vision-Language Navigation With Self-Supervised Auxiliary Reasoning Tasks. In *CVPR*.

Zhu, Y.; Gordon, D.; Kolve, E.; Fox, D.; Fei-Fei, L.; Gupta, A.; Mottaghi, R.; and Farhadi, A. 2017. Visual semantic planning using deep successor representations. In *ICCV*.