Factorizing Perception and Policy for Interactive Instruction Following

https://github.com/gistvision/moca

Kunal Pratap Singh[*,2,§] Suvaansh Bhambri[*,1] Byeonghwi Kim[*,1] Roozbeh Mottaghi[2] Jonghyun Choi[1,†]
[1]GIST, South Korea [2]Allen Institute for AI
kunals@allenai.org, suvaansh2008bhambri@gmail.com, byeonghwikim@gm.gist.ac.kr
roozbehm@allenai.org, jhc@gist.ac.kr

Abstract

*Performing simple household tasks based on language directives is very natural to humans, yet it remains an open challenge for AI agents. The 'interactive instruction following' task attempts to make progress towards building agents that jointly navigate, interact, and reason in the environment at every step. To address the multifaceted problem, we propose a model that factorizes the task into interactive perception and action policy streams with enhanced components and name it as **MOCA**, a Modular Object-Centric Approach. We empirically validate that MOCA outperforms prior arts by significant margins on the ALFRED benchmark with improved generalization.*

1. Introduction

The prospect of having a robotic assistant that can carry out daily chores based on language directives is a distant dream that has eluded the research community for decades. On recent progress in computer vision, natural language processing and embodiment, several benchmarks have been developed to encourage research on various components of such instruction following agents including navigation [2, 6, 8, 23], object interaction [30, 41], and interactive reasoning [11, 15] in visually rich 3D environments [5,22,33]. However, to move towards building realistic assistants, the agent should possess all these abilities. Taking a step forward, we address the more holistic task of *interactive instruction following* [15, 30, 34, 41] which requires an agent to navigate through an environment, interact with objects, and complete long-horizon tasks, following natural language instructions with egocentric vision.

To accomplish a goal in the interactive instruction following task, the agent should infer a sequence of actions and object interactions. While action prediction requires global semantic cues, object localisation needs a pixel-level understanding of the environment, making them semantically different tasks. In addition, in neuroscience literature [14], there is a human visual cortex model that has two pathways; the ventral stream (involved with object perception) and the dorsal stream (involved with action control). Inspired by them, we present a **M**odular **O**bject-**C**entric **A**pproach (**MOCA**) to factorize interactive perception and action policy in separate streams in a unified end-to-end framework for building an interactive instruction following agent. Specifically, our agent has the action policy module (APM) which is responsible for sequential action prediction and the interactive perception module (IPM) that localises the objects to interact with.

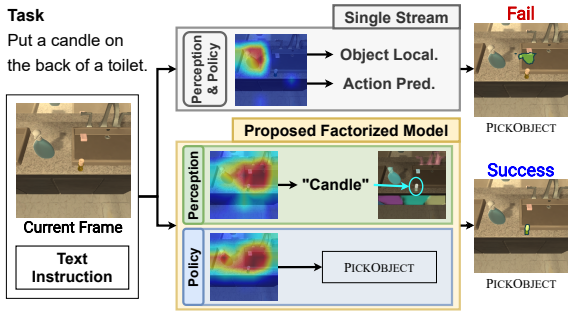


Figure 1: We divide interactive instruction following into perception and policy. Each heat-map indicates where a stream focuses on in the given visual observation. While a single stream exploits the same features for pixel-level and global understanding and thus fails to interact with the object, our factorized approach handles perception and policy separately and interacts successfully.

Figure 1 shows that our two-stream model is more beneficial than the single-stream one. The heat maps indicate the model's visual attention. For the action of 'picking up the candle,' the proposed factorized model focuses on a candle in both the streams and results in a successful interaction. In contrast, the single-stream model does not attend on the candle, implying the challenge to handle two different predictions in a single stream.

In the IPM, we propose to reason about object classes

*: equal contribution. §: work done while with GIST. †: corresponding author.

for better localisation and name it object-centric localisation (OCL). We further improve the localising ability in time by using the spatial relationship amongst the objects that are interacted with over consecutive time steps. For better grounding of visual features with textual instructions, we propose to use dynamic filters [20, 24] for its effectiveness of cross-modal embedding. We also show that these components are more effective when employed in a model that factorizes perception and policy.

We train our agent using imitation learning, specifically behavior cloning. When a trained agent's path is blocked by immovable objects like walls, tables, kitchen counters, *etc*. at inference, however, it is likely to fail to escape such obstacles since the ground truth contains only perfect expert trajectories that finish the task without any errors. To avoid such errors, we further propose an obstruction evasion mechanism in APM. Finally, we adopt data augmentation to address the sample insufficiency of imitation learning.

We empirically validate our proposed method on the recently proposed ALFRED benchmark [34] and observe that it significantly outperforms prior works in the literature by large margins in all evaluation metrics.

We summarize our contributions as follows:

- We propose to factorize perception and policy for embodied interactive instruction following tasks.
- We also present an object-centric localisation and an obstruction evasion mechanism for the task.
- We show that this agent outperforms prior arts by large margins in all metrics.
- We present qualitative and quantitative analysis to demonstrate our method's effectiveness.

2. Related Work

Embodied Instruction Following. Vision and language navigation tasks require an agent to reach a goal by following natural or templated language instructions in a simulated environment through visual observations [2, 6, 8, 29]. [2] proposed the Vision-and-Language Navigation (VLN) task on the Room2Room (R2R) benchmark where an agent navigates on a fixed underlying navigation graph based on natural language instructions. Substantial improvements [13, 21, 24, 25, 28, 40] have been achieved on this benchmark by various proposals such as progress monitoring [27], augmenting trajectories [13] and environment dropout [37]. Vision and Language Navigation in Continuous Environments (VLN-CE) [23] lifts the assumption of known navigation-graph and perfect agent localisation from R2R [2]. Recently, [35] presented ALFWorld which contains TextWorld [10] based environments corresponding to the ones in [34] that allows agents to learn in an abstract space before transfer to actual embodied environments.

Interactive Instruction Following is a much complex paradigm that combines the navigation aspect of tasks like VLN with the interactive abilities of a manipulation agent [3]. The recently introduced ALFRED [34] benchmark serves as a suitable testbed for this task. It requires an agent to navigate via egocentric visual observations and also interact with objects by producing a pixel-wise mask to complete a task in an embodied environment. Shridhar *et al*. [34] proposed a single-stream Seq-to-Seq model with progress monitoring [27] for this task. Even though similar models perform well on VLN [2, 27], it fails to generalize to unseen environments in interactive instruction following task indicating its difficulty and need of extensive investigation to develop a well-performing agent. [32] present a planner-based geometry-aware approach for the task. However they split the training data itself to create train, validation and test folds, and do not have any open source code or splits, due to which we omit them in comparison. Recently, Nguyen *et al*. [31] presented an approach wherein they relax the egocentric vision constraint of the task by collecting multiple views per time step, essentially making it similar to panoramic views in VLN [2]. They process these visual features via hierarchical attention with the instructions. Here, we propose to factorize the task into perception and policy to effectively learn an agent for this task. Note that we do not relax any constraint set of the original ALFRED benchmark and still outperform prior arts [31, 34].

Two-stream Architectures. [4, 12, 36, 38] have shown the success of multi-stream architectures for capturing different features from given inputs. Inspired by these, we also propose a two-stream architecture. Contrary to these works, we do not combine the streams to produce a single output but perform two semantically different tasks i.e. Interactive Perception and Action Policy. Recently [7, 19] decouple learning embodied tasks into two parts. Firstly, a perfect perception policy is trained using gridworlds [19] or giving direct access to the environment's state [7]. This is followed by, training the agent on a visually realistic environment to see by imitating the perfect perception policy.

Visual Grounding. Previous visual grounding methods leverage a pre-trained segmentation model [9, 16–18, 39, 42] to generate a set of candidate regions and then predict the best candidate proposal corresponding to the language query. However, these works have been used for localising a single object in one image with a given language description. We extend this to the embodied domain and localise multiple objects in a continuous stream of visual observations given a set of instructions. We propose to split the object localisation into two stages; object class prediction and mask generation (Sec. 3.2.2) and leverage a pre-trained instance segmentation model [16]. This is in contrast with [34] which upsamples a vision-language-action embedding via deconvolution layers to produce a class-agnostic mask. As we show in subsequent sections, this

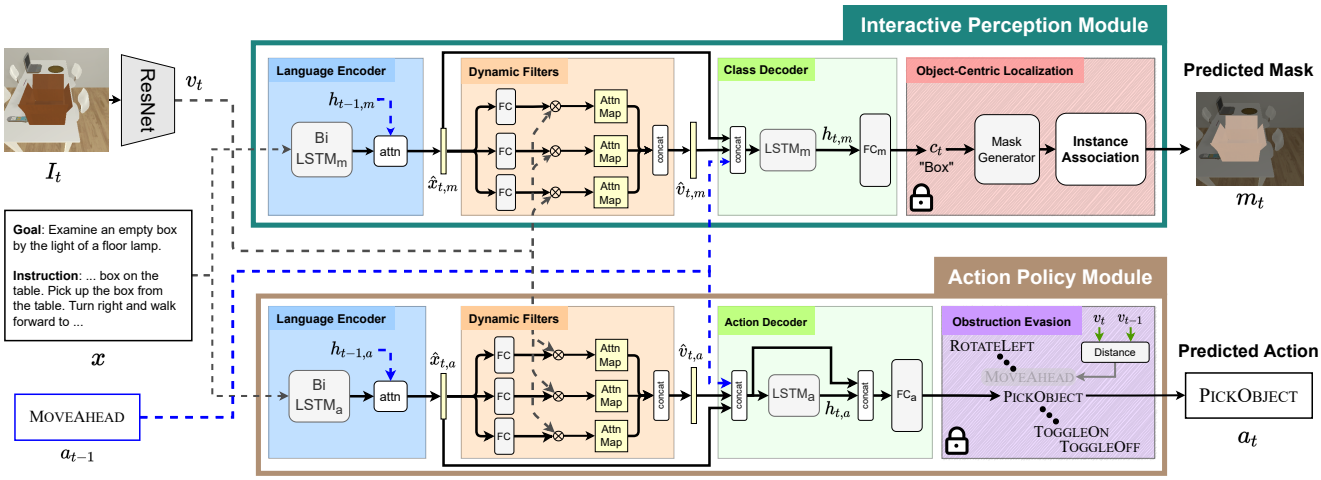


Figure 2: **Model Architecture.** The input frame at the time step, t, and language instructions are denoted by I_t and x. Blue dashed lines denote the path of the action at the previous time step. Subscripts m and a denote that a component belongs to IPM or APM, respectively. ResNet-18 encodes I_t, denoted by v_t. Dynamic filters convolve over visual features, v_t, to give attended visual features, $\hat{v}_{t,m}$ and $\hat{v}_{t,a}$. $h_{t,m}$ and $h_{t,a}$ denote the hidden states of the class and action decoder. The target class, c_t, and the action, a_t, are predicted based on attended visual and language features with the previous action. The 'lock' symbols in the components indicate their use at inference only.

results in poorly localised masks.

Previous works [34] have used concatenation for combining vision and language embeddings. However, this fails to fully capture the cross-modal correspondence. [24] produces dynamic convolution filters that are applied to panoramic visual features to produce action outputs for VLN on R2R benchmark. Motivated by their work, we use dynamic filters for grounding language features with egocentric visual features for interactive instruction following.

3. Approach

An interactive instruction following agent performs a sequence of navigational steps and object interactions based on egocentric visual observations it receives from the environment. These actions and interactions are based on natural language instructions that the agent must follow to accomplish the task.

We approach this by factorizing the model into two streams, *i.e.* interactive perception and action policy, and train the entire architecture in an end-to-end fashion. Figure 2 presents a detailed overview of MOCA.

3.1. Factorizing Perception and Policy

Action prediction requires global scene-level understanding of the visual observation to abstract it to a resulting action. On the other hand, for object interaction, the agent needs to focus on both scene-level and object-specific features to achieve precise localisation [4, 26, 36].

Given the contrasting nature of the two tasks, MOCA has separate streams for action prediction and object localisation. The two streams are the Interactive Perception Module (IPM) and Action Policy Module (APM). Subscripts a and m in following equations indicate whether a component belongs to APM or IPM, respectively. APM is responsible for sequential action prediction. It takes in the instructions to exploit the detailed action-oriented information. IPM localises the pixel-wise mask whenever the agent needs to interact with an object in case of manipulation actions. IPM tries to focus more on object-centric information in the instructions for localisation and interaction. Both IPM and APM receive the egocentric visual observation features at every time step.

3.2. Interactive Perception Module (IPM)

The ability to interact with objects in the environment is key to interactive instruction following, since accomplishing each task requires multiple interactions. The interactive perception module (IPM) facilitates this by predicting a pixel-wise mask to localise the object to interact with.

First, the language encoder in IPM encodes the language instructions and generates attended language features. For grounding the visual features to the language features, we use language guided dynamic filters for generating the attended visual features (Sec. 3.2.1). Then, to temporally align the correct object with their corresponding interaction actions amongst the ones present in the language input, we use previous action embedding along with the visual and language input. For example, in the statement, *Wash the spatula, put it in the first drawer*, the agent first needs to wash the spatula in the sink, for which we have two object classes, namely *spatula* and *sink* that the agent needs to interact with. But this has to be done in a particular order. If the action is PUTOBJECT, the agent needs to predict the sink's mask whereas if it is PICKOBJECT, it needs to predict

the spatula's mask. As shown in Figure 2, the hidden state $h_{t,m}$ of the class decoder, LSTM$_m$, is updated with three different inputs concatenated as:

$$h_{t,m} = \text{LSTM}_m([\hat{v}_{t,m}; \hat{x}_{t,m}; a_{t-1}]) \quad (1)$$

where [;] denotes concatenation. $\hat{x}_{t,m}$ and $\hat{v}_{t,m}$ are the attended language and visual features, respectively. Finally, the class decoder's current hidden state $h_{t,m}$ is then used to predict the mask m_t. This is done by invoking the *object-centric localisation* (Sec. 3.2.2), which helps the agent to accurately localise the object of interest.

3.2.1 Language Guided Dynamic Filters

Visual grounding helps the agent to exploit the relationships between language and visual features. This reduces the agent's dependence on any particular modality while encountering unseen scenarios.

It is a common practice to concatenate flattened visual and language features [17, 18, 34]. However, it might not perfectly capture the relationship between visual and textual embeddings, leading to poor performance of interactive instruction following agents [34].

Dynamic filters are conditioned on language features making them more adaptive towards varying inputs. This is in contrast with traditional convolutions which have fixed weights after training and fail to adapt to diverse instructions. Hence, we propose to use dynamic filters for the interactive instruction following task.

Particularly, we use a filter generator network comprising of fully connected layers to generate dynamic filters which attempt to capture various aspects of the language from the attended language features. Specifically, the filter generator network, f_{DF}, takes the language features, x, as input and produces N_{DF} dynamic filters. These filters convolve with the visual features, v_t, to output multiple joint embeddings, $\hat{v}_t = \text{DF}(v_t, x)$, as:

$$\begin{aligned} w_i &= f_{DF_i}(x), \;\; i \in [1, N_{DF}], \\ \hat{v}_{i,t} &= v_t * w_i, \\ \hat{v}_t &= [\hat{v}_{1,t}; \ldots; \hat{v}_{N_{DF},t}], \end{aligned} \quad (2)$$

where N_{DF}, $*$ and [;] denote the number of dynamic filters, convolution and concatenation operation respectively. We empirically investigate the benefit of using language-guided dynamic filters in Sec. 4.2.

3.2.2 Object-Centric Localisation

The IPM performs object interaction by predicting a pixel-wise interaction mask of the object of interest. We bifurcate the task of mask prediction; *target class prediction* and *instance association*. This bifurcation enables us to leverage the quality of pre-trained instance segmentation models while also ensuring accurate localisation. We refer to this mechanism as 'object-centric localisation (OCL).' We empirically validate the OCL in Sec. 4.2 and 4.3.

Target Class Prediction. As the first step of OCL, we take an object-centric viewpoint to interaction by explicitly encoding the ability to reason about object categories in our agent. To achieve this, MOCA first predicts the target object class, c_t, that it intends to interact with at the current time step t. Specifically, FC$_m$ takes as input the hidden state, $h_{t,m}$, of the class decoder and outputs the target object class, c_t, at time step, t, as shown in Equation 3. The predicted class is then used to acquire the set of instance masks corresponding to the predicted class from the mask generator.

$$c_t = \underset{k}{\text{argmax}} \; \text{FC}_m(h_{t,m}), \quad k \in [1, N_{class}], \quad (3)$$

where FC$_m(\cdot)$ is a fully connected layer and N_{class} denotes the number of the classes of a target object. The target object prediction network is trained as a part of the IPM with the cross-entropy loss with ground-truth object classes.

Instance Association. At inference, given the predicted object class, we now need to choose the correct mask instance of the desired object. We use a pre-trained mask generator to obtain the instance masks and confidence scores. A straightforward solution is to pick the highest confidence instance as it gives the best quality mask of that object. This works well when the agent interacts with the object for the first time. However, when it interacts with the same object over an interval, it is more important to *remember* the object the agent has interacted with, since its appearance might vary drastically due to multiple interactions. Thus, the sole confidence based prediction may result in failed interactions as it lacks memory.

To address all the scenarios, we propose a two-way criterion to select the best instance mask, *i.e.*, 'confidence based' and 'association based.' Specifically, the agent predicts the current time step's interaction mask $m_t = m_{\hat{i},c_t}$ with the center coordinate, $d^*_t = d_{\hat{i},c_t}$, where $\hat{i}$ is obtained as:

$$\hat{i} = \begin{cases} \underset{i}{\text{argmax}} \;\; s_{i,c_t}, & \text{if } c_t \neq c_{t-1}, \\ \underset{i}{\text{argmin}} \;\; ||d_{i,c_t} - d^*_{t-1}||_2, & \text{if } c_t = c_{t-1}, \end{cases} \quad (4)$$

where c_t is the predicted target object class and d_{i,c_t} the center of a mask instance, m_{i,c_t}, of the predicted class.

Figure 3 illustrates an example, wherein the agent is trying to open a drawer and put a knife in it, the same drawer is interacted with over multiple time steps. Table 4 in Sec. 4.2 shows ablation study of our instance association scheme.

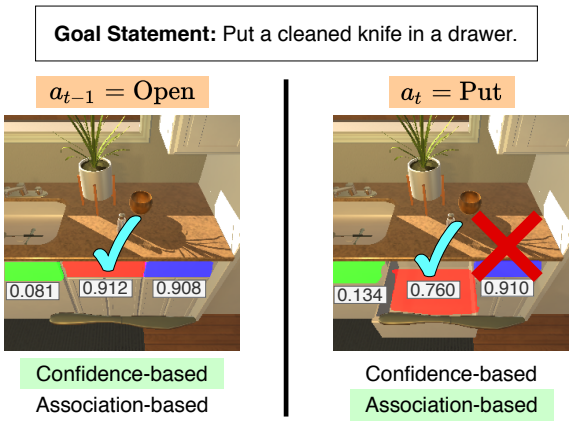


Figure 3: **Qualitative Illustration of Instance Association (IA).** The masks of the drawers are colored with their confidences. ✓ denotes the object interacted with at that time step. × denotes the object replaced by IA. Using the single-fold confidence-based approach could make the agent interact with different drawers since the closed drawer has higher confidence. IA helps the agent to interact with the same drawer and place the knife.

3.3. Action Policy Module (APM)

The Action Policy Module (APM), depicted by the lower block in Figure 2, is responsible for predicting the action sequence. It takes visual features and instructions as input. The attended language features are generated by the language encoder in APM. Same as IPM, we employ language guided dynamic filters for generating attended visual features (Sec. 3.2.1). Although we use a similar architecture for IPM, the information captured by dynamic filters is different from that of APM due to difference in language encodings used for both. The action decoder then takes attended visual and language features, along with the previous action embedding to output the action decoder hidden state, $h_{t,a}$. Finally, a fully connected layer is used to predict the next action, a_t as follows:

$$\begin{aligned} u_a &= [\hat{v}_{t,a}; \hat{x}_{t,a}; a_{t-1}], \quad h_{t,a} = \text{LSTM}_a(u_a) \\ a_t &= \underset{k}{\text{argmax}}(\text{FC}_a([u_a; h_{t,a}]), \quad k \in [1, N_{action}] \end{aligned} \tag{5}$$

where $\hat{v}_{t,a}$, $\hat{x}_{t,a}$ and a_{t-1} denote attended visual features, attended language features, and previous action embedding, respectively. FC_a, takes as input $\hat{v}_{t,a}$, $\hat{x}_{t,a}$, a_{t-1}, and $h_{t,a}$ and predicts the next action, a_t. Note N_{action} denotes the number of actions. We keep the same action space as [34].

The objective function of the APM is the cross entropy with the action taken by expert for the visual observation at each time step as ground-truth.

Obstruction Evasion. The agent learns to not encounter any obstacles during training based on the expert ground truth actions. However, during inference, there are various situations when the agent gets stranded around immovable objects. To address such unanticipated situations, we propose an 'obstruction evasion' mechanism in the APM to avoid obstacles at inference time.

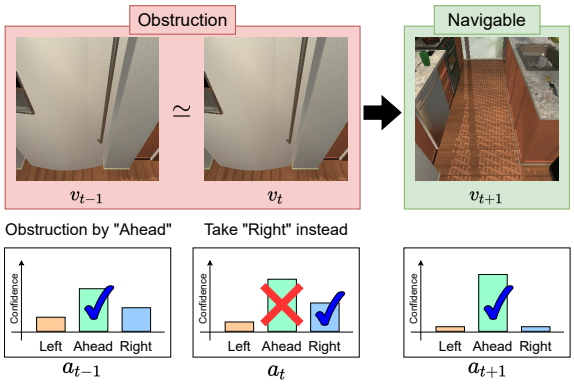


Figure 4: **Obstruction Evasion**. Each plot includes the actions with the top-3 probabilities. ✓ denotes the action taken at that time step. AHEAD with × shows that our agent detects an obstruction at the time step, t, by the criteria in Equation 6. Therefore, our agent predicts the second best action, RIGHT, to escape by removing AHEAD from the action space.

While navigating in the environment, at every time step, the agent computes the distance between visual features at the current time step, v_t, and the previous time step, v_{t-1} with a tolerance hyper-parameter ϵ as following:

$$d(v_{t-1}, v_t) < \epsilon, \tag{6}$$

where $d(v_{t-1}, v_t) = ||v_{t-1} - v_t||_2^2$. When this equation holds, the agent removes the action that causes the obstruction from the action space so that it can escape:

$$a_t = \underset{k}{\text{argmax}}\, \text{FC}_a([u_a; h_{t,a}]), \quad k \in [1, N_{action}] - \{k'\} \tag{7}$$

where k' is the index of a_{t-1}. u_a and FC_a are same as Equation 5. We empirically investigate its effect in Sec. 4.2.

4. Experiments

We present quantitative comparisons and show that we outperform prior works [31, 34] with large margins. We also perform extensive ablation studies and additional analyses over the empirical significance of various components of MOCA and discuss qualitative examples to highlight the importance of our design choices.

Dataset. For training and evaluating on the interactive instruction following task, we use the recently proposed ALFRED benchmark that runs in AI2-THOR [22]. The scenes in ALFRED are divided into 'train', 'validation' and 'test' sets. To evaluate the generalization ability, the validation and test scenes are split into two sections; *seen* and *unseen* folds. Scenes in the seen folds of validation and test data are subsets of those in the train fold. Scenes in the unseen validation and test folds are distinct from the train fold and

Model	Validation				Test			
	Seen		*Unseen*		*Seen*		*Unseen*	
	Task	Goal-Cond	Task	Goal-Cond	Task	Goal-Cond	Task	Goal-Cond
Shridhar *et al*. [34]	3.70 (2.10)	10.00 (7.00)	0.00 (0.00)	6.90 (5.10)	3.98 (2.02)	9.42 (6.27)	0.39 (0.80)	7.03 (4.26)
Nguyen *et al*. [31]	N/A	N/A	N/A	N/A	12.39 (8.20)	20.68 (18.79)	4.45 (2.24)	12.34 (9.44)
MOCA (Ours)	**25.85** (**18.95**)	**34.92** (**26.44**)	**5.36** (**3.19**)	**16.18** (**10.44**)	**26.81** (**19.52**)	**33.20** (**26.33**)	**7.65** (**4.21**)	**15.73** (**11.24**)
Input Ablations								
No Language	2.00 (1.59)	10.85 (5.69)	0.00 (0.00)	4.11 (1.60)	0.59 (0.29)	6.37 (4.24)	0.20 (0.03)	6.82 (3.43)
No Vision	0.12 (0.05)	6.16 (5.11)	0.00 (0.00)	7.26 (6.41)	0.07 (0.03)	4.31 (3.34)	0.20 (0.07)	6.92 (4.72)
Goal-Only	3.90 (2.59)	11.43 (8.65)	0.49 (0.12)	8.40 (4.66)	3.59 (2.39)	10.03 (7.47)	1.11 (0.40)	8.70 (4.96)
Instructions-Only	5.98 (4.52)	14.56 (11.16)	0.49 (0.27)	7.97 (5.09)	6.20 (3.96)	12.44 (9.45)	0.85 (0.36)	7.84 (4.62)
Human	-	-	-	-	-	-	91.00 (85.80)	94.50 (87.60)

Table 1: **Task and Goal-Condition Success Rate.** For each metric, the corresponding path weighted metrics are given in (parentheses). The highest values per fold and metric are shown in **blue**. 'N/A' denotes 'not available' as the scores are not reported in the leaderboard.

from each other. The dataset provides both high-level goal statement and low-level step-by-step instructions. We provide the detailed description of the ALFRED benchmark and our implementation details in the supplementary.

Evaluation Metrics. We follow the evaluation metrics proposed in [34], *i.e*., Success Rate, denoted by *Task*, and Goal Condition Success Rate, denoted by *Goal-Cond*. Additionally, to measure the efficiency of an agent, the above metrics are penalized by the length of the path to compute a path-length-weighted (PLW) score for each metric [1]. For more details on evaluation metrics, kindly refer [34].

4.1. Quantitative Analysis

We first conduct quantitative analysis of the performance on task success rate (Task) and goal-condition success rate (Goal-Cond) and summarise the results in Table 1 with previous methods. As shown in the figure, MOCA shows significant improvement over the prior arts [31, 34] on all metrics. The higher success rate in the unseen scenes indicates its ability to generalize in novel environments. We achieve an improvement of 14.42% and 3.20% in Seen and Unseen Task SR over Nguyen *et al*. [31] that won ALFRED challenge in ECCV 2020. Note that Nguyen *et al*. [31] is a challenge entry, they neither report validation set results nor have a code release, hence the comparison was omitted.

MOCA outperforms them in both *Seen* and *Unseen* 'Goal-Condition' metrics and gives an improvement of 12.52% and 3.39%, respectively. The superior performance of MOCA on both overall task success rate and goal condition indicates its understanding of short-term sub-tasks, as well as long-horizon full tasks. [34] lacks long term task completion ability as indicated by its poor performance on Task Success Rate. As indicated in the parenthesis in Table 1, MOCA provides better Path Length Weighted results for all the metrics which shows the efficiency of our agent. We would also like to acknowledge that in the ALFRED public leaderboard[1], the highest entry is at 9.42 unseen success rate, but it is only an anonymous leaderboard entry with no manuscript or code, thereby we omit it in comparison. We present sub-goal and task type ablations in the supplementary.

[1] https://leaderboard.allenai.org/alfred/submissions/public

4.2. Ablation Study

Input Ablations. We ablate the inputs to our model in Table 1 to investigate the vision and language bias of MOCA. When the agent is only given visual inputs (*No Language*) *i.e*. zeroing out language input, we observe that it's able to perform some tasks in the seen environments by memorising familiar visual sequences, but fails to generalize in the unseen environment.

No Vision setting is able to finish some goal conditions by following navigation instructions, but the lack of visual input handicaps the interaction ability of the agent, hence it drastically fails in both seen and unseen folds.

Goal-Only setting highlights the ability of MOCA to utilise the goal-statement better as compared to Shridhar *et al*. [34]. Since our Action Policy Module (APM) does not utilise the goal-statement as it lacks action-specific information, the action prediction ability of this setting is equivalent to the *No-Language* setting. However, since the goal-statement is used in the Interactive Perception Module (IPM), it allows the agent to perform accurate object interactions and hence achieves much better performance than *No-Language*. This result is a direct benefit of the perception and policy factorization discussed in Sec. 3.1.

Instruction-Only ablation in Table 1 indicates the performance when the agent does not receive the goal-statement. The instructions drastically improve the action prediction ability over the *Goal-Only* setting as the APM can now leverage the detailed action information. However, the IPM is deprived of its language input which depletes the target-class prediction ability (Sec. 3.2.2) of object-centric localisation. This results in many failed interactions and thus it performs worse than our full model and *Goal-Only* setting.

It is also worth noting that for input ablations, the agent is deprived of the dynamic filters for either APM or IPM,

Input		Val-Seen		Val-Unseen	
IPM	APM	Task	Goal-Cond.	Task	Goal-Cond.
G	I	25.85 (18.95)	34.92 (26.44)	5.36 (3.19)	16.18 (10.44)
G,I	I	29.76 (22.33)	39.40 (30.58)	5.97 (3.52)	18.25 (11.78)
G	G,I	28.05 (20.96)	35.89 (28.24)	5.36 (3.21)	17.26 (10.56)
G,I	G,I	26.34 (18.20)	34.28 (25.68)	5.36 (2.72)	16.23 (9.28)

Table 2: **Stream Input Ablations for Interactive Perception Module (IPM) and Action Policy Module (APM).** For each metric, we report the corresponding path weighted scores in parentheses. Each "G" and "I" denotes a goal statement and step-by-step instructions and "G,I" the concatenation of them.

#	FPP	OCL	DF	DA	**Val-Seen** Task	**Val-Unseen** Task
(a)	✓	✓	✓	✓	25.85 (18.95)	5.36 (3.19)
(b)		✓	✓	✓	22.32 (16.17)	4.51 (2.59)
(c)	✓	✓	✓		15.85 (10.02)	2.92 (1.35)
(d)		✓	✓		12.56 (7.05)	2.68 (1.32)
(e)	✓	✓			14.63 (9.80)	2.19 (1.23)
(f)		✓			11.71 (5.42)	1.83 (0.82)
(g)	✓		✓		3.90 (2.40)	0.50 (0.30)
(h)			✓		3.30 (1.70)	0.40 (0.20)

Table 3: **Ablation Study for Each Component of the Proposed Model.** FPP denotes factorizing perception and policy. OCL denotes object-centric Localisation. DF denotes language-guided dynamic filters. DA denotes data augmentation. For each metric, we report task success rates with corresponding path weighted scores in parentheses. The absence of checkmark denotes that the corresponding component is removed.

or both, due to which it fails to perform well on unseen environments in all input ablation settings.

Stream Input Ablations. As mentioned before, we use the goal statement as the input to IPM and instructions for the APM for our experiments. However, we perform an empirical study to show that our framework is not sensitive to this particular choice and can generalize beyond it. We investigate the language inputs with different goal and instruction combinations in Table 2.

We replace the input to APM and/or IPM by a concatenation of goal and instructions similar to [34] and report the task success rate on the resulting combinations. As shown in Table 2, we do not observe any performance degradation, which indicates that our approach is not sensitive to the choice of language inputs. Note that it is possible to optimize the choice of language inputs for minor performance gains, but we keep the current combination for the ease of analysis. Moreover, our goal is to contribute a general framework for interactive instruction following task which is agnostic to language instruction type, that can generalize beyond ALFRED [34].

Model Ablations. To investigate the significance of each component with empirical studies, we perform a series of ablations on MOCA and summarize the results in Table 3. We only present the task success rate due to space constraints. We present the full table in supplementary. # (a) represents our full model. We begin by showing that factorization is important for models both with (# (a) vs. (b)) and without (# (c) vs. (d)) data augmentation. For this ablation, we take the concatenation of goal and instructions as the language input and perform action and mask prediction from a single stream similar to [34] while keeping other modules the same. Note that the presence of (✓) in 'FPP' column indicates whether the model is two-stream(✓) or single-stream (no ✓). We also find that data augmentation is important (# (a) vs. (c)) in training a better and more generalizable agent for the task.

Next, we ablate over the language guided dynamic filters (Sec. 3.2.1). Removing them leads to a decrease in both seen and unseen metrics (# (c) vs. (e)). This drop can be attributed to the lack of cross-modal correspondence between visual and language inputs. We also show that dynamic filters are less effective (# (g) vs. (h)) without factorization. This solidifies our understanding that a two-stream architecture is better-suited for interactive instruction following.

Finally, we ablate over object-centric localisation (OCL) (Sec. 3.2.2). We observe that the performance drastically drops (# (c) vs. (g)) on both seen and unseen folds due to poor localisation, highlighting the effectiveness of our object-centric design. Note that the large drop also indicates the importance of object localisation, and hence interaction in the task. Additionally, we also show that OCL is more effective with factorization (# (e) vs. (f)), further highlighting the importance of factorization for our agent's superior performance. For this ablation, to remove OCL, we directly upsample the joint vision-language-action embedding using deconvolution layers to predict the mask similar to [34].

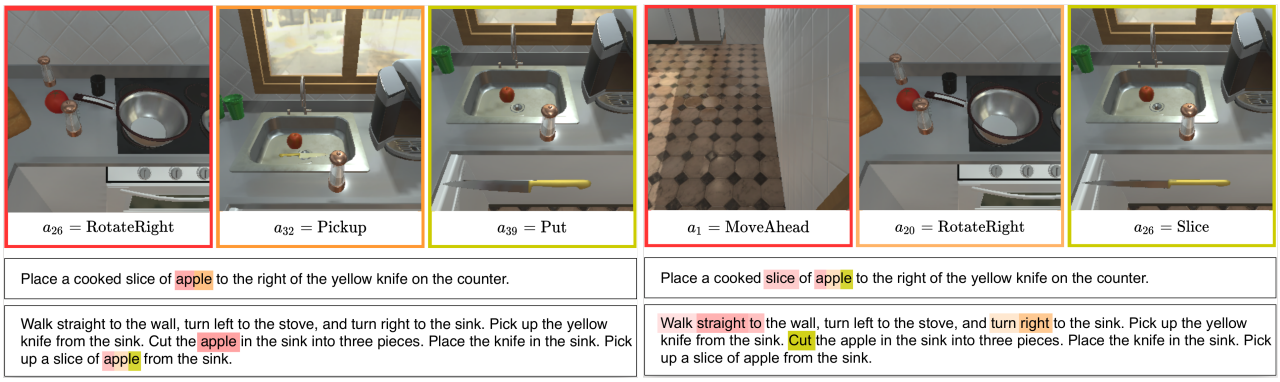
Table 4 ablates obstruction evasion from Sec. 3.3. The performance drop indicates that it helps the agent to avoid obstacles effectively. We also ablate over the Instance-Association (IA) presented in Sec. 3.2.2. For this setting, instead of picking the mask instance for the predicted target class using IAT, we pick a random instance of that class. This setting achieves almost half the performance of MOCA which implies that merely predicting the right object class is not enough, the correct instance must be selected.

Model	Val-Seen		Val-Unseen	
	Task	Goal-Cond	Task	Goal-Cond
MOCA	25.85 (18.95)	34.92 (26.44)	5.36 (3.19)	16.18 (10.44)
– w/o I.A.	23.66 (17.47)	32.48 (25.18)	5.12 (3.04)	15.85 (10.32)
– w/o O.E.	20.00 (15.08)	28.26 (22.67)	3.53 (2.38)	14.25 (10.53)

Table 4: **Ablation for Instance Association and Obstruction Evasion.** Both the components are ablated on the validation set.

4.3. Qualitative Analysis

Factorizing Perception and Policy. We present a qualitative example of the benefit of factorizing perception and



(a) MOCA w/o factorizing perception and policy (b) MOCA

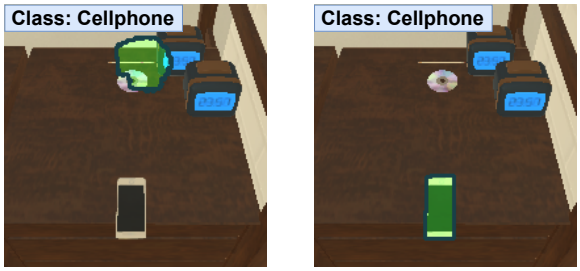
Figure 5: **Language Attention for Single-Stream and Two-Stream Models.** The colors of frame borders and words denote that the agent at the particular frame focuses on the same-colored words. a_t denotes the action taken at the time step, t. (a) Without factorization, the language attention keeps focusing on *apple* irrespective of the action taken. (b) With factorization, the language attention focuses on the words that correspond to the action taken at that time step.

policy. In Figure 5a, for the single stream model *i.e.* without factorization, the language attention focuses on the objects mentioned in the goal statement, such as *apple* on all three shown time steps, even though it is not relevant to the current action ignoring all other action-specific information in the instructions.

However, when perception and policy are factorized and we use a two-stream model, it can effectively encode the representations for both interactive perception and policy. Therefore, the attention mechanism focuses on the correct words for both navigation and interactive actions. Figure 5b qualitatively illustrates this result. For example, at $t = 20$ MOCA attends over *turn right* when it predicts RotateRight. At $t = 26$ when our agent intends to slice the apple, it attends over *Cut*. Note that the only difference between the models is factorization of perception and policy.

Object-Centric Localisation. We also conduct qualitative analysis of the object localisation ability (Sec. 3.2.2) of MOCA. *Object-Centric Localisation* (OCL) allows our method to reason about object classes (Sec. 3.2.2) which ensures interaction with the correct object. This is in contrast with [34] that upsamples a linear embedding via a deconvolution network and predicts class-agnostic masks, thereby not preserving any information about object category. In Figure 6a, for Ours w/o OCL setting, we replace OCL by deconvolution layers similar to [34]. Since it lacks the ability to reason about object class, it predicts inaccurate masks even though the objects are fully observable.

In contrast, in Figure 6b, our full method successfully predicts what objects it intends to interact with (*i.e.*, the cellphone). Identifying the correct objects enables it to predict an accurately localised mask with the mask generator's help. We present further qualitative example videos of MOCA's task completion ability in the supplementary.



(a) MOCA w/o OCL (b) MOCA

Figure 6: **Qualitative Comparison of Object Localisation.** Green regions denote the masks predicted by the models. The ground-truth object class the agent needs to interact with is shown on the top-left corner. OCL denotes object-centric localisation.

5. Conclusion

We explore the problem of interactive instruction following. To address this compositional task, we propose a model that factorizes the task into interactive perception and action policy. We also propose improved components for object localisation and obstacle avoidance. Our method provides a framework that can be adopted by future works on ALFRED and beyond. Our approach outperforms all prior arts by significant margins with superior generalization. We present extensive analysis and insights that can benefit the general paradigm of instruction following.

Acknowledgement. This work was partly supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No.2019R1C1C1009283) and Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-01842, Artificial Intelligence Graduate School Program (GIST)), (No.2019-0-01351, Development of Ultra Low-Power Mobile Deep Learning Semiconductor With Compression/Decompression of Activation/Kernel Data, 25%) and (No. 2021-0-02068, Artificial Intelligence Innovation Hub).

References

[1] Peter Anderson, Angel X. Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and Amir R. Zamir. On evaluation of embodied navigation agents. arXiv:1807.06757, 2018. 6

[2] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*, 2018. 1, 2

[3] Dhruv Batra, Angel X. Chang, Sonia Chernova, Andrew J. Davison, Jia Deng, Vladlen Koltun, Sergey Levine, Jitendra Malik, Igor Mordatch, Roozbeh Mottaghi, Manolis Savva, and Hao Su. Rearrangement: A challenge for embodied ai. arXiv:2011.01975, 2020. 2

[4] Dong Cao and Lisha Xu. Cross-enhancement transform two-stream 3d convnets for pedestrian action recognition of autonomous vehicles. arXiv:1908.08916, 2019. 2, 3

[5] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. arXiv:1709.06158, 2017. 1

[6] Devendra Singh Chaplot, Kanthashree Mysore Sathyendra, Rama Kumar Pasumarthi, Dheeraj Rajagopal, and Ruslan Salakhutdinov. Gated-attention architectures for task-oriented language grounding. In *AAAI*, 2017. 1, 2

[7] Dian Chen, Brady Zhou, Vladlen Koltun, and Philipp Krähenbühl. Learning by cheating. In *CoRL*, 2020. 2

[8] Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *CVPR*, 2019. 1, 2

[9] Kan Chen, Rama Kovvuri, and Ram Nevatia. Query-guided regression network with context policy for phrase grounding. In *ICCV*, 2017. 2

[10] Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew J. Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. Textworld: A learning environment for text-based games. In *CGW@IJCAI*, 2018. 2

[11] Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied Question Answering. In *CVPR*, 2018. 1

[12] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *CVPR*, 2016. 2

[13] Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. Speaker-follower models for vision-and-language navigation. In *NeurIPS*, 2018. 2

[14] Melvyn A Goodale and A David Milner. Separate visual pathways for perception and action. In *Trends Neurosci.*, 1992. 1

[15] Daniel Gordon, Aniruddha Kembhavi, Mohammad Rastegari, Joseph Redmon, Dieter Fox, and Ali Farhadi. Iqa: Visual question answering in interactive environments. In *CVPR*, 2018. 1

[16] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 2

[17] Ronghang Hu, Marcus Rohrbach, Jacob Andreas, Trevor Darrell, and Kate Saenko. Modeling relationships in referential expressions with compositional modular networks. In *CVPR*, 2017. 2, 4

[18] Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell. Natural language object retrieval. In *CVPR*, 2016. 2, 4

[19] Unnat Jain, Iou-Jen Liu, Svetlana Lazebnik, Aniruddha Kembhavi, Luca Weihs, and Alexander Schwing. Gridtopix: Training embodied agents with minimal supervision. arXiv:2105.00931, 2021. 2

[20] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc Gool. Dynamic filter networks. In *NeurIPS*, 2016. 2

[21] Liyiming Ke, Xiujun Li, Yonatan Bisk, Ari Holtzman, Zhe Gan, Jingjing Liu, Jianfeng Gao, Yejin Choi, and Siddhartha Srinivasa. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In *CVPR*, 2019. 2

[22] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. AI2-THOR: An Interactive 3D Environment for Visual AI. arXiv:1712.05474, 2017. 1, 5

[23] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. arXiv:2004.02857, 2020. 1, 2

[24] Federico Landi, Lorenzo Baraldi, Massimiliano Corsini, and Rita Cucchiara. Embodied vision-and-language navigation with dynamic convolutional filters. In *BMVC*, 2019. 2, 3

[25] Xiujun Li, Chunyuan Li, Qiaolin Xia, Yonatan Bisk, Asli Çelikyilmaz, Jianfeng Gao, Noah A. Smith, and Yejin Choi. Robust navigation with language pretraining and stochastic sampling. In *EMNLP/IJCNLP*, 2019. 2

[26] Sheng Liu, Zhou Ren, and Junsong Yuan. Sibnet: Sibling convolutional encoder for video captioning. In *ACM MM*, 2018. 3

[27] Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. Self-monitoring navigation agent via auxiliary progress estimation. In *ICLR*, 2019. 2

[28] Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. The regretful agent: Heuristic-aided navigation through progress estimation. In *CVPR*, 2019. 2

[29] Matt MacMahon, Brian Stankiewicz, and Benjamin Kuipers. Walk the talk: Connecting language, knowledge, and action in route instructions. In *AAAI*, 2006. 2

[30] Dipendra Misra, John Langford, and Yoav Artzi. Mapping instructions and visual observations to actions with reinforcement learning. In *EMNLP*, 2017. 1

[31] Van-Quang Nguyen and Takayuki Okatani. A hierarchical attention model for action learning from realistic environments and directives. ECCV EVAL Workshop, https://askforalfred.com/EVAL/, 2020. 2, 5, 6

[32] Homagni Saha, Fateme Fotouhif, Qisai Liu, and Soumik Sarkar. A modular vision language navigation and manipulation framework for long horizon compositional tasks in indoor environment. arXiv:2101.07891, 2021. 2

[33] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *ICCV*, 2019. 1

[34] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *CVPR*, 2020. 1, 2, 3, 4, 5, 6, 7, 8

[35] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. arXiv:2010.03768, 2020. 2

[36] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NeurIPS*, 2014. 2, 3

[37] Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *NAACL*, 2019. 2

[38] Matthew Tesfaldet, Marcus A Brubaker, and Konstantinos G Derpanis. Two-stream convolutional networks for dynamic texture synthesis. In *CVPR*, 2018. 2

[39] Liwei Wang, Yin Li, and Svetlana Lazebnik. Learning deep structure-preserving image-text embeddings. In *CVPR*, 2016. 2

[40] Xin Wang, Wenhan Xiong, Hongmin Wang, and William Yang Wang. Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation. In *ECCV*, 2018. 2

[41] Yuke Zhu, Daniel Gordon, Eric Kolve, Dieter Fox, Li Fei-Fei, Abhinav Gupta, Roozbeh Mottaghi, and Ali Farhadi. Visual semantic planning using deep successor representations. In *ICCV*, 2017. 1

[42] Charles Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014. 2