Inner Monologue: Embodied Reasoning through Planning with Language Models

Anonymous Author(s)
Affiliation
Address
email

Abstract: Recent works have shown how the reasoning capabilities of Large Language Models (LLMs) can be applied to domains beyond natural language processing, such as planning and interaction for robots. These embodied problems require an agent to understand many semantic aspects of the world: the repertoire of skills available, how these skills influence the world, and how changes to the world map back to the language. LLMs planning in embodied environments need to consider not just what skills to do, but also how and when to do them - answers that change over time in response to the agent's own choices. In this work, we investigate to what extent LLMs used in such embodied contexts can reason over sources of feedback provided through natural language, without any additional training. We propose that by leveraging environment feedback, LLMs are able to form an *inner monologue* that allows them to more richly process and plan in robotic control scenarios. We investigate a variety of sources of feedback, such as success detection, scene description, and human interaction. We find that closed-loop language feedback significantly improves high level instruction completion on three domains, including simulated and real table top rearrangement tasks and long-horizon mobile manipulation tasks in a kitchen environment in the real world. Project website: https://inner-monologue.github.io

Keywords: Large Language Models, Grounding Models, Multi-modal Control, Task and Motion Planning, Mobile Manipulation

1 Introduction

Intelligent and flexible embodied interaction requires robots to be able to deploy large repertoires of basic behaviors in appropriate ways, sequence these behaviors as needed for long horizon tasks, and also recognize when to switch to a different approach if a particular behavior or plan is unsuccessful. High-level planning, perceptual feedback, and low-level control are just a few of the sub-tasks that would need to be seamlessly combined together to perform the sort of reasoning required for an embodied agent, such as a robot, to intelligently act in the world. While conventionally these challenges have been approached from the perspective of planning (e.g., TAMP [1]) or hierarchical learning (e.g., HRL [2]), effective high-level reasoning about complex tasks also requires knowledge and semantics. Can we obtain this knowledge and semantic reasoning capability from learned models? One of the remarkable observations in recent machine learning research is that, when appropriately conditioned (e.g., prompted), large language models can not only generate fluent textual descriptions, but can carry out some degree of deduction and respond to questions that appear to require reasoning and inference [3, 4, 5]. This raises an intriguing possibility: beyond their ability to interpret natural language instructions, can language models further serve as reasoning models that combine multiple sources of feedback and become interactive problem solvers for embodied tasks, such as robotic manipulation?

Imagine the "inner monologue" that happens when a person tries to solve some task: "I have to unlock the door; let me try to pick up the key and put it in the lock... no, wait, it doesn't fit, I'll try another one... that one worked, now I can turn the key." The "thought process" in this case involves choices about the best immediate action to solve the high-level task ("pick up the key"), observations about the outcomes of attempted actions ("it doesn't fit"), and corrective actions that are taken in response to these observations ("I'll try another one"). Language models can be quite proficient at generating such logically consistent "inner monologues," but require additional mechanisms to actually ground the choices of actions and the observations in the real world, so that the available primitives ("pick up the key") reflect the present capabilities of the robot, and the observations ("it doesn't fit") reflect the actual outcomes of attempting to execute those primitives.

Our work is concerned with devising a method that provides this kind of grounding, so as to harness the capability of language models to generate logically consistent inner monologues for interactively solving robotic manipulation tasks, by only few-shot prompting and without additional training. To accomplish this, we utilize a

Submitted to the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.

number of visual-language and perception models as well as human feedback as ways to provide grounding to a language model that decides on the most appropriate skills to carry out some user instruction. The robotic primitives are provided by a repertoire of skills that a robot has acquired previously (e.g., via reinforcement or imitation learning) that can be invoked by a language model. In addition we observe that, similarly to recent work [6], natural language provides a universal and interpretable interface for such grounding of model communication and allows them to incorporate their conclusions in an overarching inner-monologue language model. While prior work has investigated using language models as planners [7, 8] or incorporating multimodal-informed perception through language [6], to the best of our knowledge no work has studied the critical link of not only planning with language, but also informing *embodied feedback with language*, which we investigate in this work.

We study methods and sources of feedback for closing the agent-environment loop via inner monologue and their impact on downstream execution success and new capabilities arising from such interaction. In particular, we combine multiple perception models that perform various tasks such as language-conditioned classification or language-based scene description, together with feedback provided by a human user that the robot is cooperating with. To execute the commands given by a user, we utilize a set of pre-trained robotic manipulation skills together with their textual descriptions that can be invoked by a language model. All of these components (perception models, robotic skills, and human feedback) interact with each other via natural language in a form of an inner robot monologue to successfully perform the user instruction.

Finally, we show that such an inner monologue can accomplish complex, long-horizon tasks in simulation as well as on two real-world robotic platforms.

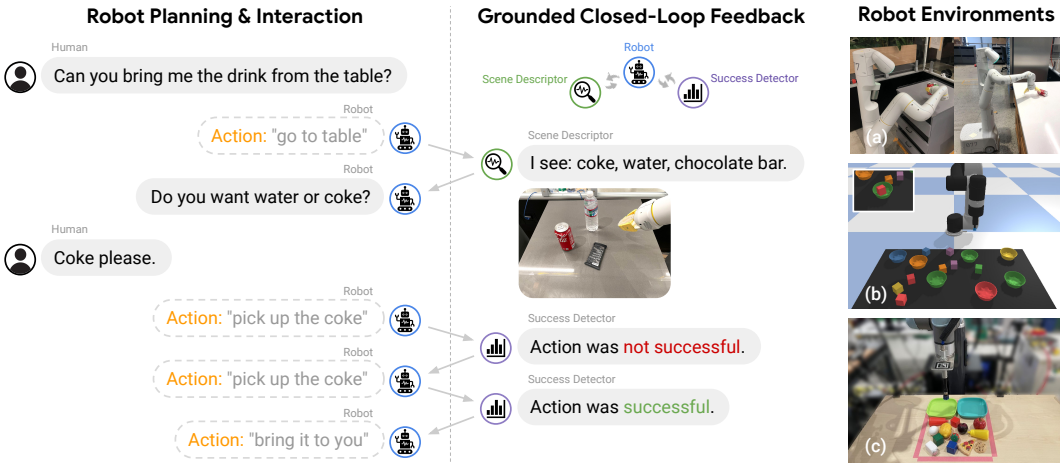

Figure. 1: Inner Monologue enables grounded closed-loop feedback for robot planning with large language models by leveraging a collection of perception models (e.g., scene descriptors and success detectors) in tandem with pretrained language-conditioned robot skills. Experiments show our system can reason and replan to accomplihs complex long-horizon tasks for (a) mobile manipulation and (b,c) tabletop manipulation in both simulated and real settings.

2 Related Work

Task and Motion Planning. Task and motion planning [9, 10] requires simultaneously solving a high-level, discrete task planning problem [11, 12, 13], and a low-level, continuous motion planning problem [14]. Traditionally, this problem has been solved through optimization [15, 16] or symbolic reasoning [11, 13], but more recently machine learning has been applied to aspects of the problem via learned representations, learned task-primitives, and more [17, 18, 19, 20, 21, 22, 23, 24, 25]. Some works utilize language for planning and grounding [26, 27, 28, 29, 30, 31]. Others have approached the problem through hierarchical learning [32, 33, 21, 34, 35, 36, 37]. In this work, we leverage pre-trained LLMs and their semantic knowledge, along with trained low-level skills, to find feasible plans.

Task Planning with Large Language Models (LLMs). Recent task planning approaches leverage LLMs for long-horizon tasks by conditioning on high level instructions to generate a sequence of intermediate, shorter horizon steps which are executable by an lower level agent. Huang et al. [7] prompts GPT-3 [4] and Codex [38] with the most similar example containing a high level instruction and its intermediate actionable steps to propose an action plan for a new desired high level task. Each step is mapped to the closest step executable by an agent as ranked by a similarity score computed by the RoBERTa model [39]. SayCan [8] instead grounds the proposed step from an LLM by multiplying the step's LLM probability with the value

function for such step, which serves as a proxy for affordance [21]. However, both approaches effectively produce the plan while assuming that each proposed step is executed successfully by the agent. As a result, these approaches may not be robust in handling intermediate failures in dynamic environments or with poor lower level policies. We explore in Inner Monologue ways to incorporate additional information about the environment into the LLM as we produce each step in the plan.

Fusing Vision, Language, and Control in Robotics. Various works have investigated strategies for the challenging problem of fusing vision, language, and control [40, 41, 42, 43, 44, 45]. While pretrained LLMs typically train only on text data, pretrained visual-language models (e.g., CLIP [46]) are trained on joint image(s) and corresponding text captions via variants of masked language modeling (MLM) objective [47, 48, 49, 50], contrastive loss [51, 52, 46] or other supervised objectives[53, 54]. CLIP has been employed in several robotics and embodied settings in zero-shot manner [55], or combined with Transporter networks [56] as in CLIPort [57]. Socratic Models [6] combines several foundation models (e.g., GPT-3 [4], ViLD [58]) and language-conditioned policies, using language as the common interface, and demonstrated manipulating objects in a simulated vision-based robotic manipulation environment.

3 Leveraging Embodied Language Feedback with Inner Monologue

In this section, we introduce how LLMs can act as interactive problem solvers and incorporate embodied environment observations into grounded planning through a process we refer to as Inner Monologue.

3.1 Problem Statement

Our setting consists of an embodied robotic agent attempting to perform a high-level natural language instruction i. This robotic agent is only capable of executing short-horizon skills from a library of previously trained policies $\pi_k \in \Pi$ with short language descriptions ℓ_k, which may be trained with reinforcement learning or behavioral cloning. The "planner," which is a pretrained LLM [7, 8], attempts to find a sequence of skills to accomplish the instruction. To observe the environment, the planner has access to textual feedback o from the environment that can be appended to the instruction or requested by the planner. The observation o may be success detection, object detection, scene description, visual-question answering, or even human feedback. Our work studies to what extent the LLM planner is able to reason over and utilize such feedback to "close the loop" with the environment and improve planning.

3.2 Inner Monologue

We formulate an "inner monologue" by continually substituting information from the various sources of feedback into the LLM planning language prompts as the robot interacts with the environment. While LLMs have demonstrated exceptional planning capabilities for embodied control tasks, prior works have found it crucial to ground LLM predictions with external components such as affordance functions [8] in order to produce useful plans that are executable by robots. However, LLMs used in this context have thus far remained one-directional – providing a list of skills, without making corrections or leveraging opportunities to re-plan accordingly. In contrast, Inner Monologue studies settings where grounded environment feedback is provided directly to the LLM in a closed-loop fashion. This promotes improved LLM reasoning in complex long-horizon settings, even before any external affordance-based grounding methods are applied.

Our analysis assumes textual feedback is provided to the planner, but does not assume a single specific method of fusing LLM planning with low-level robotic control or a specific method of extracting environment feedback into language. Rather than focusing on a particular algorithmic implementation, our aim is to provide a case study on the value of incorporating different types of feedback into closed-loop LLM-based planning. Thus, Inner Monologue in Section 4 utilizes language feedback within separate systems that incorporate different LLMs, different methods of fusing planning with control, different environments and tasks, and different methods of acquiring control policies. We note that in our specific implementations of Inner Monologue, we use pre-trained LLMs for planning that are not finetuned, but rather evaluated solely with few-shot prompting; the full prompts can be found in the Appendix.

3.3 Sources of Feedback

In theory any type of environment feedback can inform the LLM planner, as long as it can be expressed through language. We focus on the specific forms of feedback shown in Fig. 2, which can be broken down into task-specific feedback, such as success detection, and scene-specific feedback (either "Passive" or

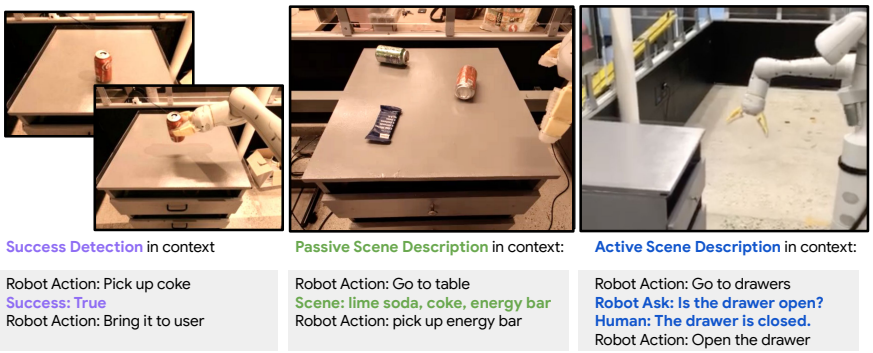

Figure. 2: Various types of textual feedback. **Success Detection** gives task-specific task completion information, **Passive Scene Description** gives structured semantic scene information at every planning step, and **Active Scene Description** gives unstructured semantic information only when queried by the LLM planner.

"Active"), which describes the scene. Specific instantiations and implementation details of each type of feedback can be found in Sec 4.1, Sec 4.2, and Sec 4.3 respectively for each domain.

Success Detection. Semantic success detection is a binary classification problem of whether the low-level skill π_k succeeded. Engineered success detectors can operate on ground-truth state in simulation, while learned success detectors can be trained on real examples of successes and failures in the real world [59, 60, 61, 62, 63]. We use the output of Success Detectors in language form, which we refer to as *Success* feedback.

Passive Scene Description. While there are many ways to describe the semantics contained within a scene, we use the term Passive Scene Description to broadly describe sources of scene feedback that are consistently provided and follow some structure. Passive Scene Description covers all sources of environment grounding feedback that are automatically provided and injected into the LLM prompt without any active prompting or querying by the LLM planner, such as Object Recognition [64, 65, 66, 67] or Scene Description related to the high-level task. We refer to the language outputs of such Object Recognizers as *Object* feedback and the outputs of Task-Progress Scene Descriptors as *Scene* feedback.

Active Scene Description. As the proactive counterpart to Passive Scene Description, Active Scene Description encompasses sources of feedback that are provided directly in response to active queries by the LLM planner. In this case, the LLM can directly ask a question about the scene, and this question can be answered either by a person, or by another pretrained model, such as a Visual Question Answering (VQA) model [68, 69, 70, 71]. While the previous types of feedback are strictly structured and narrow in their scope, in the Active Scene Description setting the LLM can receive unstructured answers to open ended questions about the scene, allowing it to actively gather information relevant to the scene and task. The combined output we send to the LLM planner includes both the LLM-generated question along with the human-provided VQA response; we refer to this type of feedback as *Human* feedback.

4 Experimental Results

In order to study how different sources of environment feedback can support a rich inner monologue that enables complex robotic control, we analyze diverse long-horizon manipulation and navigation tasks in simulation and in the real world. As Inner Monologue is not dependent on a specific LLM or a type of grounding feedback, we study different Inner Monologue implementations in three environments with different LLM planning methods and different sources of feedback from the environment. Below, we show results for a tabletop manipulation environment in sim (Section 4.1) and real (Section 4.2) as well as a mobile manipulation environment in real (Section 4.3). For more details about the experiment setup and results, please refer to the Appendix.

4.1 Simulated Tabletop Rearrangement

We experiment with vision-based block manipulation tasks in a Ravens-based [56] simulation environment to evaluate our method against several baselines and ablate across varying amounts of textual feedback. Given a number of blocks and bowls on a table, a robotic arm containing a gripper is tasked with rearranging these objects in some desired configuration, specified by natural language (e.g. "Putting the blocks in the bowls with matching colors"). We evaluate each method on four seen tasks and four unseen tasks, where seen tasks may be used for training (in the case of supervised baseline) or used as few-shot prompting for LLM planner.

This instantiation of Inner Monologue uses (i) InstructGPT [4, 72] as the LLM for multi-step planning [7, 8], (ii) scripted modules to provide language feedback in the form of Object Recognition (*Object*), Success Detection (*Success*), and Task-Progress Scene Description (*Scene*), and (iii) a pre-trained language-conditioned pick-and-place primitive (similar to CLIPort [57] and Transporter Nets [56]). *Object* feedback informs the LLM planner about the objects present in the scene, and the variant using only *Object* feedback is similar to the demonstrated example in [6] in this environment. *Success* feedback informs the planner about success/failure of the most recent action. However, in the presence of many objects and test-time disturbances, the complex combinatorial state space requires the planner to additionally reason about the overall task progress (e.g. if the goal is to stack multiple blocks, the unfinished tower of blocks may be knocked over by the robot). Therefore, Task-Progress Scene Description (*Scene*) describes the semantic sub-goals inferred by the LLM towards completing the high-level instruction that are achieved by the agent so far. For the variant that uses *Object* + *Scene* feedback, due to the additional reasoning complexity, we find that adding chain-of-thought [5, 73, 74] can improve the consistency between inferred goals and achieved goals. Example of the prompt can be found in Figure 3.

				+LLM	**+Inner Monologue**	
	Tasks	**CLIPort**	+oracle	*Object*	*Object + Success*	*Object + Scene*
Seen Tasks	"Pick and place"	24.0%	74.0%	80.0%	90.0%	**94.0%**
	"Stack all the blocks"	2.0%	32.0%	4.0%	10.0%	**26.0%**
	"Put all the blocks on the [x] corner/side"	2.0%	32.0%	30.0%	28.0%	**30.0%**
	"Put all the blocks in the [x] bowl"	32.0%	94.0%	52.0%	46.0%	**56.0%**
Unseen Tasks	"Put all the blocks in different corners"	0.0%	0.0%	20.0%	20.0%	**26.0%**
	"Put the blocks in their matching bowls"	0.0%	0.0%	56.0%	70.0%	**82.0%**
	"Put the blocks on mismatched bowls"	0.0%	0.0%	62.0%	76.0%	**86.0%**
	"Stack all the blocks on the [x] corner/side"	0.0%	0.0%	0.0%	4.0%	**6.0%**

Table. 1: Success rates for various methods, averaged across 50 episodes in Ravens-based environment with test-time disturbances. CLIPort + oracle indicates that CLIPort was provided a "termination" oracle (see Section 4.1). Although CLIPort can receive visual feedback from the environment, we show that LLM-informed feedback can effectively enable the planner to retry/replan in the presence of failures, while enjoying the generalization benefits of LLMs to unseen tasks.

We additionally compare to a multi-task CLIPort policy directly trained on long-horizon task instructions (i.e., without using LLM for planning). Because CLIPort is a single-step policy and does not terminate spontaneously during policy rollout, we report CLIPort evaluations with oracle termination (i.e., repeat until oracle indicates task completion) and fixed-step termination (i.e., repeat for k steps). While Inner Monologue terminates when the LLM stops generating new steps, we similarly set the maximum number of steps to be k for practical considerations. We use $k = 15$. To simulate real-world disturbances and evaluate the system's robustness to disturbances, we add Gaussian noise to multiple levels of the system at test time: $\mathcal{N}(0,3)$ for pixel observation, $\mathcal{N}(0,2.5)$ for policy primitive (i.e., pick-place pixel heatmaps), $\mathcal{N}(0,0.02m)$ for place locations.

Analysis. As shown in Table 1, all variants of Inner Monologue perform well on seen tasks, and by leveraging rich semantic knowledge in the pre-trained LLM, the performance can be directly translated to unseen tasks without further training. Furthermore, aligned with our hypothesis, Inner Monologue with *Object* + *Scene* performs the best because of its ability to keep track of all goal conditions and currently achieved goals. Finally, we observe that non-hierarchical and solitary systems such as CLIPort (i) struggle at generalizing to unseen long-horizon tasks under test-time disturbances, and (ii) on training tasks, an oracle is also often required to indicate task completion for good performance.

4.2 Real-World Tabletop Rearrangement

We evaluate Inner Monologue on a real-world robot platform designed to resemble the simulation experiments presented in Section 4.1 using motion primitives for tabletop pick and place. The setup consists of a UR5e robot arm equipped with a wrist-mounted Intel RealSense RGB-D camera overlooking a workspace of diverse objects – from toy blocks to food items to condiments (shown in Fig. 4). We use an instantiation of Inner Monologue using (i) InstructGPT [4, 72] as the LLM for multi-step planning, (ii) a pretrained open-vocabulary object recognition with MDETR [75] to generate a list of currently visible objects and list of previously visible objects that are no longer visible (*Object*), (iii) heuristics on the object bounding box predictions from MDETR for Success Detection (*Success*), and (iv) a zero-shot pick and place policy that uses an LLM to parse target objects from a language command (e.g., given by the planner) and then executes scripted suction-based picking and placing primitives at the center of the target objects' bounding boxes. Aside from the pretraining of the LLM and MDETR (which are available out-of-the-box), the system does not require any model finetuning to perform pick and place tasks with new objects.

We investigate two tasks: (i) a simple 3-block stacking task where 2 blocks are already pre-stacked, and (ii) a more complex long-horizon sorting task to place food in one plate and condiments in another (where categorizing food versus condiments is autonomously done by the LLM planner). Since default pick and place performance is generally quite high on the system, we artificially inject Gaussian noise into the policy actions (i.e., add standard deviation σ=4mm clipped at 2σ) to stress test recovery from failures via re-planning with grounded closed-loop feedback. Results are presented in Tab. 2. Note that the system is also subject to noisy object and success detections due to the additional challenge of real-world perception and clutter.

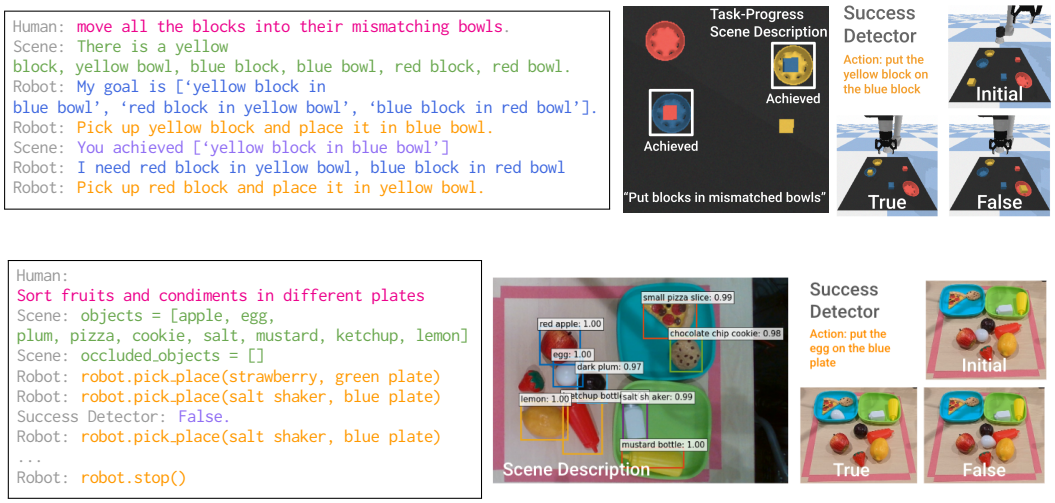

Figure. 3: Inner Monologue with a **scene descriptor** and a **success detector**, planning with InstructGPT to run language-conditioned **robot policies** in simulated and real environments with various objects on a table. It is able to respond to free-form human instructions (in **magenta**). (top: simulated tabletop rearrangement, bottom: real tabletop rearrangement)

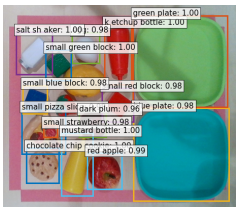

Figure. 4: Object detection on an example setting.

	LLM	+Inner Monologue		
Task Family	*Object*	*Object*	*Success*	*Object + Success*
Finish 3-block stacking	20%	40%	40%	**100%**
Sort fruits from bottles	20%	50%	40%	**80%**
Total	20%	45%	40%	**90%**

Table. 2: Inner Monologue (with grounded closed-loop feedback from object recognition and success detection) on a real pick and place robot exceeds the performance of baseline alternatives, as measured by average task success rates over 10 runs.

Analysis. We compare different variants of Inner Monologue with different LLM-informed closed-loop feedback, as well as an open-loop variant that only runs object recognition once at the beginning of the task (similar to the system demonstrated in [6]). The partial 3-block stacking task highlights an immediate failure mode of this baseline, where the initial Scene Description struggles to capture a complete representation of the scene (due to clutter and occlusion) to provide as input to the multi-step planner. As a result, the system only executes one pick and place action – and cannot recover from mistakes. To address these shortcomings, Inner Monologue (*Object + Success*) leverages closed-loop Scene Description and Success Detection after each step, which allows it to successfully replan and recover from policy mistakes.

Additional ablations with Inner Monologue also show that (i) common failures induced by lack of closed-loop Scene Description are largely due to initially occluded objects not being part of the LLM generated plans, and (ii) failures induced by lack of Success Detection come from not retrying pick and place actions that have failed due to policy noise. Overall, we observe that both components are complementary and important in maintaining robust recovery modes for real rearrangement tasks.

4.3 Real-World Mobile Manipulator in a Kitchen Setting

We implement Inner Monologue in a robotic system using the kitchen environment and task definitions described in SayCan [8]. A mobile high-degree-of-freedom manipulator robot with RGB observations is placed in an office kitchen to interact with several common objects using continuous closed-loop control.

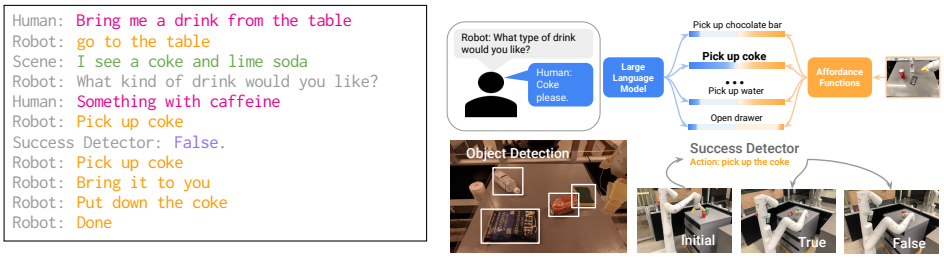

Figure. 5: Inner Monologue with a **scene descriptor** and a **success detector**, using SayCan [8] to combine a PALM [3]-based planner with learned affordance functions to run language-conditioned **robot policies** in a real world kitchen environment. It is able to respond to free-form human instructions and human-provided VQA or feedback(in **magenta**).

The baseline, SayCan [8], is a method that plans and acts in diverse real world scenarios by combining an LLM with value functions of underlying control policies. While SayCan creates plans that are grounded by the affordances of value functions, the LLM predictions in isolation are never given any closed-loop feedback.

We use an instantiation of Inner Monologue that uses (i) PALM [3] as the LLM for multi-step planning, (ii) value functions from pre-trained control policies for affordance grounding [8], (iii) a learned visual classification model for success detection feedback (*Success*), (iv) human-provided object recognition feedback (*Object*), and (v) pre-trained control policies for relevant skills in the scene. We also perform a case study where we allow the agent to ask questions and source *Human* feedback directly; results are shown in Fig. 7a and the Appendix.

We evaluate methods on 120 evaluations over three task families: 1) four manipulation tasks, 2) two dexterous manipulation tasks utilizing drawers, and 3) two long-horizon combined manipulation and navigation tasks.

In order to better study how Inner Monologue improves reasoning in especially challenging scenarios, we consider an experiment variant where we add adversarial disturbances during control policy executions that cause skill policy rollouts to fail. These disturbances may be fairly simple and just require the policy to try again, or it may complex enough that the policy needs to re-plan and select a completely new skill. While these failures occur naturally even without perturbances, the adversarial disturbances creates a consistent comparison between methods that requires retrying or re-planning to accomplish the original instruction.

		Inner Monologue	
Task Family	**SayCan**	*Success*	*Object + Success*
No Disturbances			
Manipulation	50.0%	62.5%	**75.0%**
Mobile Manipulation	50.0%	50.0%	**75.0%**
Drawers	83.3%	83.3%	**100.0%**
With Disturbances			
Manipulation	12.5%	25.0%	**33.3%**
Mobile Manipulation	0.0%	25.0%	**75.0%**
Drawers	0.0%	**44.4%**	**44.4%**
Total	30.8%	48.7%	**60.4%**

Table. 3: Success rate of Inner Monologue compared to an open-loop version on 120 evaluations across several family of tasks in our real world setting. We consider a standard setting and adversarial setting with external human disturbances (+ *Dist*).

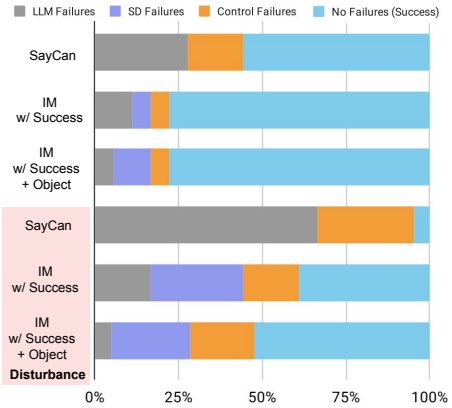

Figure. 6: Failure causes on 120 evaluations. When disturbances are added, only the Inner Monologue variants consistently complete the instructions.

Analysis. The results of real robot experiments are shown in Table 3. Without adversarial disturbances, the baseline method SayCan performs reasonably on all three task families, yet incorporating LLM-informed feedback on skill success/failure and presence of objects allows the Inner Monologue to effectively retry or re-plan under natural failures, providing further improvement to the baseline. The most notable difference is in the cases with adversarial disturbances when a policy failure is forced to occur. Without any LLM-informed feedback SayCan has success rate close to 0% since it is does not have explicitly high-level retry behavior. Inner Monologue significantly outperforms SayCan because of its ability to invoke appropriate recovery modes depending on the environment feedback. In-depth analysis on the failure causes indicates that success and object recognition feedback can effectively reduce LLM planning failures and thus overall failure rate, albeit at the cost of introducing new failure modes to the system.

4.4 Emergent Capabilities

We find that Inner Monologue demonstrates impressive reasoning and replanning behaviors beyond the prompts given, as well as a certain level of robustness to prompt structure change and typos. Although not explicitly prompted, the LLM planner can react to human interaction that changes the high-level goal mid-task. Fig. 7a demonstrates a challenging case, where *Human* feedback changes the goal mid-task, and then changes the goal yet again by saying "finish the previous task". We can see that the planner the feedback correctly by switching tasks twice. In Fig. 7b we see interesting emergent behavior as well: the robot's instructed task changes in the middle of the episode, but the planner recognizes the change and generates a new "Robot thought: Goal state is..." statement allowing it to solve the new task. This structure had not been seen in the example prompts, where human instructions are shown only at the beginning. We prompted the language model following certain conventions. For instance, in the Simulated tabletop domain, the convention is `[Robot action, Scene, and Robot thought]`. In practice we found that the LLM planner is robust to occasionally swapping the order of feedback. The LLM Planner is also robust to typos in the *Human* environment feedback, potentially inheriting this generalization capability from the LLM used [3, 4]. For further analysis on emergent behavior, please refer to the Appendix.

(a) Real Kitchen Environment **(b)** Simulated Tabletop Rearrangement

Figure. 7: Closed-loop environment feedback enables emergent re-planning behavior to solve changing tasks.

5 Limitations

Limiting assumptions and failure modes. In Section 4.1 and Section 4.3, we assume access to oracle scene descriptors in the form of human observers or scripted systems to provide textual description back to the LLM planner. As for failure modes, Inner Monologue may fail due to several sources of errors: (1) success detections, (2) LLM planning errors, and (3) control errors. False negative predictions from the success detector lead to additional retry attempts, while false positive predictions add adversarial partial observability to the environment. In some instances, we found that the LLM planners ignored the environment feedback and still proposed policy skills involving objects not present in the scene.

Limitations of results and future work. The performance of low-level control policies limits not only overall high-level instruction completion performance, but also limits the scope of tasks that the LLM is able to reason over: no matter how much the LLM reasoning improves, it can still be bottlenecked by what low-level control policies are able to achieve. Future work remains open on several fronts. First, with advances in image/video captioning and visual-question answering, a fully automated system of Inner Monologue can be implemented without a human in the loop as an oracle. Second, improvements can be made on how to aggregate potentially inaccurate sources of information, such as using text to describe the uncertainty of the feedback modules, or including additional feedback modules for safety and ethics for the proposed plans.

6 Conclusion

In this work, we investigated the role that environment feedback plays for LLMs reasoning in tasks involving embodied robotic planning and interaction. We presented a general formulation Inner Monologue that combines different sources of environment feedback with methods fusing LLM planning with robotic control policies. Then, we studied how these methods scaled to three different robotic manipulation settings in simulation and the real world. We found that environment feedback significantly improves high level instruction completion, especially in challenging scenarios with adversarial disturbances. Finally, we analyze emergent capabilities of Inner Monologue that highlight how closed-loop language feedback enables replanning even in complex unseen settings.

References

[1] L. P. Kaelbling and T. Lozano-Pérez. Integrated task and motion planning in belief space. *The International Journal of Robotics Research*, 32(9-10):1194–1227, 2013.

[2] A. G. Barto and S. Mahadevan. Recent advances in hierarchical reinforcement learning. *Discrete event dynamic systems*, 13(1):41–77, 2003.

[3] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

[4] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[5] J. Wei, X. Wang, D. Schuurmans, M. Bosma, E. Chi, Q. Le, and D. Zhou. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

[6] A. Zeng, A. Wong, S. Welker, K. Choromanski, F. Tombari, A. Purohit, M. Ryoo, V. Sindhwani, J. Lee, V. Vanhoucke, et al. Socratic models: Composing zero-shot multimodal reasoning with language. *arXiv preprint arXiv:2204.00598*, 2022.

[7] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*. PMLR, 2022.

[8] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[9] L. P. Kaelbling and T. Lozano-Pérez. Hierarchical planning in the now. In *Workshops at the Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[10] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel. Combined task and motion planning through an extensible planner-independent interface layer. In *2014 IEEE international conference on robotics and automation (ICRA)*, 2014.

[11] R. E. Fikes and N. J. Nilsson. Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, 1971.

[12] E. D. Sacerdoti. A structure for plans and behavior. Technical report, SRI International, Menlo Park California Artificial Intelligence Center, 1975.

[13] D. Nau, Y. Cao, A. Lotem, and H. Munoz-Avila. Shop: Simple hierarchical ordered planner. In *Proceedings of the 16th international joint conference on Artificial intelligence*, 1999.

[14] S. M. LaValle. *Planning algorithms*. Cambridge university press, 2006.

[15] M. Toussaint. Logic-geometric programming: An optimization-based approach to combined task and motion planning. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[16] M. A. Toussaint, K. R. Allen, K. A. Smith, and J. B. Tenenbaum. Differentiable physics and stable modes for tool-use and manipulation planning. *Robotics: Science and Systems Foundation*, 2018.

[17] B. Eysenbach, R. R. Salakhutdinov, and S. Levine. Search on the replay buffer: Bridging planning and reinforcement learning. *Advances in Neural Information Processing Systems*, 2019.

[18] D. Xu, S. Nair, Y. Zhu, J. Gao, A. Garg, L. Fei-Fei, and S. Savarese. Neural task programming: Learning to generalize across hierarchical tasks. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[19] D. Xu, R. Martín-Martín, D.-A. Huang, Y. Zhu, S. Savarese, and L. F. Fei-Fei. Regression planning networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[20] T. Silver, R. Chitnis, N. Kumar, W. McClinton, T. Lozano-Perez, L. P. Kaelbling, and J. Tenenbaum. Inventing relational state and action abstractions for effective and efficient bilevel planning. *arXiv preprint arXiv:2203.09634*, 2022.

[21] D. Shah, P. Xu, Y. Lu, T. Xiao, A. Toshev, S. Levine, and B. Ichter. Value function spaces: Skill-centric state abstractions for long-horizon reasoning. *ICLR*, 2022. URL https://openreview.net/pdf?id=vgqS1vkkCbE.

[22] A. Srinivas, A. Jabri, P. Abbeel, S. Levine, and C. Finn. Universal planning networks: Learning generalizable representations for visuomotor control. In *International Conference on Machine Learning*, pages 4732–4741. PMLR, 2018.

[23] T. Kurutach, A. Tamar, G. Yang, S. J. Russell, and P. Abbeel. Learning plannable representations with causal infogan. *Advances in Neural Information Processing Systems*, 31, 2018.

[24] A. Akakzia, C. Colas, P.-Y. Oudeyer, M. Chetouani, and O. Sigaud. Grounding language to autonomously-acquired skills via goal generation. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=chPj_I5KMHG.

[25] S. Pirk, K. Hausman, A. Toshev, and M. Khansari. Modeling long-horizon tasks as sequential interaction landscapes. *arXiv preprint arXiv:2006.04843*, 2020.

[26] T. Kollar, S. Tellex, D. Roy, and N. Roy. Toward understanding natural language directions. In *2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 259–266. IEEE, 2010.

[27] S. Tellex, T. Kollar, S. Dickerson, M. Walter, A. Banerjee, S. Teller, and N. Roy. Understanding natural language commands for robotic navigation and mobile manipulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 25, pages 1507–1514, 2011.

[28] M. Bollini, S. Tellex, T. Thompson, N. Roy, and D. Rus. Interpreting and executing recipes with a cooking robot. In *Experimental Robotics*, pages 481–495. Springer, 2013.

[29] S. Tellex, R. Knepper, A. Li, D. Rus, and N. Roy. Asking for help using inverse semantics. 2014.

[30] T. Kollar, S. Tellex, D. Roy, and N. Roy. Grounding verbs of motion in natural language commands to robots. In *Experimental robotics*, pages 31–47. Springer, 2014.

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

[32] S. Nair and C. Finn. Hierarchical foresight: Self-supervised learning of long-horizon tasks via visual subgoal generation. *ArXiv*, abs/1909.05829, 2020.

[33] F. Xia, C. Li, R. Martín-Martín, O. Litany, A. Toshev, and S. Savarese. Relmogen: Integrating motion generation in reinforcement learning for mobile manipulation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[34] C. Li, F. Xia, R. Martin-Martin, and S. Savarese. Hrl4in: Hierarchical reinforcement learning for interactive navigation with mobile manipulators. In *Conference on Robot Learning*, 2020.

[35] Y. Jiang, S. Gu, K. Murphy, and C. Finn. Language as an abstraction for hierarchical deep reinforcement learning. In *NeurIPS*, 2019.

[36] D. Hafner, K.-H. Lee, I. Fischer, and P. Abbeel. Deep hierarchical planning from pixels. *arXiv preprint arXiv:2206.04114*, 2022.

[37] S. Mirchandani, S. Karamcheti, and D. Sadigh. Ella: Exploration through learned language abstraction. *Advances in Neural Information Processing Systems*, 34:29529–29540, 2021.

[38] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. d. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[39] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[40] C. Paxton, Y. Bisk, J. Thomason, A. Byravan, and D. Foxl. Prospection: Interpretable plans from language by predicting the future. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6942–6948. IEEE, 2019.

[41] S. Stepputtis, J. Campbell, M. Phielipp, S. Lee, C. Baral, and H. Ben Amor. Language-conditioned imitation learning for robot manipulation tasks. *Advances in Neural Information Processing Systems*, 33:13139–13150, 2020.

[42] V. Blukis, R. A. Knepper, and Y. Artzi. Few-shot object grounding and mapping for natural language robot instruction following. *arXiv preprint arXiv:2011.07384*, 2020.

[43] C. Lynch and P. Sermanet. Language conditioned imitation learning over unstructured data. *Robotics: Science and Systems*, 2021. URL https://arxiv.org/abs/2005.07648.

[44] Y. Chen, R. Xu, Y. Lin, and P. A. Vela. A joint network for grasp detection conditioned on natural language commands. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4576–4582. IEEE, 2021.

[45] O. Mees, L. Hermann, and W. Burgard. What matters in language conditioned robotic imitation learning. *arXiv preprint arXiv:2204.06252*, 2022.

[46] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[47] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[48] J. Lu, D. Batra, D. Parikh, and S. Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32, 2019.

[49] Z. Wang, J. Yu, A. W. Yu, Z. Dai, Y. Tsvetkov, and Y. Cao. Simvlm: Simple visual language model pretraining with weak supervision. *arXiv preprint arXiv:2108.10904*, 2021.

[50] A. Suglia, Q. Gao, J. Thomason, G. Thattai, and G. Sukhatme. Embodied bert: A transformer model for embodied, language-guided visual task completion. *arXiv preprint arXiv:2108.04927*, 2021.

[51] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255, 2020.

[52] A. Jain, M. Guo, K. Srinivasan, T. Chen, S. Kudugunta, C. Jia, Y. Yang, and J. Baldridge. Mural: multimodal, multitask retrieval across languages. *arXiv preprint arXiv:2109.05125*, 2021.

[53] J. Sun, D.-A. Huang, B. Lu, Y.-H. Liu, B. Zhou, and A. Garg. Plate: Visually-grounded planning with transformers in procedural tasks. *IEEE Robotics and Automation Letters*, 7(2):4924–4930, 2022.

[54] F. Sener and A. Yao. Zero-shot anticipation for instructional activities. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 862–871, 2019.

[55] A. Khandelwal, L. Weihs, R. Mottaghi, and A. Kembhavi. Simple but effective: Clip embeddings for embodied ai. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14829–14838, 2022.

[56] A. Zeng, P. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, V. Sindhwani, and J. Lee. Transporter networks: Rearranging the visual world for robotic manipulation. *Conference on Robot Learning (CoRL)*, 2020.

[57] M. Shridhar, L. Manuelli, and D. Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, pages 894–906. PMLR, 2022.

[58] X. Gu, T.-Y. Lin, W. Kuo, and Y. Cui. Open-vocabulary object detection via vision and language knowledge distillation. *arXiv preprint arXiv:2104.13921*, 2021.

[59] I. Lenz, H. Lee, and A. Saxena. Deep learning for detecting robotic grasps. *The International Journal of Robotics Research*, 34(4-5):705–724, 2015.

[60] F.-J. Chu, R. Xu, and P. A. Vela. Real-world multiobject, multigrasp detection. *IEEE Robotics and Automation Letters*, 3(4):3355–3362, 2018.

[61] D. Kalashnikov, J. Varley, Y. Chebotar, B. Swanson, R. Jonschkowski, C. Finn, S. Levine, and K. Hausman. Mt-opt: Continuous multi-task robotic reinforcement learning at scale. *arXiv preprint arXiv:2104.08212*, 2021.

[62] T. Migimatsu and J. Bohg. Grounding predicates through actions. *arXiv preprint arXiv:2109.14718*, 2021.

[63] Y. Cui, S. Niekum, A. Gupta, V. Kumar, and A. Rajeswaran. Can foundation models perform zero-shot task specification for robot manipulation? In *Learning for Dynamics and Control Conference*, pages 893–905. PMLR, 2022.

[64] M. Liang and X. Hu. Recurrent convolutional neural network for object recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3367–3375, 2015.

[65] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[66] Z. Zou, Z. Shi, Y. Guo, and J. Ye. Object detection in 20 years: A survey. *arXiv preprint arXiv:1905.05055*, 2019.

[67] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020.

[68] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

[69] L. Zhou, H. Palangi, L. Zhang, H. Hu, J. Corso, and J. Gao. Unified vision-language pre-training for image captioning and vqa. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13041–13049, 2020.

[70] H. Cai, C. Gan, T. Wang, Z. Zhang, and S. Han. Once for all: Train one network and specialize it for efficient deployment. In *International Conference on Learning Representations*, 2020. URL https://arxiv.org/pdf/1908.09791.pdf.

[71] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, et al. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022.

[72] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022.

[73] A. K. Lampinen, I. Dasgupta, S. C. Chan, K. Matthewson, M. H. Tessler, A. Creswell, J. L. McClelland, J. X. Wang, and F. Hill. Can language models learn from explanations in context? *arXiv preprint arXiv:2204.02329*, 2022.

[74] M. Nye, A. J. Andreassen, G. Gur-Ari, H. Michalewski, J. Austin, D. Bieber, D. Dohan, A. Lewkowycz, M. Bosma, D. Luan, et al. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*, 2021.

[75] A. Kamath, M. Singh, Y. LeCun, G. Synnaeve, I. Misra, and N. Carion. Mdetr-modulated detection for end-to-end multi-modal understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1780–1790, 2021.