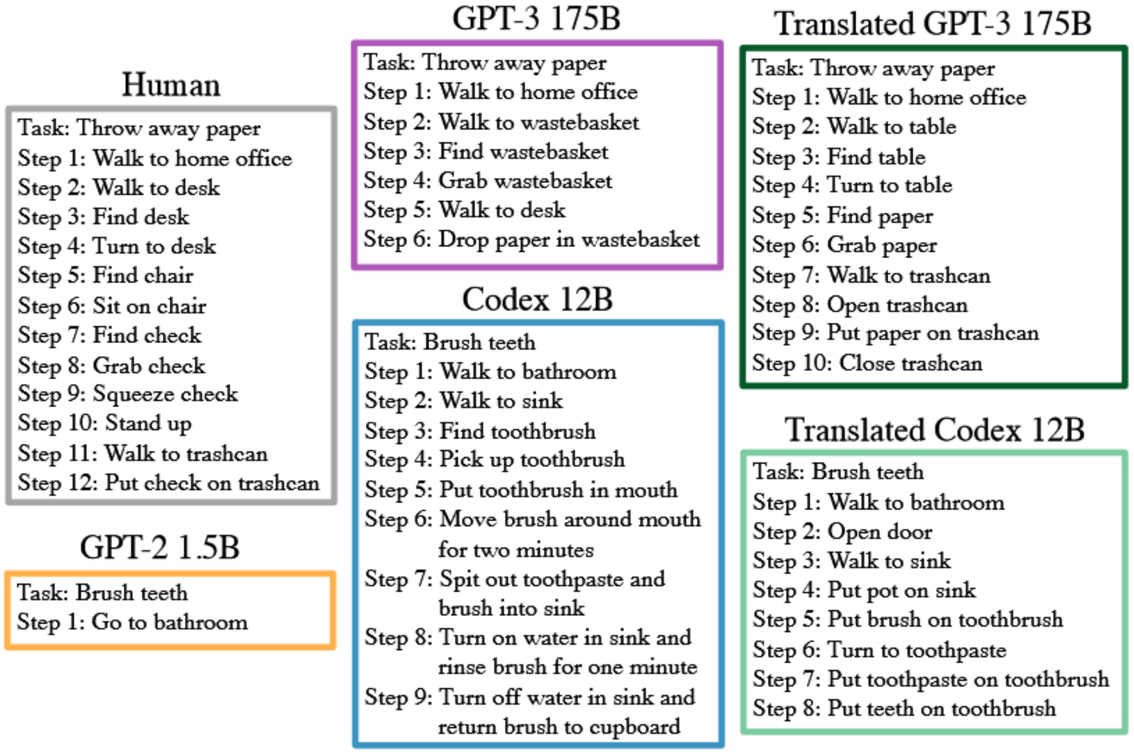
Language Models as Zero-Shot Planners: Extracting Actionable Knowledge for Embodied Agents

W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, "Language Models as Zero-Shot Planners: Extracting Actionable Knowledge for Embodied Agents." arXiv, Mar. 08, 2022. Accessed: Apr. 27, 2023. [Online]. Available: http://arxiv.org/abs/2201.07207

Backgrounds

- Trained on large corpora of human-produced language, the LLMs contain a lot of world knowledge.
- If prompted appropriately, the learned world knowledge is enough for LLMs to effectively decompose high-level tasks into mid-level plans without any further training.
- However, the produced plans often cannot map precisely to admissible actions, given an interactive, embodied environments.



Evaluated Environment: Virtual Home

- Simulator for activities in a household
- Pattern of Actions:
  - **[action] 〈arg〉 (idx)**
  - 42 atomic actions, such as "walk" and "open"
  - arg for specifying an interaction (objects or rooms)
  - idx to specifying the exact arg (multiple instances of the same object class)
- Tasks:
  - 292 distinct high-level tasks
  - 88 tasks for evaluation
  - 204 tasks as demonstration set

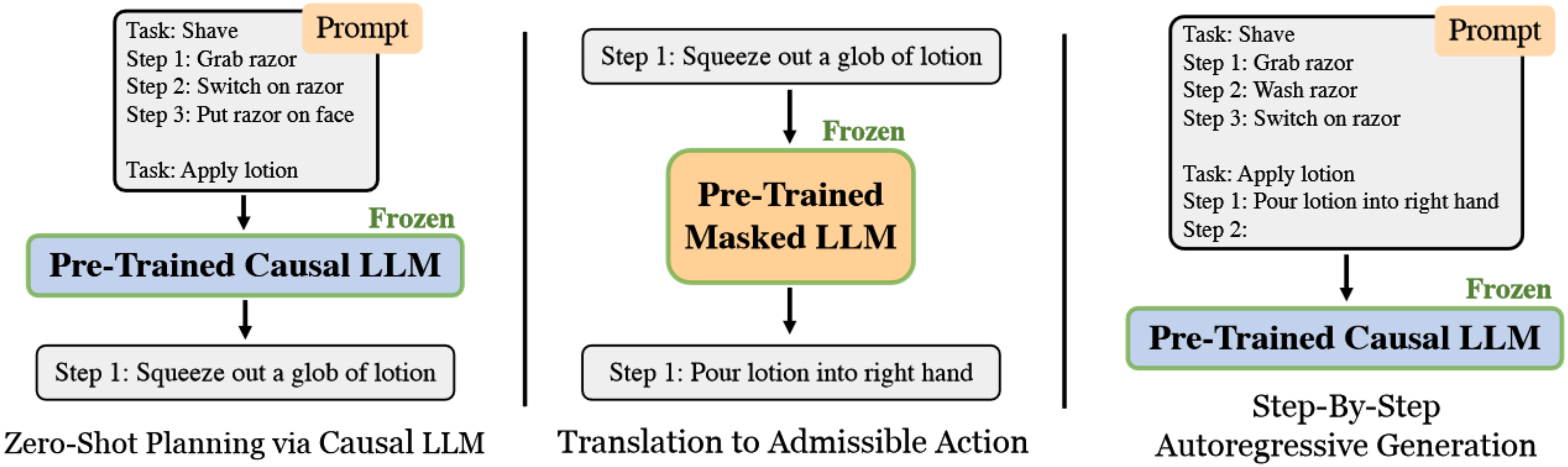

```
[WALK] ⟨living_room⟩(1)
[WALK] ⟨television⟩(1)
[FIND] ⟨television⟩(1)
[SWITCHON] ⟨television⟩(1)
[FIND] ⟨sofa⟩(1)
[SIT] ⟨sofa⟩(1)
[TURNTO] ⟨television⟩(1)
[WATCH] ⟨television⟩(1)
```

Methods

1. Prompt the LLM with a task example that is similar to the query task.

2. Map the model' s output phrases to the most semantically-similar admissible action (RoBERTa)

3. Replace the output of the model with the admissible action and generate the whole plan autoregressively.



Methods

Algorithm 1: Generating Action Plans from Pre-Trained Language Models

Notation Summary:
LM_P: text completion language model (also referred as **Planning LM**)
LM_T: text embedding language model (also referred as **Translation LM**)
$\{(T_i, E_i)\}_{i=1}^N$: demonstration set, where T is task name and E is example plan for T
C: cosine similarity function
P: mean token log probability under LM_P
Input: query task name Q, e.g. "make breakfast"
Output: action plan consisting of admissible env actions, e.g. "open fridge"

Extract most similar example (T^*, E^*) whose T^* maximizes $C(LM_T(T), LM_T(Q))$
Initialize prompt with $(T^* + E^* + Q)$
while max step is not reached **do**
  Sample LM_P with current prompt to obtain k single-step action phrases
  for each sample $\hat{a}$ **and** each admissible env action a_e **do**
    Calculate ranking score by $C(LM_T(\hat{a}), LM_T(a_e)) + \beta \cdot P(\hat{a})$
  end for
  Append highest-scoring env action a_e^* to prompt
  Append a_e^* to output
  if $> 50\%$ samples are 0-length **or** highest score $< \epsilon$ **then**
    break
  end if
end while

$$C(f(\hat{a}), f(a_e)) := \frac{f(\hat{a}) \cdot f(a_e)}{\|f(\hat{a})\| \|f(a_e)\|}$$

where f is an embedding function.

$$\operatorname*{argmax}_{a_e} \left[\max_{\hat{a}} C(f(\hat{a}), f(a_e)) + \beta \cdot P_\theta(\hat{a}) \right]$$

where β is a weighting coefficient.

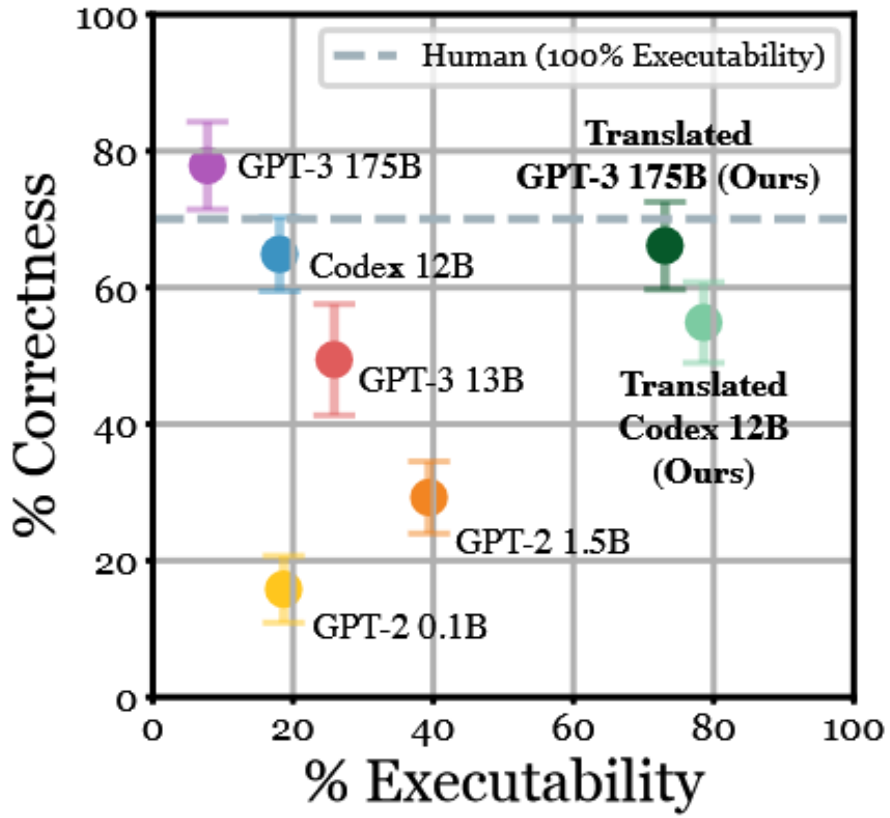
Experiments

Metrics

1. Executability: whether the action plan is valid for the environment.
2. Correctness: evaluation of 10 humans
3. LCS: the longest common subsequence between human annotations and LLM outputs

Language Model	Executability	LCS	Correctness
Vanilla GPT-2 117M	18.66%	3.19%	15.81% (4.90%)
Vanilla GPT-2 1.5B	39.40%	7.78%	29.25% (5.28%)
Vanilla Codex 2.5B	17.62%	15.57%	63.08% (7.12%)
Vanilla GPT-Neo 2.7B	29.92%	11.52%	65.29% (9.08%)
Vanilla Codex 12B	18.07%	16.97%	64.87% (5.41%)
Vanilla GPT-3 13B	25.87%	13.40%	49.44% (8.14%)
Vanilla GPT-3 175B	7.79%	17.82%	77.86% (6.42%)
Human	100.00%	N/A	70.05% (5.44%)
Fine-tuned GPT-3 13B	66.07%	34.08%	64.92% (5.96%)
OUR FINAL METHODS			
Translated Codex 12B	78.57%	24.72%	54.88% (5.90%)
Translated GPT-3 175B	73.05%	24.09%	66.13% (8.38%)

Table 1: Human-evaluated correctness and evaluation results in VirtualHome. Although action plans generated by large language models can match or even surpass human-written plans in correctness measure, they are rarely executable. By translating the naive action plans, we show an important step towards grounding LLMs in embodied environments, but we observe room to achieve this without trading executability for correctness. We also observe a failure mode among smaller models that lead to high executability. For correctness measure, standard error of the mean across 10 human annotators is reported in the parenthesis.



Experiments

Summary

1. Actions generated by vanilla LLMs are generally not very executable. While the proposed method improves the executability significantly.
2. For smaller vanilla LLMs
   a. Executability anomaly
      - Ignoring the queried task and repeating the prompts.

   b. Correctness anomaly
      - Generating shorter plans through ignoring common-sense actions
      - Task rephrasing
3. Source of Errors
   a. Translation LM fails to map compounded instructions to a succinct admissible action.
   b. Generated action plans stop too early.

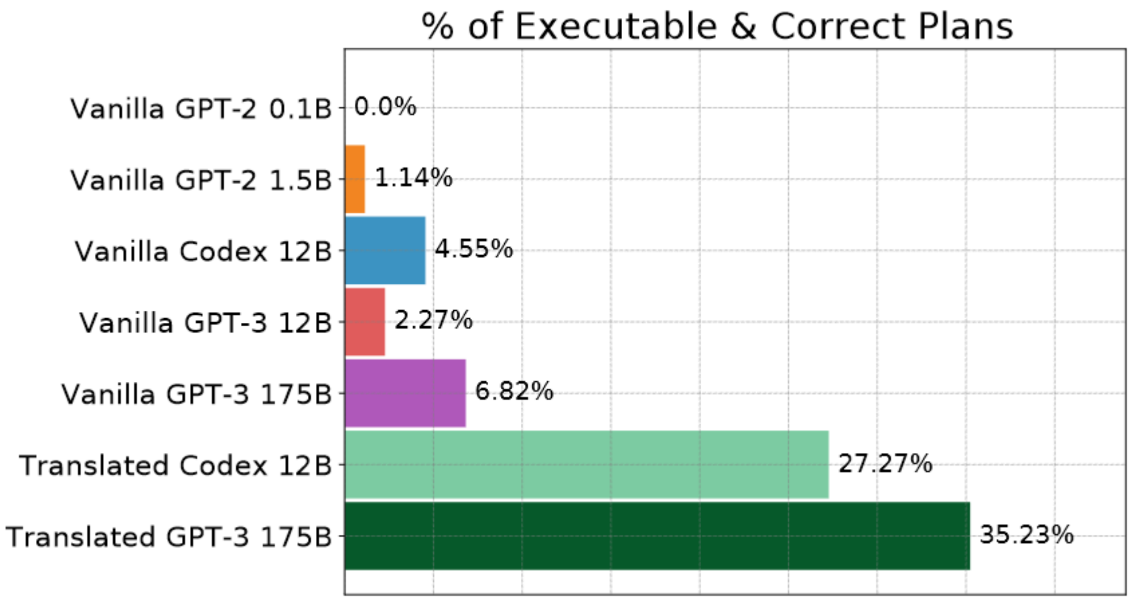
Language Model	Executability	LCS	Correctness
Vanilla GPT-2 117M	18.66%	3.19%	15.81% (4.90%)
Vanilla GPT-2 1.5B	39.40%	7.78%	29.25% (5.28%)
Vanilla Codex 2.5B	17.62%	15.57%	63.08% (7.12%)
Vanilla GPT-Neo 2.7B	29.92%	11.52%	65.29% (9.08%)
Vanilla Codex 12B	18.07%	16.97%	64.87% (5.41%)
Vanilla GPT-3 13B	25.87%	13.40%	49.44% (8.14%)
Vanilla GPT-3 175B	7.79%	17.82%	77.86% (6.42%)
Human	100.00%	N/A	70.05% (5.44%)
Fine-tuned GPT-3 13B	66.07%	34.08%	64.92% (5.96%)
OUR FINAL METHODS			
Translated Codex 12B	78.57%	24.72%	54.88% (5.90%)
Translated GPT-3 175B	73.05%	24.09%	66.13% (8.38%)

Table 1: Human-evaluated correctness and evaluation results in VirtualHome. Although action plans generated by large language models can match or even surpass human-written plans in correctness measure, they are rarely executable. By translating the naive action plans, we show an important step towards grounding LLMs in embodied environments, but we observe room to achieve this without trading executability for correctness. We also observe a failure mode among smaller models that lead to high executability. For correctness measure, standard error of the mean across 10 human annotators is reported in the parenthesis.

Ablation & Analysis

Methods	Executability	LCS
Translated Codex 12B	**78.57%**	**24.72%**
- **w/o** Action Translation	31.49%	22.53%
- **w/o** Dynamic Example	50.86%	22.84%
- **w/o** Trajectory Correction	55.19%	24.43%
Translated GPT-3 175B	**73.05%**	24.09%
- **w/o** Action Translation	36.04%	24.31%
- **w/o** Dynamic Example	60.82%	22.92%
- **w/o** Trajectory Correction	40.10%	**24.98%**

Table 2: Ablation of three proposed techniques.



reach human-level performance (65.91%)

Ablation & Analysis

Translation LM	Parameter Count	Executability	LCS
CODEX 12B AS PLANNING LM			
Avg. GloVe embeddings	-	46.92%	9.71%
Sentence Bert (base)	110M	73.21%	24.10%
Sentence Bert (large)	340M	75.16%	20.79%
Sentence RoBERTa (base)	125M	74.35%	22.82%
Sentence RoBERTa (large)	325M	**78.57%**	**24.72%**
GPT-3 175B AS PLANNING LM			
Avg. GloVe embeddings	-	47.40%	12.16%
Sentence Bert (base)	110M	**77.60%**	**24.49%**
Sentence Bert (large)	340M	67.86%	21.24%
Sentence RoBERTa (base)	125M	72.73%	23.64%
Sentence RoBERTa (large)	325M	73.05%	24.09%

Table 3: Effect of different Translation LMs on executability and LCS.

Discussion

- One possible way to finish a high-level task
    1. Dynamic Example (choose a similar task as the prompt example)
    2. Action Translation (map the ambiguous step-by-step action to a valid one)
    3. Autoregressive Trajectory Correction
- How to find a similar task?
    - Based on the similarity of two embedding vectors
    - ...
- Is there a better way for action translation?
    - Similarity of embeddings
    - ...
- Autoregressive action generation is slow.

Inner Monologue: Embodied Reasoning through Planning with Language Models

W. Huang *et al.*, "Inner Monologue: Embodied Reasoning through Planning with Language Models." arXiv, Jul. 12, 2022. Accessed: Feb. 14, 2023. [Online]. Available: http://arxiv.org/abs/2207.05608

Backgrounds

- Intelligent and flexible embodied interaction requires
  1. A large skill set
  2. Sequence skills needed for long horizon tasks
  3. Percept the environment and generate feedbacks (either after or in the execution of skills)

- Existing works involves
  - Using language models as planners
  - Incorporating multimodal-informed perception through language

- This work:
  - plan with language, execute with additional embodied feedbacks.

Methods

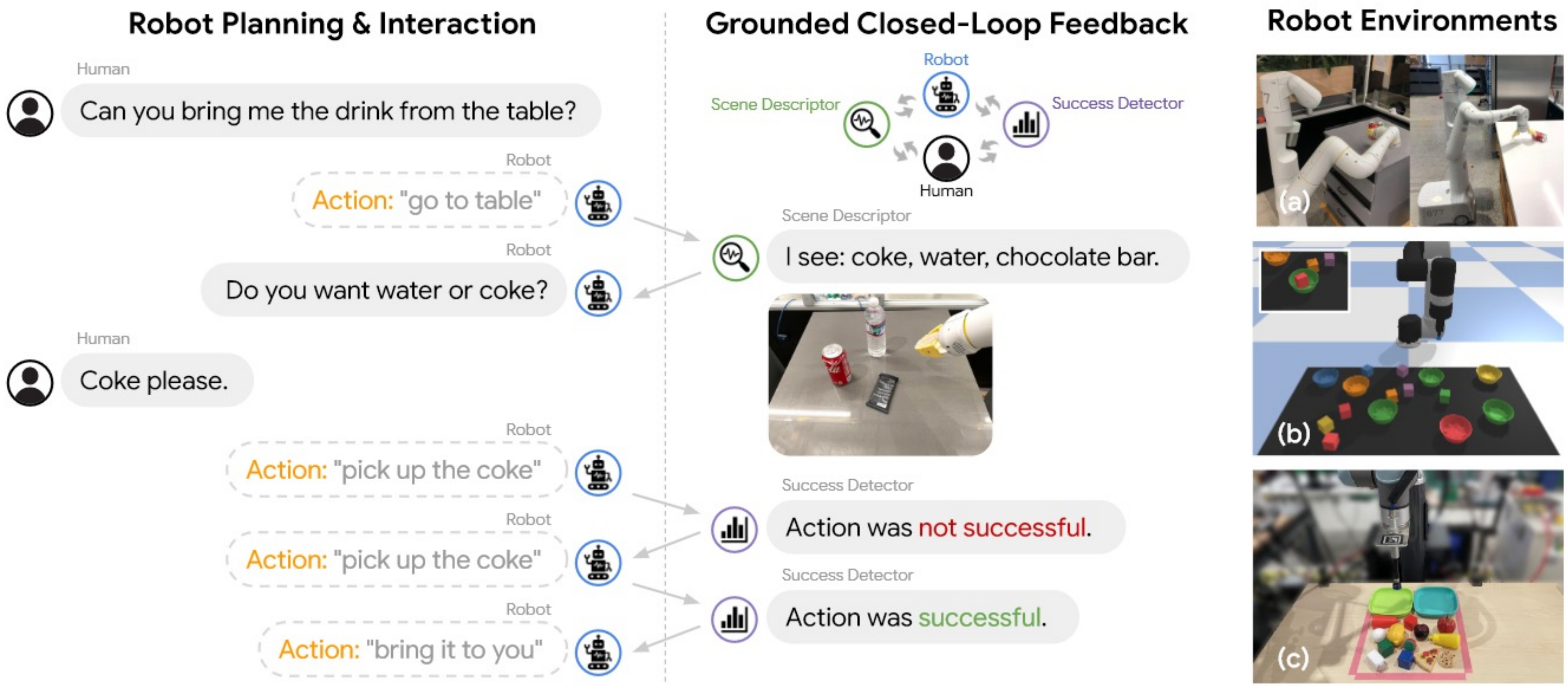
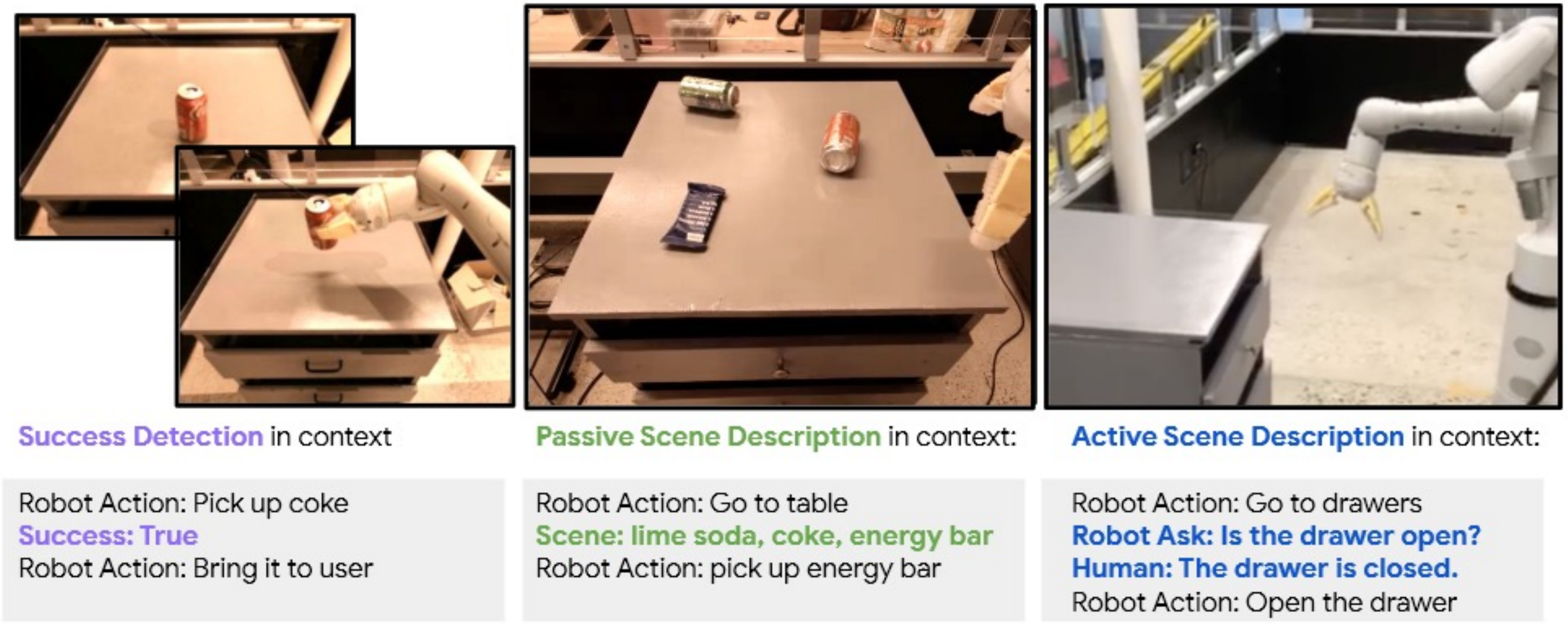


Figure 1: Inner Monologue enables grounded closed-loop feedback for robot planning with large language models by leveraging a collection of perception models (e.g., scene descriptors and success detectors) in tandem with pretrained language-conditioned robot skills. Experiments show our system can reason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

Methods

Source of Feedbacks

1. Success Detection
   - Whether the last skill execution was successful.
2. Passive Scene Description
   - Description of scene feedbacks that are consistently provided with some structure
3. Active Scene Description
   - Providing answers to the questions of the LLM



Simulated Tabletop Rearrangement

	Tasks	CLIPort	+oracle	+LLM	+Inner Monologue	
				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. 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.

CLIPort(baseline)

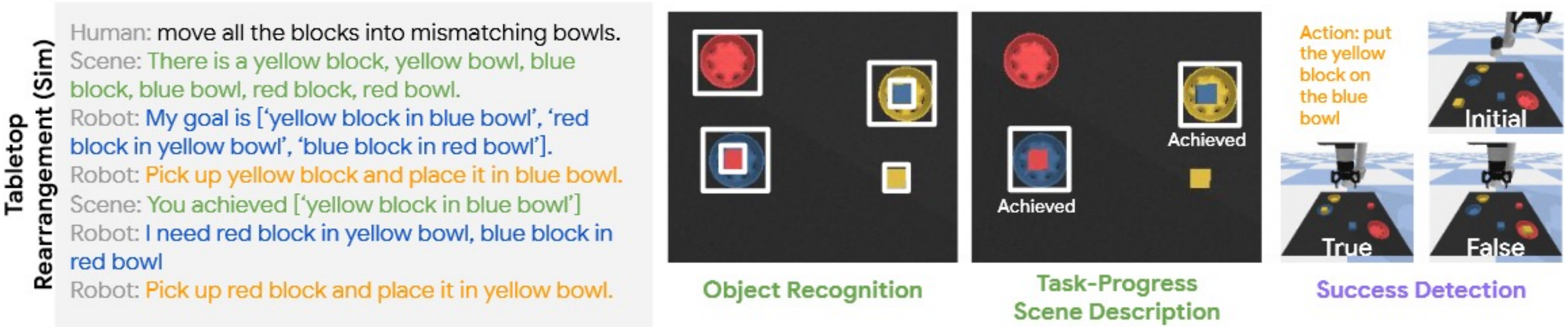
- A multi-task CLIPort policy trained on long-horizon task instructions

LLM Planner

- InstructGPT

Executor

- a pre-trained language-conditioned pick-and-place primitive



Real-World Tabletop Rearrangement

	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%**

LLM Planner

- InstructGPT

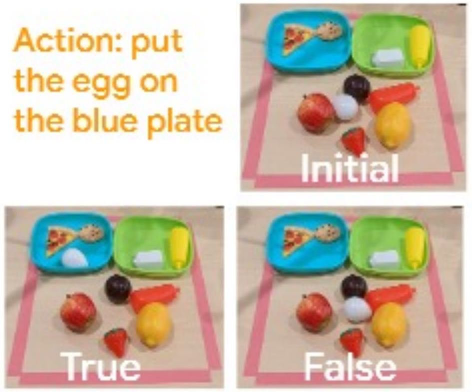
Executor

- MDETR for objection detection, LLM for parsing language command, a scripted suction-based picking and placing primitives for execution

Tabletop Rearrangement (Real)

Human: Sort fruits and condiments in different plates
Scene: objects = [apple, egg, plum, pizza, cookie, salt, mustard, ketchup, lemon]
Scene: occluded objects = []
Robot: robot.pickplace(strawberry, green plate)
Robot: robot.pickplace(salt shaker, blue plate)
Success Detector: False.
Robot: robot.pickplace(salt shaker, blue plate)
...
Robot: robot.stop()

Object Recognition (w/ Potential Occlusion)



Success Detection

Real-World Mobile Manipulator in a Kitchen Setting

Task Family	SayCan	+Inner Monologue *Success*	+Inner Monologue *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: Averaged success rate across 120 evaluations on several task families in our real-world mobile manipulation environment. We consider a standard setting and adversarial setting with external human disturbances. In all cases, LLM-informed embodied feedback is shown to be effective in improving robustness of the system, especially when low-level policies are prone to failures.

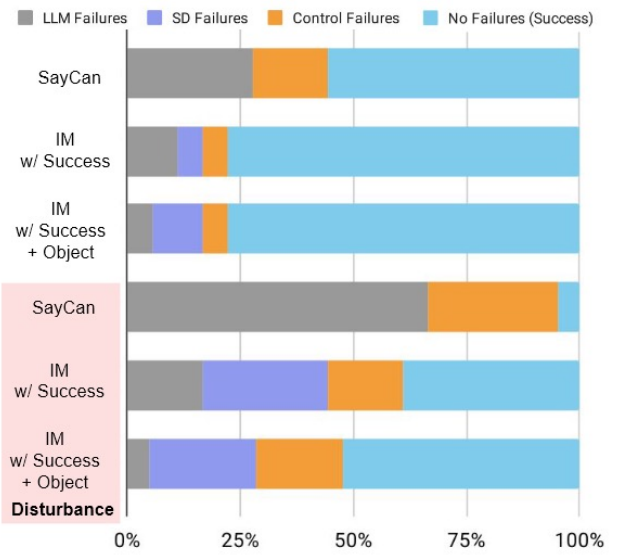


Figure 4: Failure causes on 120 evaluations. When disturbances are added (red), only the Inner Monologue variants consistently complete the instructions.

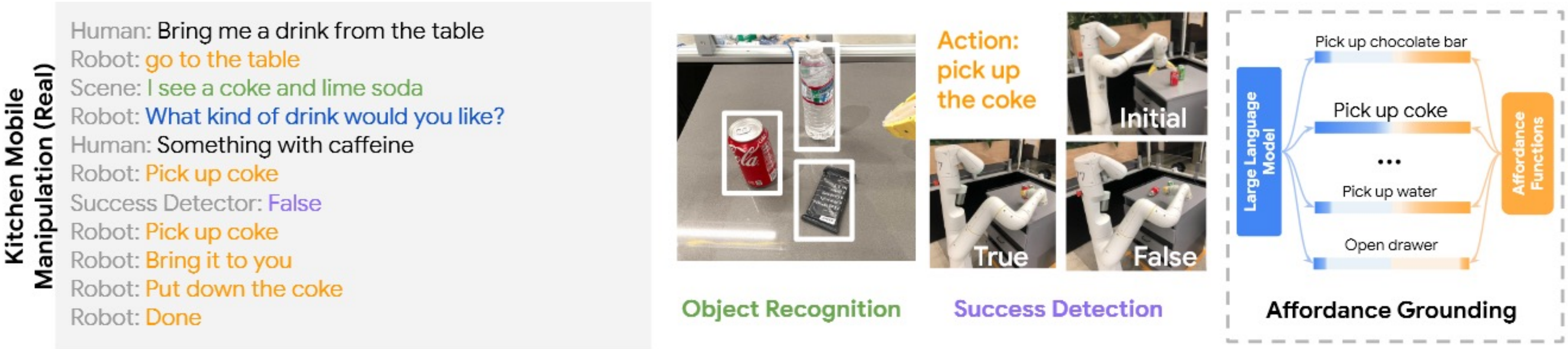
SayCan(baseline)

LLM Planner

- PALM

Executor

- pre-trained control policies for relevant skills in the scene



Pipelines for LLM & Robotic Control