Cook2LTL: Translating Cooking Recipes to LTL Formulae using Large Language Models

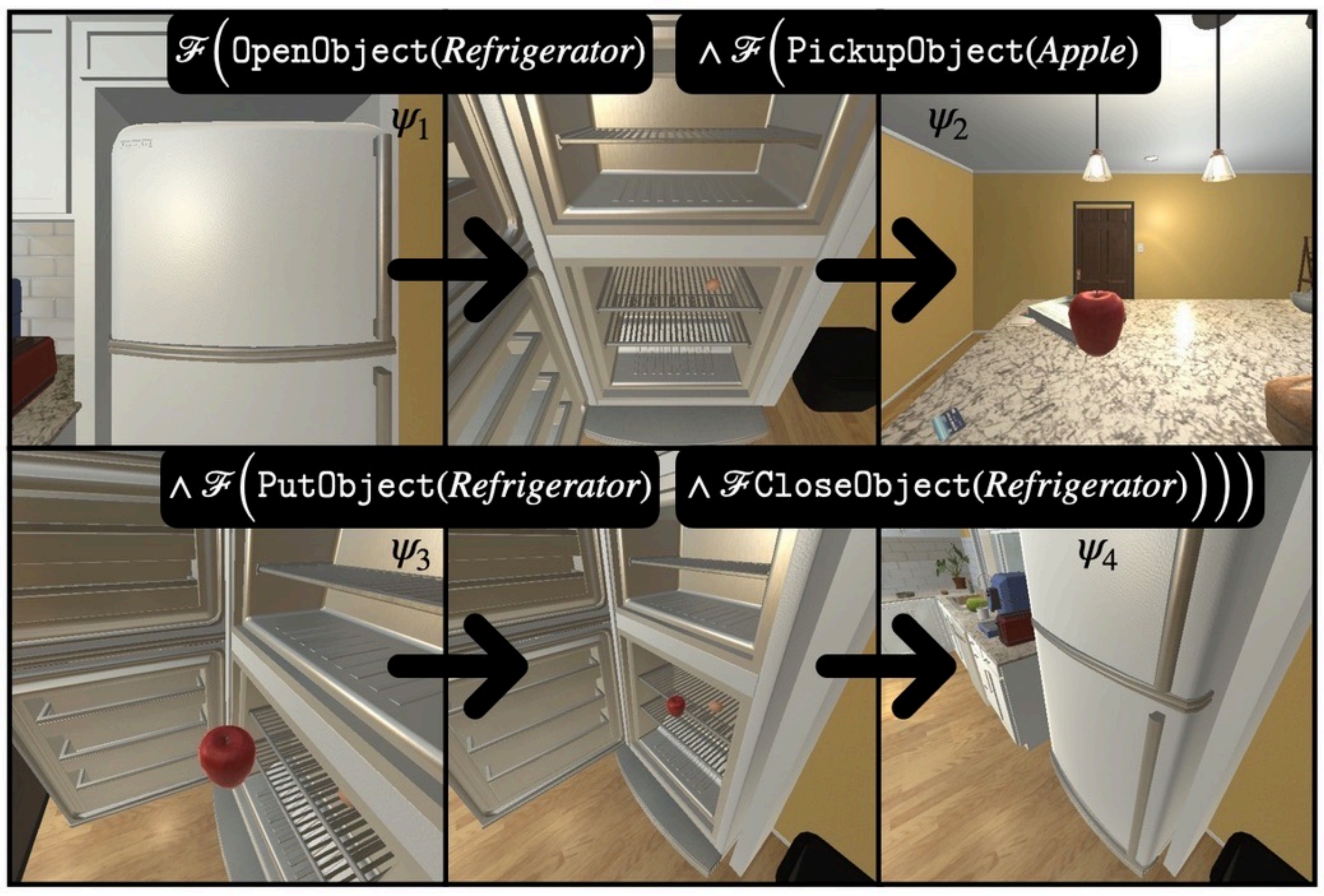
Angelos Mavrogiannis[1], Christoforos Mavrogiannis[2], Yiannis Aloimonos[1]

[1]Department of Computer Science, University of Maryland, College Park

[2]Department of Robotics, University of Michigan

ICRA2024 YOKOHAMA | JAPAN

$\phi = \mathcal{F}\texttt{Refrigerate}(Apple) = \mathcal{F}(\psi_1 \wedge \mathcal{F}(\psi_2 \wedge \mathcal{F}(\psi_3 \wedge \mathcal{F}\psi_4)))$



Goal: Given a cooking recipe in the form of natural language, extract unambiguous robot-executable plans with actions that are admissible in a kitchen environment.

Challenges

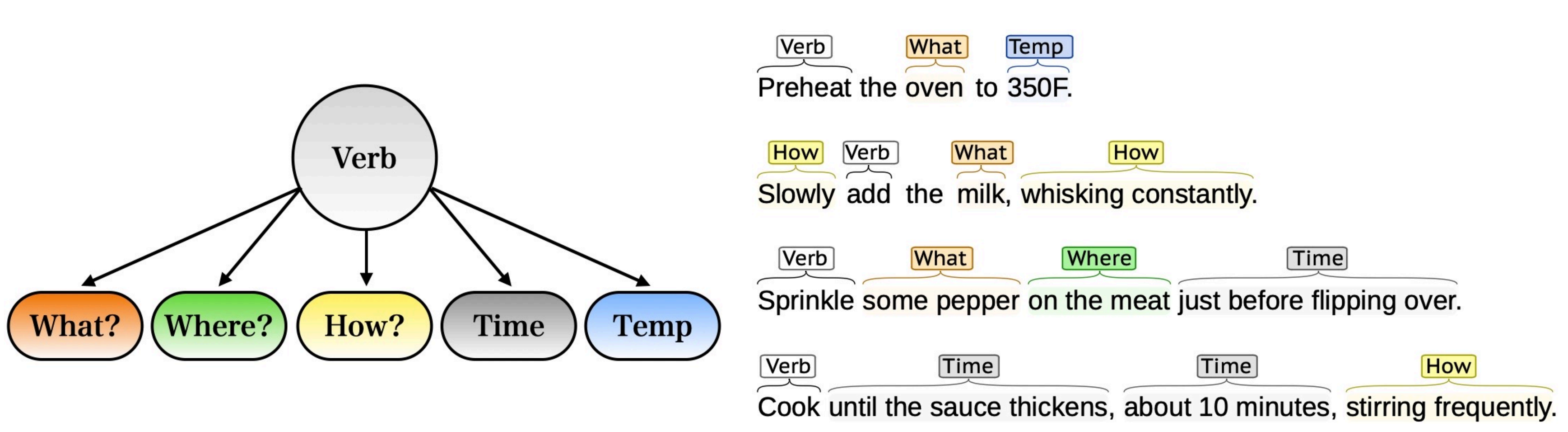
- Cooking poses a unique set of challenges to robots [1].
- Natural language has a practically infinite space of actions, while robots can only execute a small set of actions.
- The language of recipes is ambiguous, with context-implicit parts of speech, underspecified tasks, and explicit sequencing language (e.g. until, before) [2].

Approach

- Semantically parse a recipe r into a function representation for every detected high-level action.
- Reduce each high-level action $a \notin \mathcal{A}$ to a combination of primitive actions from $\mathcal{A}$.
- Cache the action reduction policy to an action library ($\mathbb{A}$) for future use.
- Translate r into an LTL formula ϕ with function representations as atomic propositions.

Named Entity Recognition (NER)

- Annotate subset of Recipe1M+ dataset [3] with salient categories $\mathcal{C}$ of an action.
- Fine-tune a BERT NER model to predict $\mathcal{C}$.



PRG

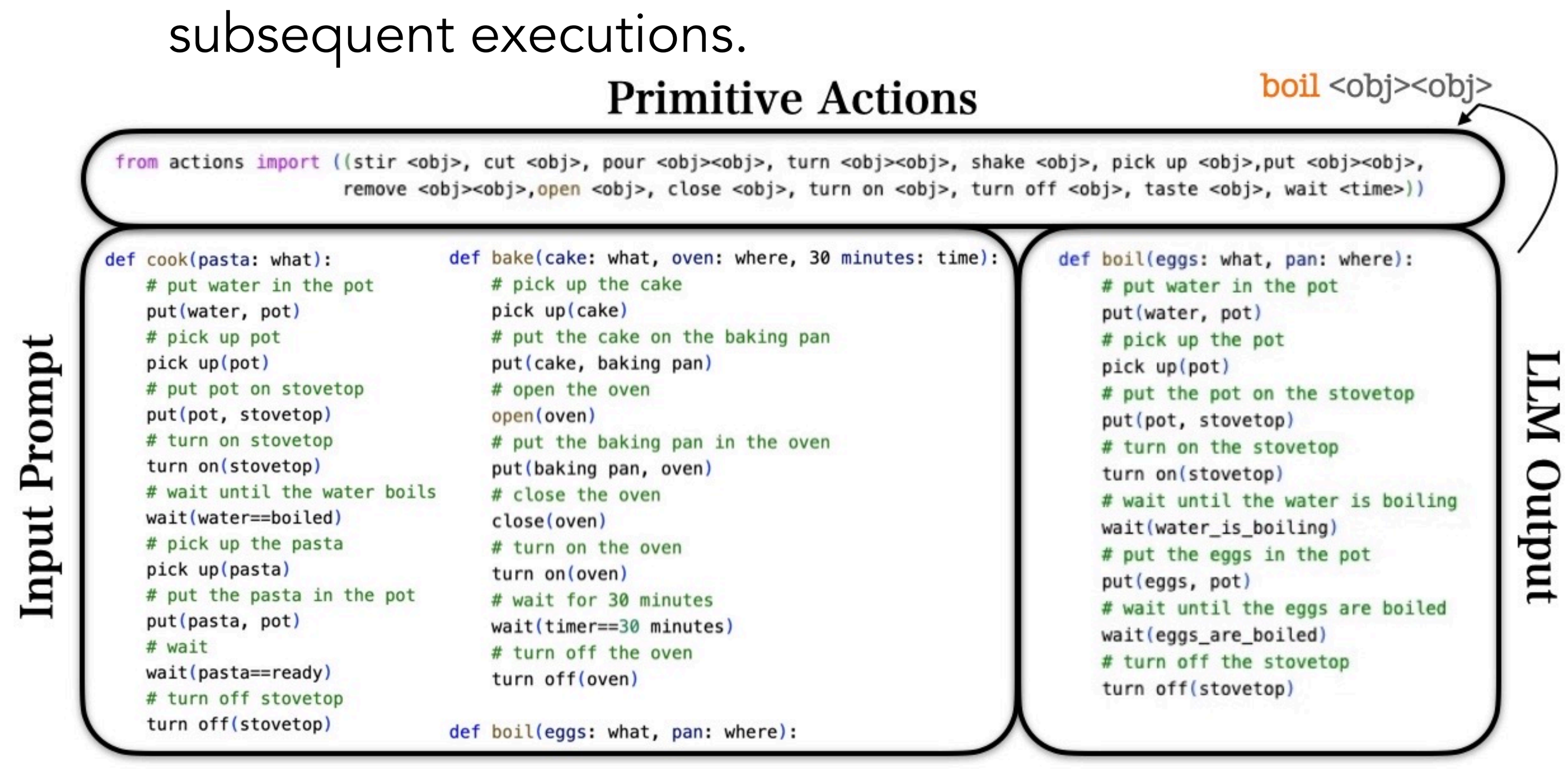
COMPUTER SCIENCE UNIVERSITY OF MARYLAND

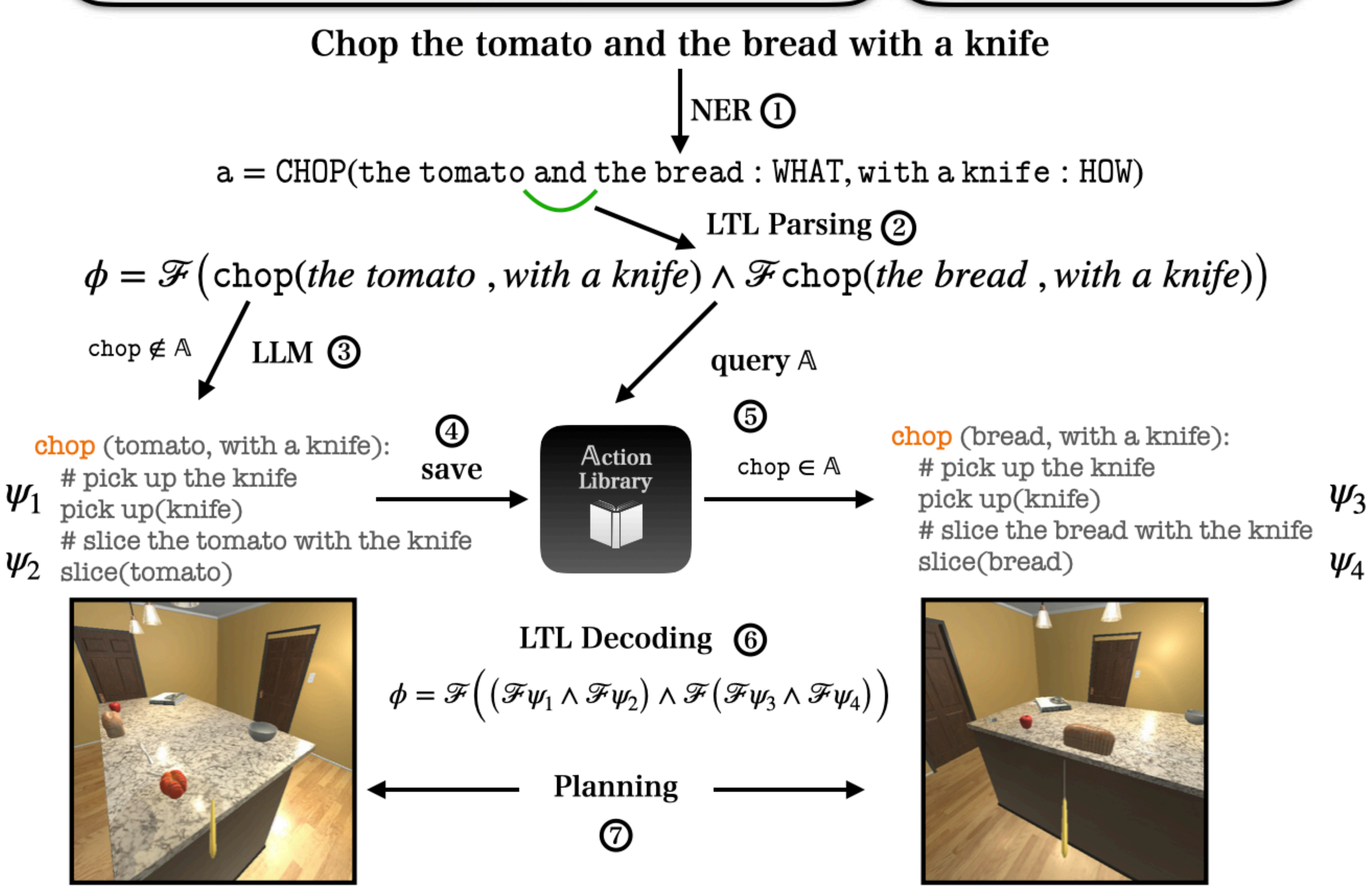
ROBOTICS

thefluentroboticslab

LLM Action Reduction

- Following ProgPrompt [4], we prompt an LLM with a pythonic import of the admissible actions in the environment and two example task plans in the form of pythonic functions.
- Once acquiring the plan for a newly seen action, we add the action to the import to enable model to invoke it in subsequent executions.





	Active Modules		
Metric	AR*	AR	Cook2LTL (AR+$\mathbb{A}$)
Executability (%)	0.91 ± 0.01	0.92 ± 0.01	$\mathbf{0.94 \pm 0.01}$
Time (min)	14.85 ± 1.05	9.89 ± 0.46	$\mathbf{6.05 \pm 0.12}$
Cost ($)	0.19 ± 0.01	0.16 ± 0.00	$\mathbf{0.11 \pm 0.00}$
API calls (#)	275 ± 0.00	231 ± 0.00	$\mathbf{134 \pm 0.00}$

Results

- We simulate Cook2LTL (AR+$\mathbb{A}$) on held out Recipe1M+ recipes and observe that it decreases LLM API calls (**-51%**), Latency (**-59%**), and Cost (**-42%**) compared to a baseline system (AR*) that queries the LLM for every newly encountered action at runtime (See table above).
- Additional simulations on 4 simple cooking tasks in an AI2-THOR [5] kitchen show that Cook2LTL is still more time-efficient but fails when the 1st LLM-generated plan is incorrect.

References

[1] Bollini et al. Interpreting and executing recipes with a cooking robot. Experimental Robotics 2013.

[2] Malamud et al. Cooking with Semantics. ACL 2014.

[3] Marin et al. A dataset for learning cross-modal embeddings for cooking recipes and food images. IEEE TPAMI 2019.

[4] Singh et al. ProgPrompt: Generating Situated Robot Task Plans using Large Language Models. CoRL 2021.

[5] Kolve et al. Ai2-THOR: An Interactive 3D environment for visual AI. RSS 2021.

We acknowledge the support of ARL and the NSF-funded NeuroPAC program.