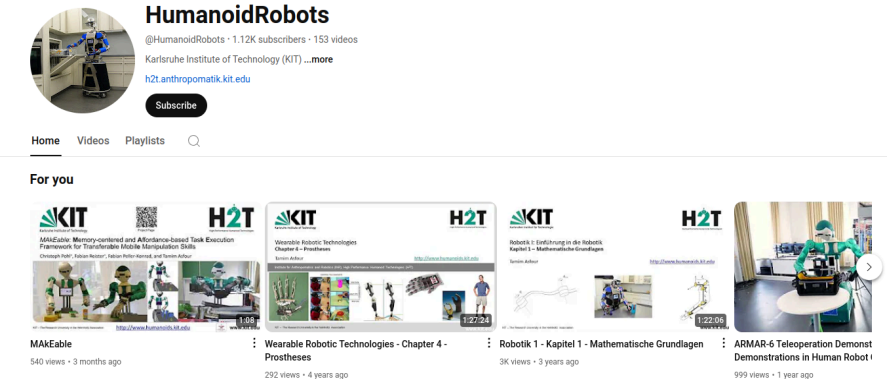
[KIT'24] AutoGPT+P: Affordance-based Task Planning using Large Language Mode

1. Link: https://arxiv.org/pdf/2402.10778
2. Arthurs and institution: Timo Birr, Christoph Pohl, Abdelrahman Younes and Tamim Asfour, KIT
3. brief intro: a German team focuses on building mobile manipulators (humanoid manipulations) https://www.youtube.com/channel/UCQC4i8MlCMCefJmPz7PmDMw



Comments:

1. How does the function $checksyntax(\Delta, \Xi)$ work in algorithm 3?
2. Comparing AutoGPT+P with INTERPRET, what are the key differences between them in ystem architectures and task capabilities?

Conclusions

The authors proposed AUTOGPT+P, a system for robotic task planning which 1. introduce an LLM-based approach to generate object affordance function 2. use (1) to automatically generate init state file, predicates file and goal file in PDDL. Meanwhile, this work still did not address the probability problem in LLM+P area for task planning.

Existing problems

1.'LLM + classical planners' can only generate plans if all objects needed to complete the task are available.

2. LLM+P has no automated error correction
3. It's vulnerable to contradictory goal definitions of the LLM.

Contributions

1.Deriving the planning domain from an affordance-based scene representation, which allows symbolic planning with arbitrary objects.

2. Handle planning with incomplete information, such as tasks with missing objects, by exploring the scene, suggesting alternatives, or providing a partial plan.
3. The affordance-based scene representation combines object detection with an Object Affordance Mapping that is automatically generated using ChatGPT.

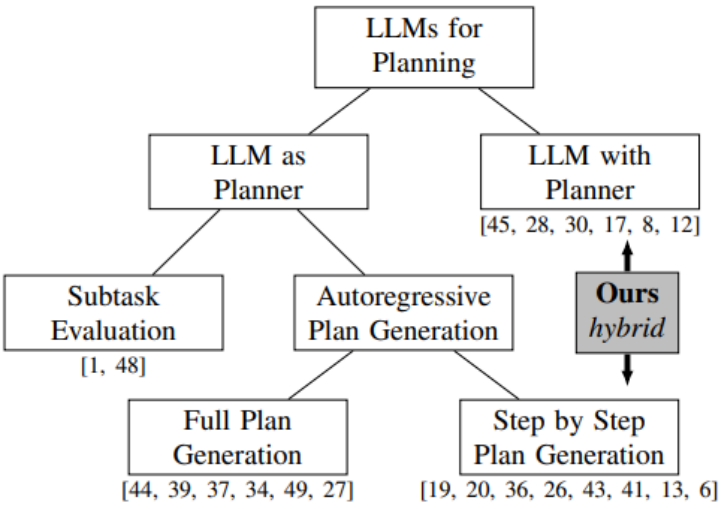
The core planning tool extends existing work by automatically correcting semantic and syntactic errors leading to a success rate of 98% on the SayCan instruction set.

Key concepts

1. generates the initial state of the PDDL problem from OAD and not from natural language
2. dynamically generates the PDDL domain based on the capabilities of the agents and the OAM,

Details

1. a literature collection of LLM+P



Approach	Planning Method	# Affordances	Substitutions	Long-Horizon	Novel Classes
Lörken and Hertzberg [31]	PDDL + Planner	2 (liftablibility & pushability)	implicit	no	no
Wang et al. [40]	Reinforcement Learning	1 (movability)	none	no	yes
Awaad et al. [3, 4, 5]	Hierarchical Task Network	(not directly listed)	explicit	yes	no
Moldovan et al. [32]	Own Algorithm similar to Monte Carlo Search	2 (tap and push)	none	no	no
Chu et al. [9, 10]	PDDL + Planner	7	implicit	no	yes
Xu et al. [46]	PDDL + Planner	4 (grasp, cut, contain, support)	implicit	no	yes
Ours	PDDL + Planner / Hybrid with LLM	16	explicit & implicit	yes	no

2. problem formulation 2.1 OAD

[4]. To this end, we represent the scene S symbolically as a set of object-affordance-pairs p_i, where each object has one or more affordances assigned to it as in Equation 1:

$$S = \{p_1, \ldots, p_n\}, \text{ with}$$
$$p_i = (o_i, k_i, a_i, b_i) \in (\mathbb{O} \times \mathbb{N}_0 \times \mathbb{A} \times [0,1]^4), \quad (1)$$

where $\mathbb{O}$ is the set of all object classes in the domain and $\mathbb{A}$ is the set of all possible affordances. b_i represents the object's bounding box in normalized coordinates. Thus, the space of all scenes can be expressed as $\mathbb{S} = \mathcal{P}(\mathbb{O} \times \mathbb{N}_0 \times \mathbb{A} \times [0,1]^4)$. The problem of deriving this representation from the image of a scene, i. e., Object Affordance Detection (OAD), can be formalized as in Equation 2:

$$OAD : \mathbb{I} \rightarrow \mathbb{S} \quad (2)$$

2.2 closed world planning and alternative suggestion

$$ClosedWorldPlanning : (\Lambda \times \mathbb{S} \times \mathcal{P}(\mathbb{R}_S)) \rightarrow A^{\mathbb{N}} \quad (6)$$

An alternative suggestion is the problem of suggesting an alternative object $alt \in O$, where O is the set of object classes present in the scene, given a user-specified task λ in natural language and a missing object class $o \in \mathbb{O}$ needed to fulfill that task. This can be written as

$$AlternativeSuggestion : \Lambda \times \mathbb{O} \rightarrow O \quad (7)$$

2.3 semantic errors as the occurrence of multiple predicates that cannot be true at the same time in a real scene.

2.4 overview of the system arch.

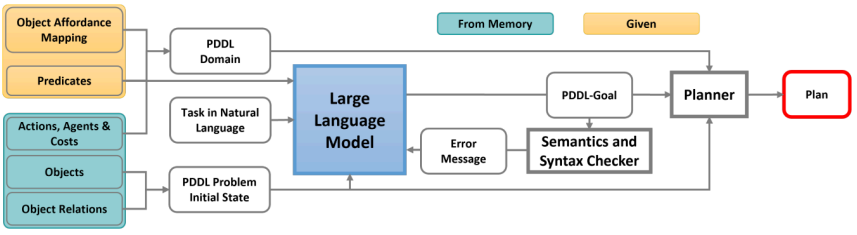


Fig. 5: Overview of the Planning Tool. Rounded boxes represent the input and the output of the components that are represented