Ground Manipulator Primitive Tasks to Executable Actions Using Large Language Models

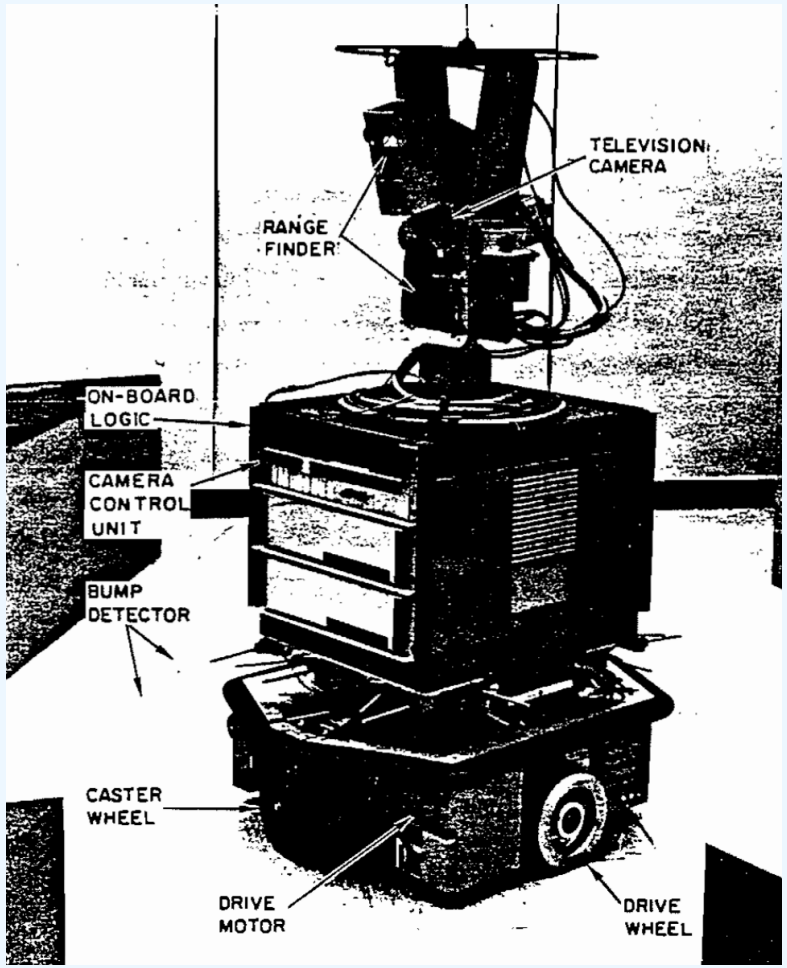
2023 AAAI Fall Symposium on

Unifying Representations for Robot Application Development

Yue Cao, C.S. George Lee
Purdue University
Elmore Family School of Electrical and Computer Engineering

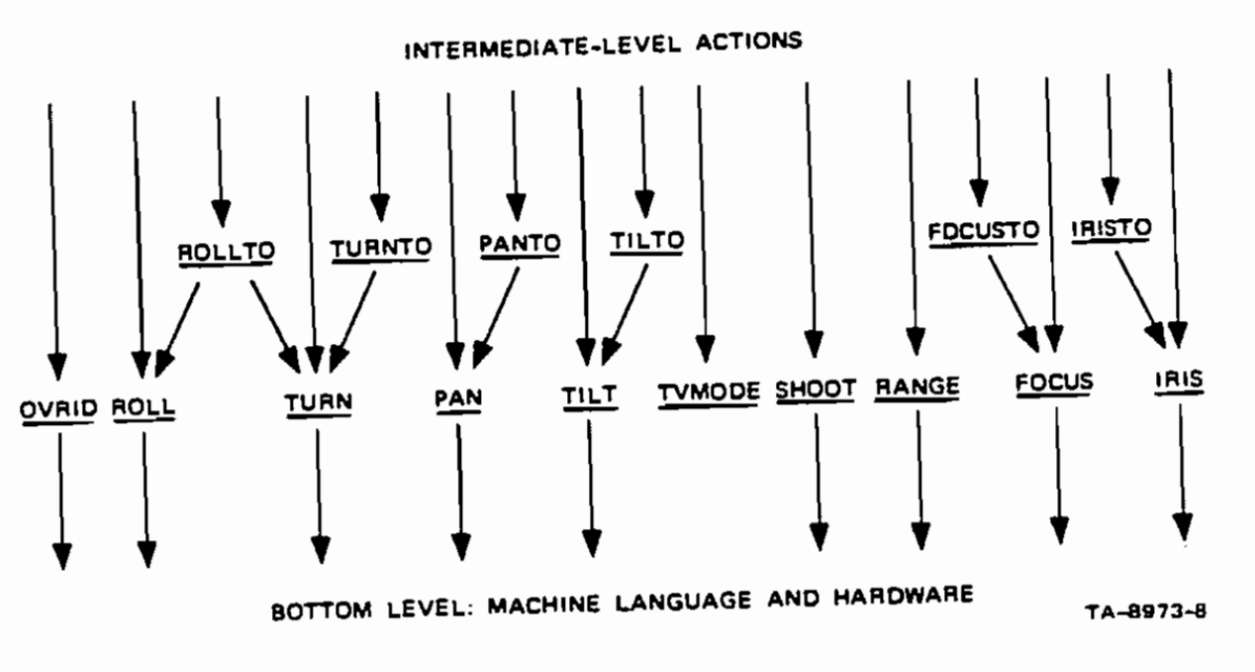
Layered Architectures

- **“Shakey” the Robot**
  1966 – 1972,
  Stanford Research Institute



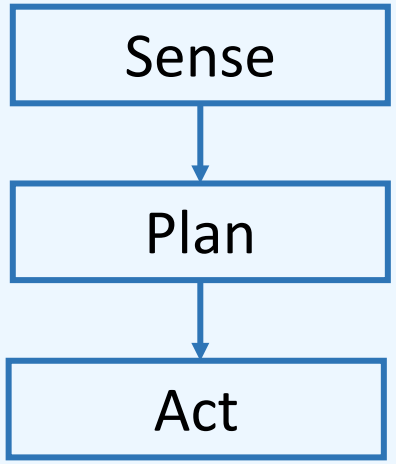
Layered Architectures

- **“Shakey” the Robot**
  Layered Design



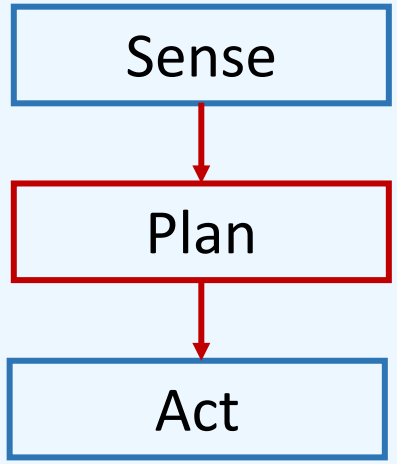
Nilsson, N. J. 1984. Shakey the Robot. Technical Report 323, SRI AI Center.

Layered Architectures



SMPA/SPA paradigm

Layered Architectures



SMPA/SPA paradigm

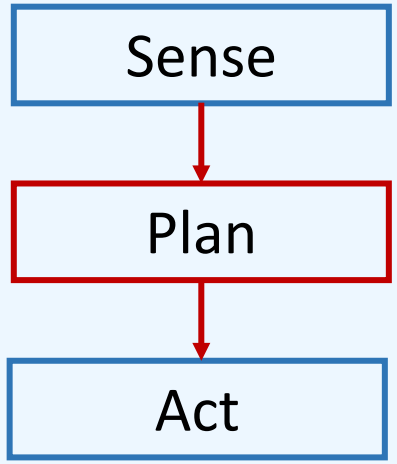
- **Debate from Rodney Brooks**

- *"The idea of planning and plan execution is just an intuition based decomposition. There is no reason it has to be that way."*

- *"Plans provide a useful level of abstraction for a designer or observer of a system but provide nothing to a robot operationally."*

Brooks, R. A. 1987. Planning is Just a Way of Avoiding Figuring Out What To Do Next. Working Papers WP-303, MIT AI Laboratory.

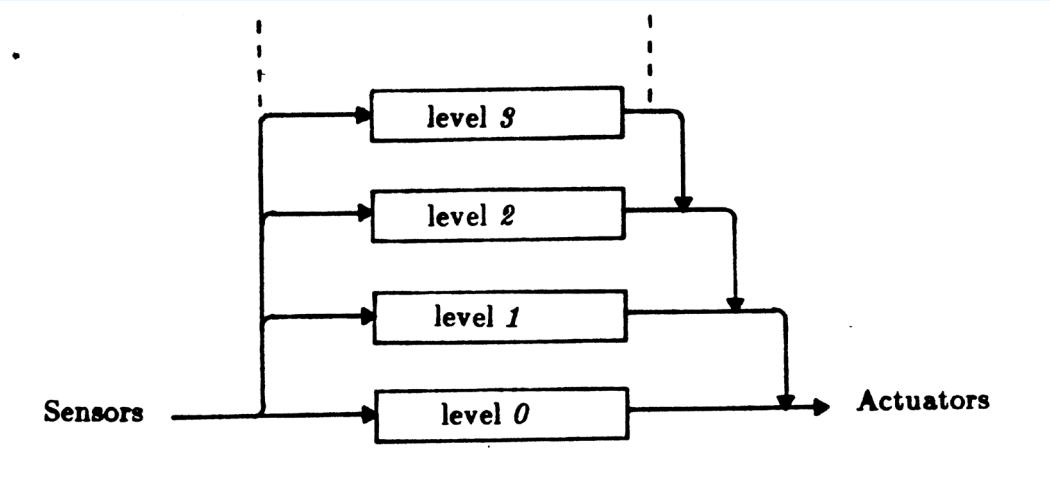
Layered Architectures



SMPA/SPA paradigm

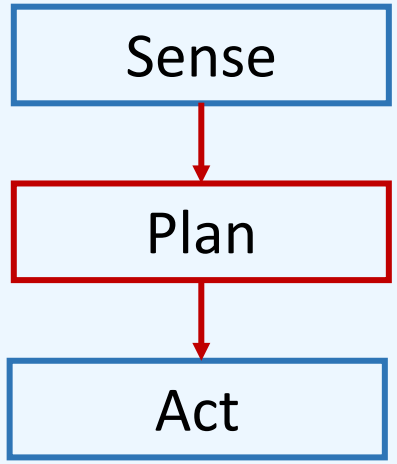
- **Solution from Rodney Brooks:**

  the Subsumption Architecture



Brooks, R. A. 1991. New Approaches to Robotics. Science, 253(5025): 1227–1232.

Layered Architectures



SMPA/SPA paradigm

- **Solution from Rodney Brooks:**

  the Subsumption Architecture

- **What happened after years?**

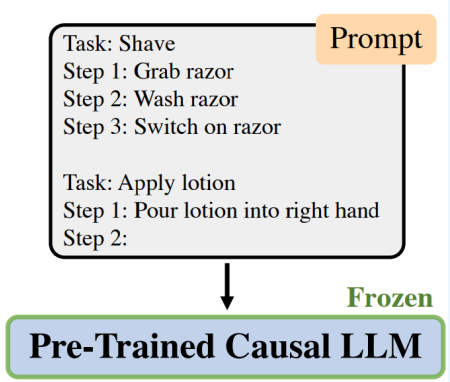
  Not too much progress.
  Most modern robots stay with layered architectures.
  But the gap between layers remains.

LLM for Robotics

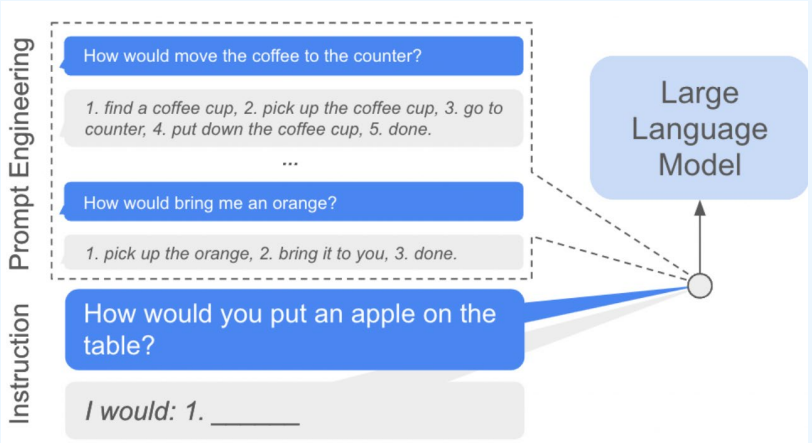
- **New technique recently?**

Large Language Models for robotics.

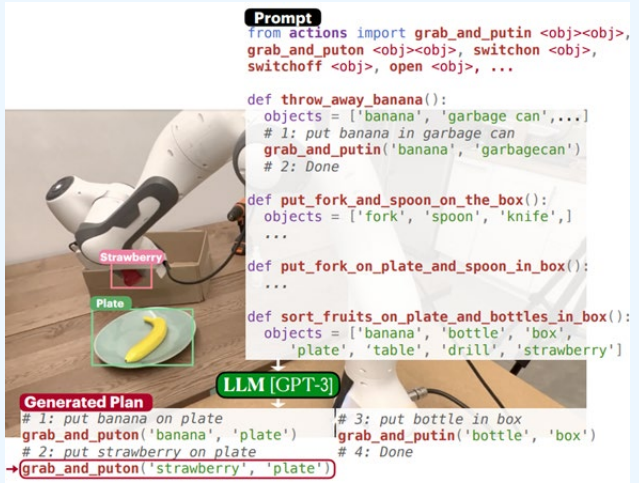
Majority of current work: in the planning layer.



Zero-shot Planner [1]



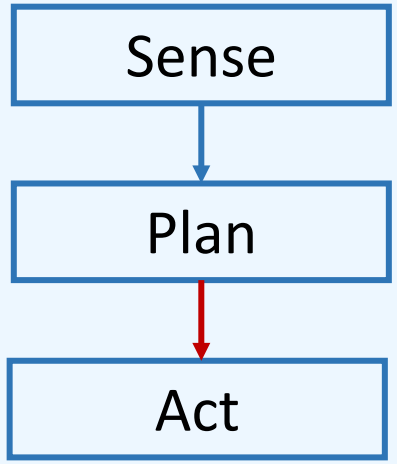
SayCan [2]



ProgPrompt [3]

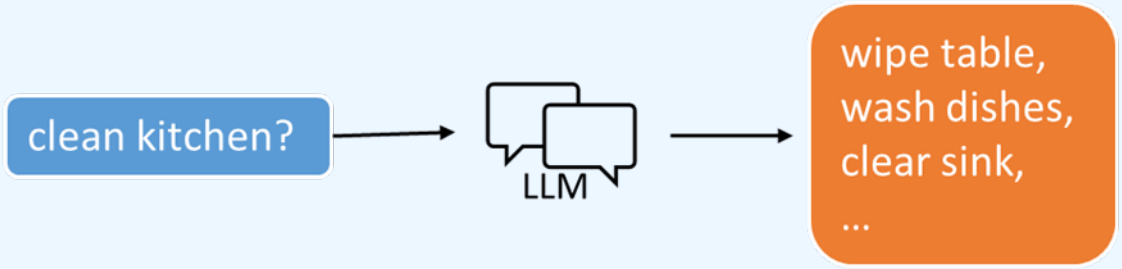
[1] Huang, W , et al. 2022. Language Models as Zero-Shot Planners: Extracting Actionable Knowledge for Embodied Agents.
[2] Ahn, M , et al. 2022. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances.
[3] Singh, I. et al. 2022. ProgPrompt: Generating Situated Robot Task Plans using Large Language Models.

LLM for Robotics



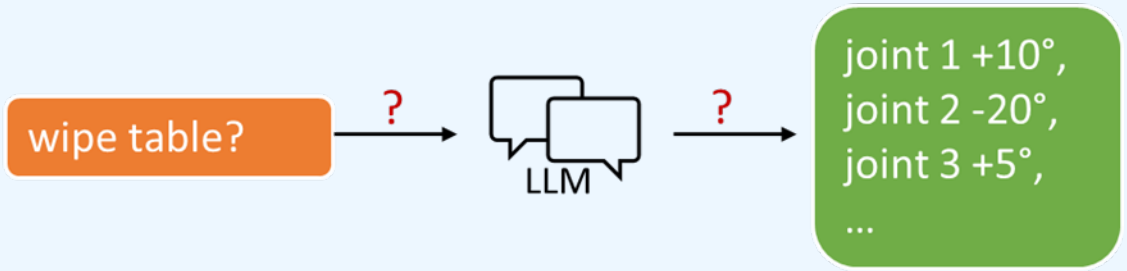
SMPA/SPA paradigm

- **A summary of many current work**



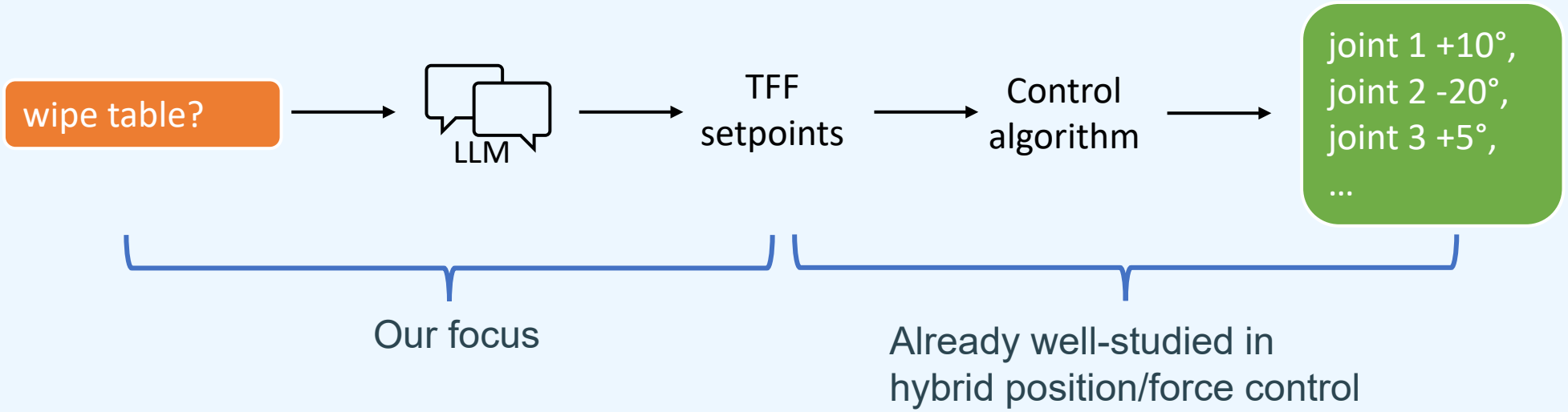
But these tasks are not yet linked with low-level actuator execution.

- **What we look for**



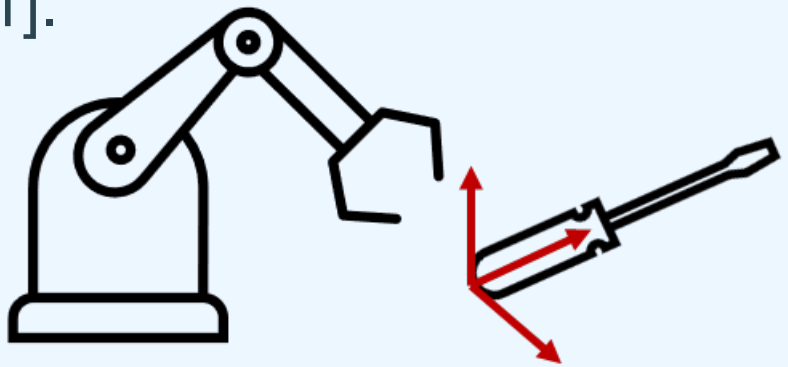
Scope and Idea

- Manipulator primitive tasks, typically contact-rich.
- LLM-based approach
- **Input:** a natural-language-described manipulator primitive task.
- **Utilize:** task frame formalism (TFF)
- **Output:** a set of position/force set-points in the task frame.



Recap: Task Frame Formalism

- A classical concept originated in [Mason, 1981].
- A frame specified on the manipulated object, robot-agnostic.
- 3 translational directions,
  3 rotational directions,
  to be either position controlled/force controlled.



The effector velocity and effector force are represented as column vectors in a six-dimensional vector space over the reals:

$$\boldsymbol{v} = (v_x v_y v_z w_x w_y w_z)^T$$

$$\boldsymbol{f} = (f_x f_y f_z g_x g_y g_z)^T$$

Captured from Mason's paper

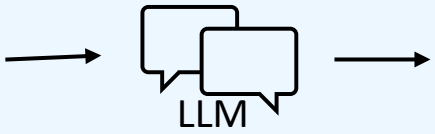
Mason, M. T. 1981. Compliance and Force Control for Computer Controlled Manipulators. IEEE Trans. Syst. Man Cybern.

Approach: Overview

- Program-function-like prompt
- Task-frame-formalism-based representation
- Few-shot inference

```python
# Source function 1
def turn_screw(translational_x, translational_y, translational_z,
               angular_x, angular_y, angular_z):
  # Coordinate setting: Z axis as the direction of screw
  translational_x=0
  translational_y=0
  translational_z=-5 # N

  angular_x=0
  angular_y=0
  angular_z=5 # rad/sec
  return translational_x, translational_y, translational_z, \
  angular_x, angular_y, angular_z

# Source function 2
def wipe_table(...)
  ...
  return

# Source function 3
def open_door_from_doorknob(...)
  ...
  return

# Target function
def turn_steering_wheel(translational_x, translational_y, translational_z,
                        angular_x, angular_y, angular_z):
```

A 3-shot prompt example




```python
# Coordinate setting: Z axis as the direction of the steering wheel
translational_x=0
translational_y=0
translational_z=0

angular_x=0
angular_y=0
angular_z=5 # rad/sec
return translational_x, translational_y, translational_z, \
angular_x, angular_y, angular_z
```

LLM Output

Approach: Breakdown

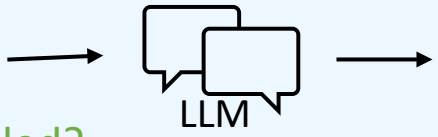
manipulator primitive task name

```python
# Source function 1
def turn_screw(translational_x, translational_y, translational_z,
               angular_x, angular_y, angular_z):
  # Coordinate setting: Z axis as the direction of screw
  translational_x=0
  translational_y=0
  translational_z=-5 # N

  angular_x=0
  angular_y=0
  angular_z=5 # rad/sec
  return translational_x, translational_y, translational_z, \
  angular_x, angular_y, angular_z

# Source function 2
def wipe_table(...)
  ...
  return

# Source function 3
def open_door_from_doorknob(...)
  ...
  return

# Target function
def turn_steering_wheel(translational_x, translational_y, translational_z,
                        angular_x, angular_y, angular_z):
```

coordinate in task frame formalism

activate directions in task frame formalism

position or force controlled?




```python
# Coordinate setting: Z axis as the direction of the steering wheel
translational_x=0
translational_y=0
translational_z=0

angular_x=0
angular_y=0
angular_z=5 # rad/sec
return translational_x, translational_y, translational_z, \
angular_x, angular_y, angular_z
```

LLM Output

generate for a new manipulator primitive task

A 3-shot prompt example

Preliminary Evaluation

- Evaluated in 30 manipulator primitive tasks* in July 2023.

1. cut pizza	11. rasp wood	21. open door from hinge
2. scrub desk with bench brush	12. scrape substance from surface	22. slide block over vertical surface
3. spear cake with fork	13. peel potato	23. turn steering wheel
4. fasten screw with screwdriver	14. slice cucumber	24. shake cocktail bottle
5. loosen screw with screwdriver	15. flip bread	25. cut banana
6. unlock lock with key	16. shave object	26. crack egg
7. fasten nut with wrench	17. use roller to roll out dough	27. press button
8. loosen nut with wrench	18. insert peg into pegboard	28. insert GPU into socket
9. spread paint with brush	19. brush across tray	29. open bottle
10. hammer in nail	20. insert straw through cup lid	30. open childproof bottle

- Overall correct rate

	0-shot	1-shot	3-shot	5-shot
GPT-3.5-turbo	0	0.30	0.70	0.67
GPT-4	0	0.47	0.63	0.83
Bard	0	0	0.47	0.70
LLaMA-2-70B	0	0	0.07	0.13

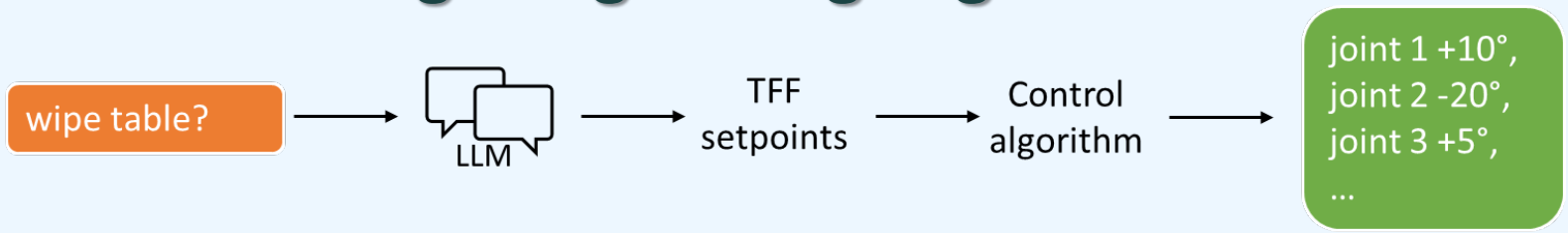
- Task correctness in 5-shot test. blue: correct, red: incorrect

	Task No.	Correctness
GPT-3.5-turbo	1-10	
	11-20	
	21-30	
GPT-4	1-10	
	11-20	
	21-30	
Bard	1-10	
	11-20	
	21-30	
LLaMA-2-70B	1-10	
	11-20	
	21-30	

AAAI Symposium Series

* Among them, Task No. 1-20 are selected from [Paulius, D, et al. 2020. A Motion Taxonomy for Manipulation Embedding. RSS.], Task No. 21-30 are created by us.

Ground Manipulator Primitive Tasks to Executable Actions Using Large Language Models



Take-Aways

- Recent work sharing similar motivation in applying LLMs for low-level actions, but different in approach and application domain:

BLOG ›

Language to rewards for robotic skill synthesis

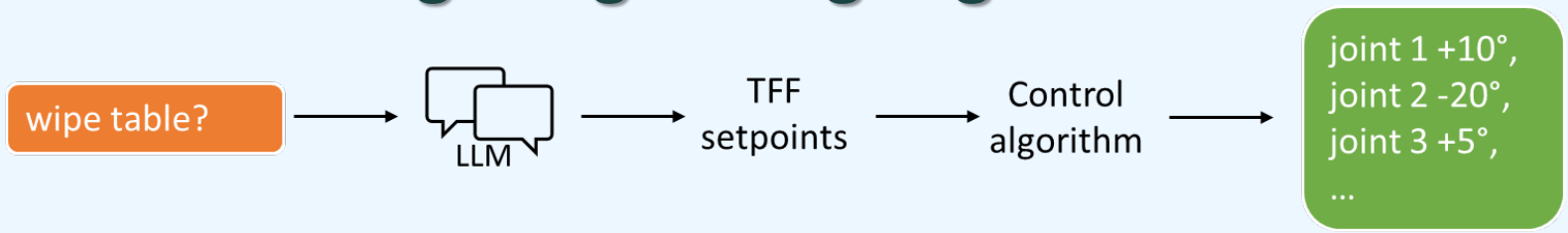
TUESDAY, AUGUST 22, 2023

Posted by Wenhao Yu and Fei Xia, Research Scientists, Google

Eureka: Human-Level Reward Design via Coding Large Language Models

Jason Ma [1 2], William Liang[2], Guanzhi Wang[1 3], De-An Huang[1],
Osbert Bastani[2], Dinesh Jayaraman[2], Yuke Zhu[1 4], Linxi "Jim" Fan [1‡], Anima Anandkumar[1 3‡]
[1]NVIDIA; [2]UPenn; [3]Caltech; [4]UT Austin; [‡]Equal Advising

Ground Manipulator Primitive Tasks to Executable Actions Using Large Language Models



Take-Aways

- This work was presented in IROS Detroit late-breaking results. Some feedback to share:

  - Why Python?
  - Numerical values make sense?
  - Any world state?
  - Evaluation metrics?

Thanks!