

Large Language Models for Autonomous Robot Planning and Acting

Research Talk at Randolph College

Yue Cao

Purdue University, ECE
yuecao@purdue.edu

Overview

1. Real Autonomous Robots
2. LLM for Planning
3. LLM for Acting

Fake Autonomous Robots

Kiwibot 2018: food delivery at
UC Berkeley campus



Fake Autonomous Robots

Kiwibot 2018: food delivery at UC Berkeley campus



What is actually behind?

San Francisco Chronicle

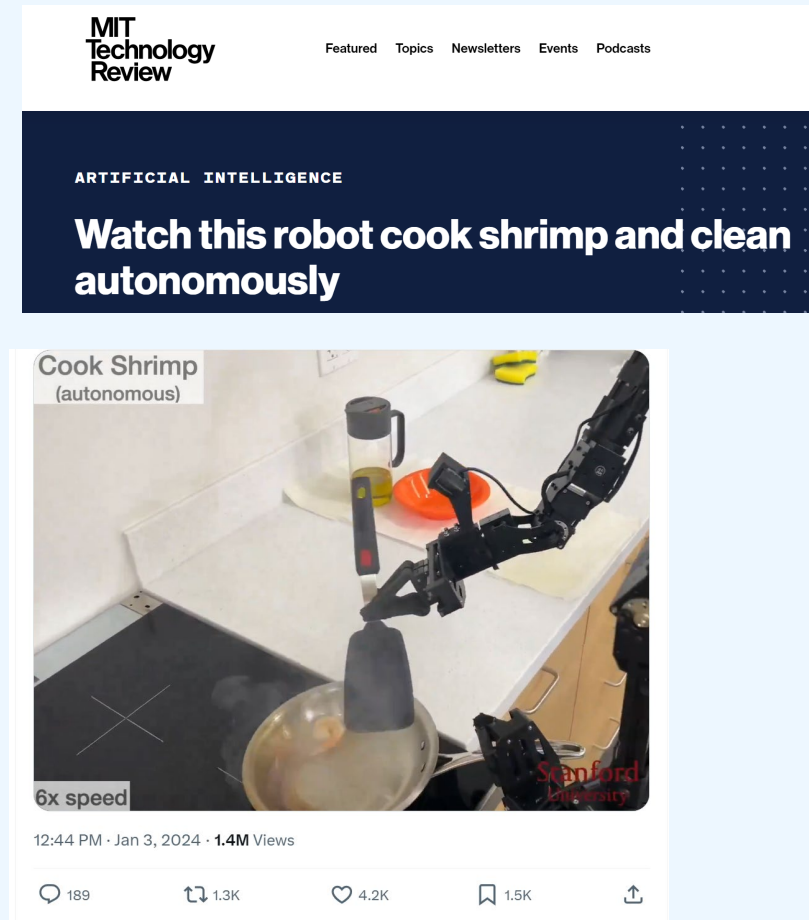
The Kiwibots do not figure out their own routes. Instead, people in Colombia, the home country of Chavez and his two co-founders, plot “waypoints” for the bots to follow, sending them instructions every five to 10 seconds on where to go.

As with other offshoring arrangements, the labor savings are huge. The Colombia workers, who can each handle up to three robots, make less than \$2 an hour, which is above the local minimum wage.

“Kiwibots win fans at UC Berkeley as they deliver fast food at slow speeds”, San Francisco Chronicle, 2019

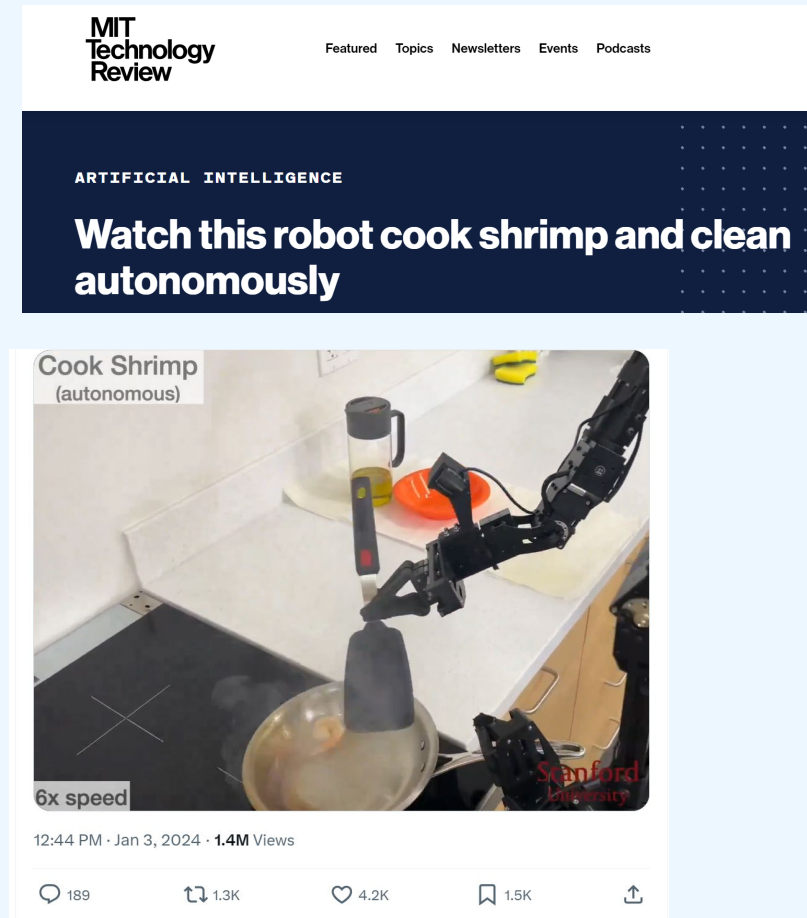
Fake Autonomous Robots

Mobile ALOHA 2024: Stanford



Fake Autonomous Robots

Mobile ALOHA 2024: Stanford



What is actually behind?

IEEE Spectrum / Robotics

What's wrong with teleoperation, then?

Nothing! Teleoperation is great. But when people see a robot doing something and it *looks autonomous but it's actually teleoperated*, that's a problem, because it's a misrepresentation of the state of the technology. Not only do people end up with the wrong idea of how your robot functions and what it's really capable of, it also means that whenever those people see *other* robots doing similar tasks autonomously, their

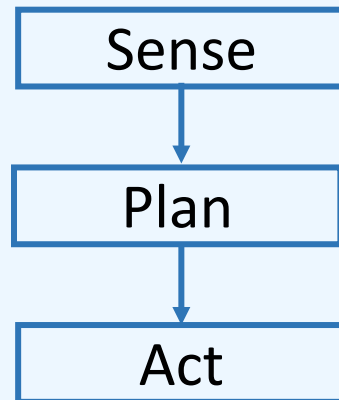
“That Awesome Robot Demo Could Have a Human in the Loop”, IEEE Spectrum, 2024

Towards Real Autonomous Robots

Two fundamental questions

1. New technology?
Artificial intelligence (AI)

2. Robot capability?



IEEE Spectrum / Robotics

What's wrong with teleoperation, then?

Nothing! Teleoperation is great. But when people see a robot doing something and it *looks* autonomous but it's *actually* teleoperated, that's a problem, because it's a misrepresentation of the **state of the technology**.

Not only do people end up with the wrong idea of how your robot functions and **what it's really capable of**, it also means that whenever those people see *other* robots doing similar tasks autonomously, their

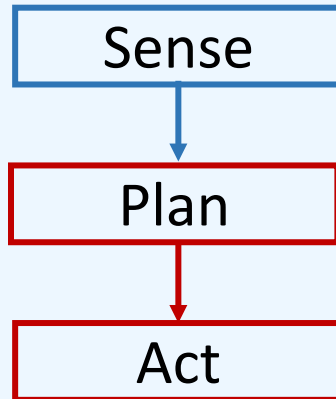
Towards Real Autonomous Robots

Two fundamental questions

1. New technology?

Artificial intelligence (AI) →

2. Robot capability?



My Research Focus

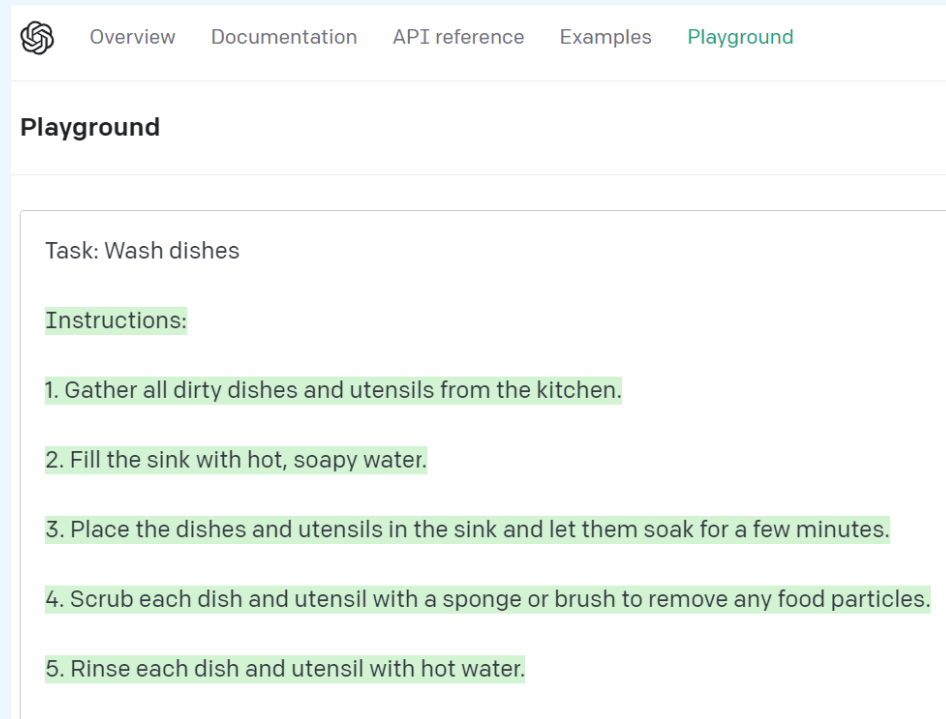
Large Language Models

For

Planning and Acting

LLM for Planning

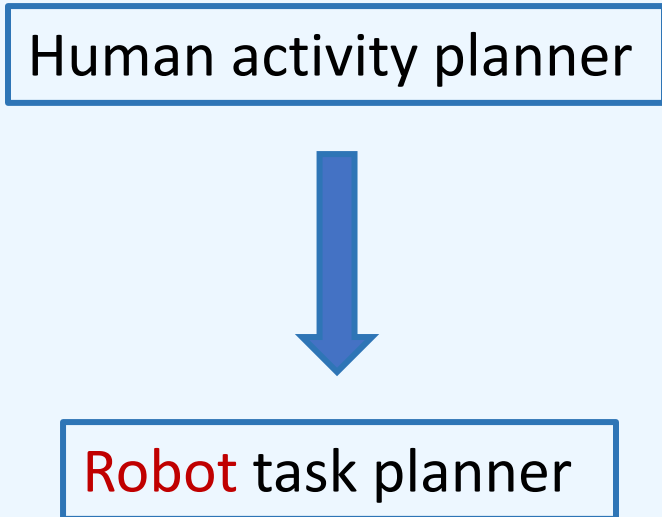
GPT can do planning for human activities



The screenshot shows the OpenAI Playground interface. At the top, there are navigation links: Overview, Documentation, API reference, Examples, and Playground. Below the navigation bar, the word "Playground" is displayed. The main content area shows a task: "Task: Wash dishes". Underneath, the word "Instructions:" is followed by a list of five steps, each on a new line and highlighted with a green background:

1. Gather all dirty dishes and utensils from the kitchen.
2. Fill the sink with hot, soapy water.
3. Place the dishes and utensils in the sink and let them soak for a few minutes.
4. Scrub each dish and utensil with a sponge or brush to remove any food particles.
5. Rinse each dish and utensil with hot water.

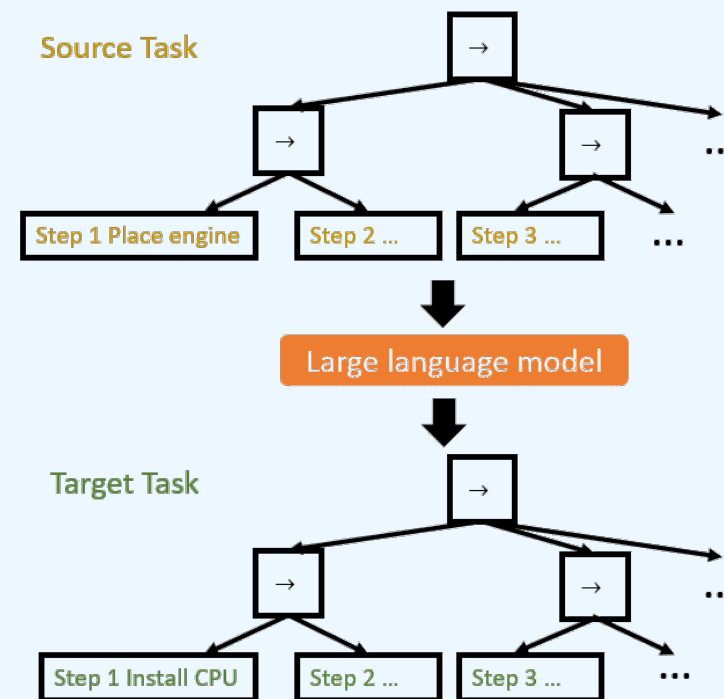
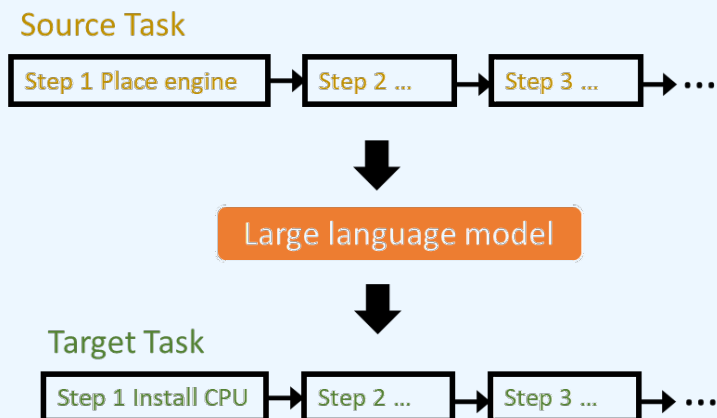
How about using LLMs in robot tasks?



LLM for Planning

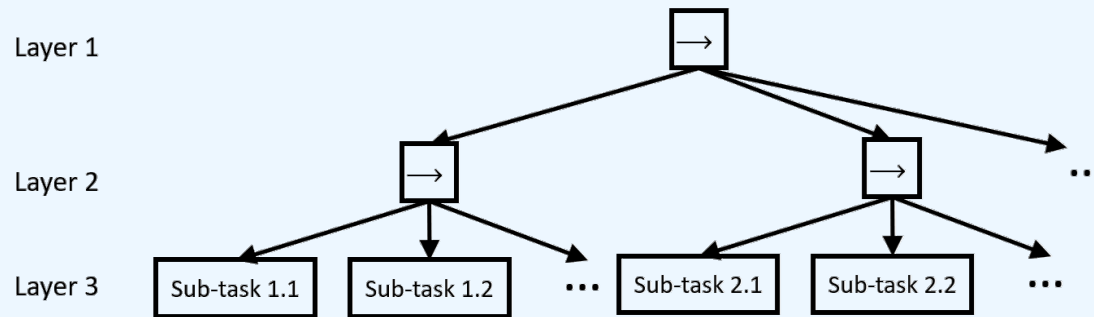
Previous works:
Generated task are **sequential**

Can we generate **modular** tasks?
Better reusability and readability



LLM for Planning

- Behavior-Tree Task Generation with LLMs
Prompt Design: Phase-Step Prompt



Source Task

Procedures:

Phase 1.

Step 1. [Sub-task 1.1] x ; Step 2. [Sub-task 1.2] x ; ...

Phase 2.

Step 1. [Sub-task 2.1] x ; Step 2. [Sub-task 2.2] x ; ...

...

Target Task: Task Description

Procedures:

[Phase 1.

Step 1. Sub-task ?; Step 2. Sub-task ?; ...

Phase 2.

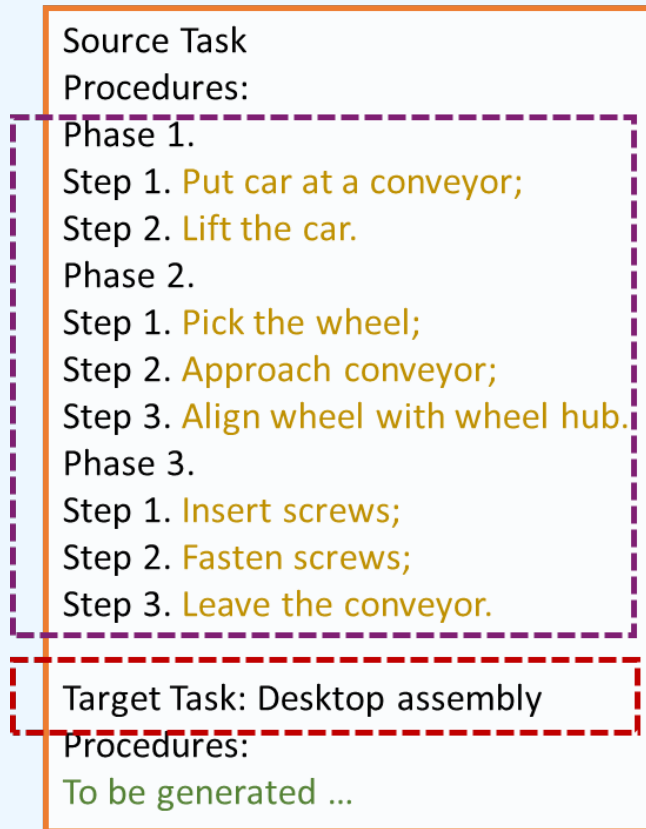
Step 1. Sub-task ?; Step 2. Sub-task ?; ...

...] y

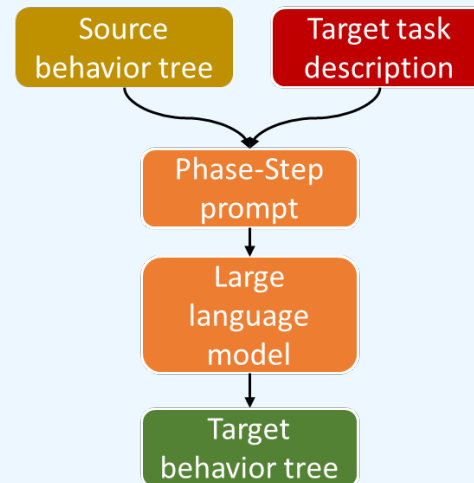
Layer 2 node: Phase; Layer 3 node: Step

LLM for Planning

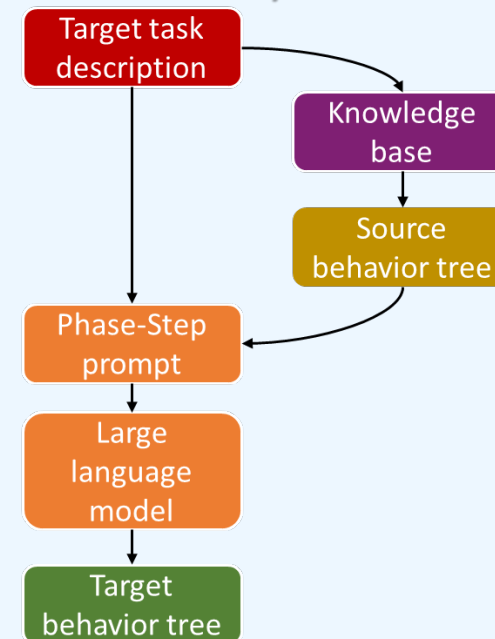
- Use domain knowledge to improve prompt



Standard way



Better way



Integrate knowledge base retrieval into the pipeline

The end-user just needs to input a short **target task description**. The source behavior tree can be automatically retrieved from a knowledge base.

LLM for Planning

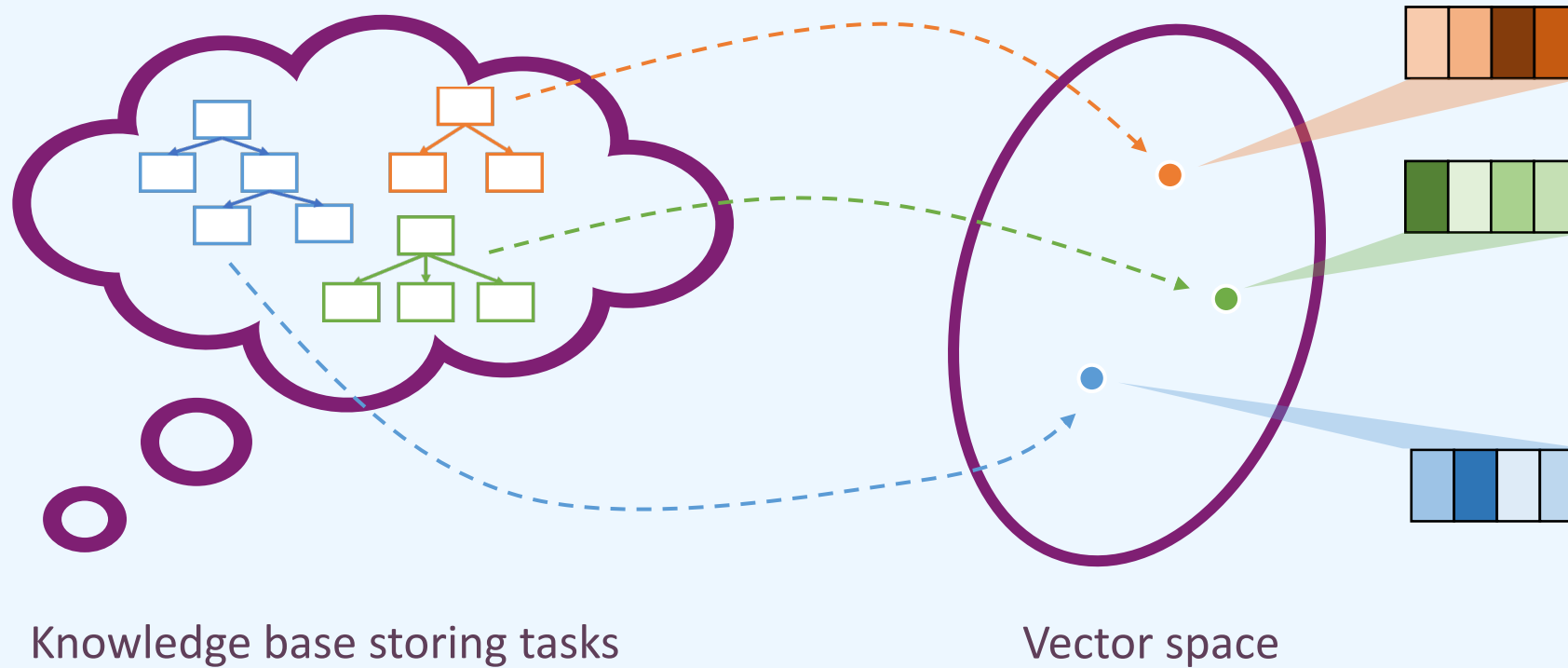
- How to retrieve **similar** tasks from a knowledge base?
- Solution:
Embedding, mapping an **entity** into a fixed length vector.



word, document, Amazon product, Netflix movie...(Entity2Vec)

LLM for Planning

- Vector Embeddings for behavior-tree tasks



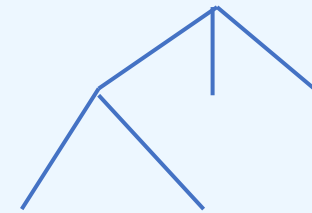
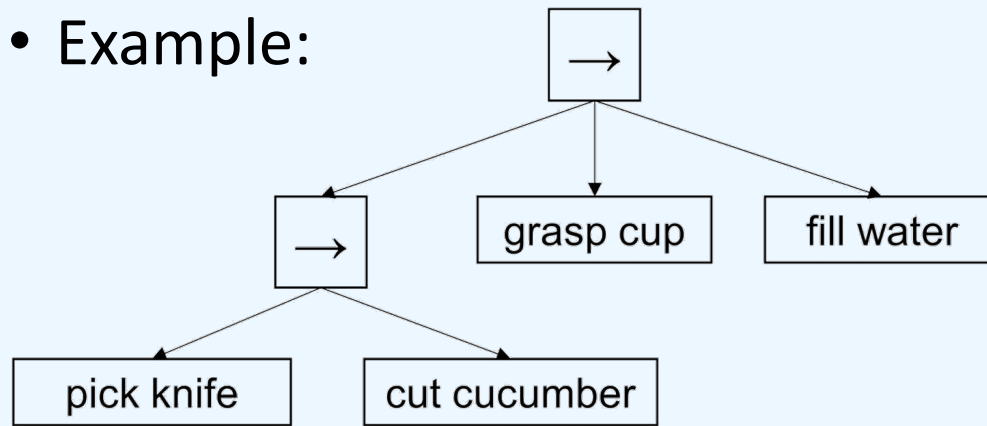
LLM for Planning

- Principle: In the vector space, **similar** tasks should be close, while **distinct** tasks should be far away.



Define
similarity?

- Example:



Structural
characteristics



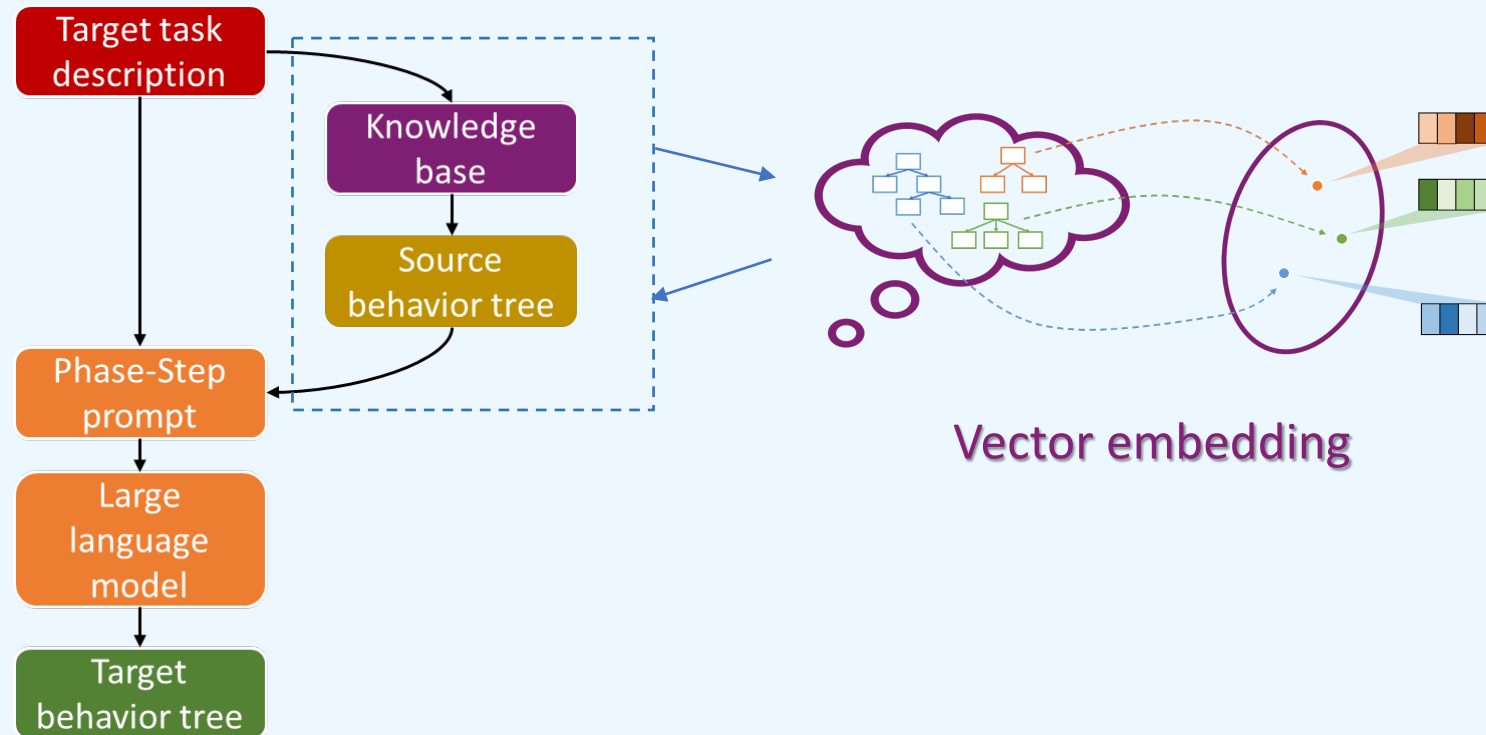
grasp cup pick knife

Semantic
characteristics

pick knife cut cucumber

LLM for Planning

- Summary

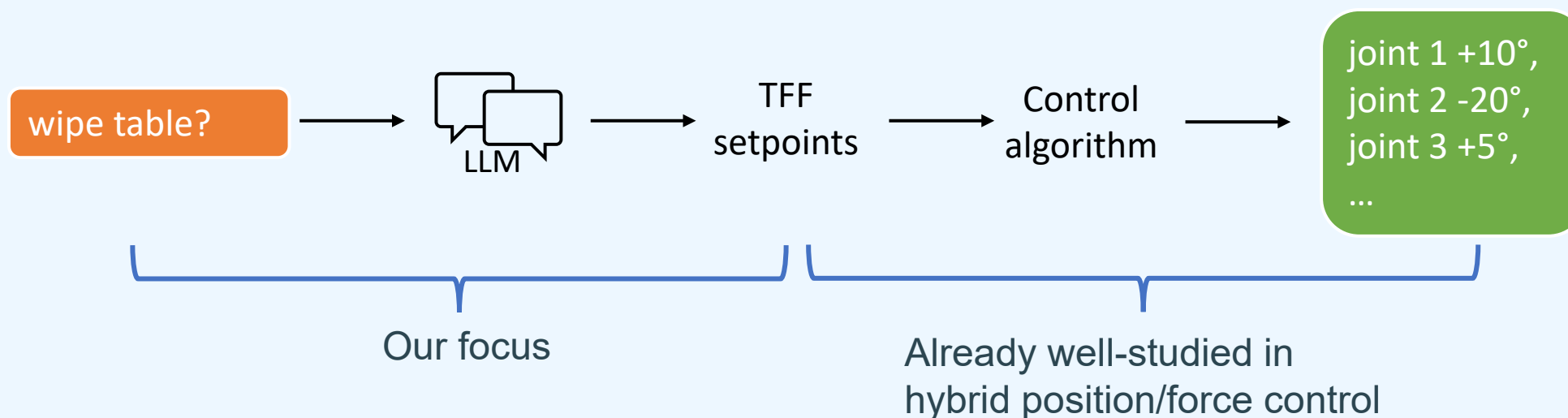


LLM for Acting

- These generated tasks are not yet linked with **low-level motor execution**.
- Using **LLMs** to realize this process?

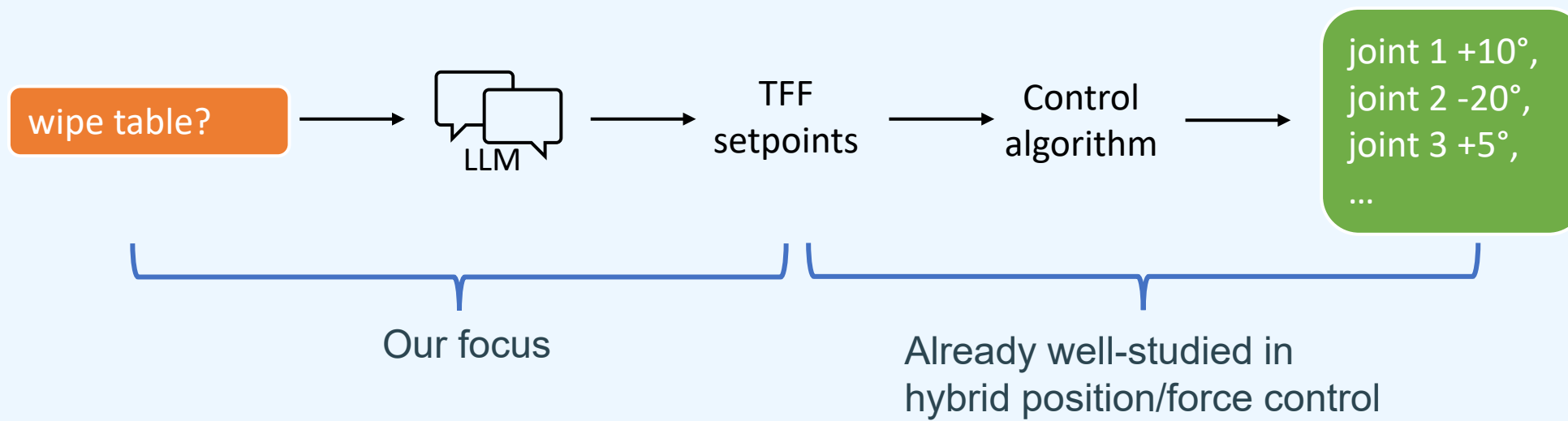
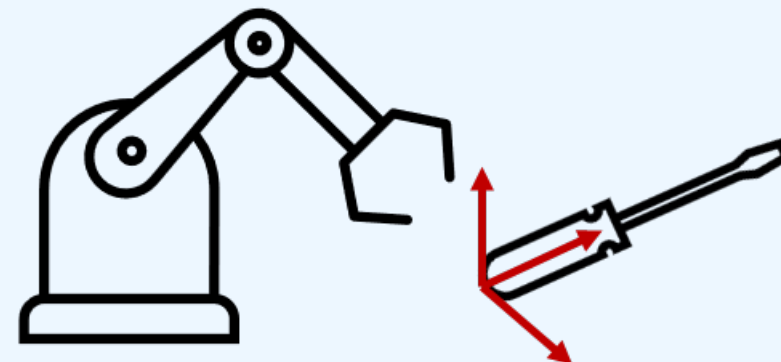
LLM for Acting

- Scope: Manipulator primitive tasks, typically contact-rich.
- **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.



LLM for Acting

- task frame formalism
- A frame specified on the manipulated object, 3 translational directions, 3 rotational directions, either position controlled/force controlled.



LLM for Acting

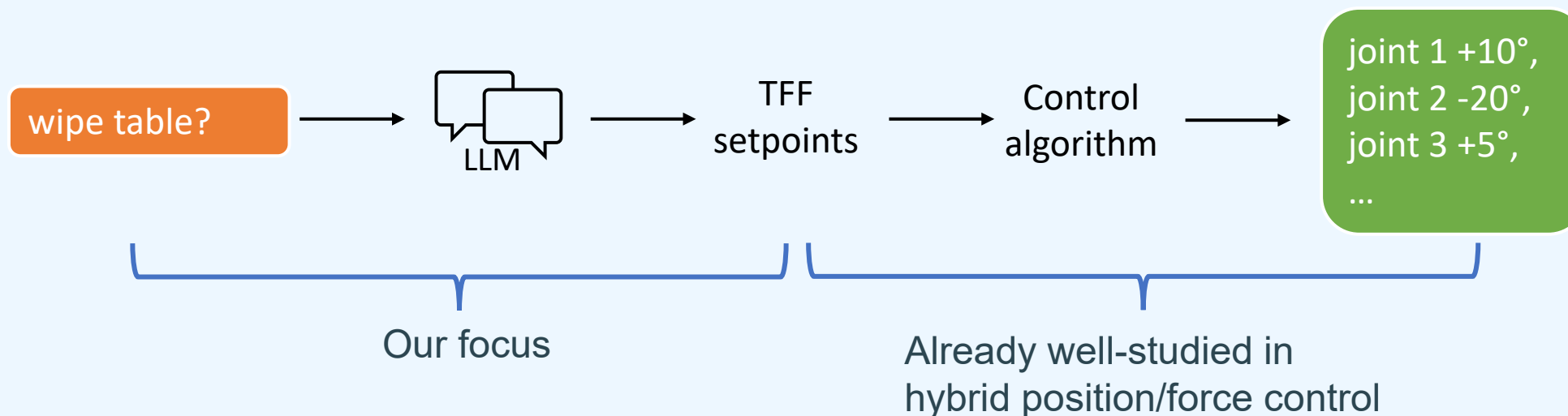
The effector velocity and effector force are represented as column vectors in a six-dimensional vector space over the reals:

$$\mathbf{v} = (v_x v_y v_z w_x w_y w_z)^T$$
$$\mathbf{f} = (f_x f_y f_z g_x g_y g_z)^T$$

Structured form!

Captured from Mason's paper

LLMs prefer **structured input**, think about the example of generating fake citations (.bib file).



LLM for Acting

```
# 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



```
# 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

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

Overview

1. Real Autonomous Robots
(not teleoperated)
2. LLM for Planning
(hierarchical planning, retrieval-augmented)
3. LLM for Acting
(structured input)

Thanks!