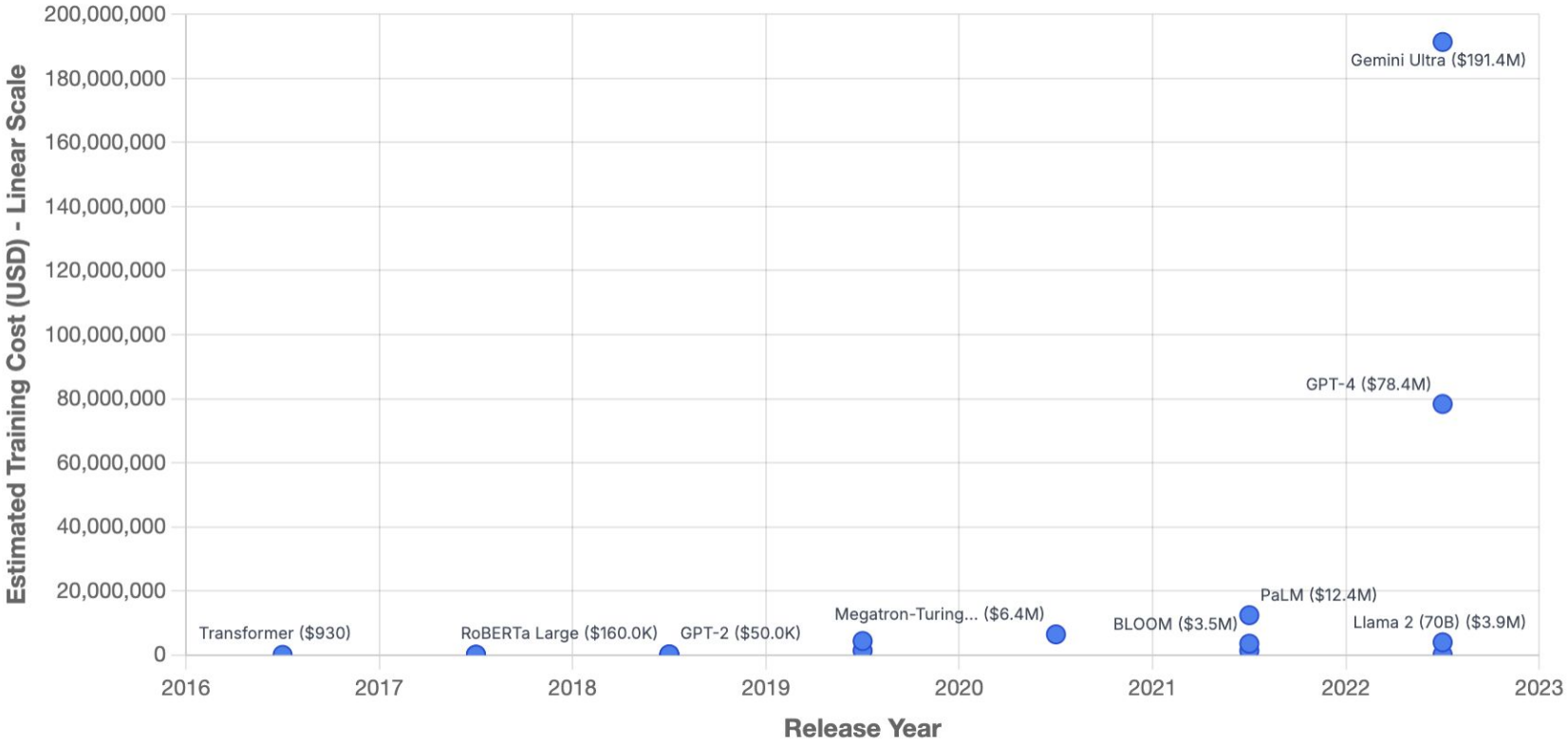
LLM for Robotics

Wenhao Yu
05/12/2025

A crash course on LLM, by LLM

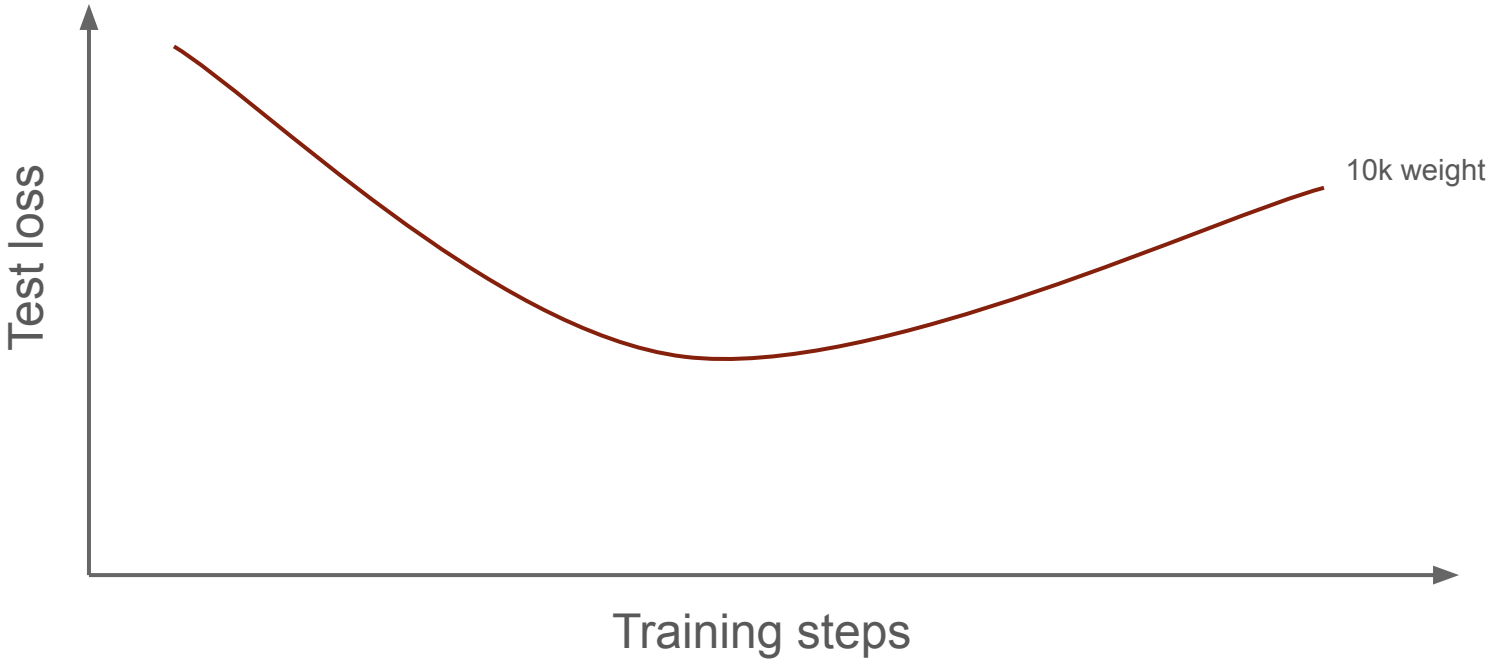
- https://gemini.google.com/
- https://aistudio.google.com/live
- Create an interactive and fun web-based tutorial about LLMs for undergrad students in CS with basic understanding of machine learning, but no prior experience with LLM theories. Provide some cool interactive visualizations whenever possible. Break it down into intro, tokenization, transformer & attention, training, and applications.

A puzzle about $

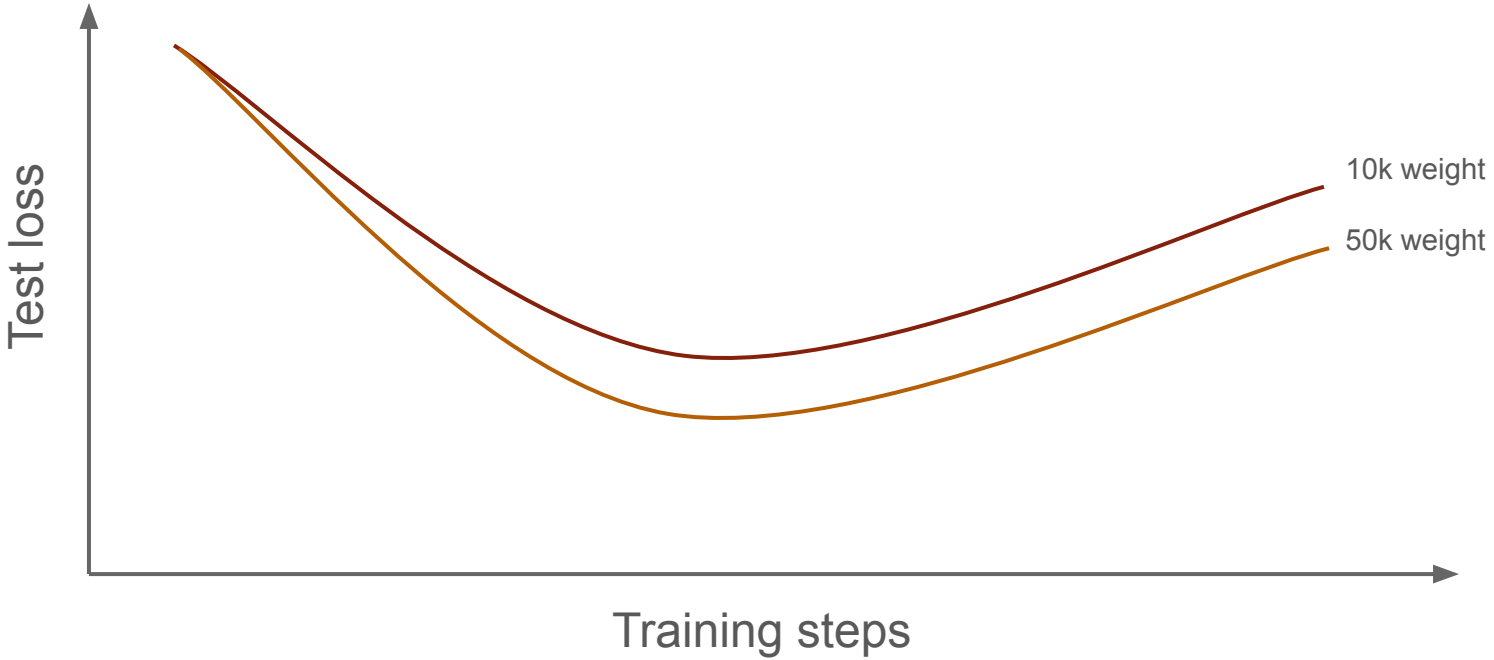


Scaling Laws

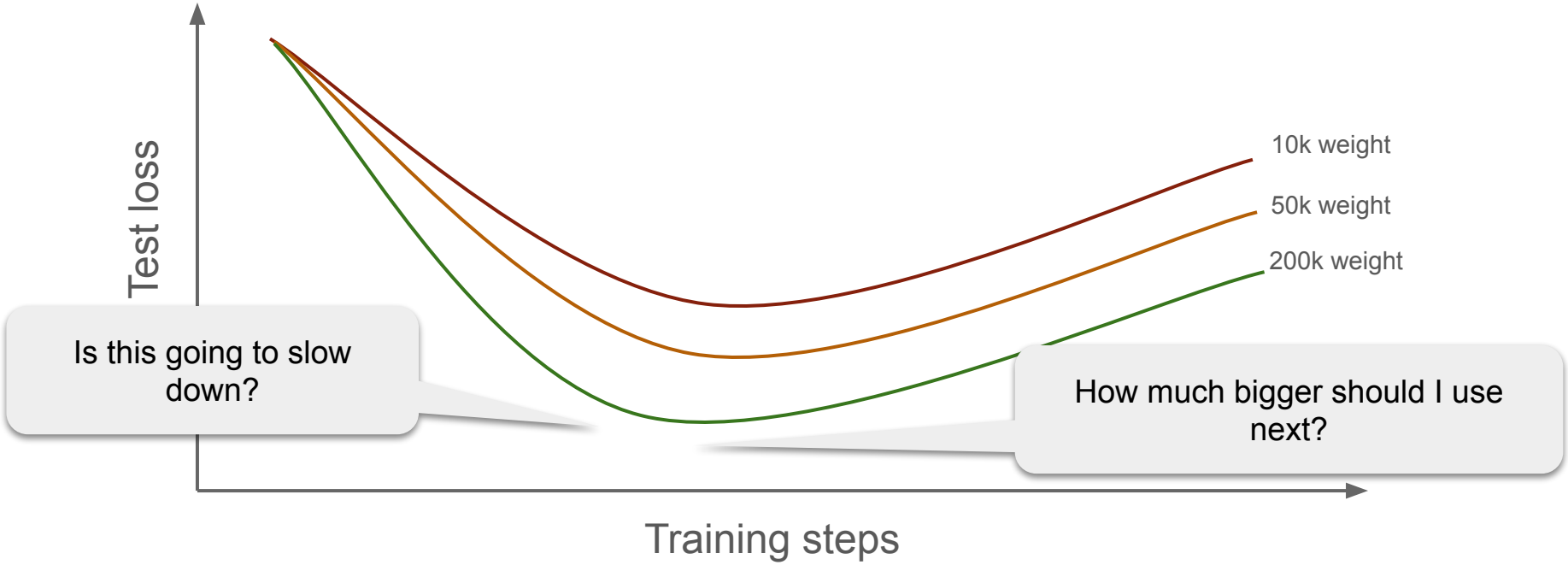
Scaling Laws



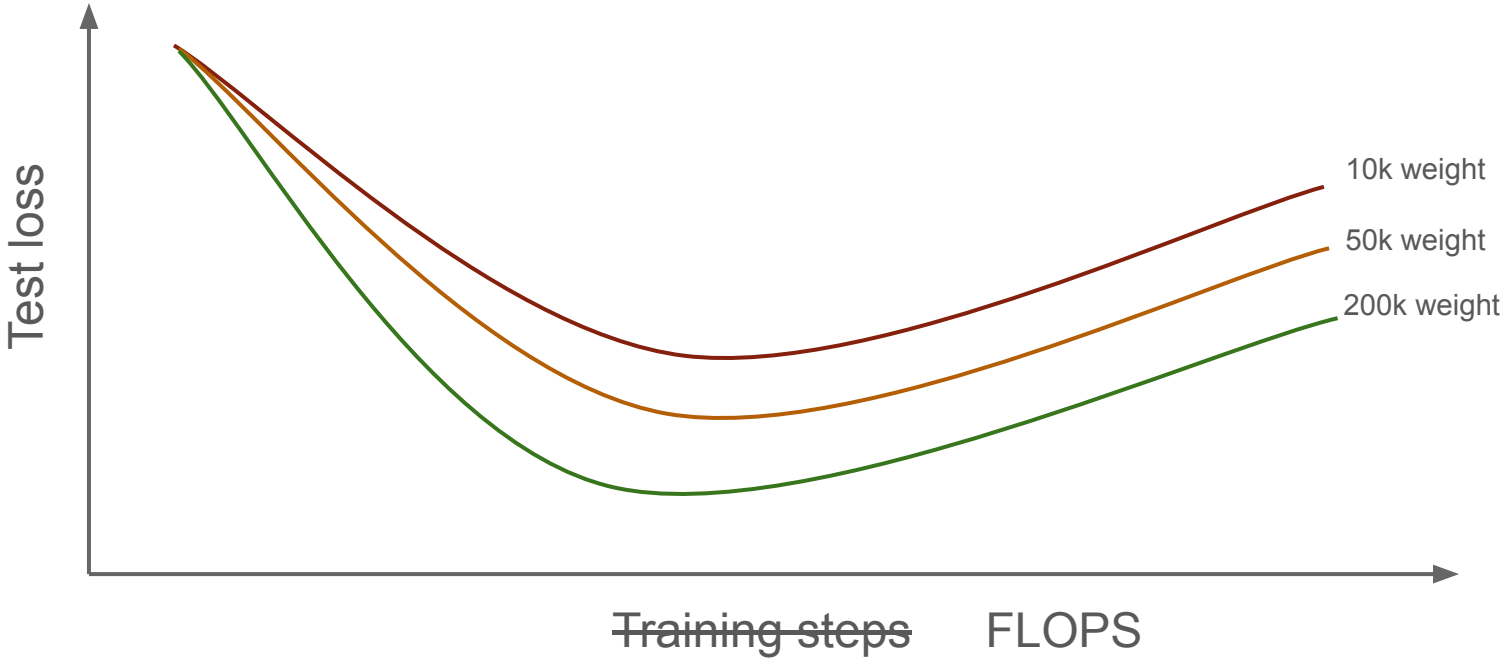
Scaling Laws



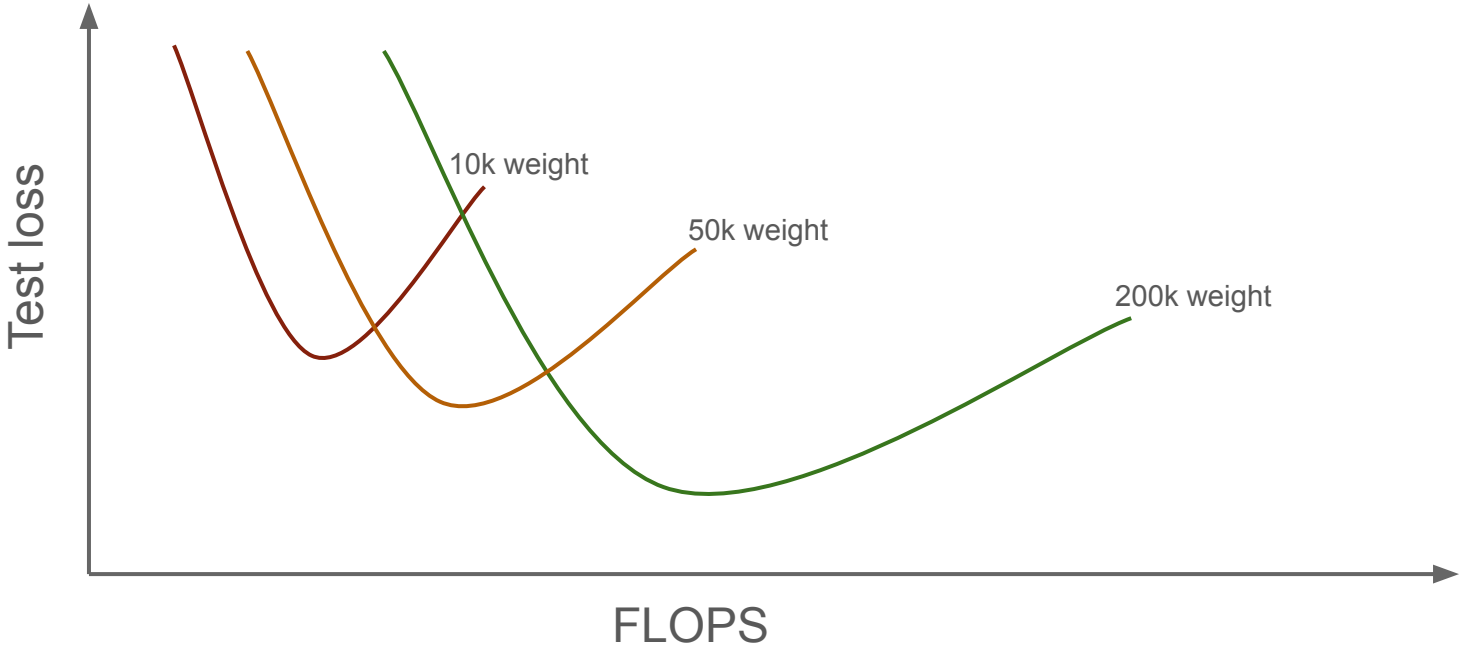
Scaling Laws



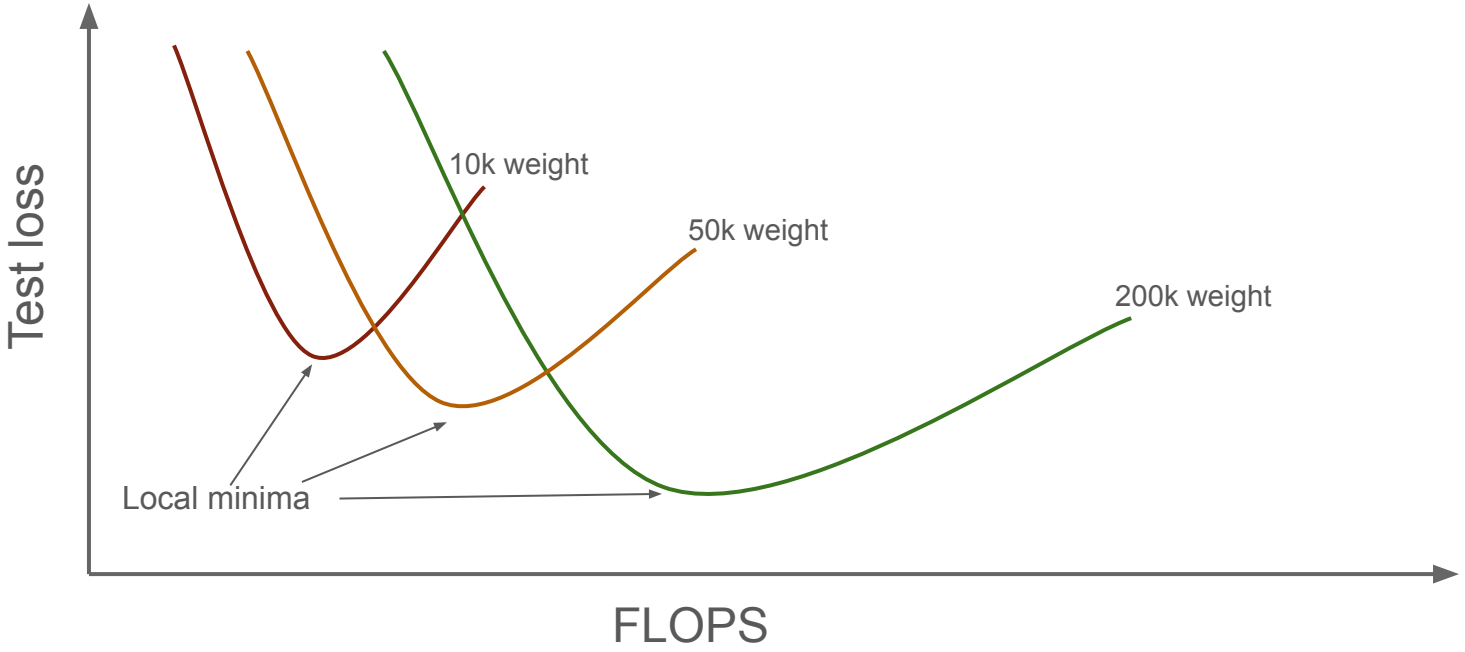
Scaling Laws



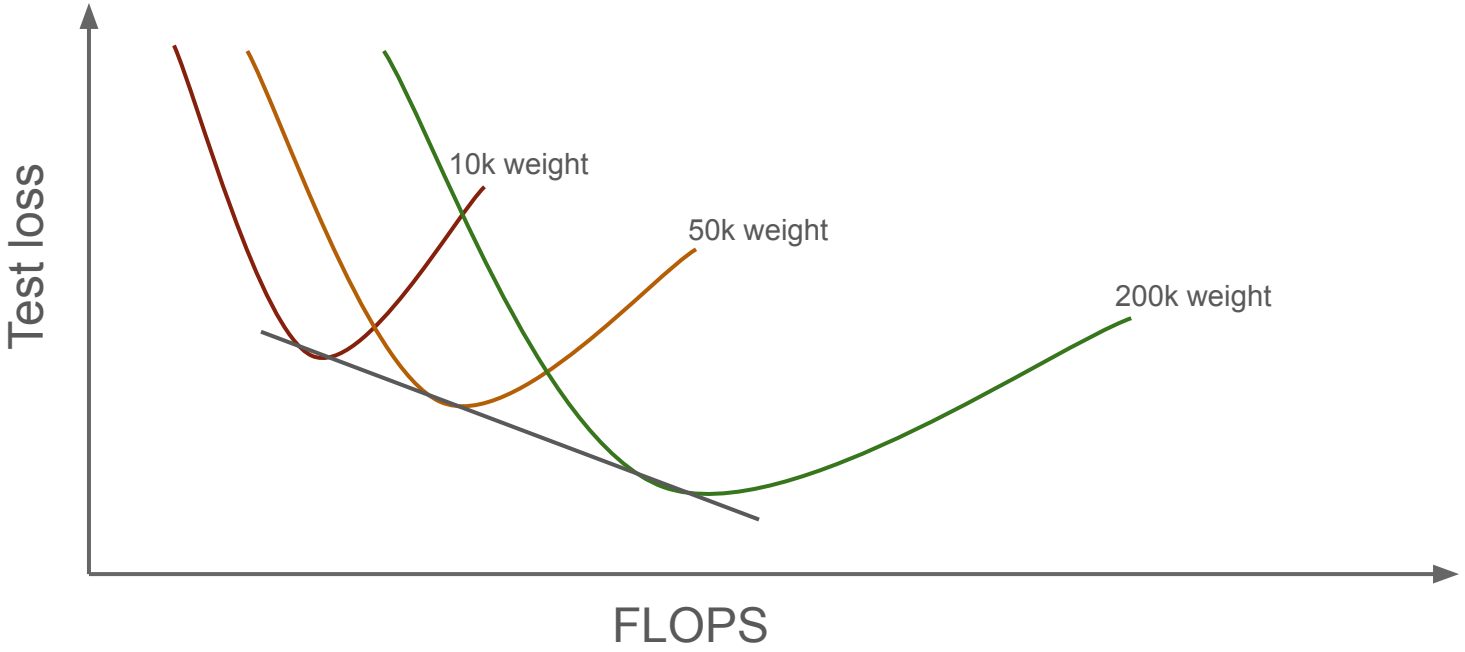
Scaling Laws



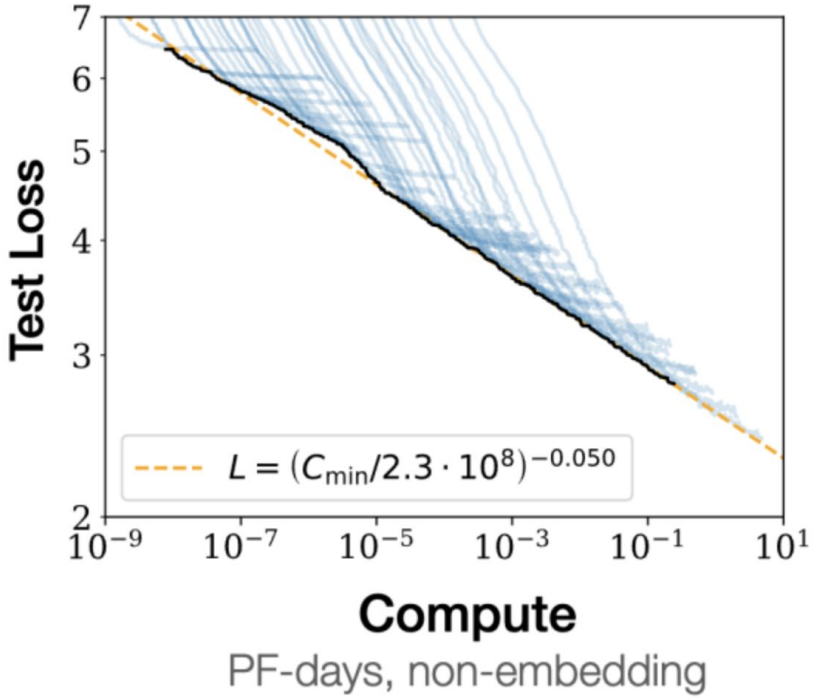
Scaling Laws



Scaling Laws

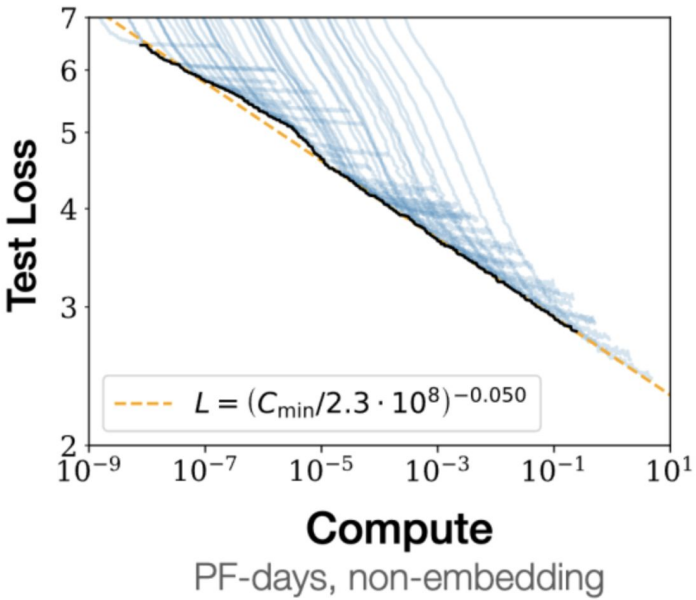


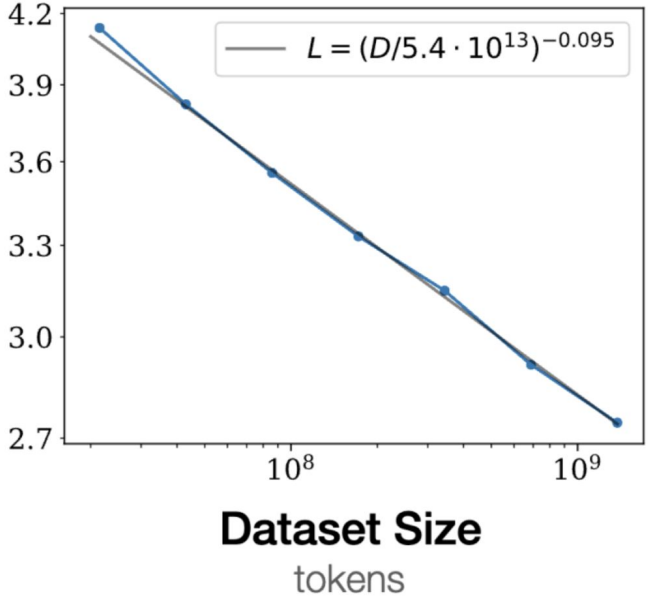
Scaling Laws



Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., ... & Amodei, D. (2020). Scaling laws for neural language models. arXiv preprint arXiv:2001.08361.

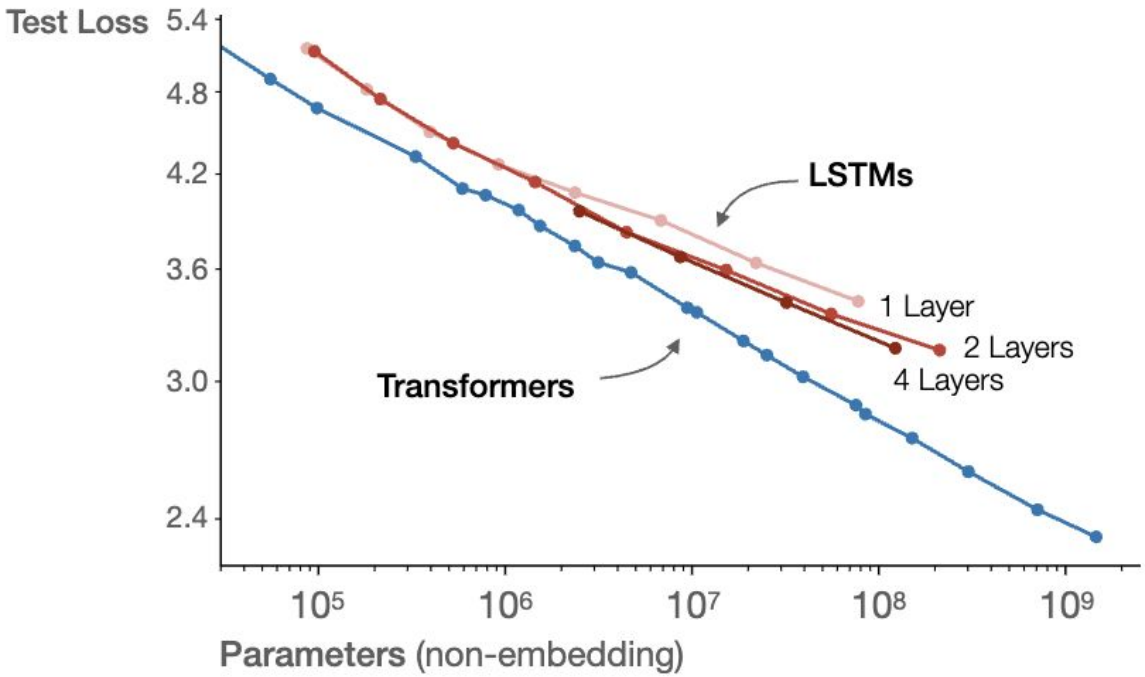
Scaling Laws



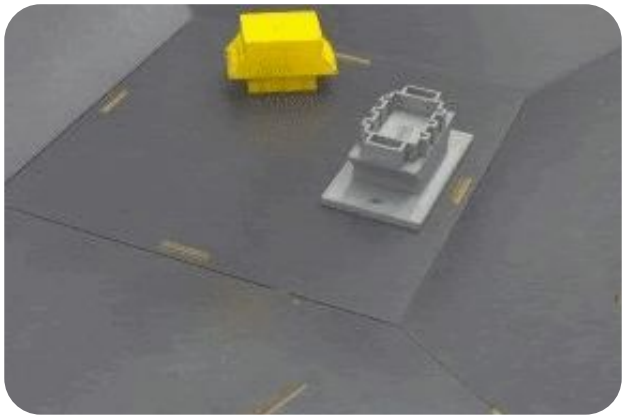


Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., ... & Amodei, D. (2020). Scaling laws for neural language models. arXiv preprint arXiv:2001.08361.

Scaling Laws



Why LLM for robotics?

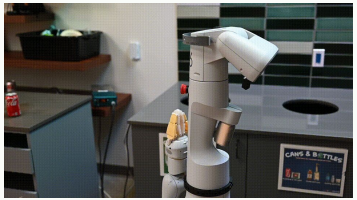


We have made long strides in *Robot Dexterity and Agility*.

However, current models cannot go much beyond the data it has seen, and collecting robot data isn't trivial.

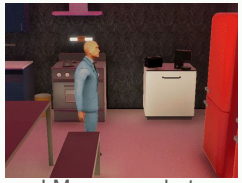
How to apply LLM to robotics?

Planner



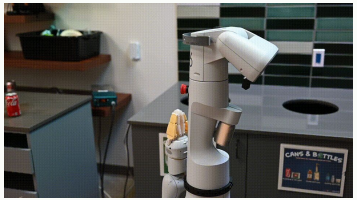
SayCan, 2022

GPT-4V(ision) for Robotics, 2023



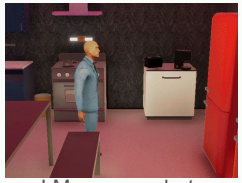
LM as zero-shot planners, 2022

Planner



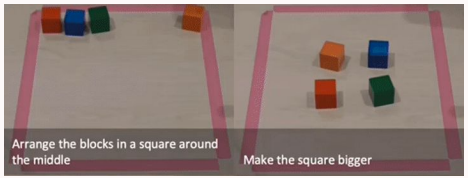
SayCan, 2022

GPT-4V(ision) for Robotics, 2023

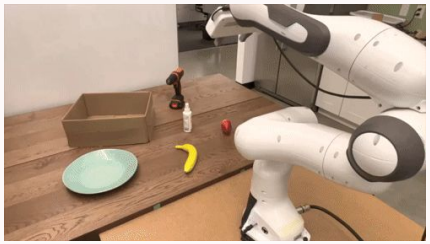


LM as zero-shot planners, 2022

Coder

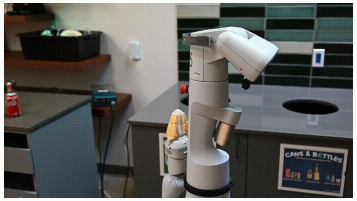


Code-as-policies, 2022



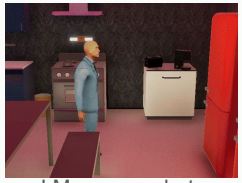
ProgPrompt, 2023

Planner



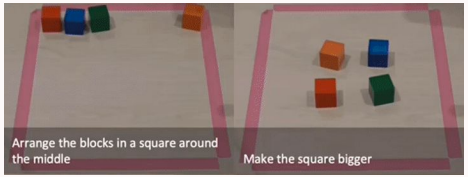
SayCan, 2022

GPT-4V(ision) for Robotics, 2023

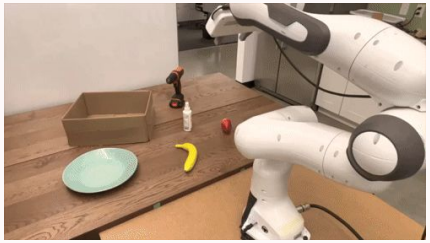


LM as zero-shot planners, 2022

Coder



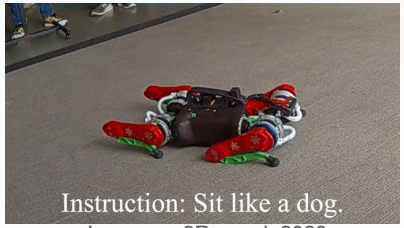
Code-as-policies, 2022



ProgPrompt, 2023

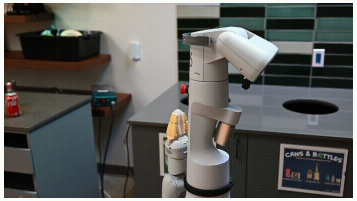
Teacher

Eureka, 2023



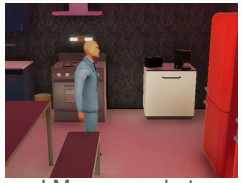
Language2Reward, 2023

Planner



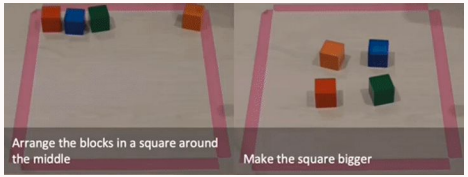
SayCan, 2022

GPT-4V(ision) for Robotics, 2023

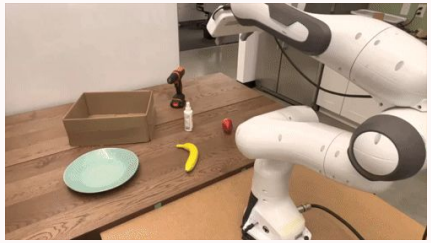


LM as zero-shot planners, 2022

Coder



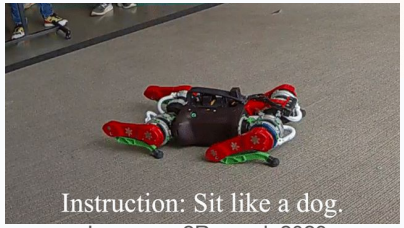
Code-as-policies, 2022



ProgPrompt, 2023

Teacher

Eureka, 2023



Language2Reward, 2023

Controller

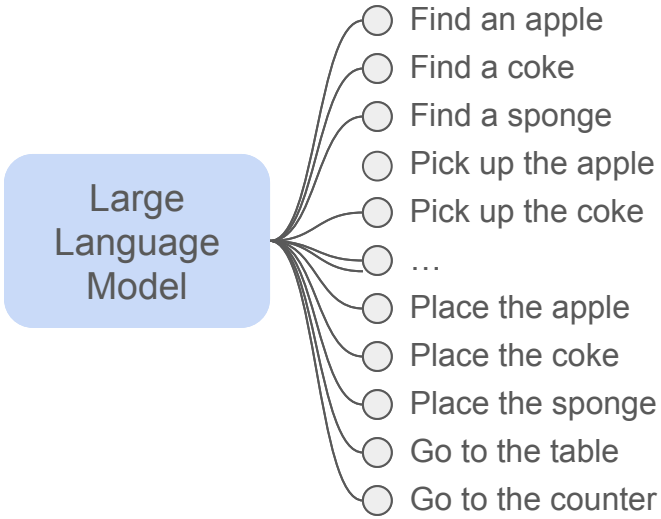
Gemini Robotics, 2025

π0.5, 2025

LLM as Planner

- Find an apple
- Find a coke
- Find a sponge
- Pick up the apple
- Pick up the coke
- …
- Place the apple
- Place the coke
- Place the sponge
- Go to the table
- Go to the counter

LLM as Planner



LLM as Planner

You are a helpful robot with following skills: “Find an apple”, “Find a coke”, …

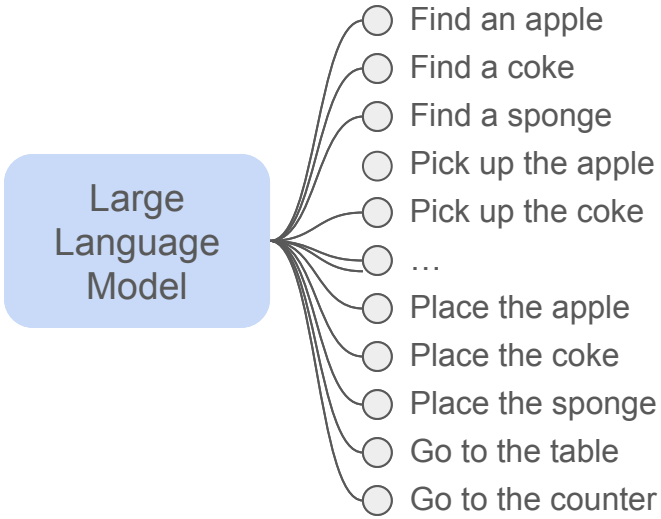
How would you move the coffee to the counter?

1. find a coffee cup, 2. pick up the coffee cup, 3. go to counter, 4. put down the coffee cup, 5. done.

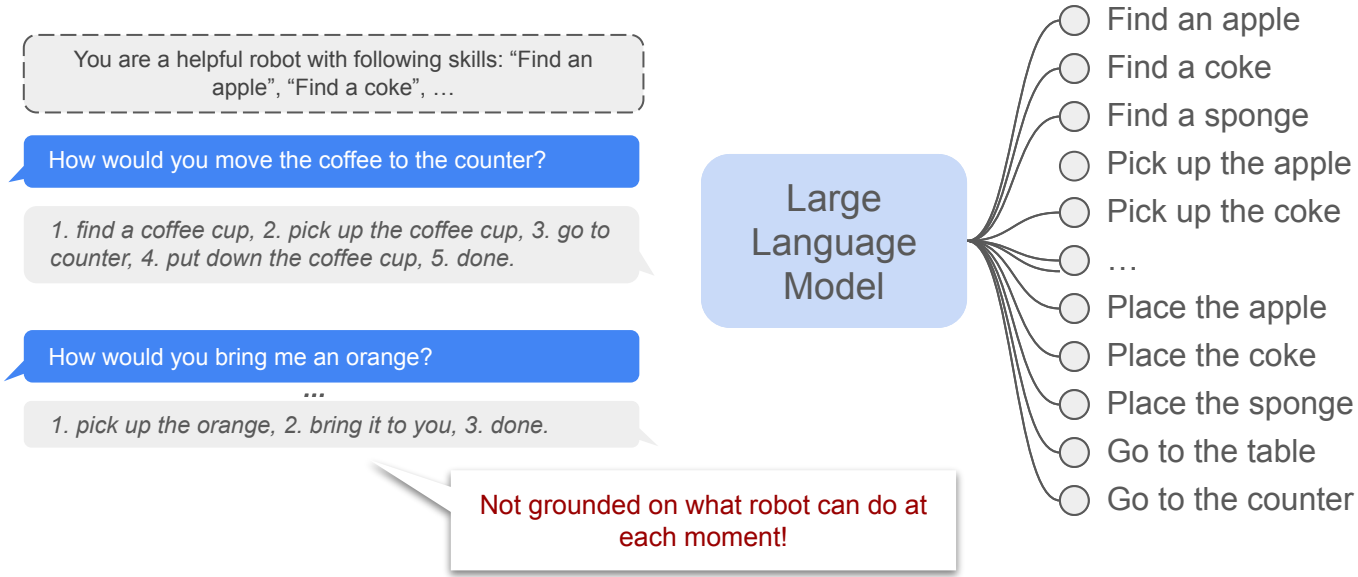
How would you bring me an orange?

...

1. pick up the orange, 2. bring it to you, 3. done.



LLM as Planner



How to ground the plan

How to ground the plan

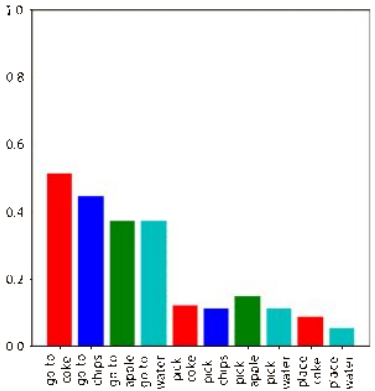
- Value function $V^{\pi}(s_t, c) = \mathbb{E}(R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots)$

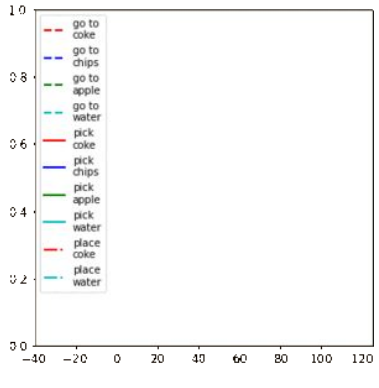
"Pick up coke can"

Total reward the robot can get

How to ground the plan

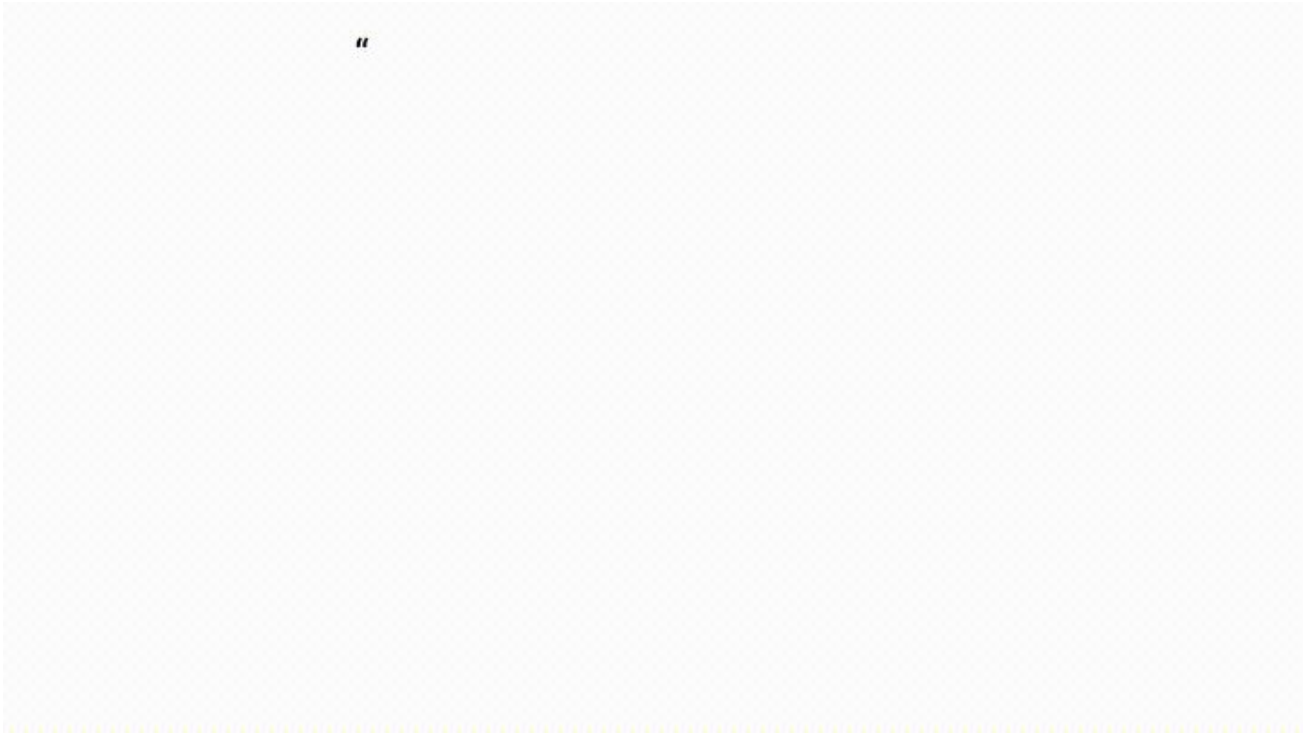
- Value function $V^{\pi}(s_t, c) = \mathbb{E}(R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots)$





How to ground the plan

- Value function $V^{\pi}(s_t, c) = \mathbb{E}(R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots)$



How to ground the plan

- Value function $V^{\pi}(s_t, c) = \mathbb{E}(R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots)$

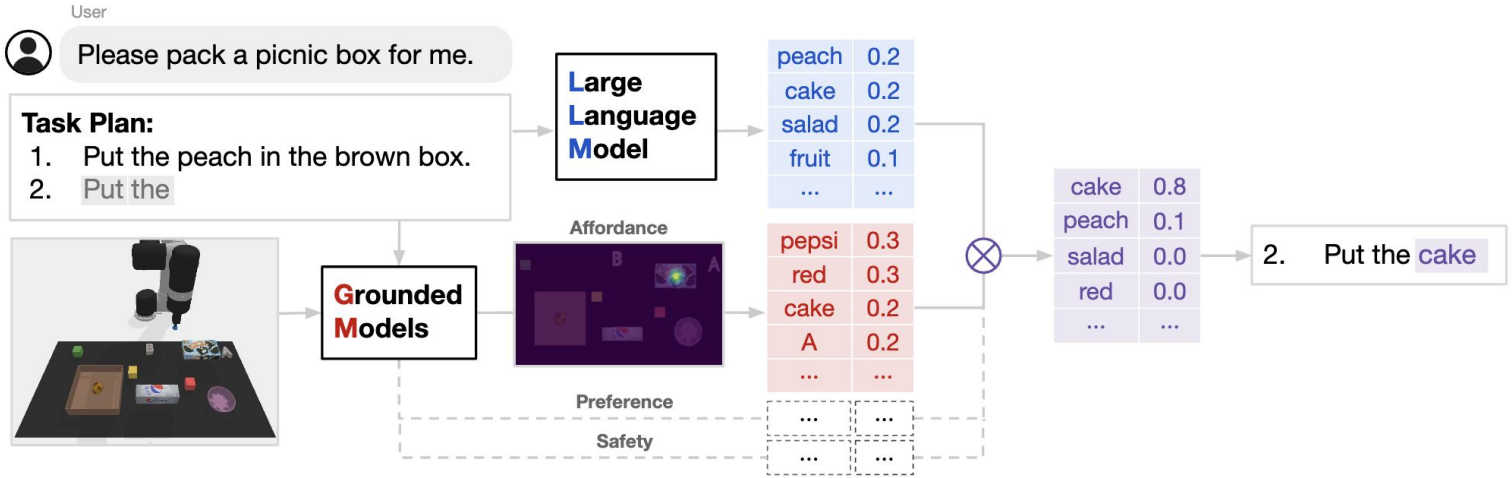
- Fine-tune a LLM/VLM to predict the affordance

[Grounded Decoding: Guiding Text Generation with Grounded Models for Embodied Agents. Huang et al. 2023]

How to ground the plan

- Value function $V^{\pi}(s_t, c) = \mathbb{E}(R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots)$

- Fine-tune a LLM/VLM to predict the affordance



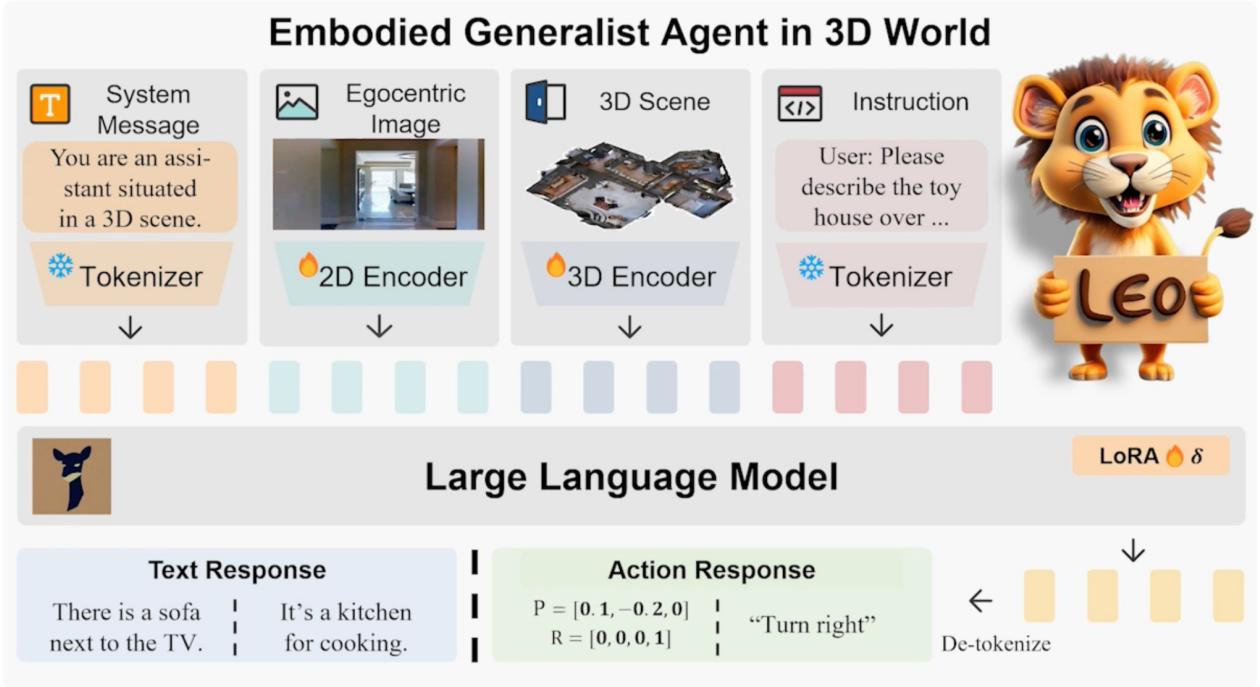
[Grounded Decoding: Guiding Text Generation with Grounded Models for Embodied Agents. Huang et al. 2023]

How to ground the plan

- Value function $V^{\pi}(s_t, c) = \mathbb{E}(R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots)$
- Fine-tune a LLM/VLM to predict the affordance
- Create Embodied foundation models

How to ground the plan

- Value function $V^{\pi}(s_t, c) = \mathbb{E}(R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots)$
- Fine-tune a LLM/VLM to pre
- Create Embodied foundatio



[An Embodied Generalist Agent in 3D World. Huang et al. 2023]

How to ground the plan

Dialogue(O-CoT):
Dialogue Context: high level task: organize the bedroom. low level task: check some objects.
Human: I need help organizing my bedroom. Can you assist me? Thought: no specific objects
Assistant: Of course! What specifically do you need help with?
Human: Can you check if there are any clean clothes hanging on the wall? Thought: clothes-13
Assistant: I'm sorry, but the clothes hanging on the wall are messy.
Human: Thanks for your help! Thought: no specific object
Assistant: You're welcome! If you need any further assistance, feel free to ask.

Scene Caption:
In this room, there is a wooden floor that is clean and flat. A tall wardrobe stand on the right side of a desk, close to a basket. The wardrobe is in front of a chair, a bed, and behind a bedside table. Clothes are hanging on a white wall, to the right of a closed curtain. The bed is wide, clean, and covered with blue blanket. The room has a comfortable and organized setting with functional furniture.

Object Scene Caption:
The showcase is supported by the wall and positioned behind, close to, and to the left of the shelf. The wall is flat, architectural, and dark in color. The shelf is made of wood and has a low size.

QA(O-CoT):
Question: Where is the printer located? Thought: printer-8 Answer: standing on the desk
Question: How many blankets are on the bed? Thought: blanket-16, blanket-17 Answer: 2
Question: What is the type of the room? Thought:wardrobe-2, desk-7, chair-11, bed-15 Answer: bedroom

Planning:
High-Level Task: Organize and tidy up the bedroom.
Low-Level Actions:
1. Clean the floor by sweeping to remove any dirt.
2. Make the bed by arranging the blanket and pillows.
3. Place any loose items or belongings into the basket.
4. Arrange items on the shelves and showcase in a tidy way.

LLM as Coder

- Many tasks can be hard to express in language-only
- LLMs have great coding abilities

LLM as Coder

- Find an apple
- Find a coke
- Find a sponge
- Pick up the apple
- Pick up the coke
- …
- Place the apple
- Place the coke
- Place the sponge
- Go to the table
- Go to the counter

LLM as Coder

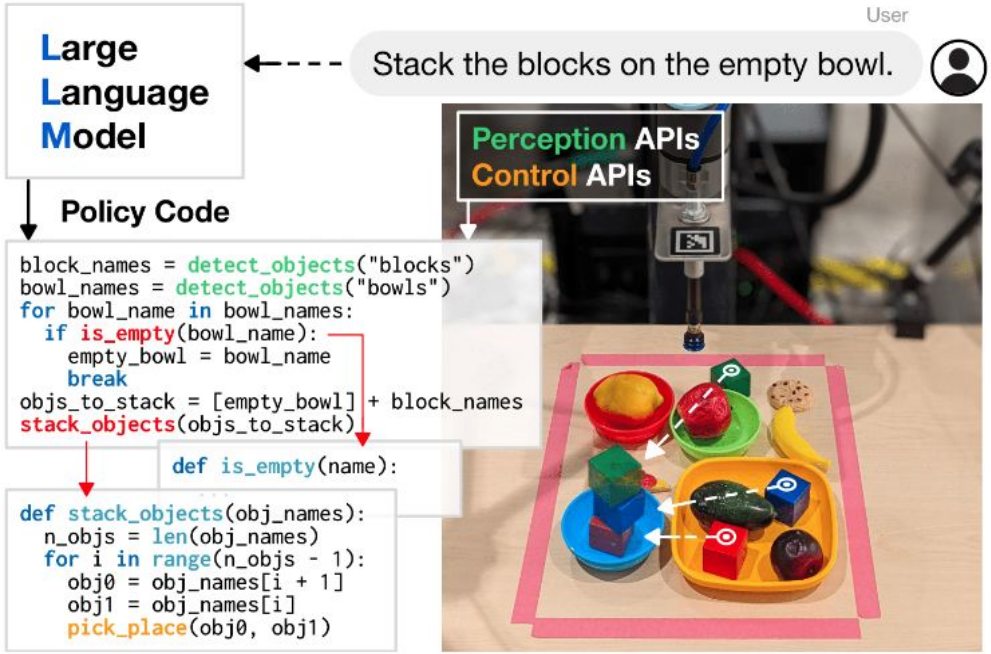
- find(X)
- pick_up(X)
- place(X)
- go_to(X)

action API

- detect_pos(O)
- detect_bbox(O)

perception API

LLM as Coder



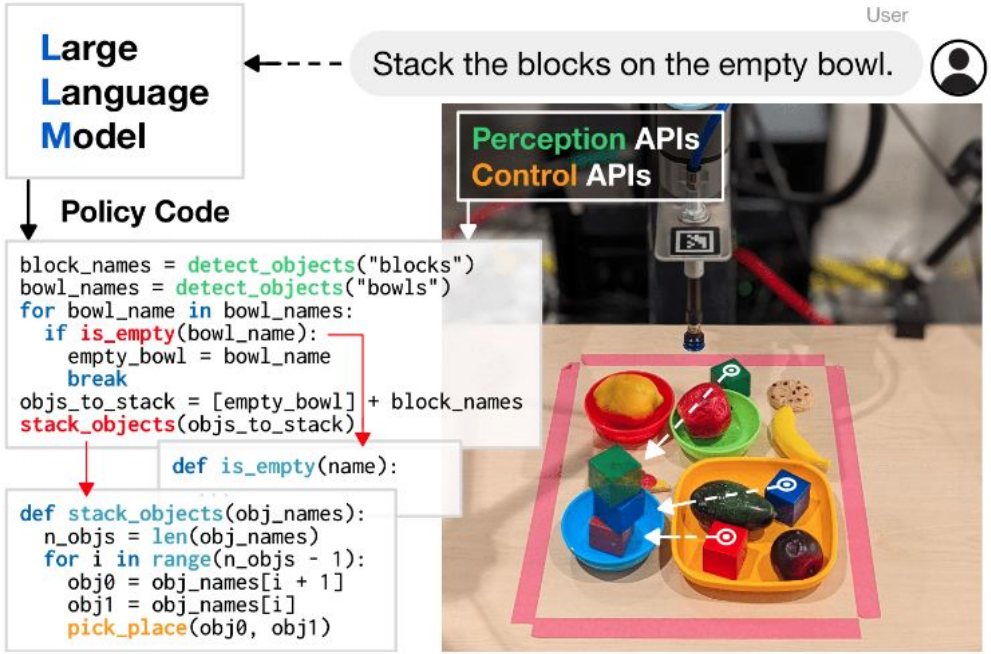
- find(X)
- pick_up(X)
- place(X)
- go_to(X)

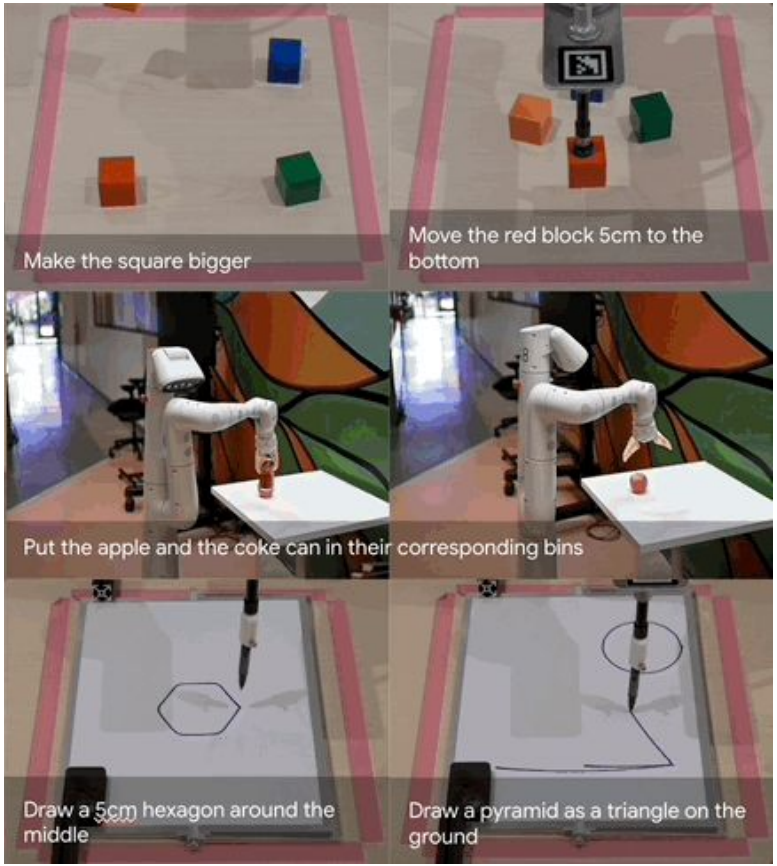
action API

- detect_pos(O)
- detect_bbox(O)

perception API

LLM as Coder





[Code as Policies. Liang et al. 2022]

LLM as Coder

- Challenge: many LLMs lack embodied reasoning ability

LLM as Coder

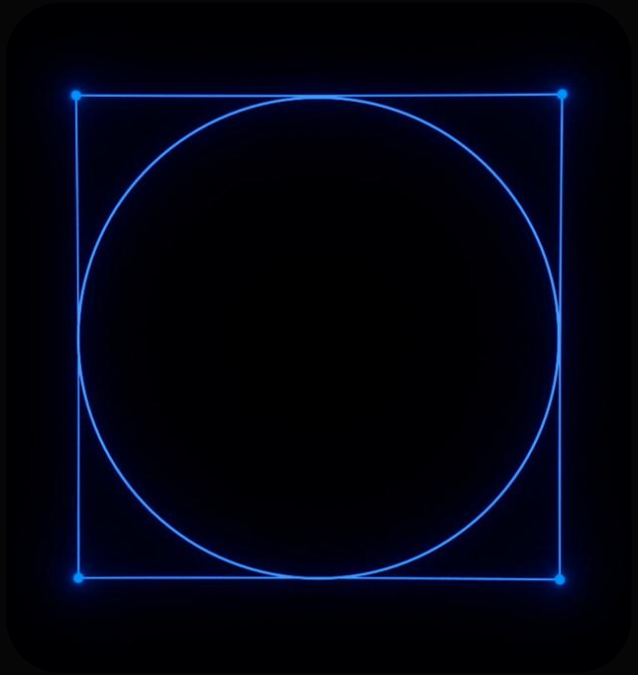
- Challenge: many LLMs lack embodied reasoning ability
- Mitigations:
  - Prompt engineering

```
import numpy as np
from shapely.geometry import *
from shapely.affinity import *
from env_utils import parse_loc_name, parse_obj_name, get_visible_obj_names, get_loc_names, get_obj_pos, get_loc_pos, get_robot_pos

# a line with 5 points from the robot to the chair with no arms.
start_pos = get_robot_pos()
chair_name = parse_loc_name('the chair with no arms', f'objects = {get_loc_names()}')
end_pos = get_loc_pos(chair_name)
line = make_line(start=start_pos, end=end_pos)
points_3d = interpolate_pts_on_line(line=line, n=2)
ret_val = points_3d
# a point 0.1m left of the bowls.
bowl_names = parse_obj_name('the bowls', f'objects = {get_visible_obj_names()}')
bowl_positions = np.array([get_obj_pos(bowl_name) for bowl_name in bowl_names])
left_obj_pos = bowl_positions[np.argmin(bowl_positions[:, 0])] + [-0.1, 0, 0]
ret_val = left_obj_pos
# a point 0.2m in behind the banana.
banana_name = parse_obj_name('the banana', f'objects = {get_visible_obj_names()}')
ret_val = get_obj_pos(banana_name) + [0, 0.2, 0]
# a 1m square around the table with 4 points.
table_name = parse_loc_name('the table', f'objects = {get_loc_names()}')
table_pos_2d = get_loc_pos(table_name)[:2]
shape_2d = make_square(size=1, center=table_pos_2d)
points_2d = get_points_from_polygon(shape_2d)
points_3d = np.c_[points_2d, np.zeros(len(points_2d))]
ret_val = points_3d
# a 1.5m circle with centered around the lounge chair with 20 points.
lounge_chair_name = parse_loc_name('the lounge chair', f'objects = {get_loc_names()}')
lounge_chair_pos = get_loc_pos(lounge_chair_name)
shape_2d = make_circle(radius=1.5, center=lounge_chair_pos)
points_2d = interpolate_pts_along_exterior(exterior=shape_2d.exterior, n=20)
points_3d = np.c_[points_2d, np.zeros(len(points_2d))]
ret_val = points_3d
```

LLM as Coder

- Challenge: many LLMs lack embodied reasoning ability
- Mitigations:
  - Prompt engineering
  - Create Embodied foundation models

Gemini Robotics- ER Embodied Reasoning



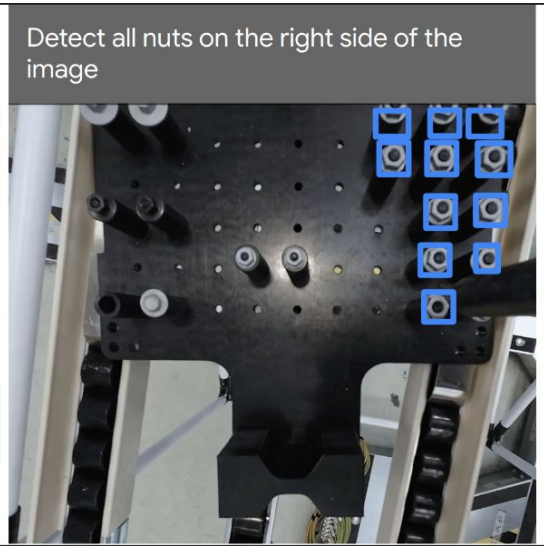
Advanced vision-language-model

Enhances Gemini's pointing and 3d detection

New capabilities of trajectory generation and grasp prediction

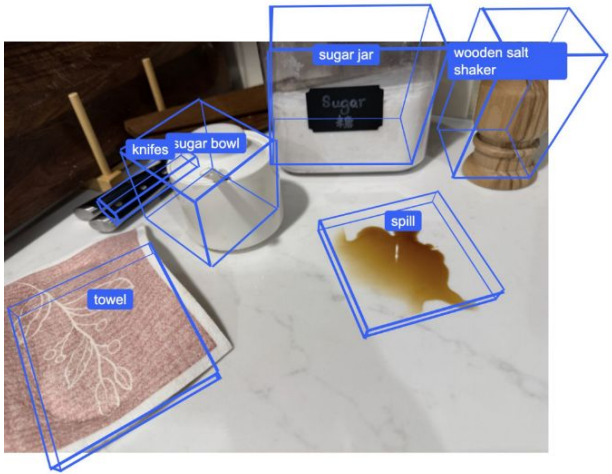
2x-3x success rate compared to Gemini 2.0

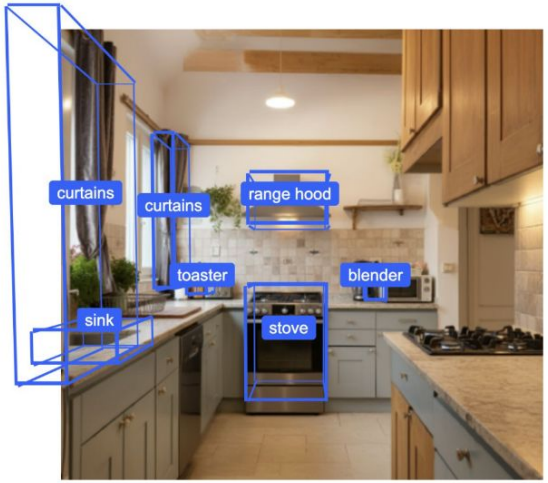
Embodied Reasoning: 2D bounding boxes

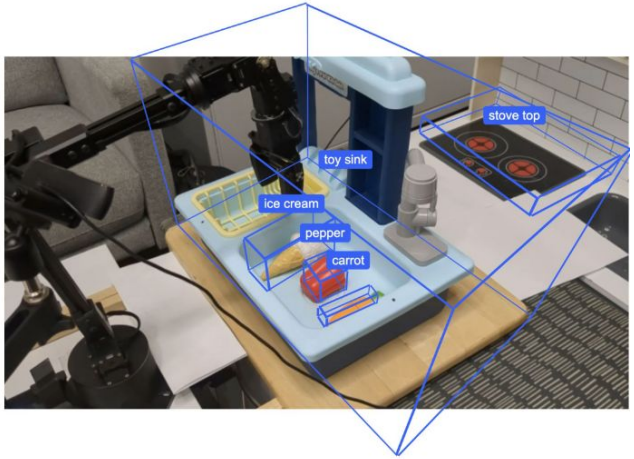


Embodied Reasoning: 3D bounding boxes

Find the 3D bounding boxes of sugar bowl, sugar jar, wooden salt shaker, knifes, spill, towel.	Detect the 3D bounding boxes of blender, toaster, curtains, sink, range hood, stove.	Detect the 3D bounding boxes of stove top, toy sink, ice cream, pepper, carrot.

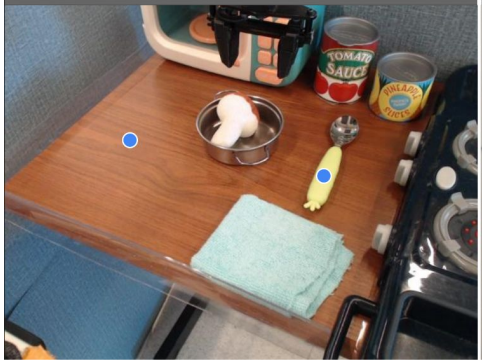






Embodied Reasoning: Pointing

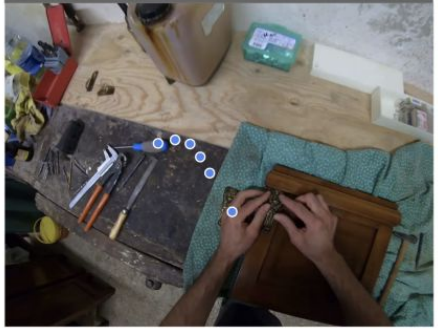
Point to the spoon handle, an empty area on the table left of the pan



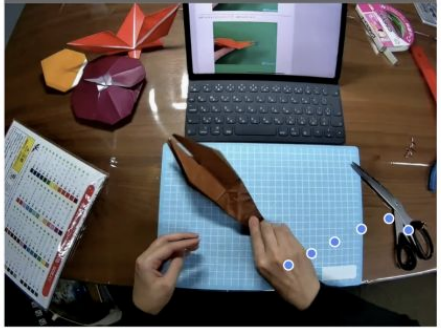
Point to all 8 cans, and where a 9th can would be placed following the grid pattern

Point to where a human would grasp this and pick it up

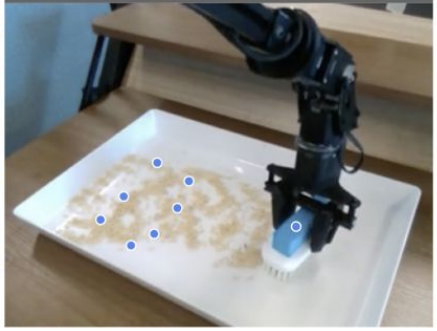
Point to the left hand and the handle of the blue screwdriver, and a trajectory of 6 points connecting them.



Point to the right hand and the handles of the scissor ... and a trajectory of 5 points from the right hand to the handles of the scissor



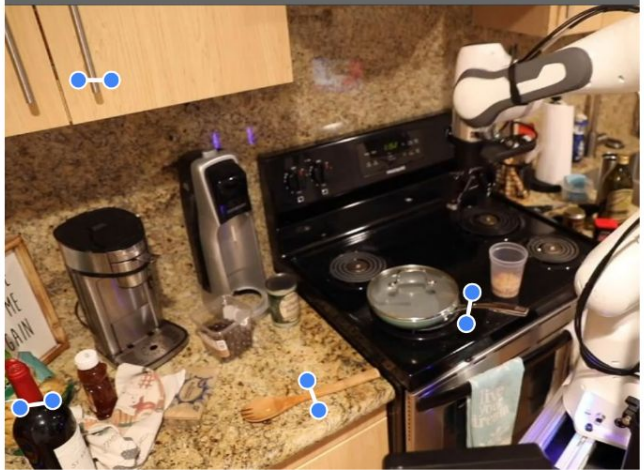
Point to the blue brush and a list of points covering the region of particles



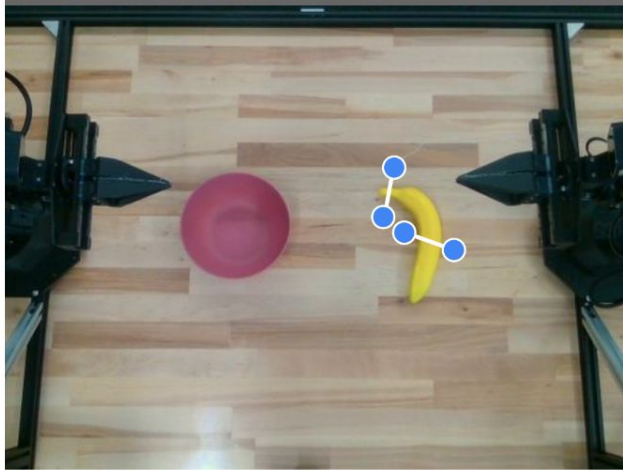
Embodied Reasoning: Grasp angles

Find the grasp points and grasp angles on the stapler handle, tape roll, finger holes of the scissors, dark blue pen, rim of the black square tray.

Find the grasp points and grasp angles on the wooden spoon handle, pan handle, neck of the wine bottle, and right handle of the cupboard.

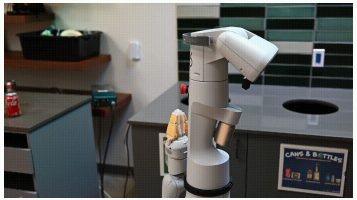


Find the grasp points and grasp angles on the stem and the middle of the banana.



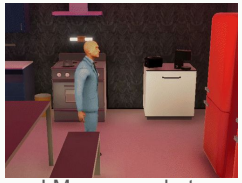
Combining these capabilities with code generation to execute on robot

Planner



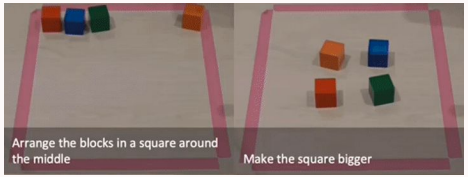
SayCan, 2022

GPT-4V(ision) for Robotics, 2023

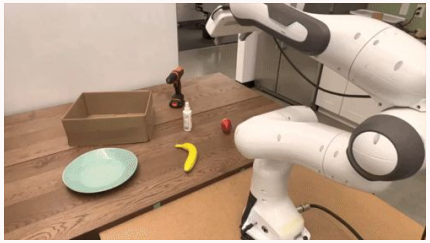


LM as zero-shot planners, 2022

Coder



Code-as-policies, 2022



ProgPrompt, 2023

Assumes low-level skills already exists through policy or API.

LLM as Teacher

- Let LLM to ‘teach’ the robot new skills by providing guidance/feedback.

LLM as Teacher

User

Stand up on two feet.

Large Language Model (LLM)

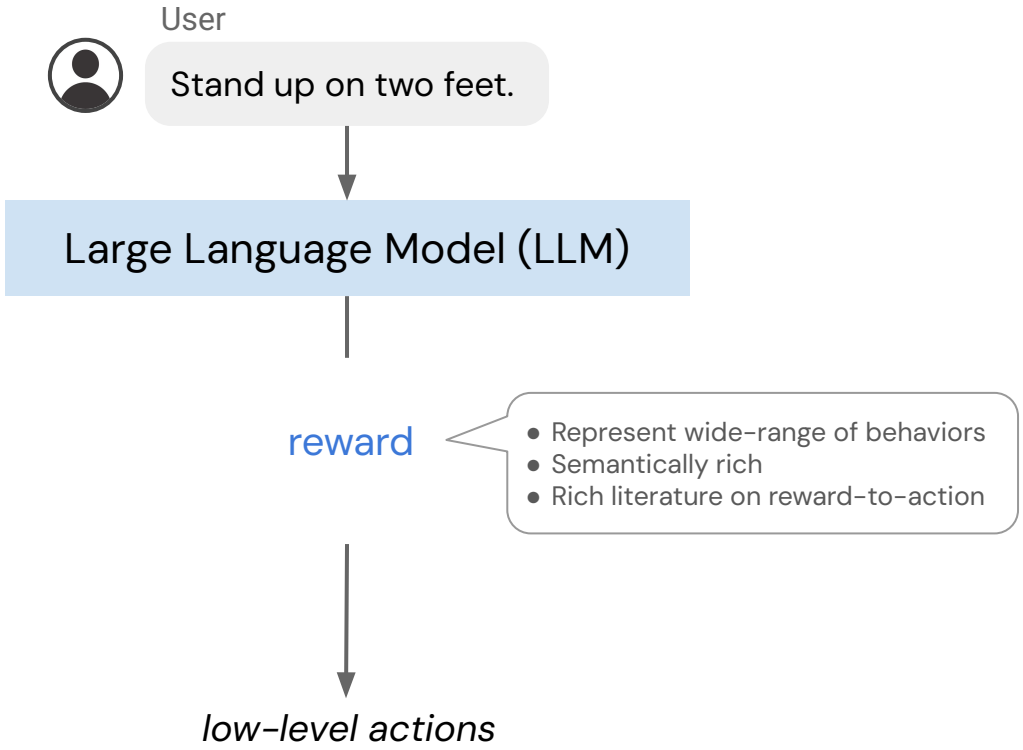
?

code?

plan?

low-level actions

Reward as interface



Reward as interface

- How to represent reward:

```
# Set torso rewards
set_torso_rewards(height=0.7, pitch=np.deg2rad(90))

# Set feet rewards
set_feet_pos_rewards('front_left', height=0.7)
set_feet_pos_rewards('back_left', height=0.0)
set_feet_pos_rewards('front_right', height=0.7)
set_feet_pos_rewards('back_right', height=0.0)
```

Reward API [Language-to-reward, 2023]

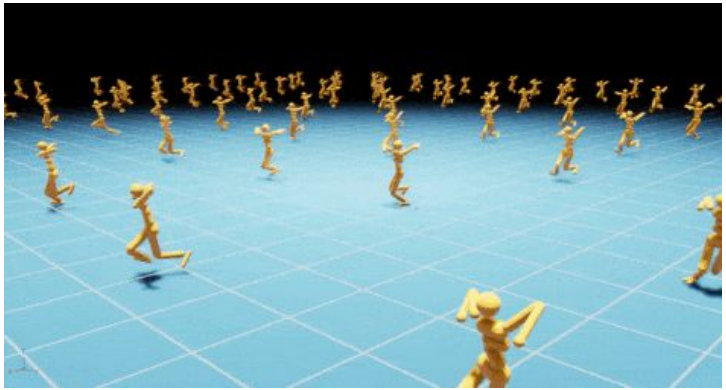
```python
def compute_reward(object_rot, goal_rot, object_angvel, object_pos, fingertip_pos):
    # Rotation reward
    rot_diff = torch.abs(torch.sum(object_rot * goal_rot, dim=1) - 1) / 2
    rotation_reward_temp = 20.0
    rotation_reward = torch.exp(-rotation_reward_temp * rot_diff)

    # Distance reward
    min_distance = torch.min(torch.norm(fingertip_pos - object_pos[:, None], dim=2), dim=1).values
    distance_reward = min_distance

    total_reward = rotation_reward + distance_reward

    reward_components = {
        "rotation_reward": rotation_reward,
        "distance_reward": distance_reward
    }

    return total_reward, reward_components
```

Reward function code [Eureka, 2024]

Reward as interface

- How to represent reward:
  - Reward APIs
  - Reward function code
  - ...
- How to get actions

Real-time MPC

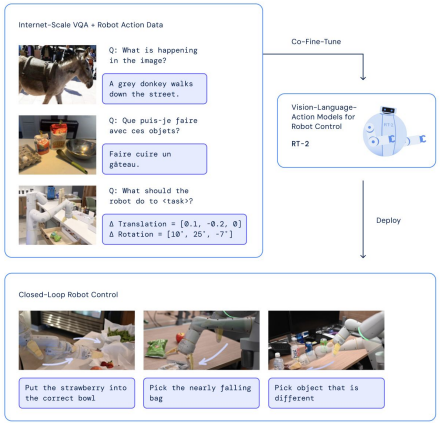

Reinforcement Learning

LLM as Teacher

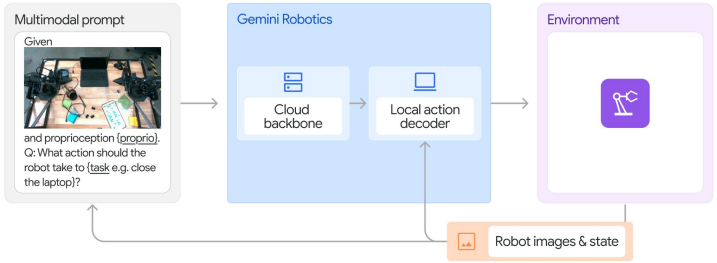
[Language-to-reward, 2023]

[DrEureka, 2024]

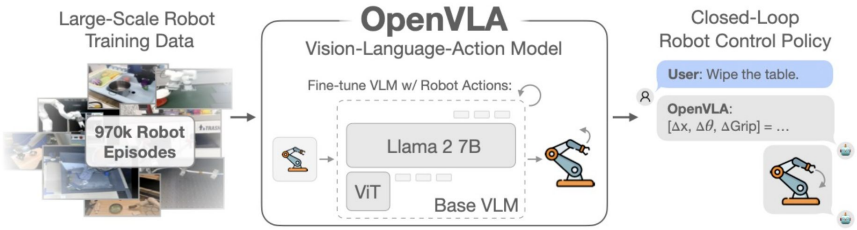
LLM as Controller (VLA)



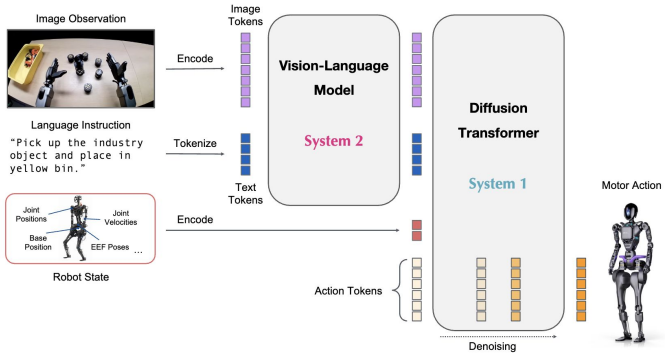
RT-2, 2023



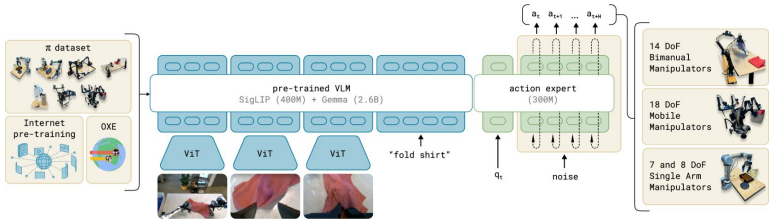
Gemini Robotics, 2025



OpenVLA, 2024

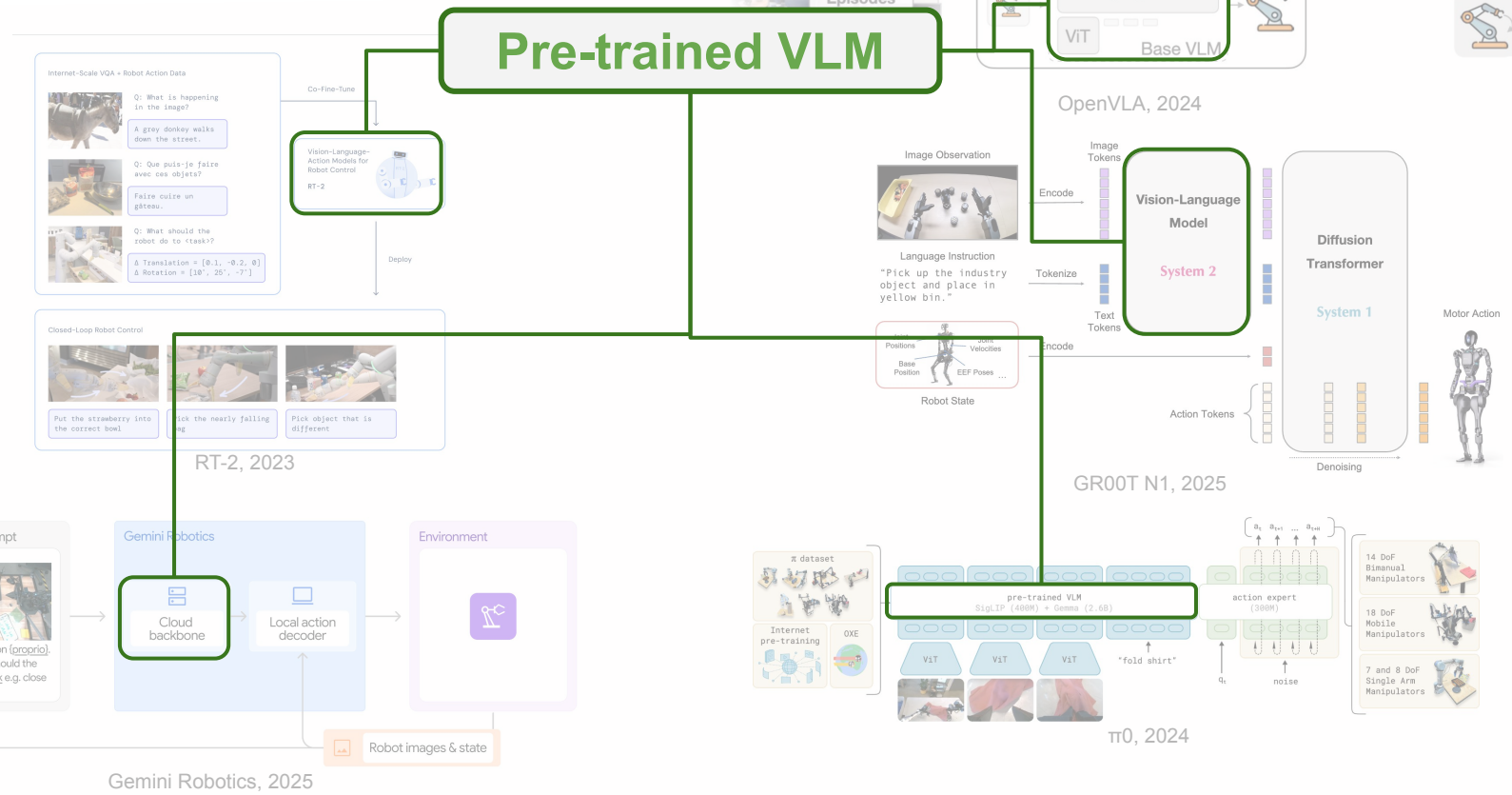


GR00T N1, 2025



π0, 2024

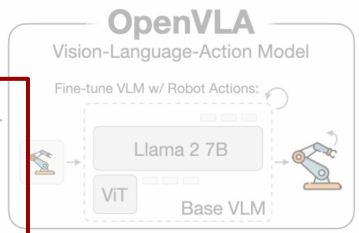
LLM as Controller (VLA)

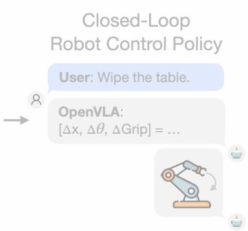


LLM as Controller (VLA)

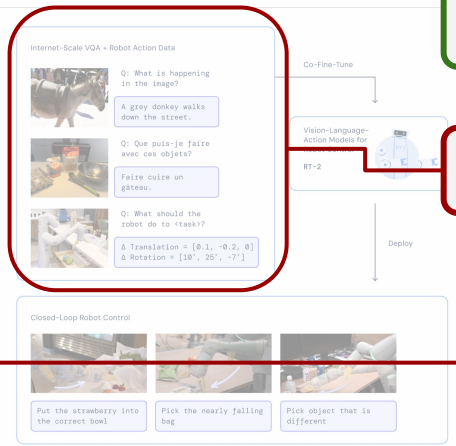
Pre-trained VLM

Large-scale dataset

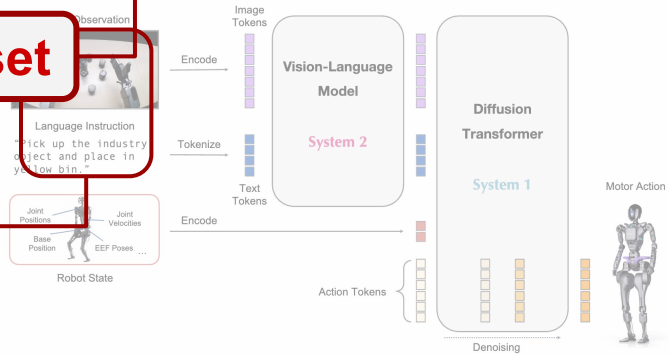




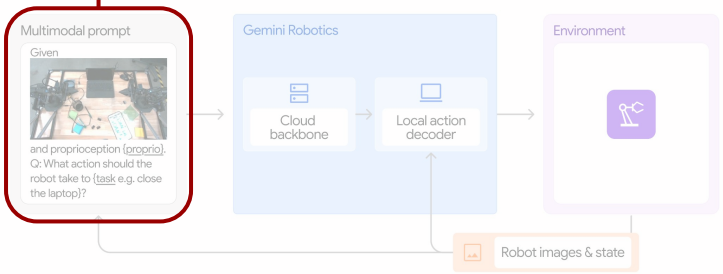
OpenVLA, 2024



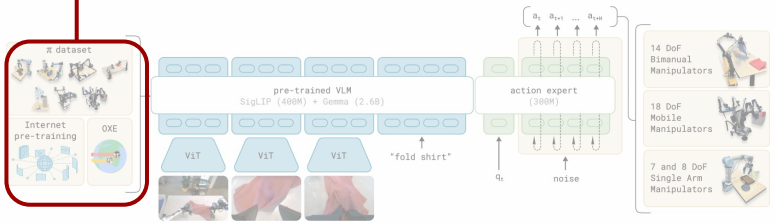
RT-2, 2023



GR00T N1, 2025



Gemini Robotics, 2025

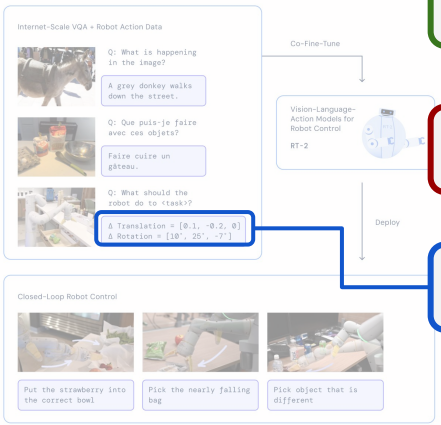


π0, 2024

LLM as Controller (VLA)

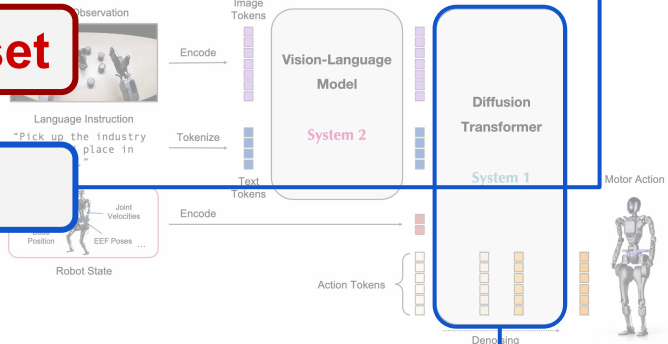
OpenVLA, 2024

Pre-trained VLM



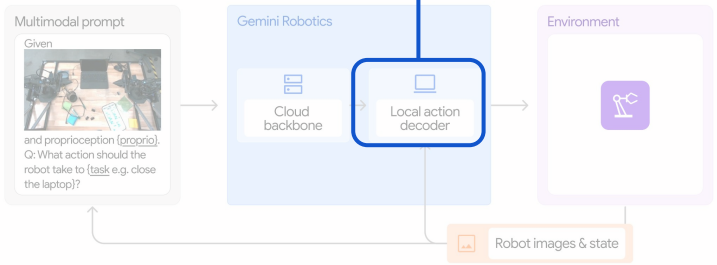
RT-2, 2023

Large-scale dataset

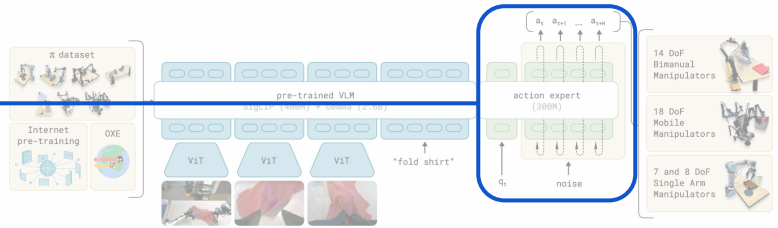


GR00T N1, 2025

Action decoder



Gemini Robotics, 2025



π0, 2024

LLM as Controller (VLA)

Pre-trained VLM

- Affect the ability of VLA to follow complex instructions, absorb diverse and large-scale dataset.
- What people use:
  - Llama, PaliGemma, Gemini, Eagle-2

Large-scale dataset

- Affects the action precision and robustness and the ability to handle diverse environments
- What people use:
  - In-house dataset, Open-X embodiment, Agibot, simulation data, video model data, human data

Action decoder

- Affects the ability to absorb diverse dataset, precision of actions, speed of inference
- What people use:
  - Diffusion head, latent model, direct tokenization

LLM as Controller (VLA)

Gemini Robotics, 2025

LLM as Controller (VLA)

Gemini Robotics, 2025

LLM as Controller (VLA)

π0.5, 2025

LLM as Controller (VLA)

Take away

- LLM/VLM has evolved dramatically over a short period of time
- Scaling law is a driving force behind the high investment
- Four ways of applying large models to robotics:
  - Planning
  - Coding
  - Teaching
  - Control

Take away

- LLM/VLM has evolved dramatically over a short period of time
- Scaling law is a driving force behind the high investment
- Four ways of applying large models to robotics:
  - Planning
  - Coding
  - Teaching
  - Control