Foundation Models For Robots and RL

Huazhe Xu
Tsinghua University

Exam grades are released!

AI This Week


Zhiting Hu
@ZhitingHu
Enjoy breakfast made by robot roommate😉
翻译帖子
0:59

Zhiting Hu @ZhitingHu · 5月10日

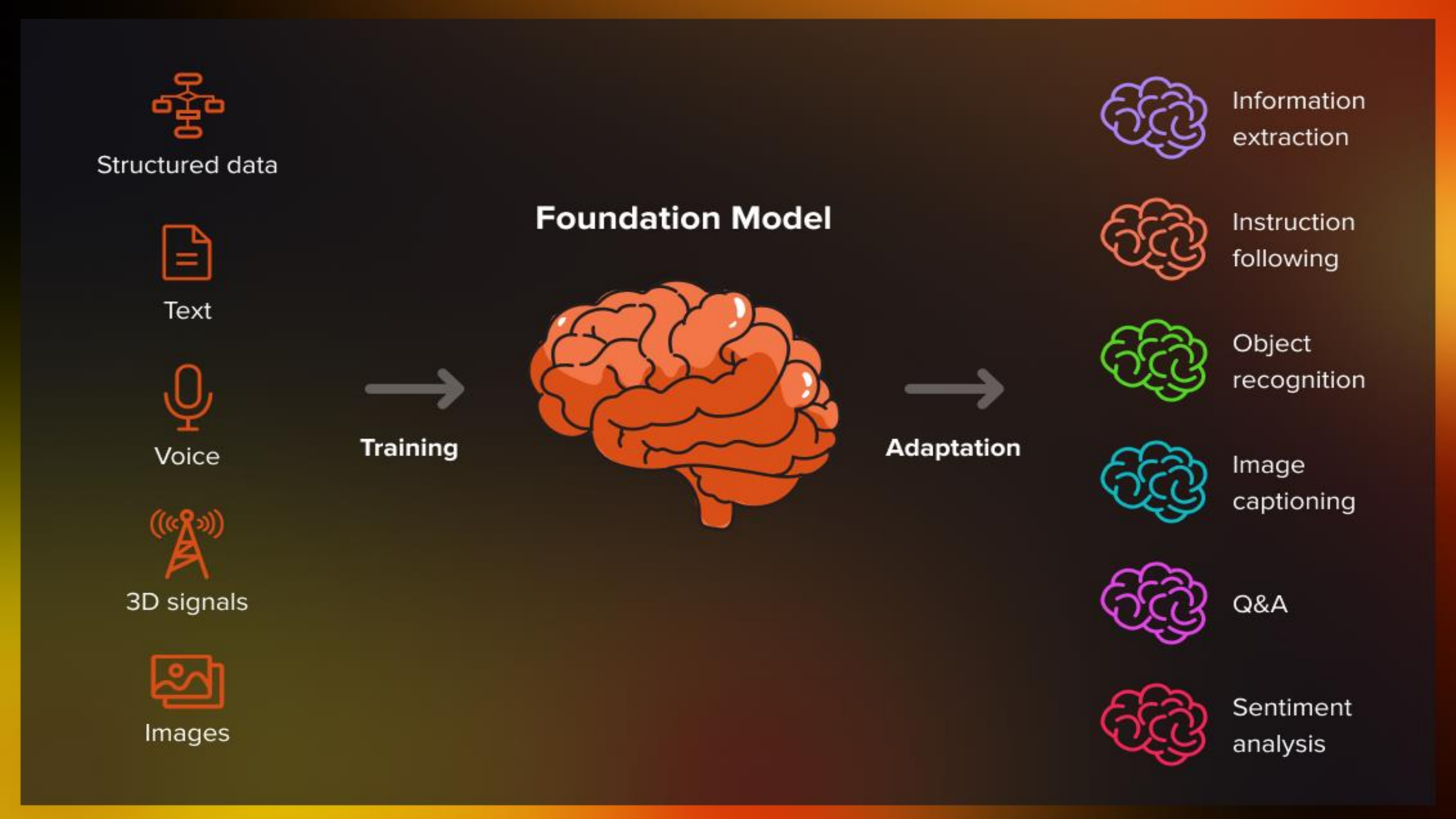
I was kidding -- this video was entirely simulated by the _world model_ we're building. 😀

It's mind-blowing how it produces high-fidelity simulations, lasting several minutes, to complete non-trivial tasks.

This showcases the potential for infinite data & experience in
显示更多

> 
>
> **Zhiting Hu** @ZhitingHu · 5月10日
>
> Enjoy breakfast made by robot roommate😉
>
> 
>

10 53 402 7万


Structured data
Text
Voice
3D signals
Images
Training
Foundation Model
Adaptation
Information extraction
Instruction following
Object recognition
Image captioning
Q&A
Sentiment analysis

Embodied AI

AI agents that interact with and learn from a physical environment

AI agents

Can be robots, virtual assistants, or other intelligent systems

Perceptual inputs

Equipped with sensors that import data from their surroundings, along with AI systems that can analyze and 'learn' from data

Interactive learning

The AI-powered agents learn from interacting with the environment until it reaches it goal

World model

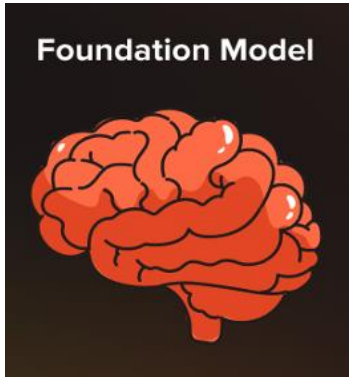
Develop an abstract representation and understanding of the spatial or temporal dimensions of our world

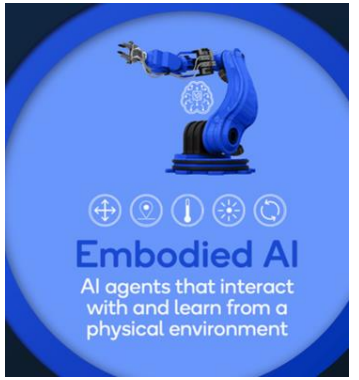
Goal

Create agents that can learn to solve complex tasks, such as motion planning and navigation, by interacting with their environment

Hottest topics combined…

- LLM/VLM + Embodied AI
- What is enabled?
  - Natural language/multi-modal interaction
  - Task planning
  - Reasoning
  - Adaptation
  - Automated embodied pipeline





In Lec12

1. LLMs for high-level planning
2. LLMs for low-level control policies
3. LLMs who write code

In Lec12

1. **LLMs as high-level planners**
2. LLMs for low-level control policies
3. LLMs who write code

Do As I Can, Not As I Say:
Grounding Language in Robotic Affordances

Michael Ahn* Anthony Brohan* Noah Brown* Yevgen Chebotar* Omar Cortes* Byron David* Chelsea Finn*
Chuyuan Fu* Keerthana Gopalakrishnan* Karol Hausman* Alex Herzog* Daniel Ho* Jasmine Hsu* Julian Ibarz*
Brian Ichter* Alex Irpan* Eric Jang* Rosario Jauregui Ruano* Kyle Jeffrey* Sally Jesmonth* Nikhil Joshi*
Ryan Julian* Dmitry Kalashnikov* Yuheng Kuang* Kuang-Huei Lee* Sergey Levine* Yao Lu* Linda Luu* Carolina Parada*
Peter Pastor* Jornell Quiambao* Kanishka Rao* Jarek Rettinghouse* Diego Reyes* Pierre Sermanet* Nicolas Sievers*
Clayton Tan* Alexander Toshev* Vincent Vanhoucke* Fei Xia* Ted Xiao* Peng Xu* Sichun Xu* Mengyuan Yan* Andy Zeng*

 Robotics at Google  Everyday Robots

I spilled my drink, can you help?

GPT3

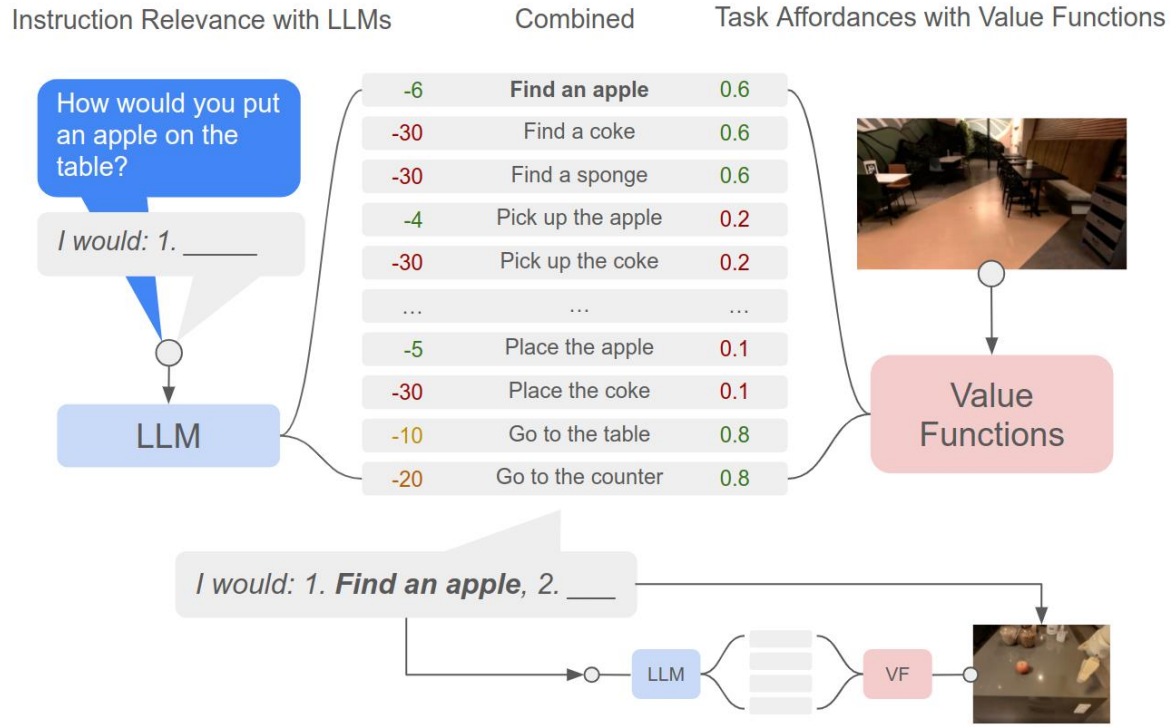
You could try using a vacuum cleaner.

LaMDA

Do you want me to find a cleaner?

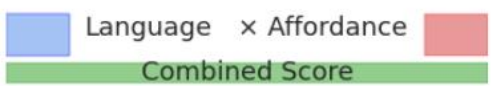
FLAN

I'm sorry, I didn't mean to spill it.



Human: I spilled my coke, can you bring me something to clean it up?

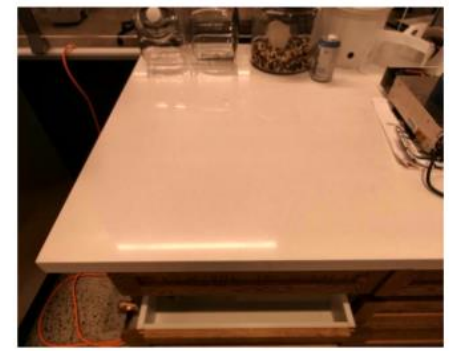
Robot: I would
1. Find a sponge
2. Pick up the sponge
3. Bring it to you
4. Done



find a sponge 1.00
go to the table 0.08
find a coke can 0.08
go to the trash can 0.05
find a water bottle 0.01

pick up the sponge 1.00
put down the sponge 0.11
bring it to you 0.04
go to the table 0.00
go to the trash can 0.00

bring it to you 1.00
put down the sponge 0.47
go to the table 0.21
go to the trash can 0.00
done 0.00



done 1.00
go to the table 0.01
find a coke can 0.00
find a sponge 0.00
go to the trash can 0.00

Instruction Family	Num	Plan	Execution
Natural Language Single Primitive	15	67%	67%
Natural Language Nouns	15	60%	53%
Natural Language Verbs	15	80%	67%
Structured Language	15	100%	87%
Embodiment	11	64%	55%
Crowd Sourced	15	73%	67%
Long-Horizon	15	47%	33%
Total	**101**	**70%**	**61%**

What’s wrong with SayCan?

How dare you to enumerate all the skills?

Inner Monologue:
Embodied Reasoning through Planning with Language Models

Wenlong Huang* Fei Xia* Ted Xiao* Harris Chan Jacky Liang Pete Florence

Andy Zeng Jonathan Tompson Igor Mordatch Yevgen Chebotar Pierre Sermanet

Noah Brown Tomas Jackson Linda Luu Sergey Levine Karol Hausman Brian Ichter

Robotics at Google

Action:
pick up
the coke

Object Recognition

Success Detection

Kitchen Mobile Manipulation (Real)

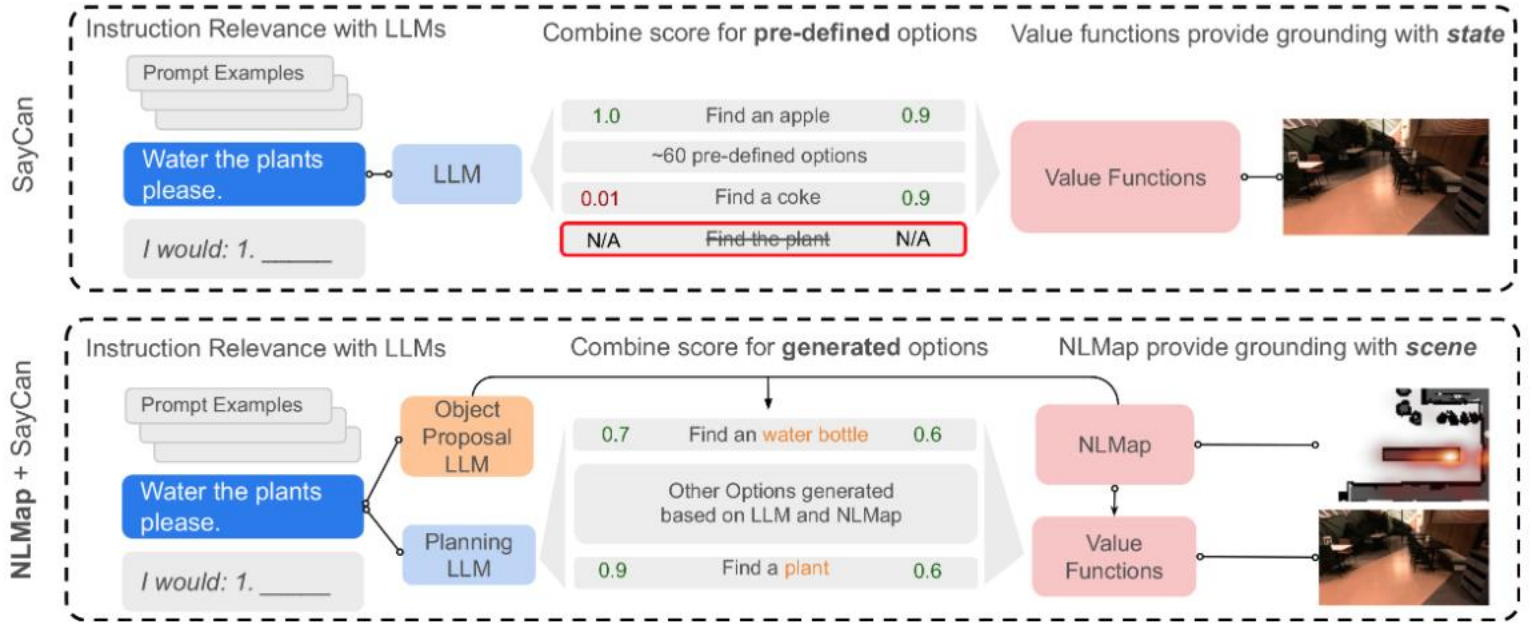
Human: Bring me a drink from the table
Robot: go to the table
Scene: I see a coke and lime soda
Robot: What kind of drink would you like?
Human: Something with caffeine
Robot: Pick up coke
Success Detector: False
Robot: Pick up coke
Robot: Bring it to you
Robot: Put down the coke

Still not satisfied?

Open-vocabulary Queryable Scene Representations for Real World Planning

Boyuan Chen Fei Xia Brian Ichter Kanishka Rao Keerthana Gopalakrishnan
Michael S. Ryoo Austin Stone Daniel Kappler

Everyday Robots Robotics at Google



Instruction

↓

Object Proposal

↓

Query Context

↓

Generate Executable Options

↓

Context Aware Planning

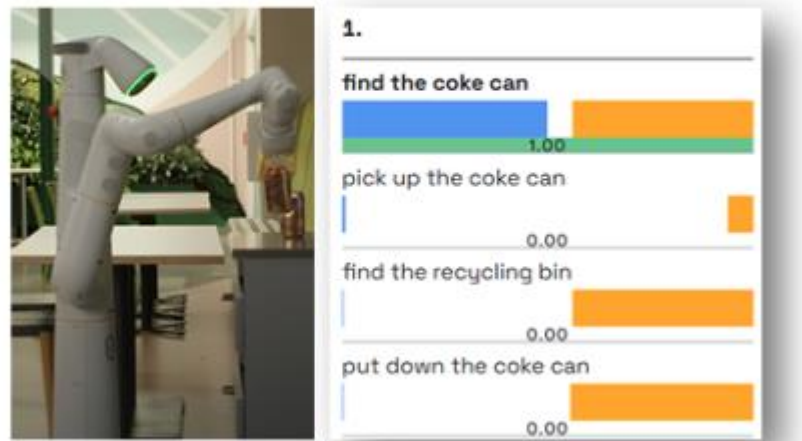
"Recycle the coke can"

"coke can", "recycle bin"

"coke can" found at (x1, y1)
"recycle bin" found at (x2, y2)

"go to coke can", "pick up coke can",
"put down coke can", ...

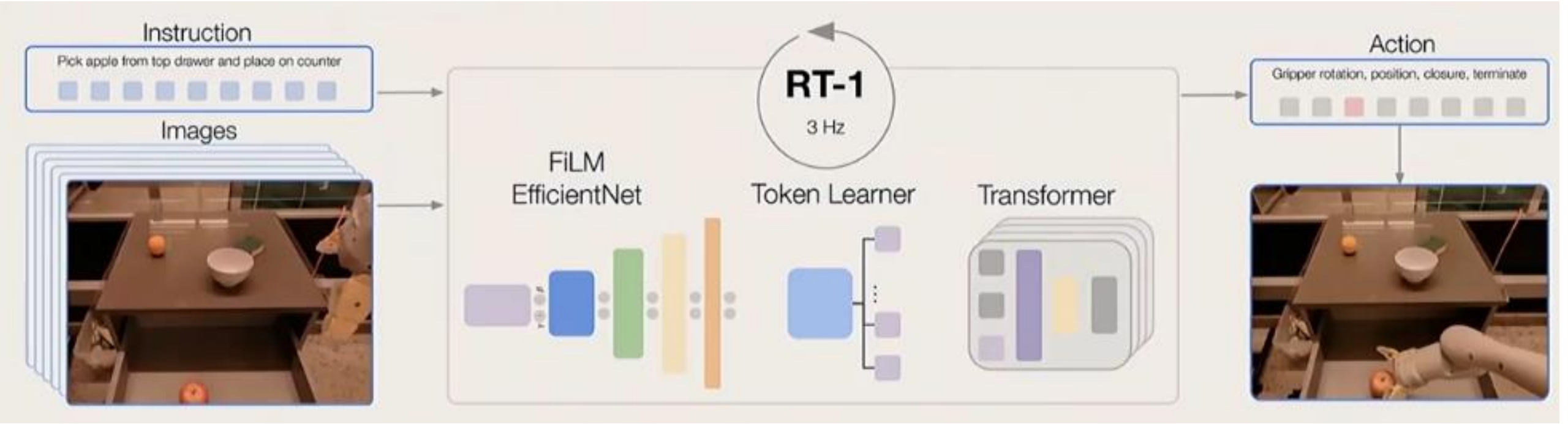
Scene: coke can, recycle bin
Robot: I should
1. find the coke can
2. pick up coke can
3. go to recycle bin
4. put down coke can



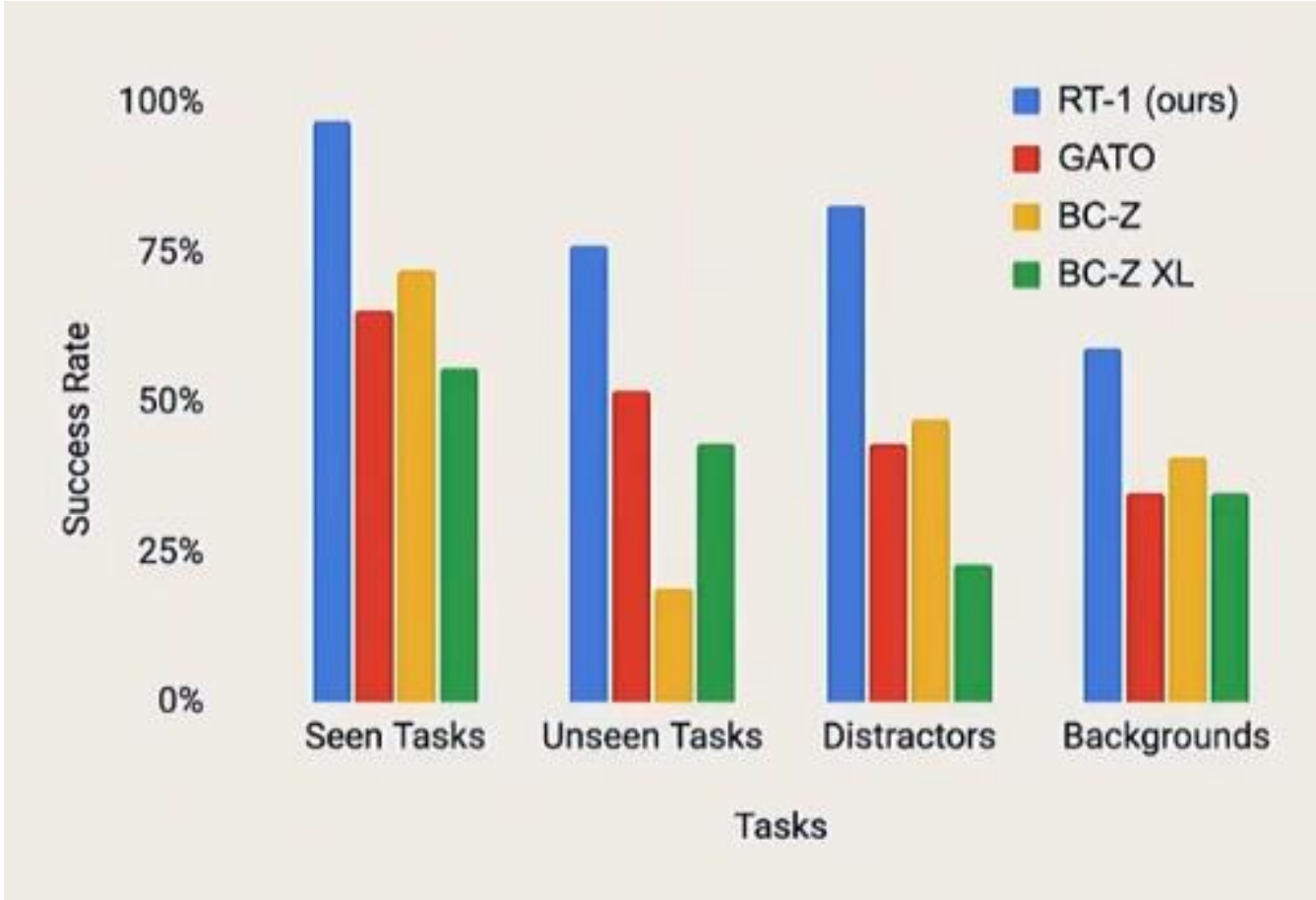
In Lec13

1. Foundation models for high-level planning
2. **Foundation models for low-level control policies**
3. LLMs who write code

Robot Transformer (RT-1)

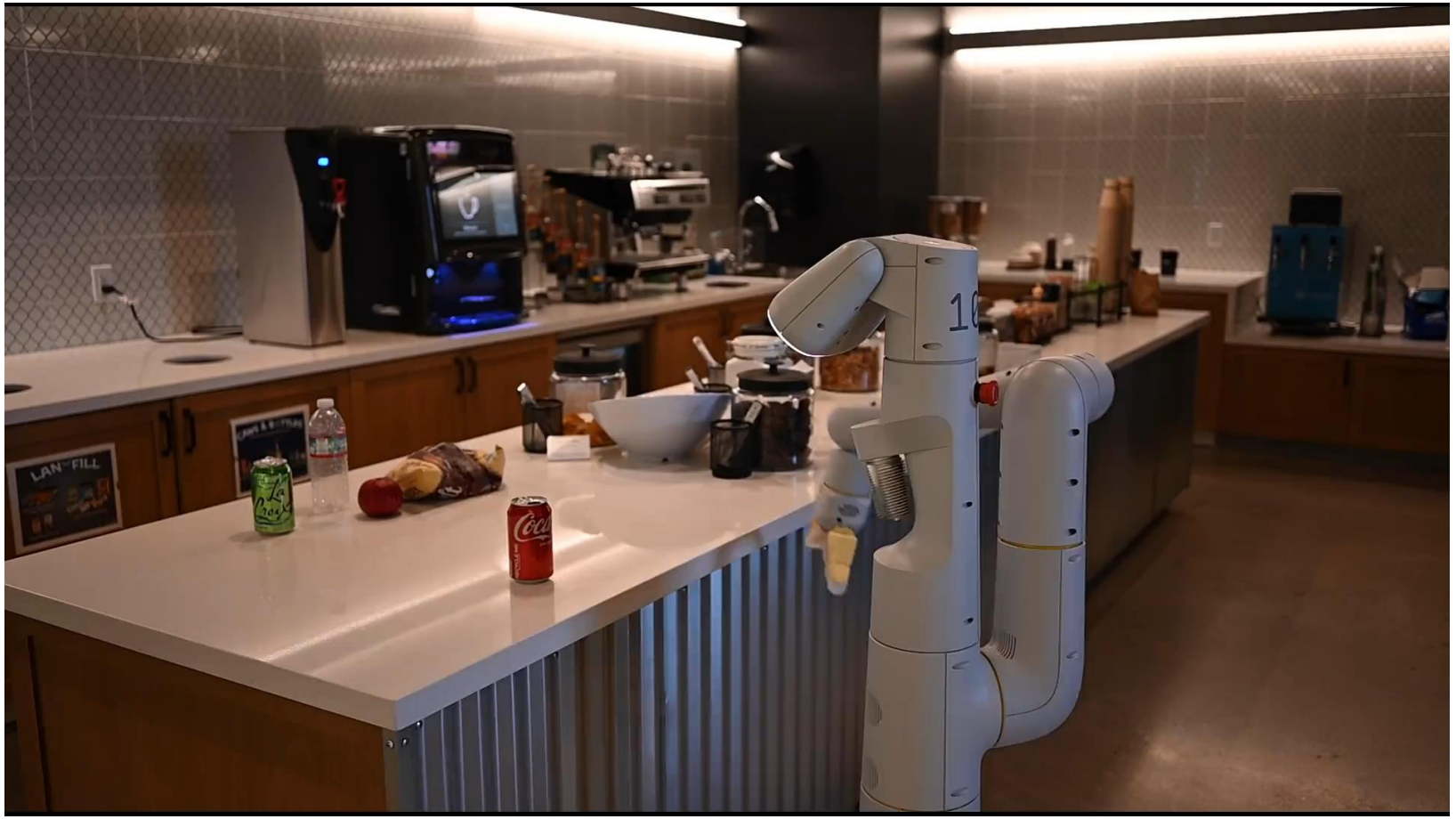


- Tokenized input and outputs
- Decoder only transformer, sparse categorical entropy objective
- Image tokenizer: Pre-trained film efficient net backbone
- Token learner for compression/faster inference
- 130000 episodes, 13 robots, over 17 months, 700 tasks


100%
75%
50%
25%
0%
Success Rate
RT-1 (ours)
GATO
BC-Z
BC-Z XL
Seen Tasks
Unseen Tasks
Distractors
Backgrounds
Tasks

- Capable of overfitting training task
- Improves with addition of simulated data
- Shows signs of cross-embodiment transfer?
  - Crux of robots' GPT moment
- Data is not diverse.

Collect data

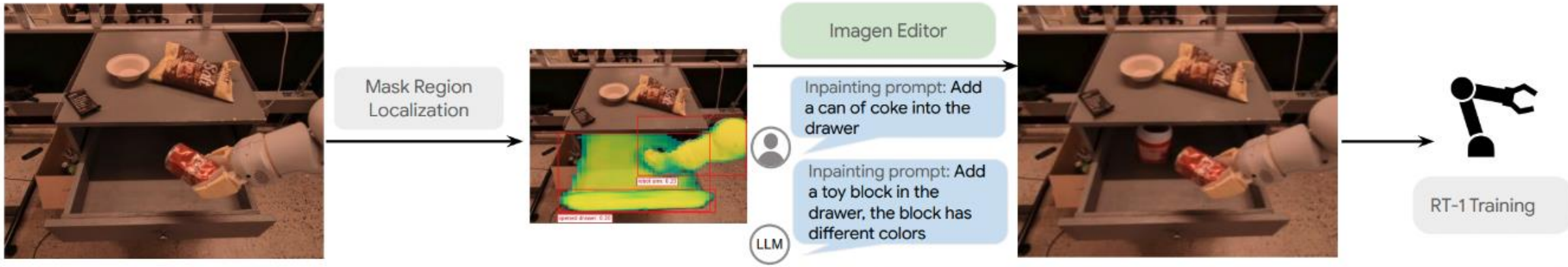


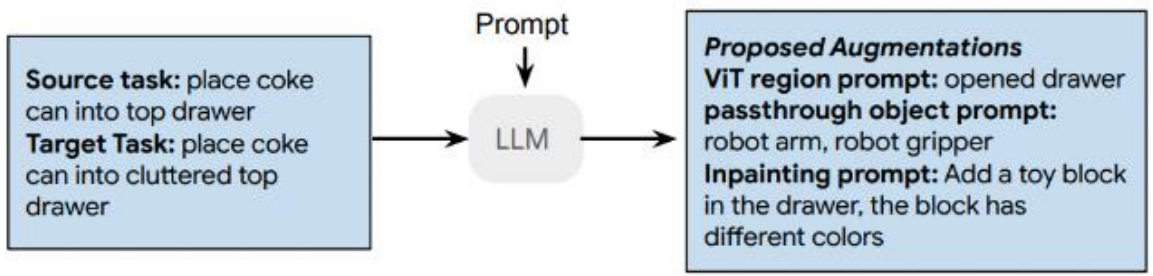
Scaling Robot Learning with Semantically Imagined Experience

Tianhe Yu Ted Xiao Austin Stone Jonathan Tompson
Anthony Brohan Su Wang Jaspiar Singh Clayton Tan Dee M
Jodilyn Peralta Brian Ichter Karol Hausman Fei Xia

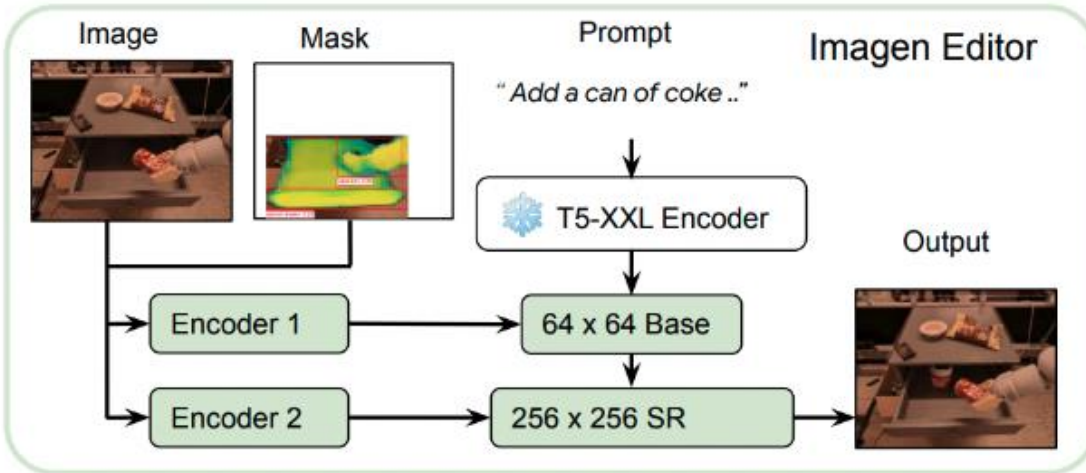
13 robots,17 months,
to collect 130k demonstrations

+ Generative models?





LLM-assisted Augmentation Prompt Generation




Robotics @ Google

Task Family / Text Instruction	NoAug	InstructionAug	ROSIE
Move object near novel object	0.86	0.78	**0.94**
move coke can/orange near lunch box	0.8	0.6	0.9
move coke can/orange near woven basket	0.7	0.6	0.9
move coke can/orange near ceramic pot	1.0	0.9	1.0
move coke can/orange near glass mason jar	0.9	0.8	1.0
move coke can/orange near orange paper plate	0.9	1.0	0.9
Pick up novel object	0.25	0.3	**0.75**
pick blue microfiber cloth	0.1	0.4	0.8
pick black microfiber cloth	0.4	0.2	0.7
Place object into novel container	0.13	0.25	**0.44**
place coke can into orange plastic plate	0.0	0.19	0.5
place coke can into blue plastic plate	0.25	0.06	0.38
Place object into sink	0.0	-	**0.6**
place coke can into sink	0.0	-	0.8
place pepsi can into sink	0.0	-	0.4
Pick up object in new backgrounds	0.33	-	**0.71**
pick coke can on an orange table cloth	0.0	-	0.4
pick pepsi can on an orange table cloth	0.0	-	0.7
pick coke can on an blue and white table cloth	0.2	-	0.7
pick pepsi can on an blue and white table cloth	0.8	-	0.8
pick coke can near the side of a sink	0.4	-	0.5
pick pepsi can near the side of a sink	0.3	-	0.7
pick coke can in front of a sink	0.4	-	0.9
pick pepsi can in front of a sink	0.5	-	1.0

ROSIE Augmented Data

Real-world Rollout

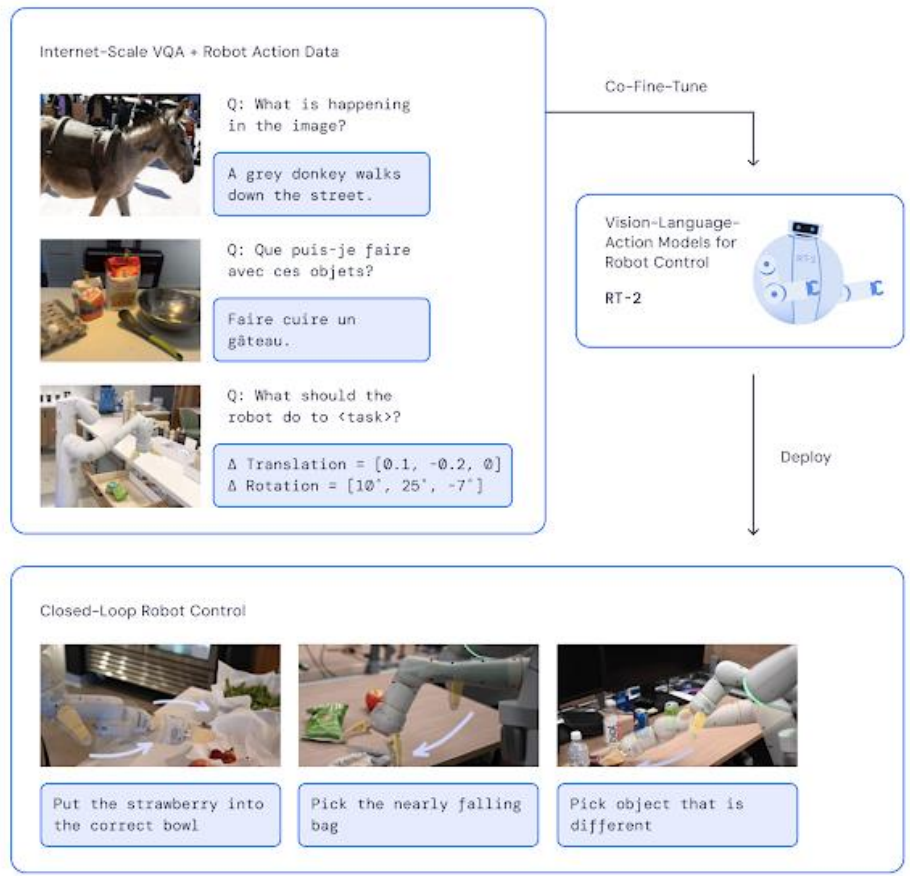
robot putting objects in drawer → *robot putting objects in sink*

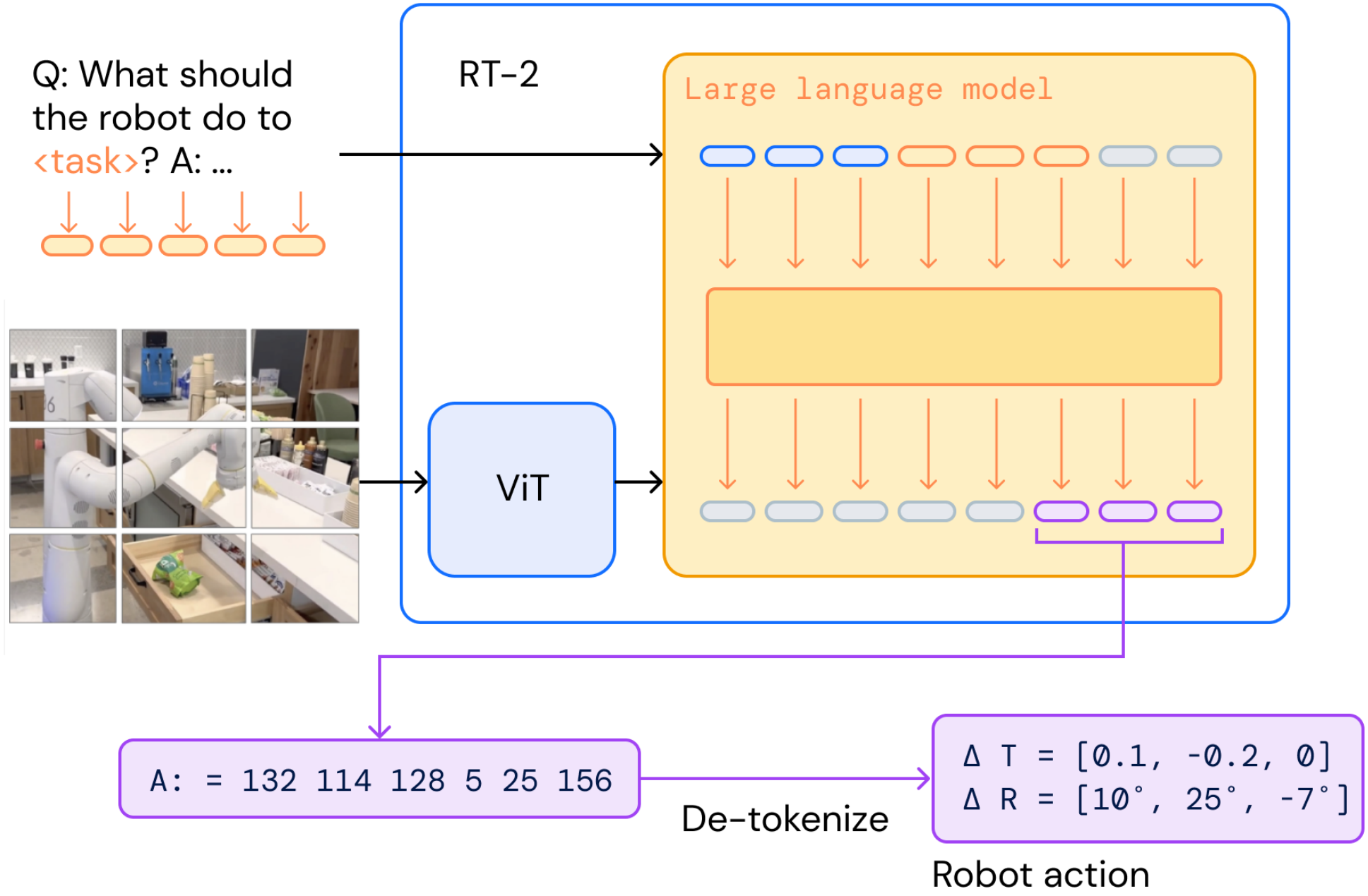

VLM

“ Terminate or continue | Δ Pos X | Δ Pos Y | Δ Pos Z | Δ Rot X | Δ Rot Y | Δ Rot Z | Gripper ”

Positional change

Rotational change




Q: What should the robot do to <task>? A: ...
RT-2
Large language model
ViT
A: = 132 114 128 5 25 156
De-tokenize
Δ T = [0.1, -0.2, 0]
Δ R = [10°, 25°, -7°]
Robot action

In Lec12

1. Foundation models for high-level planning
2. Foundation models for low-level control policies
3. **LLMs who write code**

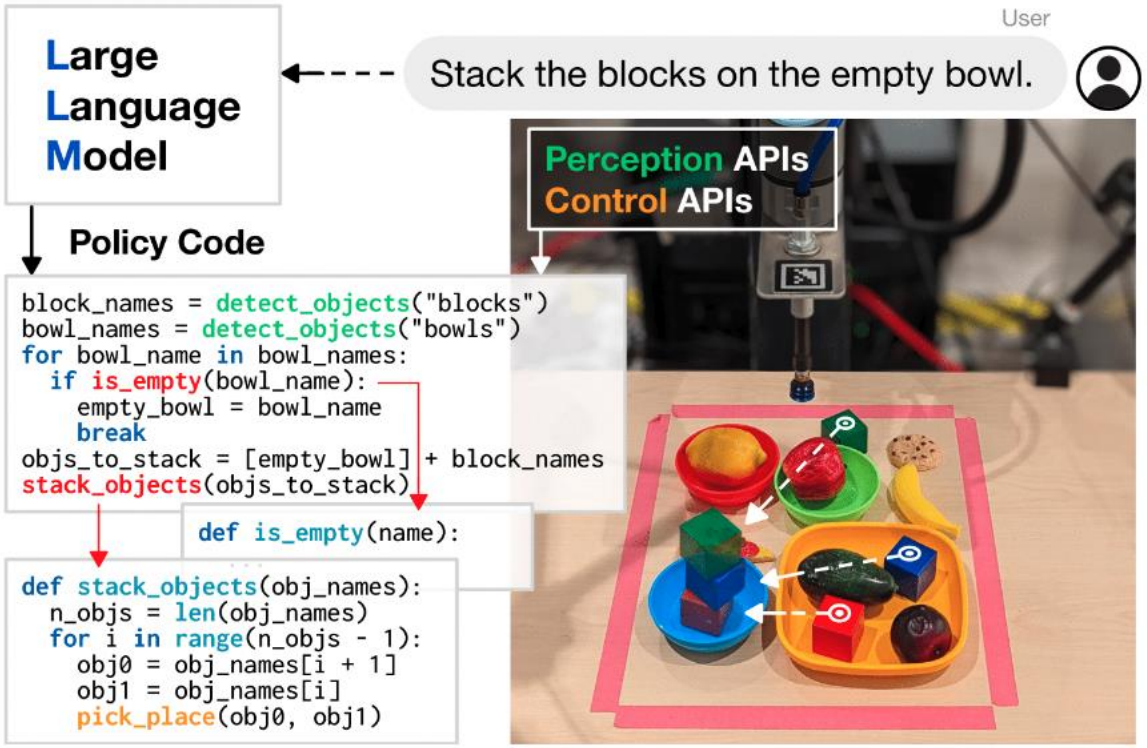
LLMs can write code…

Code as Policies:

Language Model Programs for Embodied Control

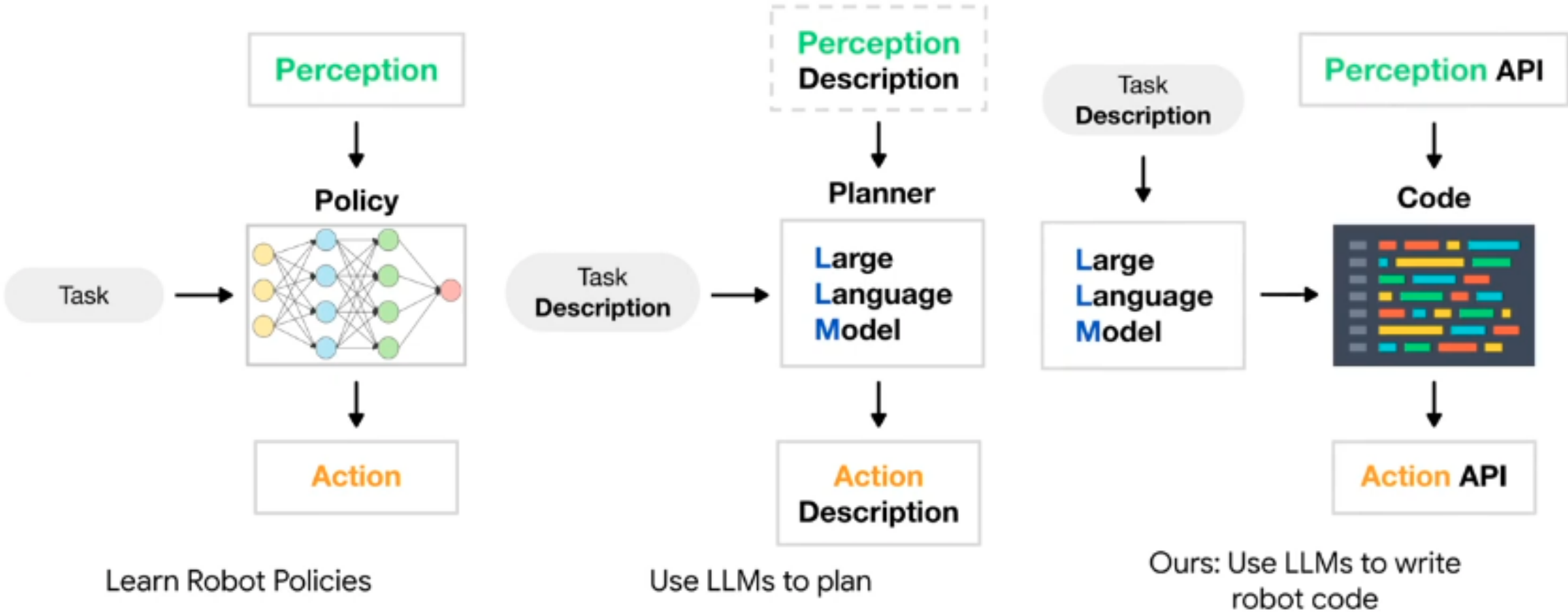
Jacky Liang Wenlong Huang Fei Xia Peng Xu Karol Hausman Brian Ichter Pete Florence Andy Zeng

Robotics at Google



Benefits

- Expressive inputs and outputs
- Solve tasks w/ few-shot prompting and zero-shot training
- Improved generalization to unseen tasks



Learn Robot Policies

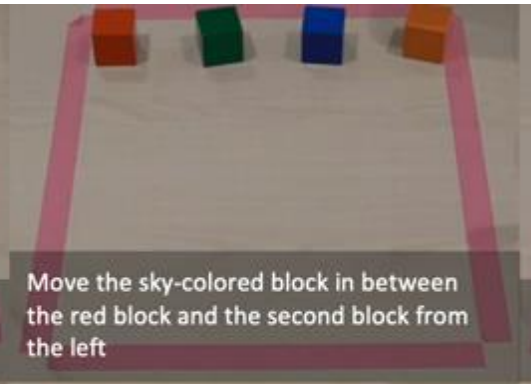
Use LLMs to plan

Ours: Use LLMs to write robot code

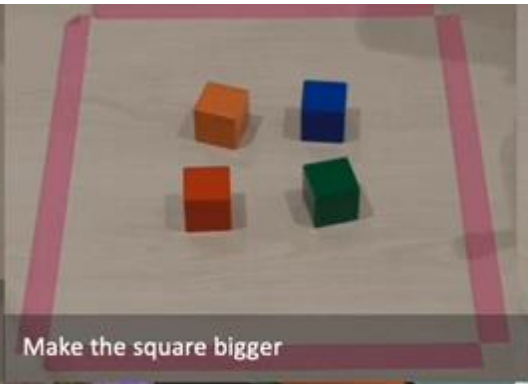
1) Put the red block to the left of the rightmost bowl
2) Now move it to the side farthest away from it
3) How many bowls are to the left of the red block?
4) Place the blocks in bowls with non matching colors
5) Put the blocks in a vertical line 20 cm long and 10 cm below the blue bowl

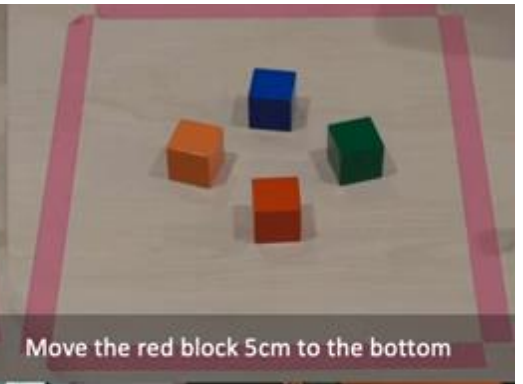
```
objects = ['red block', 'green block', 'blue block', 'red bowl', 'green bowl', 'blue bowl']
# put the red block to the left of the rightmost bowl.
rightmost_bowl_name = parse_obj_name('the rightmost bowl', f'objects = {get_obj_names()}')
if rightmost_bowl_name:
    say(f'Putting the red block to the left of the {rightmost_bowl_name}')
    left_pos = parse_position(f'a point 10cm left of the {rightmost_bowl_name}')
    put_first_on_second('red block', left_pos)
else:
    say('There are no bowls')

objects = ['red block', 'green block', 'blue block', 'red bowl', 'green bowl', 'blue bowl']
# the rightmost bowl.
bowl_names = ['red bowl', 'green bowl', 'blue bowl']
bowl_positions = get_obj_positions_np(bowl_names)
right_bowl_idx = get_right_most_idx(bowl_positions)
right_bowl_name = bowl_names[right_bowl_idx]
ret_val = right_bowl_name

# a point 10cm left of the blue bowl.
blue_bowl_name = parse_obj_name('blue bowl', f'objects = {get_obj_names()}')
blue_bowl_pos = get_obj_pos(blue_bowl_name)
left_obj_pos = blue_bowl_pos + [-0.1, 0]
ret_val = left_obj_pos

objects = ['red block', 'green block', 'blue block', 'red bowl', 'green bowl', 'blue bowl']
# blue bowl.
ret_val = 'blue bowl'

# define function: bowl_positions = get_obj_positions_np(bowl_names).
def get_obj_positions_np(obj_names):
    obj_positions = []
    for obj_name in obj_names:
        obj_positions.append(get_obj_pos(obj_name))
    return np.array(obj_positions)

# define function: right_bowl_idx = get_right_most_idx(bowl_positions).
def get_right_most_idx(points):
    return np.argmax(points[:, 0])
```


Put the blocks in a horizontal line near the top


Move the sky-colored block in between the red block and the second block from the left

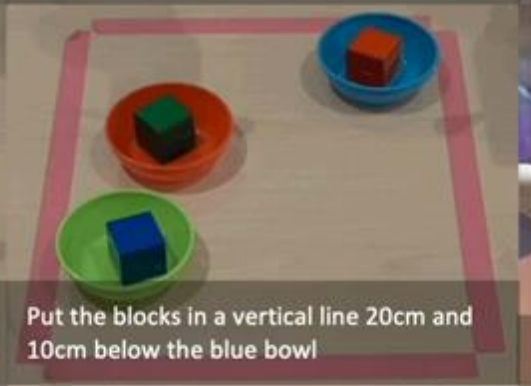

Arrange the blocks in a square around the middle


Make the square bigger


Move the red block 5cm to the bottom


Put the red block to the left of the rightmost bowl


Place the blocks in bowls with non-matching colors


Put the blocks in a vertical line 20cm and 10cm below the blue bowl


Put the apple and the coke in their corresponding bins


Move the fruits to the green plate and bottles to the blue plate


Wait until you see an egg and put it on the green plate


Draw a 5cm hexagon around the middle


Draw a pyramid as a triangle on the ground

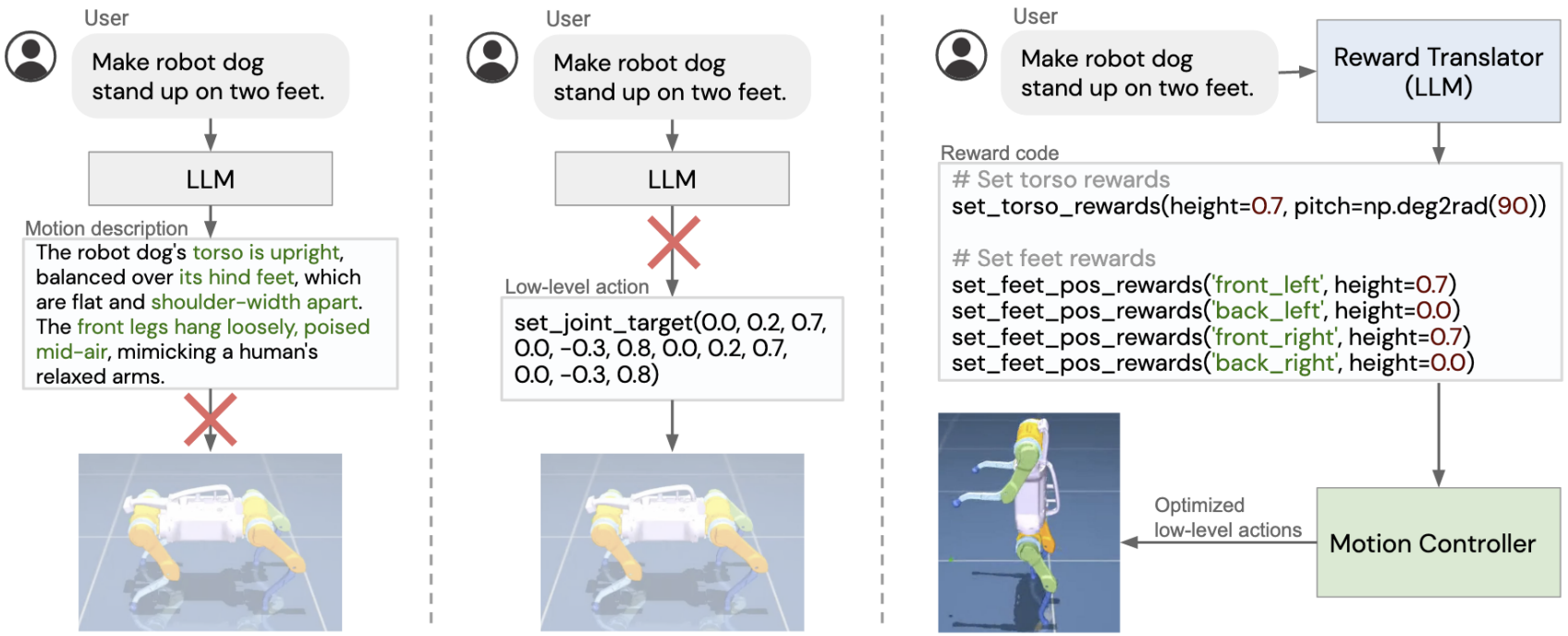
What if I need the robot to rotate a pen?

- LLMs are not as clever as we think.
- It cannot work!

What else can LLMs code up?

- Rewards!
- And … remember, you are already an RL expert!

Language2Reward

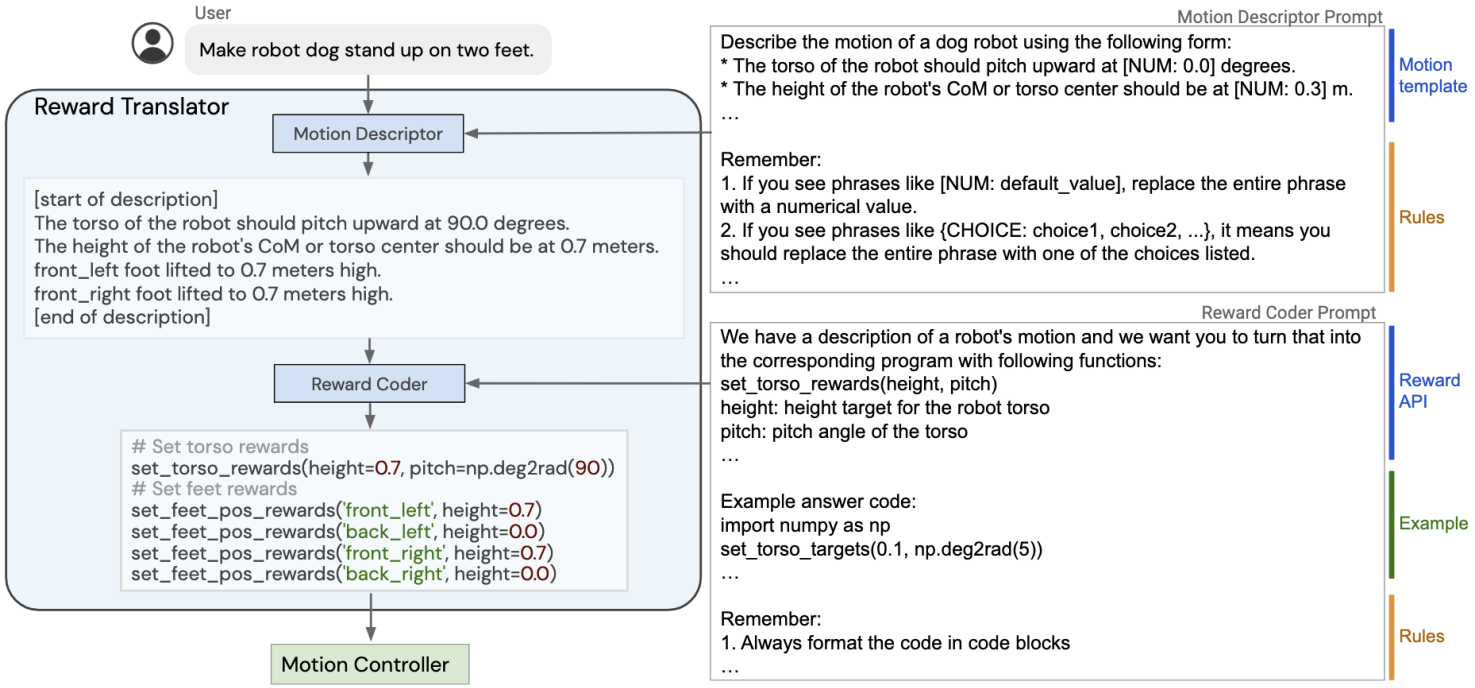


1. High-level description is hard to guide the agent's action.
2. Directly outputting the low-level action is hardware-dependent and limit availability of relevant training data.

The insight is to bridge the gap between language and low-level actions.

Language2Reward

Reward Translator



1. User describe his requirement.
2. A **Motion Descriptor** LLM takes the user input and describe the user-specified motion in natural language.
3. **Reward Coder** translates the motion into the reward parameters.
4. Leverage the in-context learning ability to achieve goal.

Language2Reward

Motion Controller

1. Map reward function generated by the Reward Translator low-level robot actions.
2. There are a few possible ways to achieve this, RL or trajectory optimization.

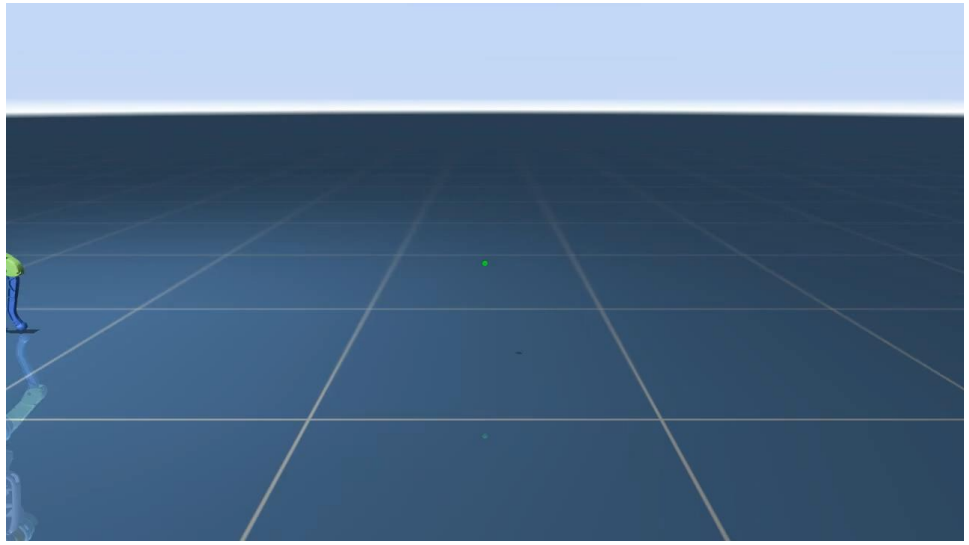
In this work, we assume the reward takes a particular form, suitable for use with MJPC:

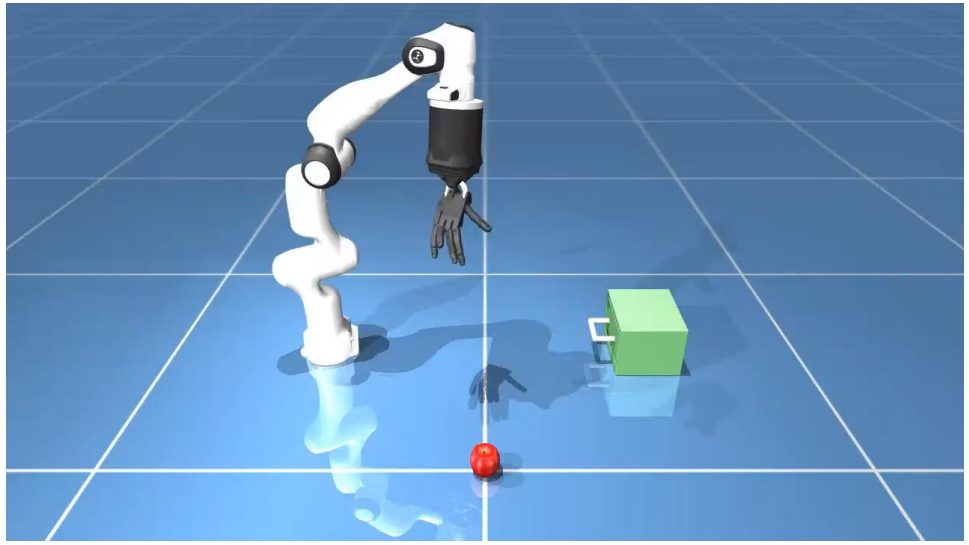
$$R(\mathbf{s}, \mathbf{a}) = -\sum_{i=0}^{M} w_i \cdot \mathbf{n}_i(r_i(\mathbf{s}, \mathbf{a}, \psi_i))$$

Language2Reward

We designed a total of 17 tasks for a simulated quadruped robot and a dexterous manipulator robot.

We demonstrate that our proposed method reliably tackles 90% of the designed tasks.



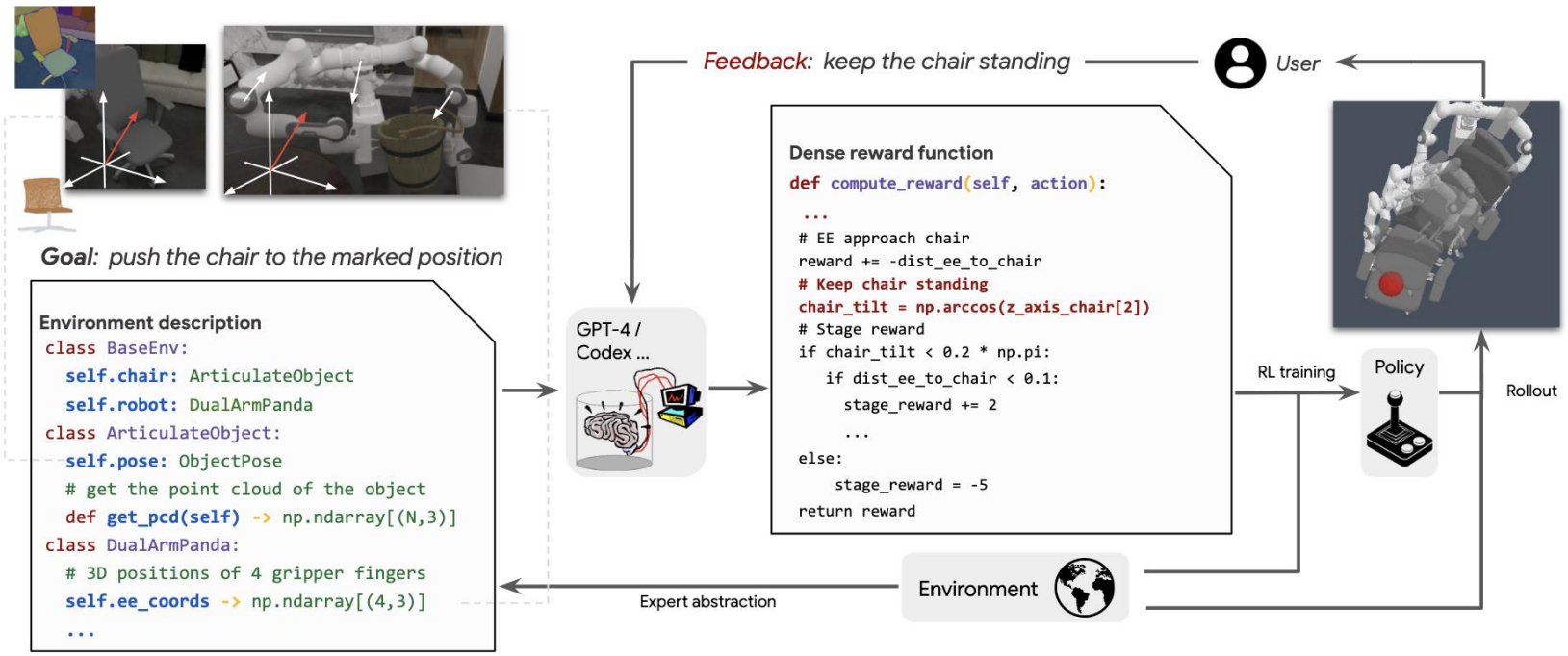


Drawbacks:

1. Essentially, L2R fills the function parameters with numbers.
2. L2R heavily relies on a large number of prompts.

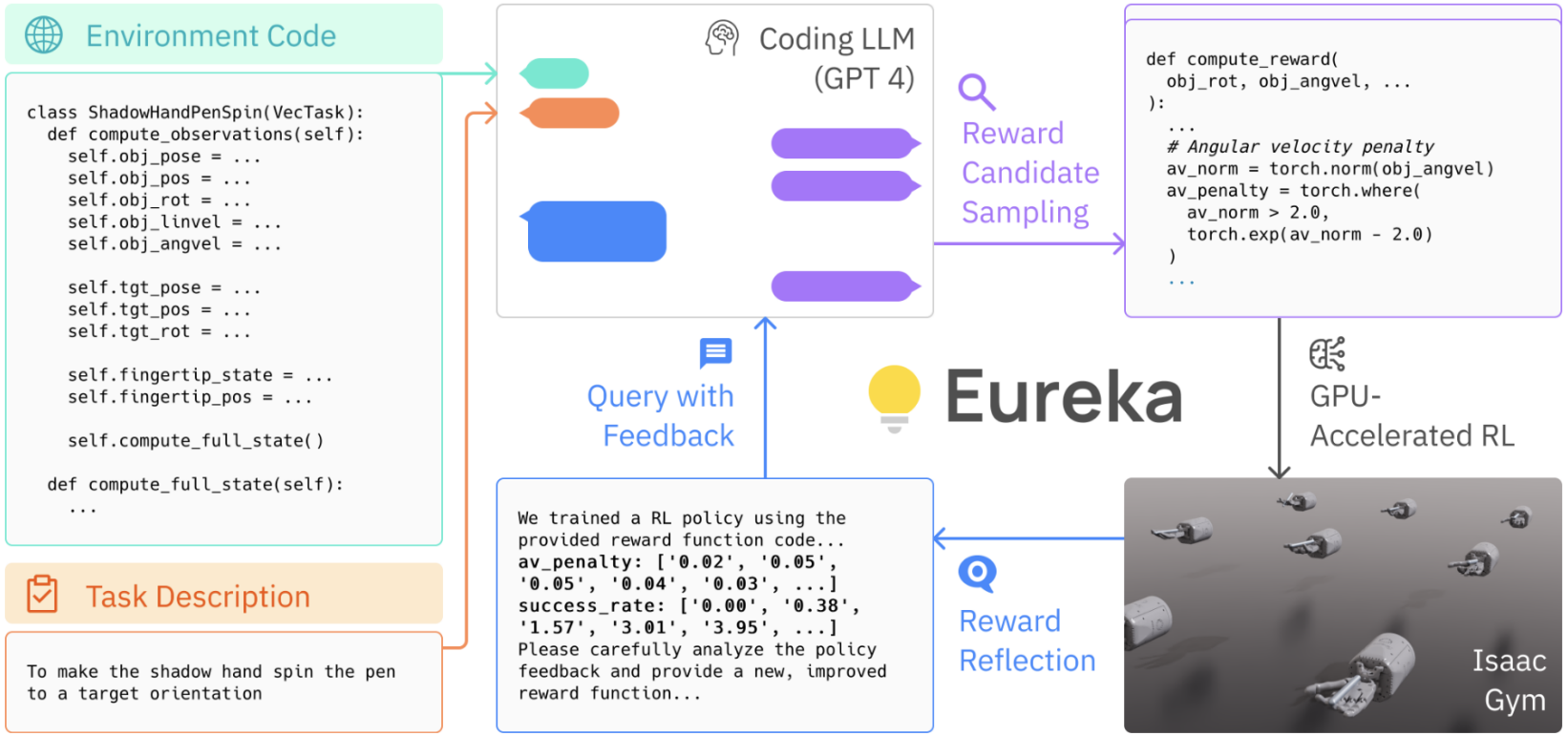
Text2reward

1. Text2reward generates dense reward functions and executable programs.
2. The free-form dense reward code has a wider coverage of tasks and can utilize established coding packages (e.g., Numpy and other calcuations.).
3. They also demonstrate that the policy trained in the simulator can be deployed on a real Franka Panda robot.



Sim2real

Eureka



1. Similar to Text2reward, Eureka takes environment code and language task description as context to zero-shot generate the executable reward function.
2. Furthermore, Eureka incorporates reward reflection mechanism to progressively improve the generated reward.

Eureka

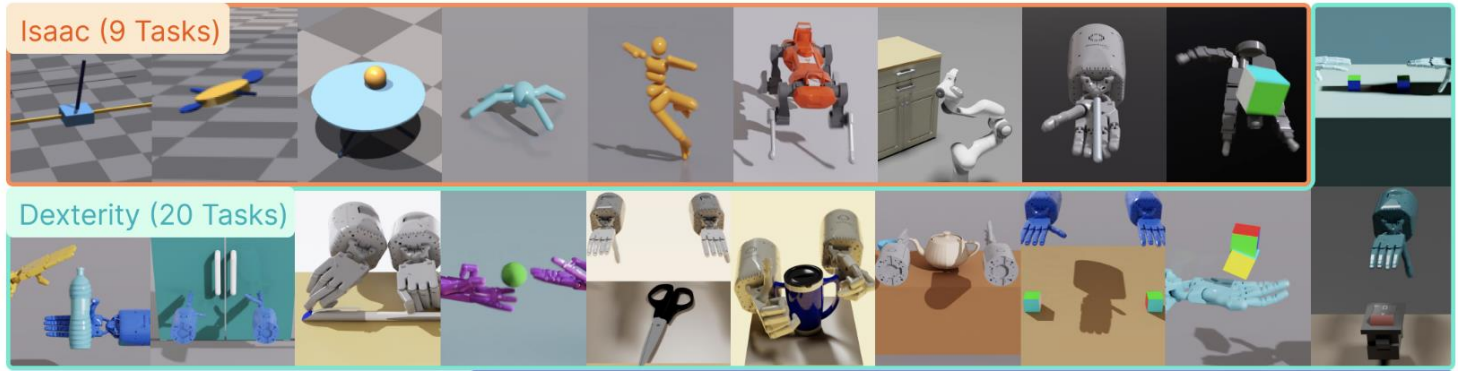
Eureka can flexibly improve the rewards with many distinct types of free-form modification:

- Changing the hyperparameter of existing reward components.
- Changing the functional form.
- Introducing new reward components.

```
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_temp = 30.0                                   Changing hyperparameter
    rotation_reward = torch.exp(-rotation_reward_temp * rot_diff)

    # Distance reward
+   min_distance_temp = 10.0
    min_distance = torch.min(torch.norm(fingertip_pos - object_pos[:, None], dim=2), dim=1).values
-   distance_reward = min_distance
+   uncapped_distance_reward = torch.exp(-min_distance_temp * min_distance)
+   distance_reward = torch.clamp(uncapped_distance_reward, 0.0, 1.0)   Changing functional form

-   total_reward = rotation_reward + distance_reward
+   # Angular velocity penalty                                    Adding new component
+   angvel_norm = torch.norm(object_angvel, dim=1)
+   angvel_threshold = 0.5
+   angvel_penalty_temp = 5.0
+   angular_velocity_penalty = torch.where(angvel_norm > angvel_threshold,
+       torch.exp(-angvel_penalty_temp * (angvel_norm - angvel_threshold)), torch.zeros_like(angvel_norm))
+
+   total_reward = 0.5 * rotation_reward + 0.3 * distance_reward - 0.2 * angular_velocity_penalty

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

    return total_reward, reward_components
```

Eureka

1. Achieve human-level performance on reward design across 29 RL environments and 10 distinct robot morphologies.



2. Solves dexterous manipulation tasks that were previously not feasible by manual reward engineering. (pen spinning with shadow hand)

What else can LLMs code up?

- Tasks in simulators

RoboGen: Towards Unleashing Infinite Data for Automated Robot Learning via Generative Simulation

Yufei Wang[*,1], Zhou Xian[*,1], Feng Chen[*,2], Tsun-Hsuan Wang[3], Yian Wang[4],
Zackory Erickson[1], David Held[1], Chuang Gan[4,5]
[1]CMU, [2]Tsinghua IIIS, [3]MIT CSAIL, [4]UMass Amherst, [5]MIT-IBM AI Lab
[*]Equal Contribution

GenSim: Generating Robotic Simulation Tasks via Large Language Models

Lirui Wang[1], Yiyang Ling*[2,3], Zhecheng Yuan*[4],
Mohit Shridhar[5], Chen Bao[6], Yuzhe Qin[3], Bailin Wang[2], Huazhe Xu[4], Xiaolong Wang[3]
MIT CSAIL[1], Shanghai Jiao Tong University[2], UCSD[3], Tsinghua University[4], UW[5], CMU[6],

Workshop on Language Grounding and Robot Learning (Workshop Best Paper), CoRL 2023

Gen2Sim: Scaling up Robot Learning in Simulation with Generative Models

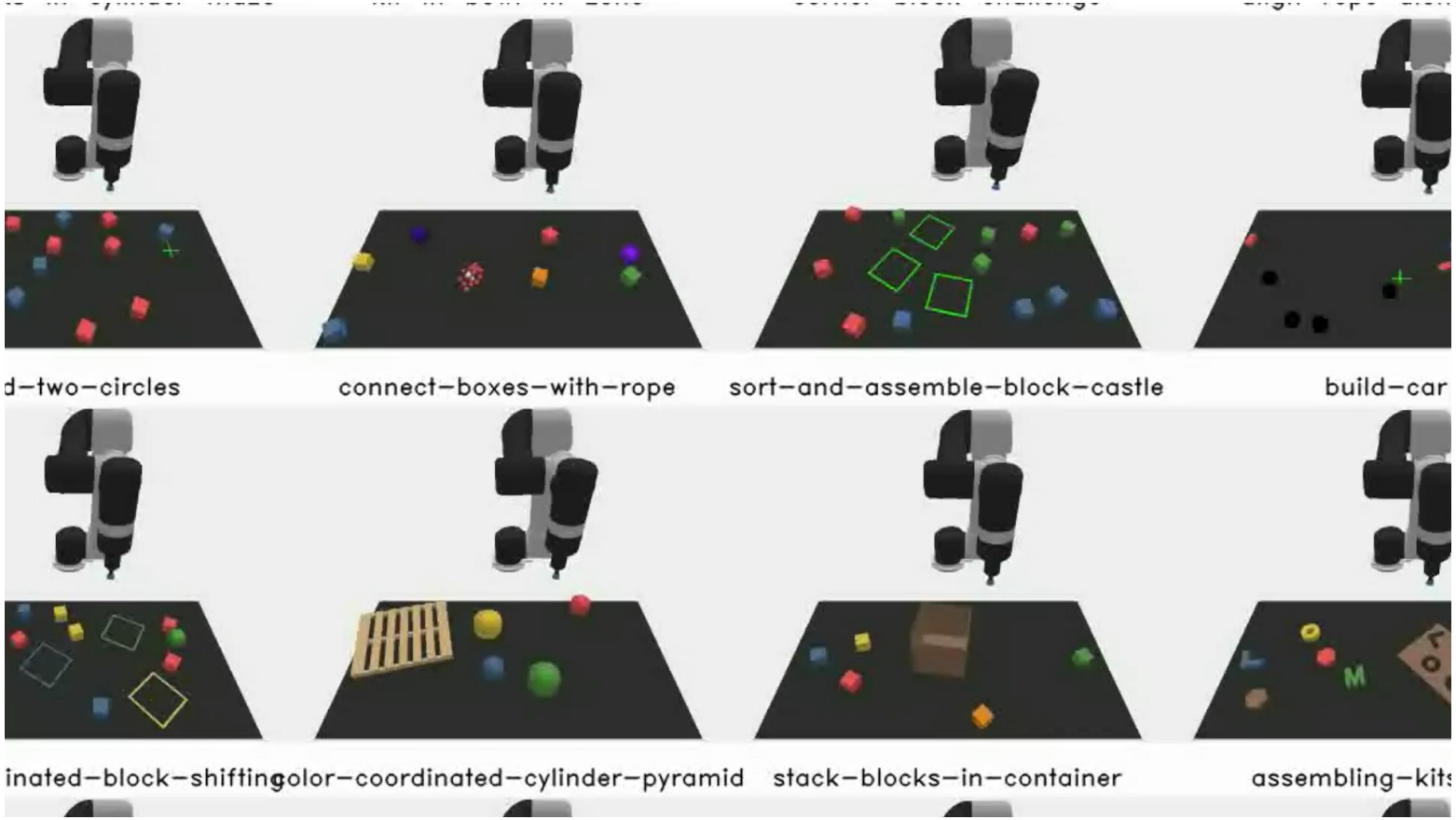
Pushkal Katara, Zhou Xian, Katerina Fragkiadaki

Carnegie Mellon University

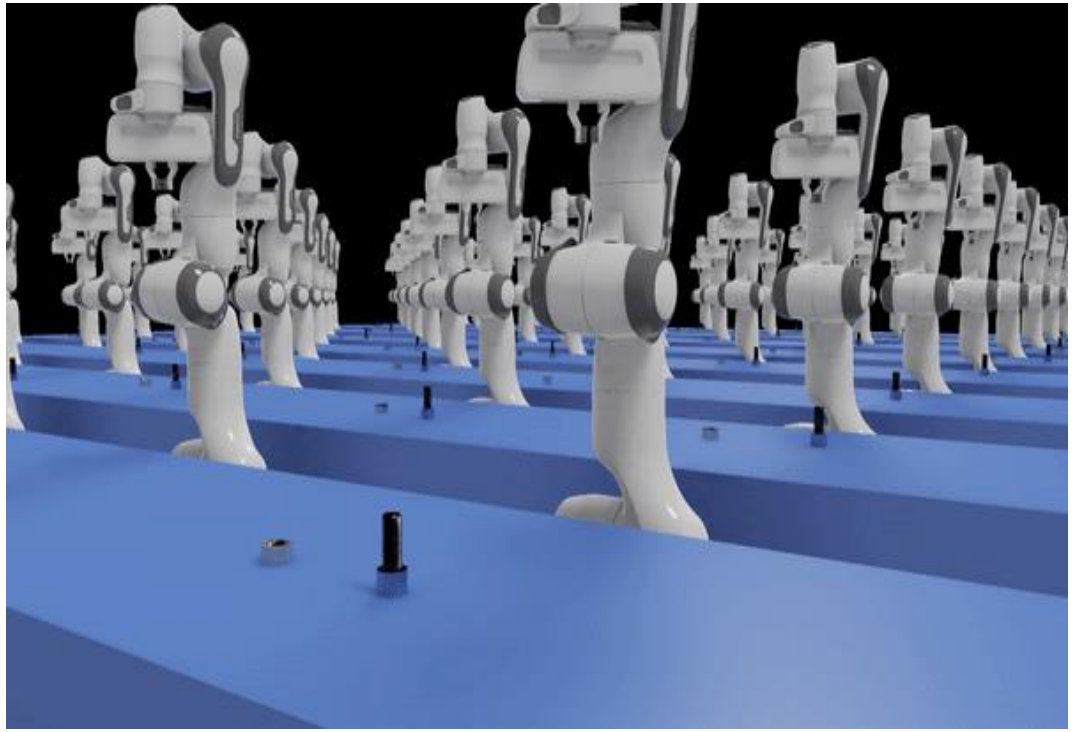
arXiv Video Github

GenSim: Generating Robotic Simulation Tasks via Large Language Models

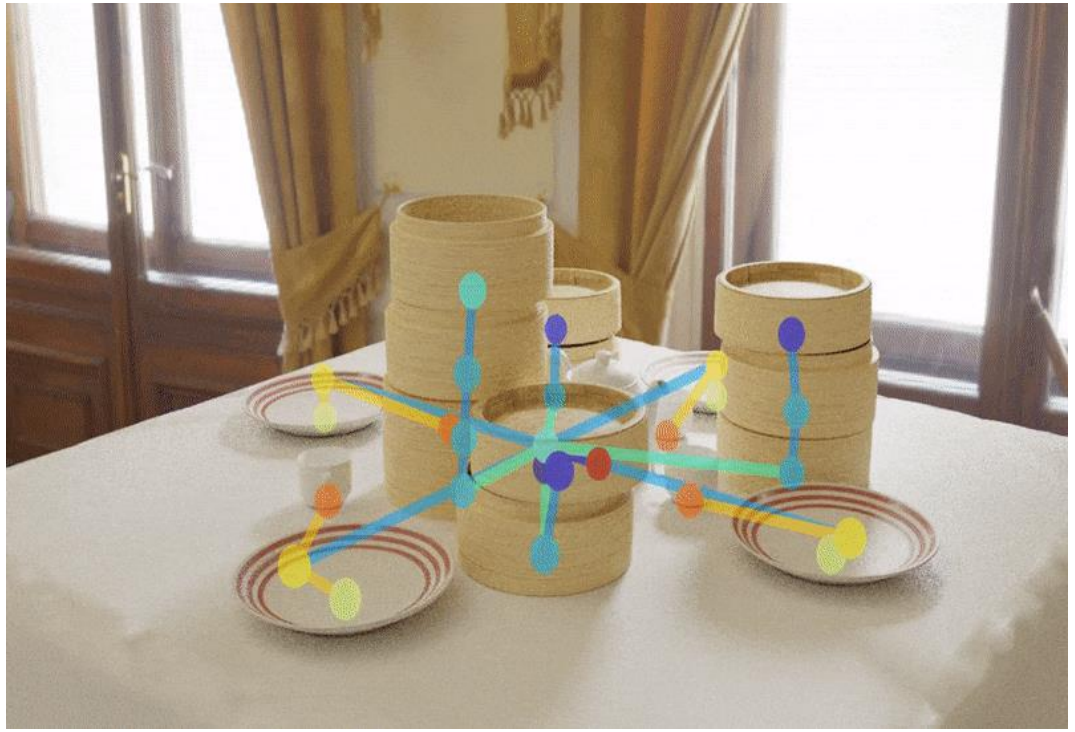
Lirui Wang, Yiyang Ling*, Zhecheng Yuan*, Mohit Shridhar, Chen Bao, Yuzhe Qin, Bailin Wang, Huazhe Xu, Xiaolong Wang



Current Methods to Scale Up Data in Simulation

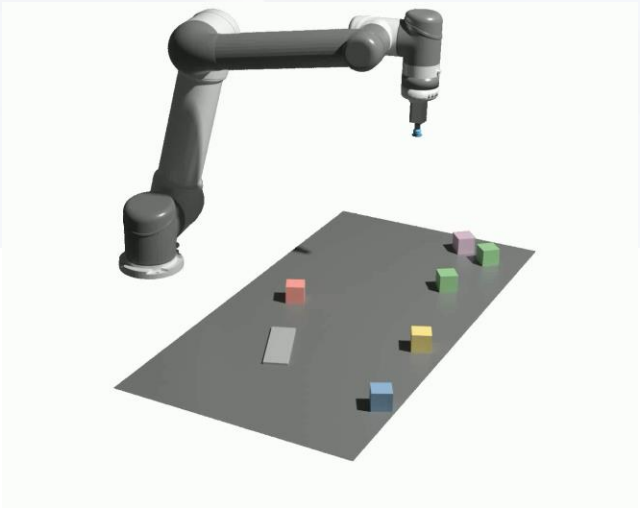


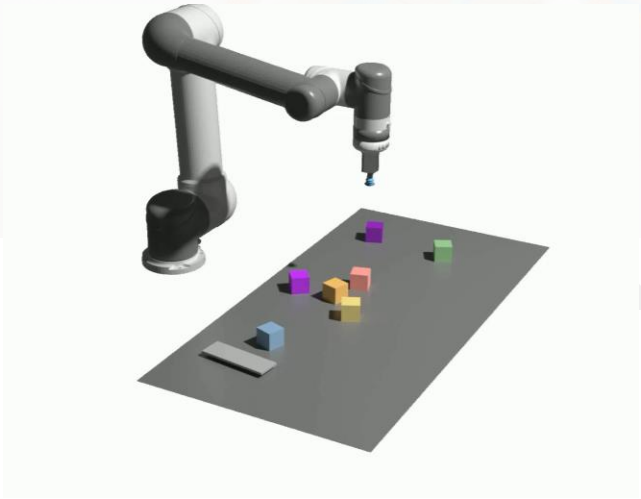
Massive Parallel Rollouts



Randomized Domains

Lack of Task Diversity





GenSim Framework

Can you generate the task “build-car”?

Large Language Model

Task Creator

GenSim Framework

Can you generate the task “build-car”?

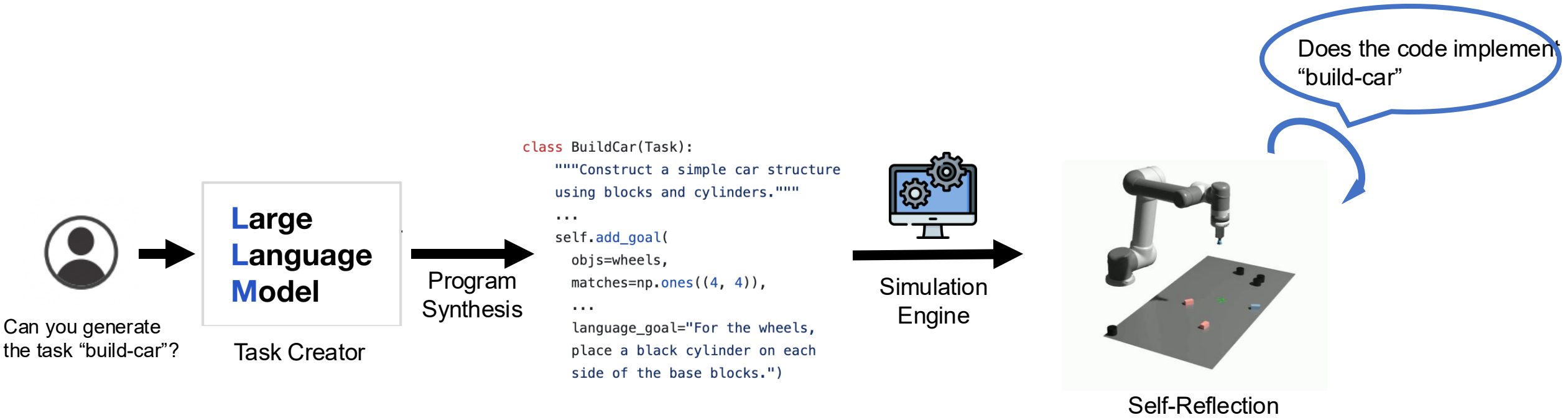
Large Language Model

Task Creator

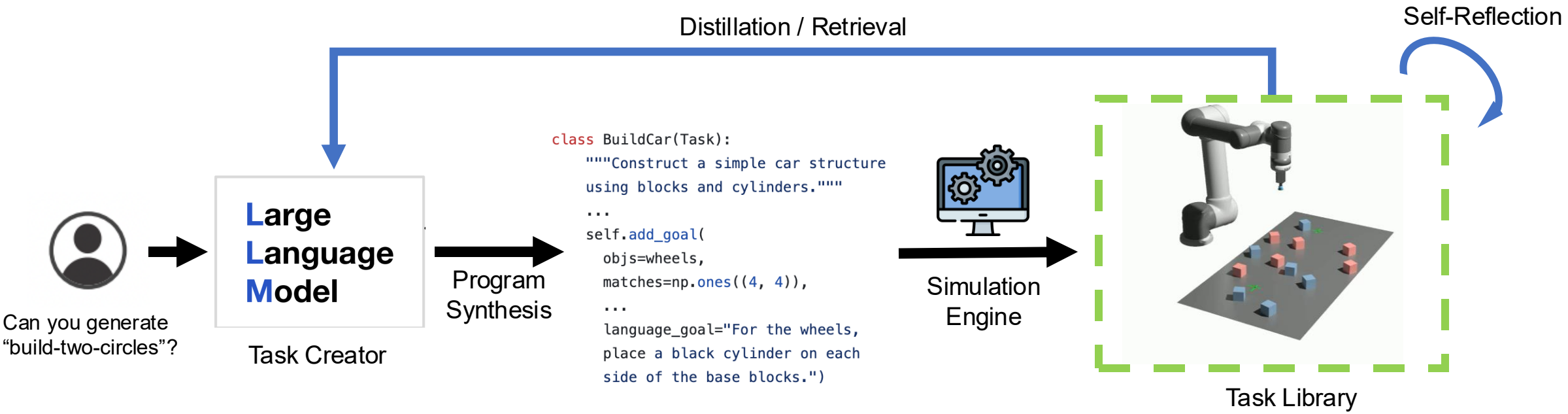
Program Synthesis

```
class BuildCar(Task):
    """Construct a simple car structure using blocks and cylinders."""

      ...
     def reset(self, env):
    # Add wheels.
    wheel_size = (0.02, 0.02, 0.02)  # x, y, z dimensions for the asset size
    wheel_urdf = 'cylinder/cylinder-template.urdf'

       ...
    for idx in range(4):
        wheel_pose = self.get_random_pose(env, wheel_size)
        wheel_id = env.add_object(wheel_urdf, wheel_pose, color=utils.COLORS['black'])
        wheels.append(wheel_id)

       ...

    self.add_goal(
      objs=wheels,
      matches=np.ones((4, 4)),
      ...
      language_goal="For the wheels, place a black cylinder on each side of the base blocks.")
```

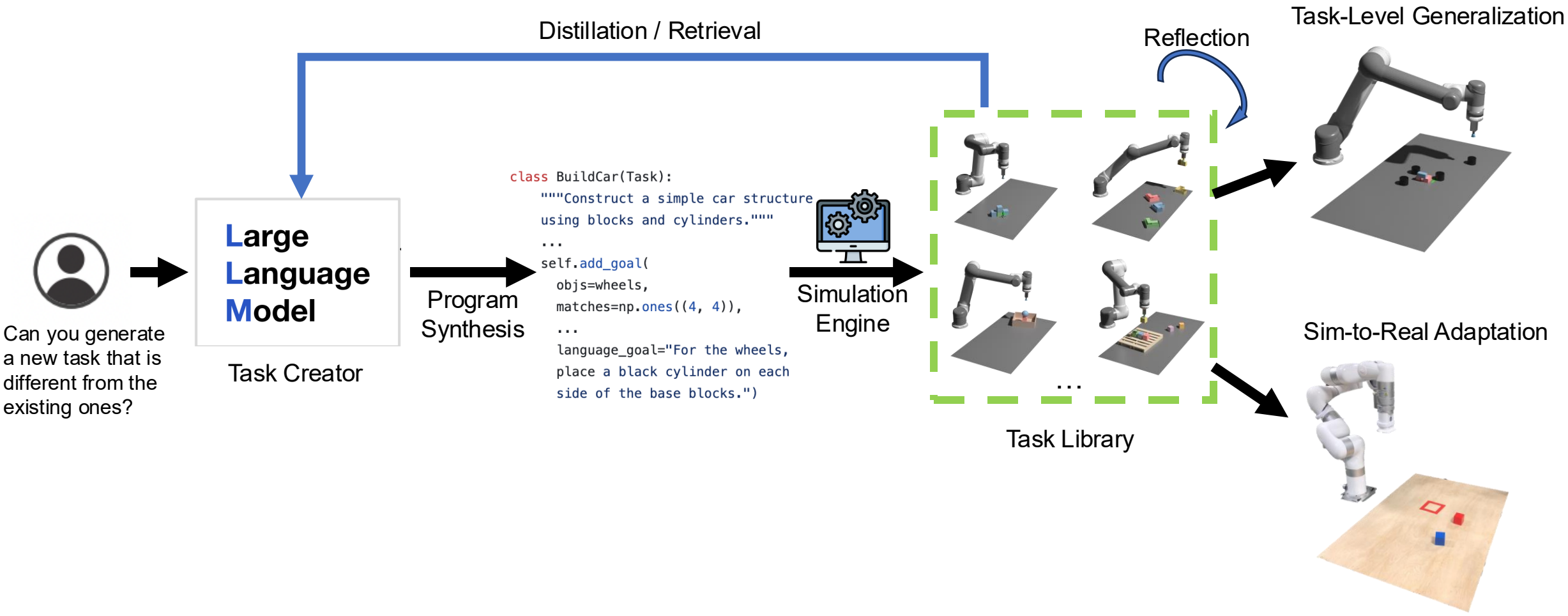
GenSim Framework



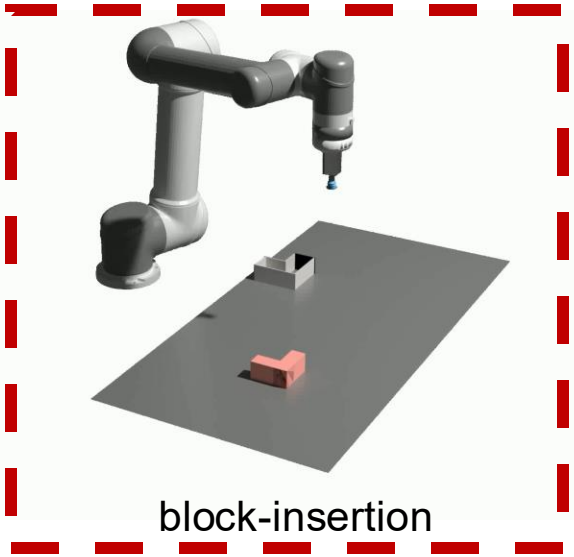
GenSim Framework

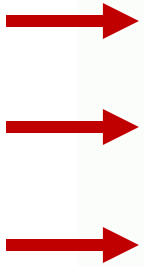


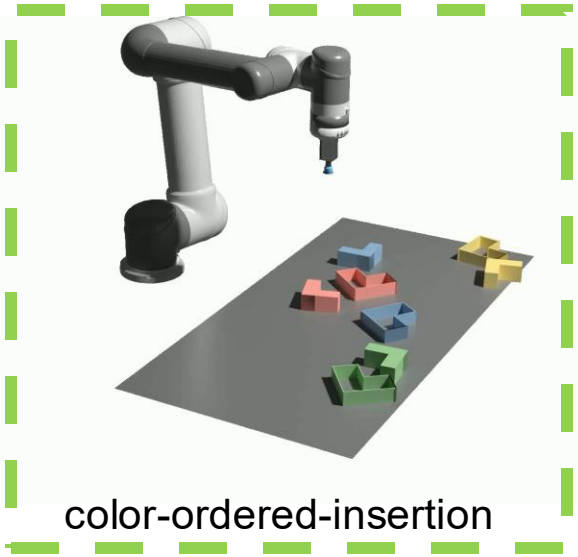
GenSim Framework



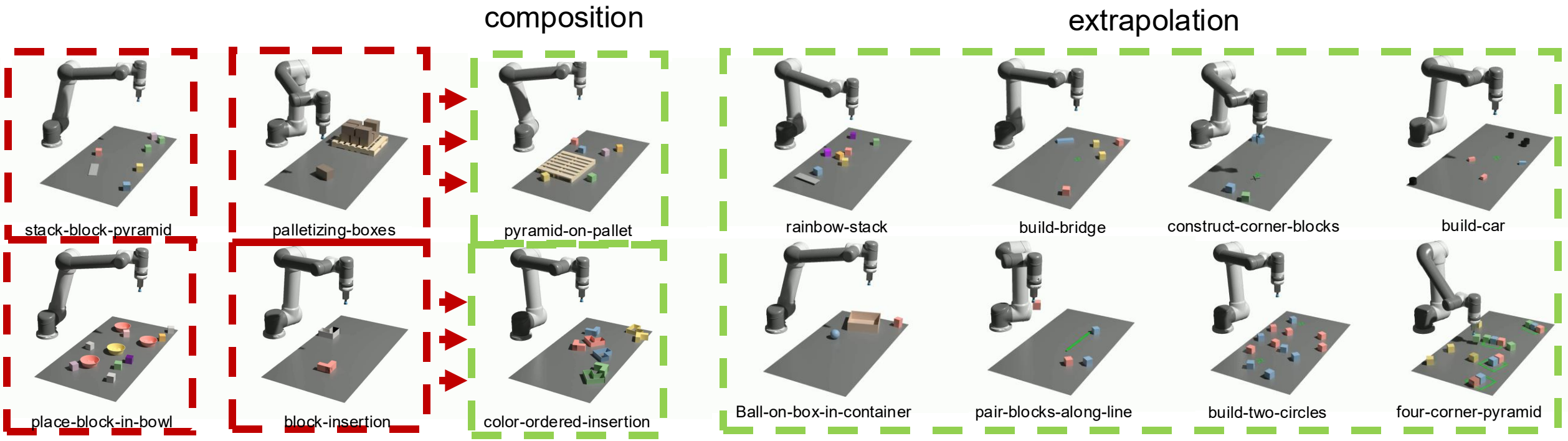
Generated Tasks





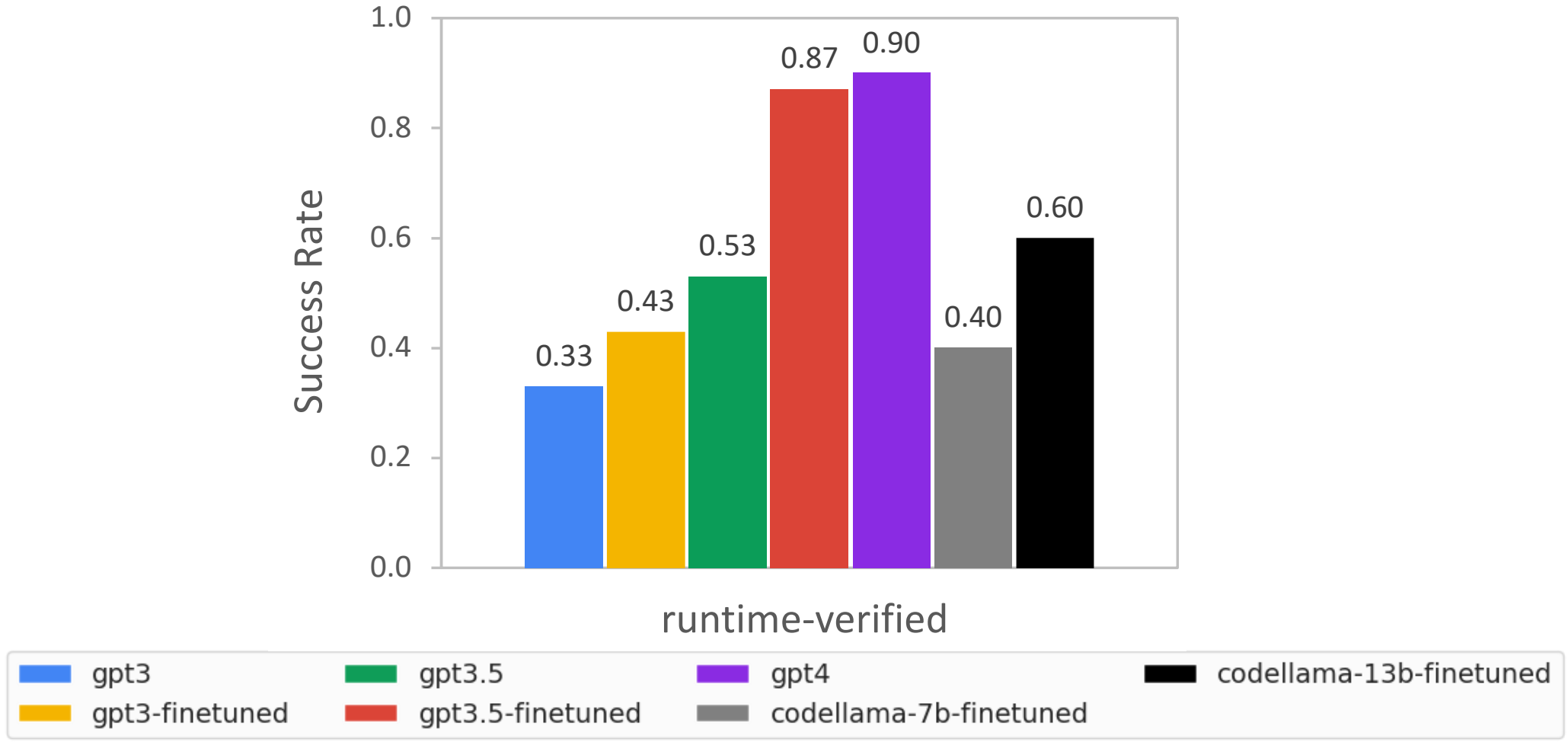


Generated Tasks



Experiments

LLM Task Generation Evaluation



Thank you!