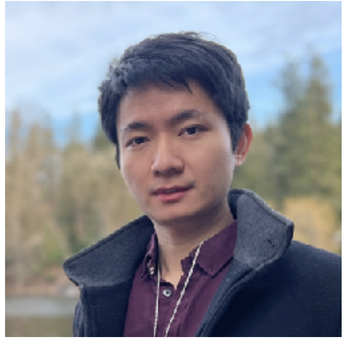
Hierarchy in Imitation and Reinforcement Learning

CS 224R

Course reminders

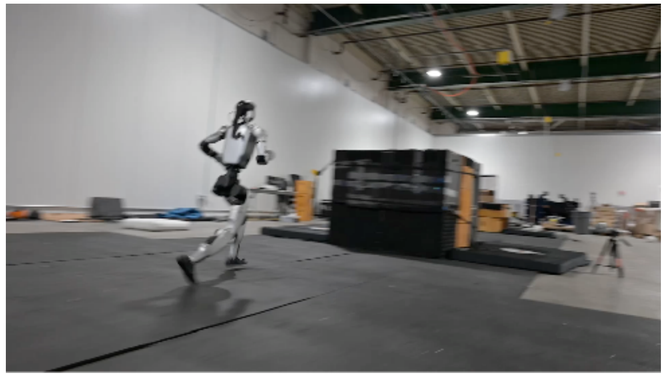
- Project milestone due Friday.
- Guest lecture on Friday!



Guanya Shi

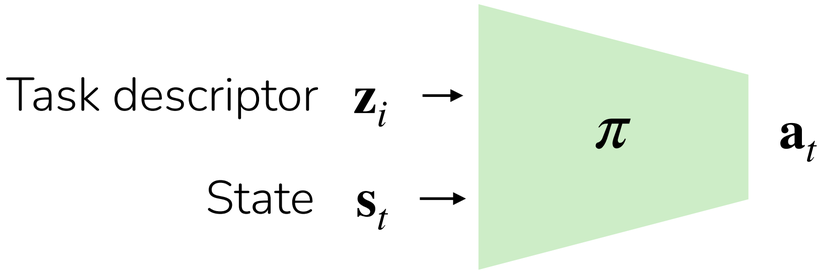
Amazon Frontier AI & Robotics (FAR)

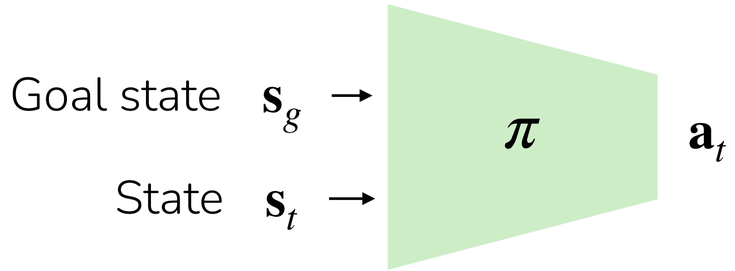
Assistant Prof. at CMU



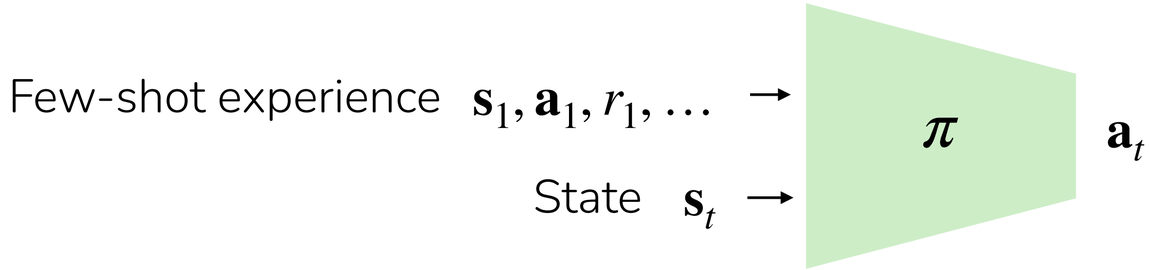
Brief recap

Multi-task imitation and reinforcement learning





Meta-reinforcement learning



Today: can we string together behaviors from multiple (sub)tasks?

Plan for today

1. What's the problem
    a. Why long-horizon tasks are hard
    b. Why hierarchy may help
2. Key design choices
    a. How to represent skills/goals?
    b. When to move on from one goal to the next?
    c. Supervision for each level
3. Example systems

Long horizon tasks

🤖 Cook focaccia.

🖥️ Fix a bug that is causing neural network training loss to explode.

🚗 Drive to Yosemite.

✏️ Give feedback on an 8-page report.

Why are these hard?

1. Very large # of states visited.
2. Many opportunities to make mistakes.
3. Many opportunities to get “stuck”.

The main idea

Can we break up hard tasks into easier subtasks?

Bake a cheesecake <- doing this is really hard

Buy ingredients

Go to the store

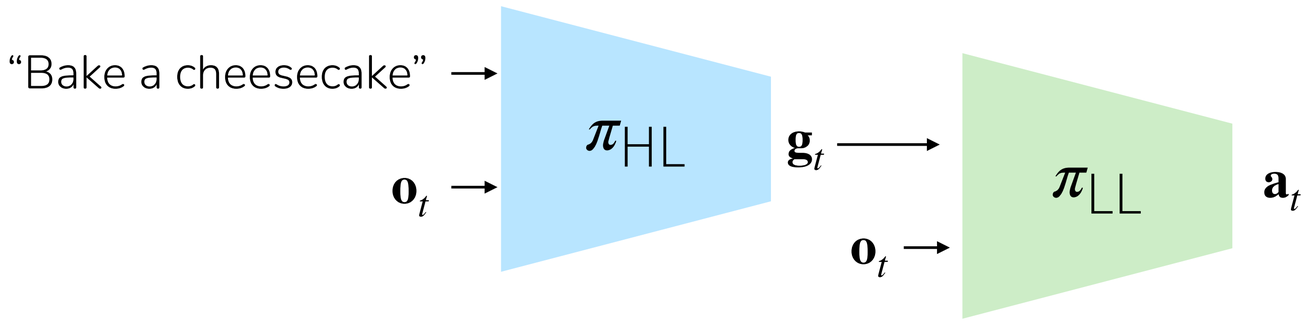
Walk to the door

Take a step <- doing this is easy

Contract quad muscle

The main idea

Can we break up hard tasks into easier subtasks?



High-level policy predicts intermediate goals $\mathbf{g}_t$

Low-level policy tries to accomplish $\mathbf{g}_t$ by predicting actions $\mathbf{a}_t$

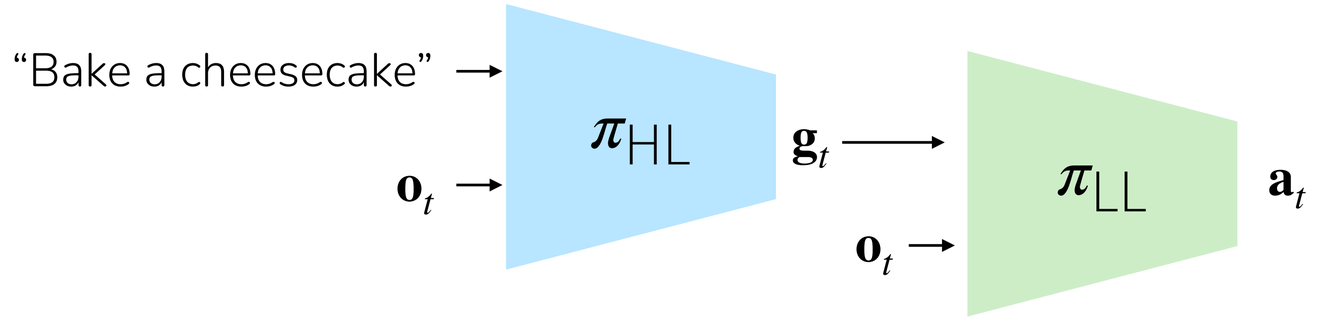
It operates at a *higher frequency* than the high-level policy.

Many names for $\mathbf{g}_t$: subgoals, subtasks, skills, options, high-level actions

Could have more than 2 levels of hierarchy!

The main idea

Can we break up hard tasks into easier subtasks?



High-level policy predicts intermediate goals $\mathbf{g}_t$

Low-level policy tries to accomplish $\mathbf{g}_t$ by predicting actions $\mathbf{a}_t$

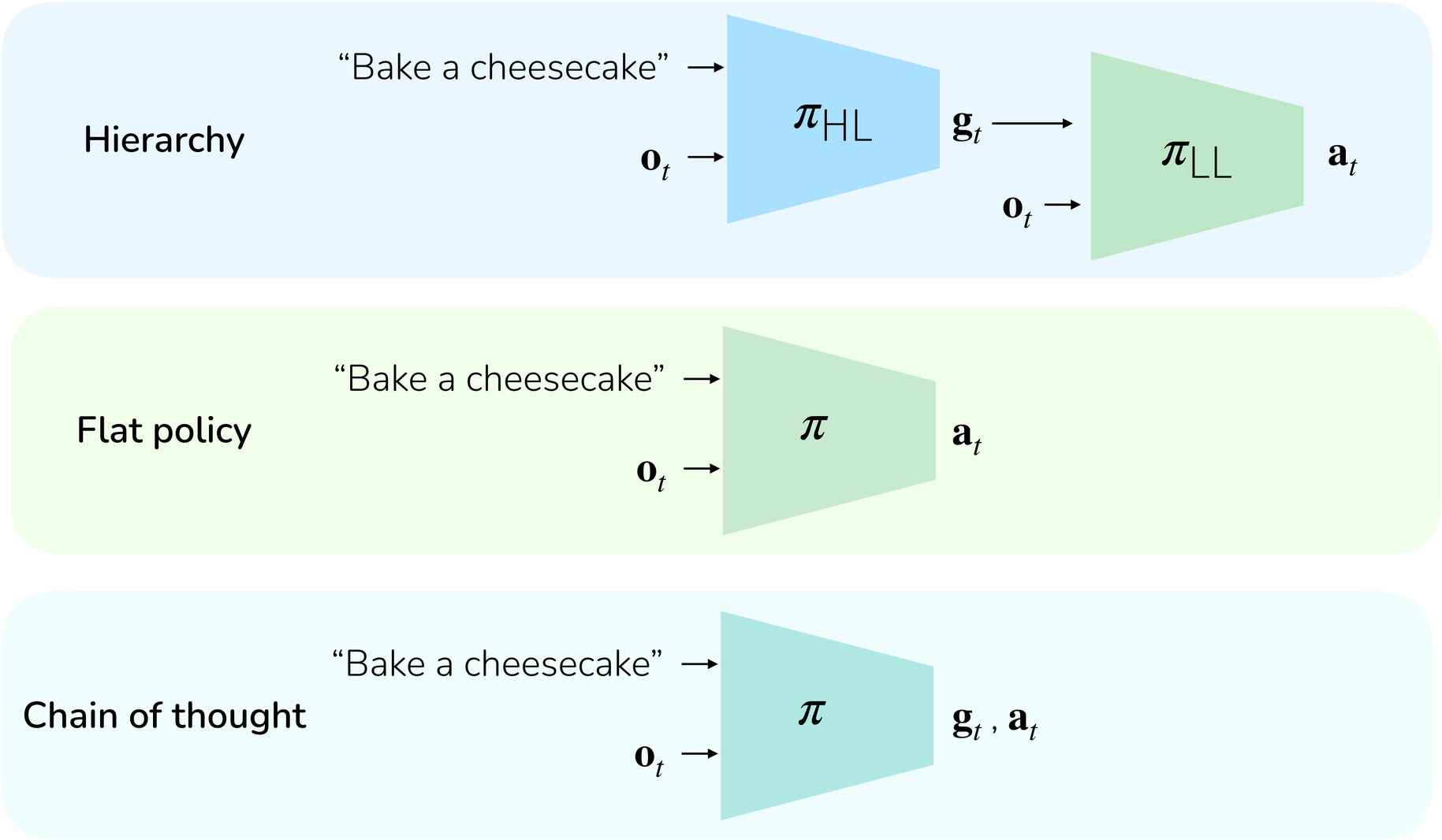
Rolling out the hierarchical policy:

0. Observe initial observation $\mathbf{o}_1$ at $t = 1$
1. Plan goal $\mathbf{g}_t$ according to $\pi_{\text{HL}}(\cdot \mid \mathbf{o}_t, \text{prompt})$
2. Until new goal is selected:
    a. Execute predicted action $\mathbf{a}_t$ according to $\pi_{\text{LL}}(\cdot \mid \mathbf{o}_t, \mathbf{g}_t)$
    b. $t \leftarrow t + 1$, observe new observation $\mathbf{o}_t$

Why may hierarchy help?

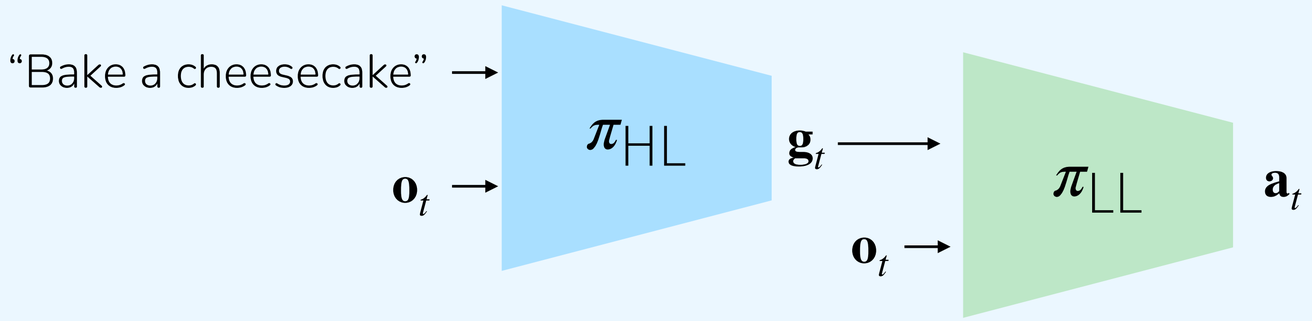
1. Supervision signal on how to complete the task
2. More direct knowledge sharing across similar subtasks
3. **In RL**: structured exploration in higher-level space
4. **Practical advantage**: Help with latency requirements by running polices at different frequencies

Do we *really* need multiple policies?

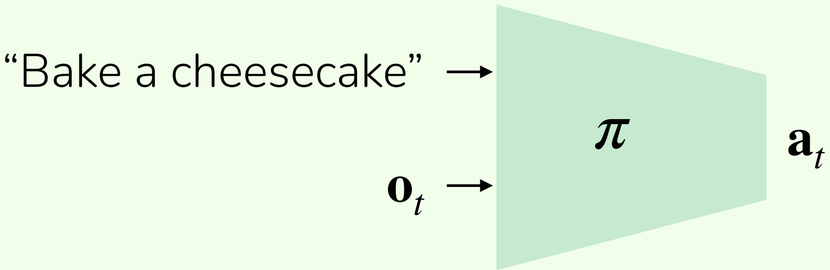


Do we *really* need multiple policies?

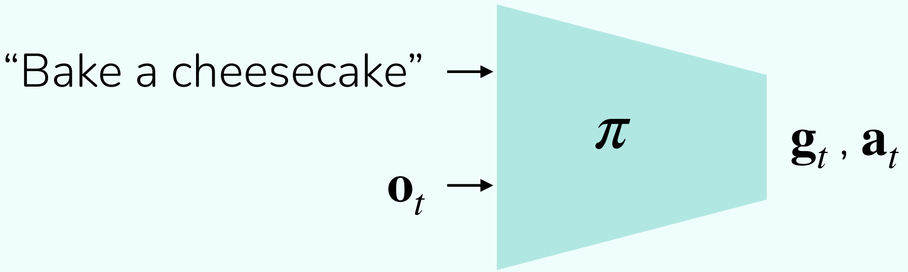
Hierarchy



Flat policy



Chain of thought



Can benefit from same supervision!

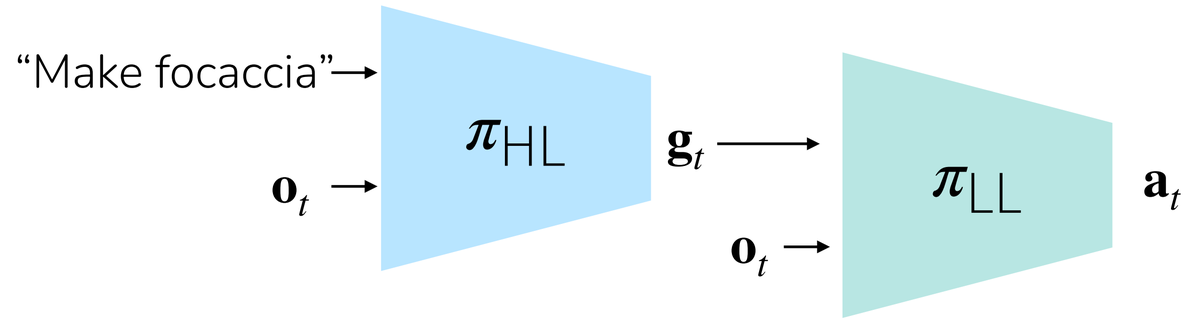
May be too computationally expensive (e.g. 50 Hz control)

No conclusive empirical comparison (yet).

Plan for today

1. What's the problem
    a. Why long-horizon tasks are hard
    b. Why hierarchy may help
2. **Key design choices**
    a. How to represent skills/goals?
    b. Supervision for each level
    c. When to move on from one goal to the next?
3. Example systems

What goal representation would you use?



Think-pair-share: for one task, discuss (1) intermediate goals, (2) the best representation for those goals

🤖 1. Cook Italian dishes.

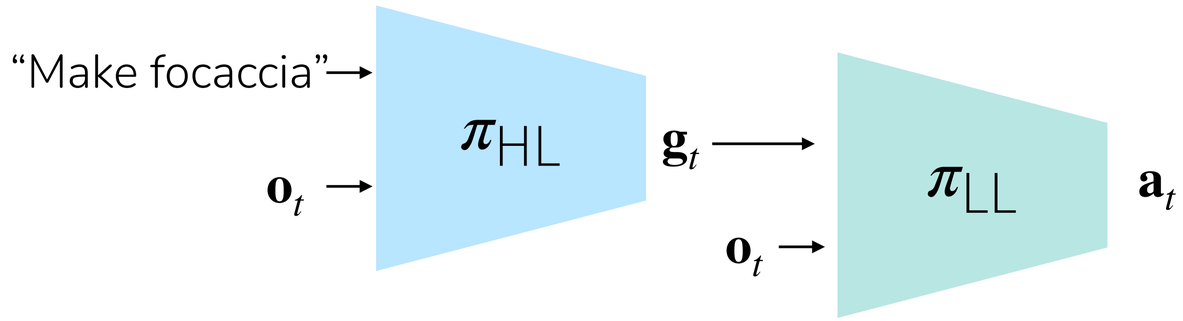
🚲 2. Bike to various campus locations

e.g. “Bike to the dish entrance.”

✏️ 3. Draft legal briefs.

🖥️ 4. Plan and book 1-week vacations for users.

What goal representation would you use?



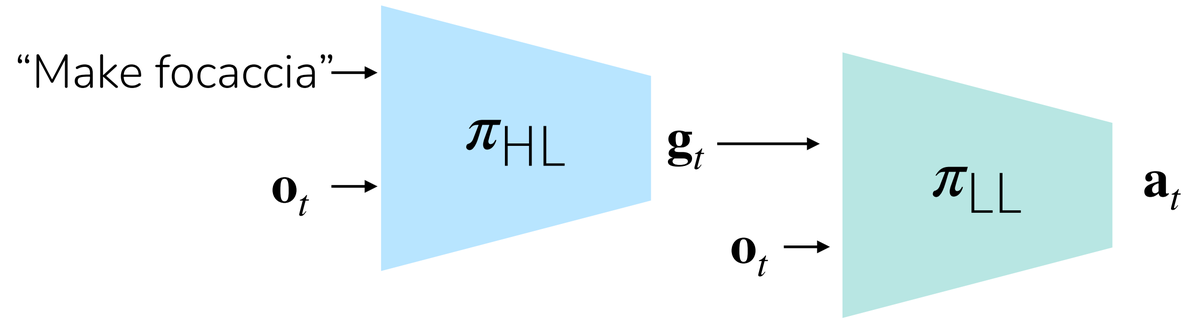
Summary:

1. Best goal representations will likely be domain specific!
2. Properties of good goal representations:
    a. Expressive (can communicate many different low-level behaviors)
    b. Structured (similar behaviors should have similar $\mathbf{g}$)
    c. Appropriate level of abstraction (not too hard for low-level or for high-level)

Plan for today

1. What's the problem
    a. Why long-horizon tasks are hard
    b. Why hierarchy may help
2. Key design choices
    a. How to represent skills/goals?
    b. Supervision for each level
    c. When to move on from one goal to the next?
3. Example systems

How to supervise each level?



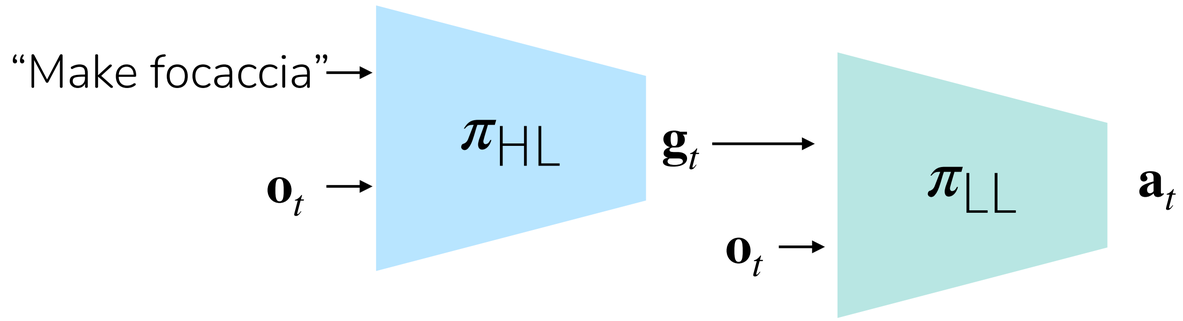
Why not train everything end-to-end, with latent goal representation?

Result: a flat policy

Key: Think carefully about where the benefits are coming from!

Generally good to do in any kind of research!!

How to supervise each level?



Low-level policy:

Trained to accomplish $\mathbf{g}$, not the original task!

Key question: For which distribution of goals? Ideally whatever the high-level policy will output

… but we haven’t learned a high-level policy yet.

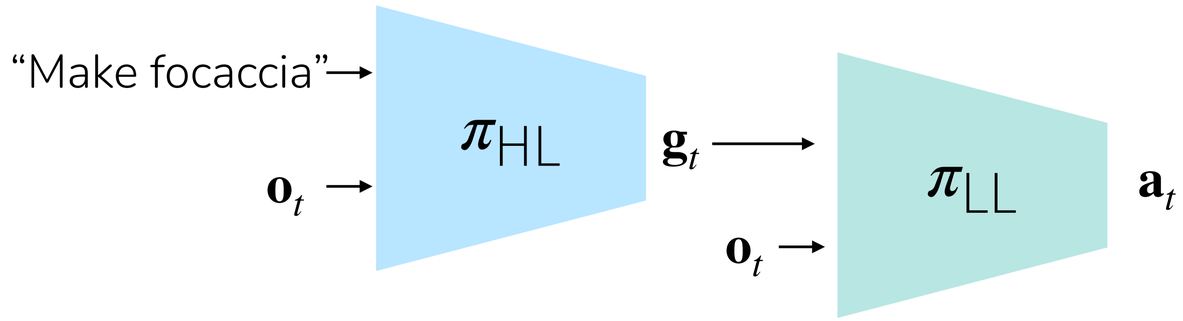
High-level policy:

Trained to accomplish original, long-horizon task

Key question: For which low-level policy? Ideally with the learned low-level policy

… but we haven’t learned a low-level policy yet.

How to supervise each level?



Low-level policy:

Trained to accomplish $\mathbf{g}$, not the original task!

High-level policy:

Trained to accomplish original, long-horizon task

Key: Can be trained separately first. But, at least one policy should be adapted to the deficiencies of the other.

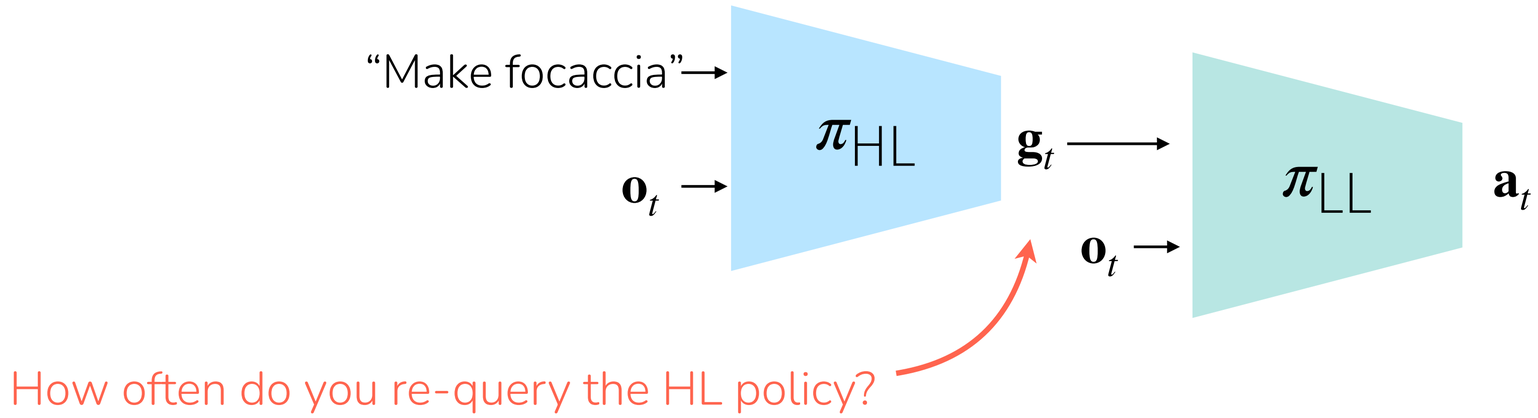
(if not both fine-tuned jointly)

Note: LLMs are often good high-level policies!

Plan for today

1. What's the problem
    a. Why long-horizon tasks are hard
    b. Why hierarchy may help
2. Key design choices
    a. How to represent skills/goals?
    b. Supervision for each level
    c. When to move on from one goal to the next?
3. Example systems

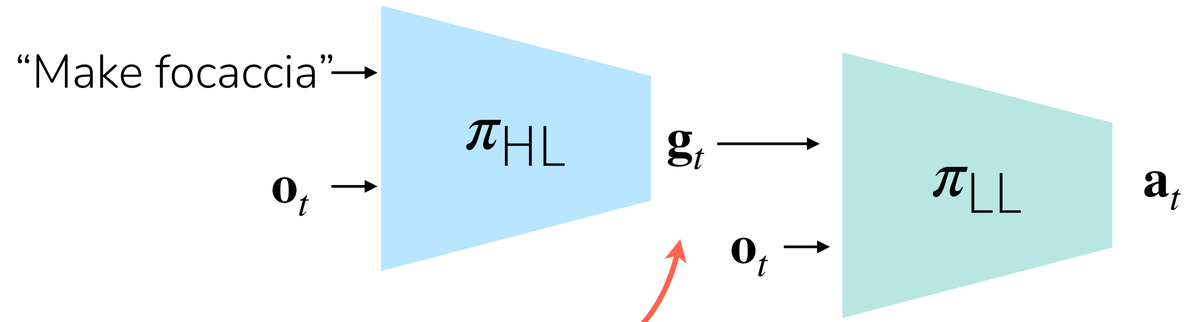
When to move on from one subgoal to the next?



Rolling out the hierarchical policy:

0. Observe initial observation $\mathbf{o}_1$ at $t = 1$
1. Plan goal $\mathbf{g}_t$ according to $\pi_{\text{HL}}(\cdot \mid \mathbf{o}_t, \text{prompt})$
2. Until new goal is selected:
    a. Execute predicted action $\mathbf{a}_t$ according to $\pi_{\text{LL}}(\cdot \mid \mathbf{o}_t, \mathbf{g}_t)$
    b. $t \leftarrow t + 1$, observe new observation $\mathbf{o}_t$

When to move on from one subgoal to the next?



Option 1: When the low-level policy has completed $\mathbf{g}_t$

e.g. by estimating progress towards $\mathbf{g}$

+ ideal in principle

- hard to estimate when done
- need to account for mistakes that require redoing past goals

Option 2: After fixed number of n timesteps.

e.g. by choosing fairly frequent interval to replan

+ simple

- tradeoff between more compute vs more delay in transition to new goal
- with small n, more burden on HL policy

Errors in estimating when done —>

agent easily gets perpetually stuck.

These errors are more fatal!

Errors in predicting subtask —>

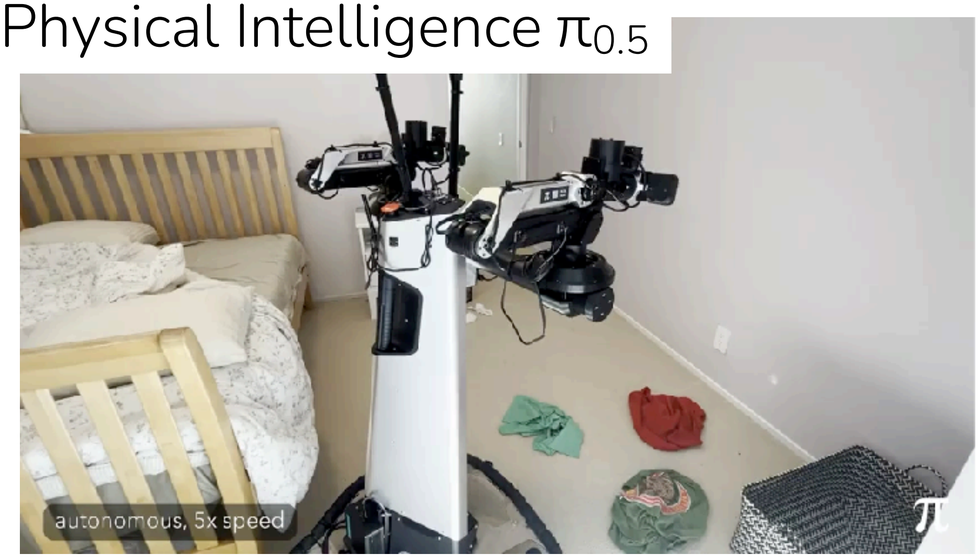
LL policy may take wrong actions for n steps.

Plan for today

1. What's the problem
    a. Why long-horizon tasks are hard
    b. Why hierarchy may help
2. Key design choices
    a. How to represent skills/goals?
    b. Supervision for each level
    c. When to move on from one goal to the next?
3. **Example systems**

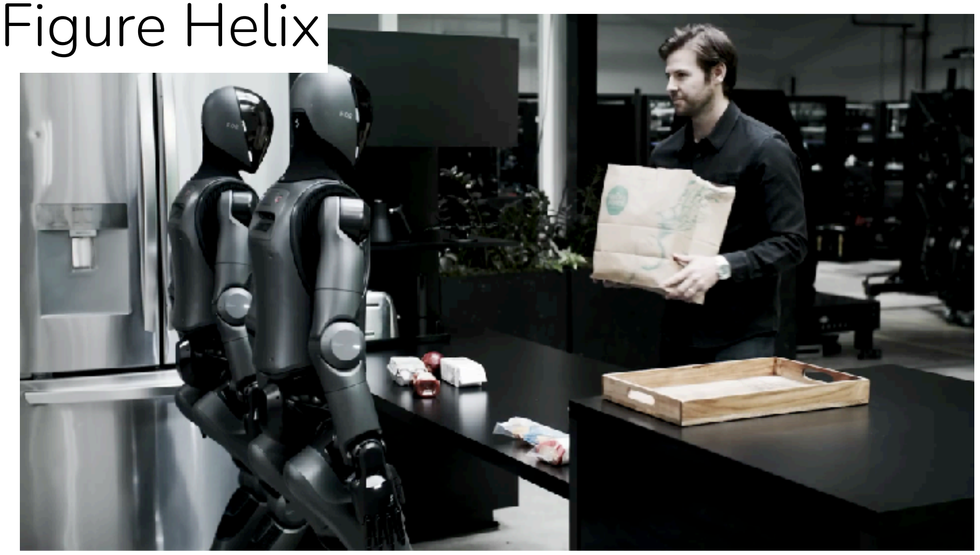
Hierarchy is hot 🌶️

Physical Intelligence $\pi_{0.5}$

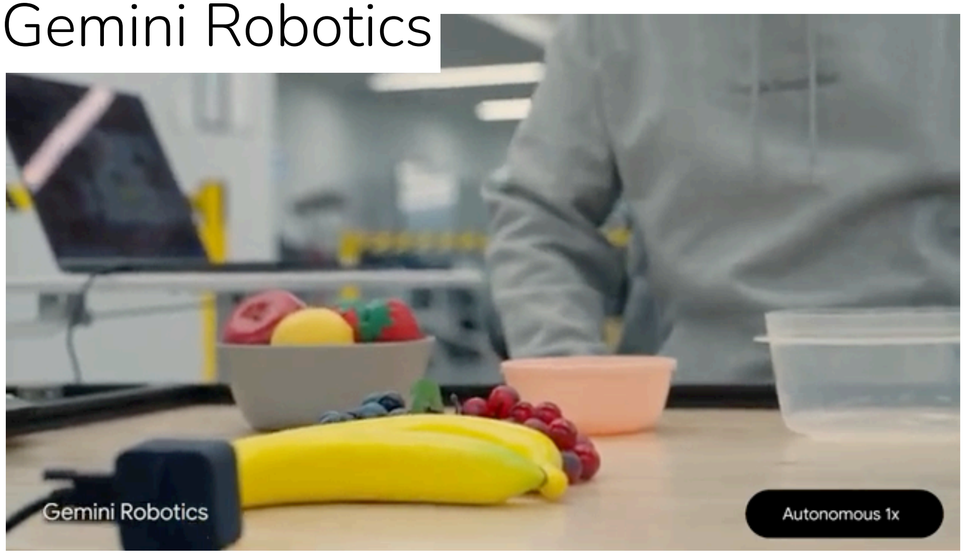


NVIDIA Gr00t N1

Figure Helix

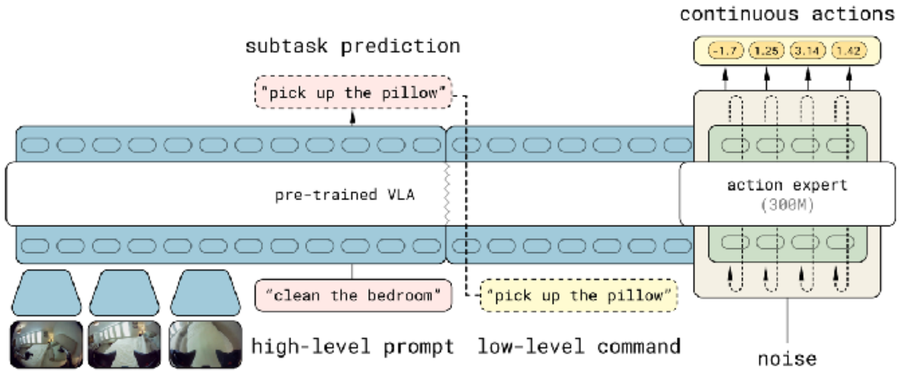


Gemini Robotics



Hierarchy is hot 🌶️

Physical Intelligence $\pi_{0.5}$



NVIDIA Gr00t N1

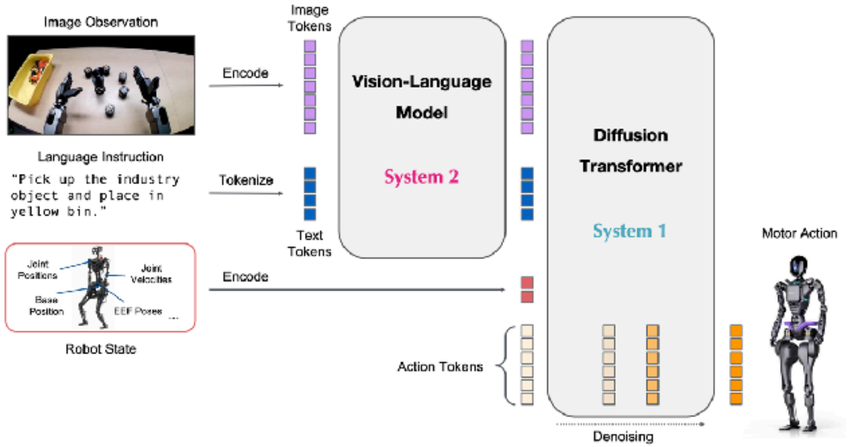
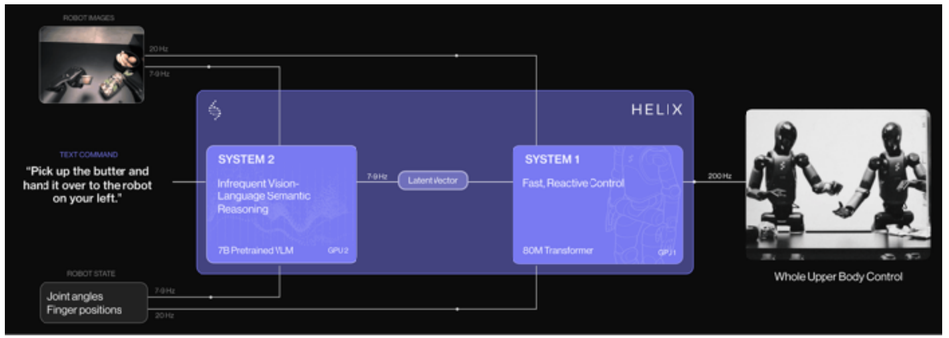
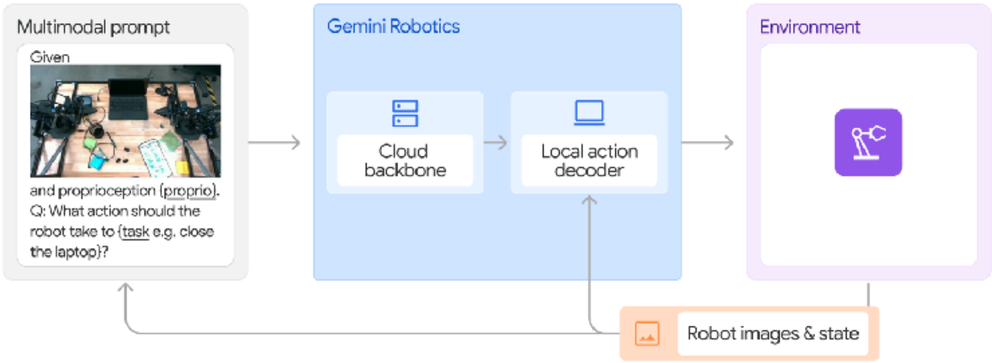


Figure Helix



Gemini Robotics



Let’s cover a few examples

1. Hierarchical imitation learning
    a. with language subgoals
    b. with image subgoals
2. Hierarchical reinforcement learning
    a. with language subgoals
    b. with state subgoals

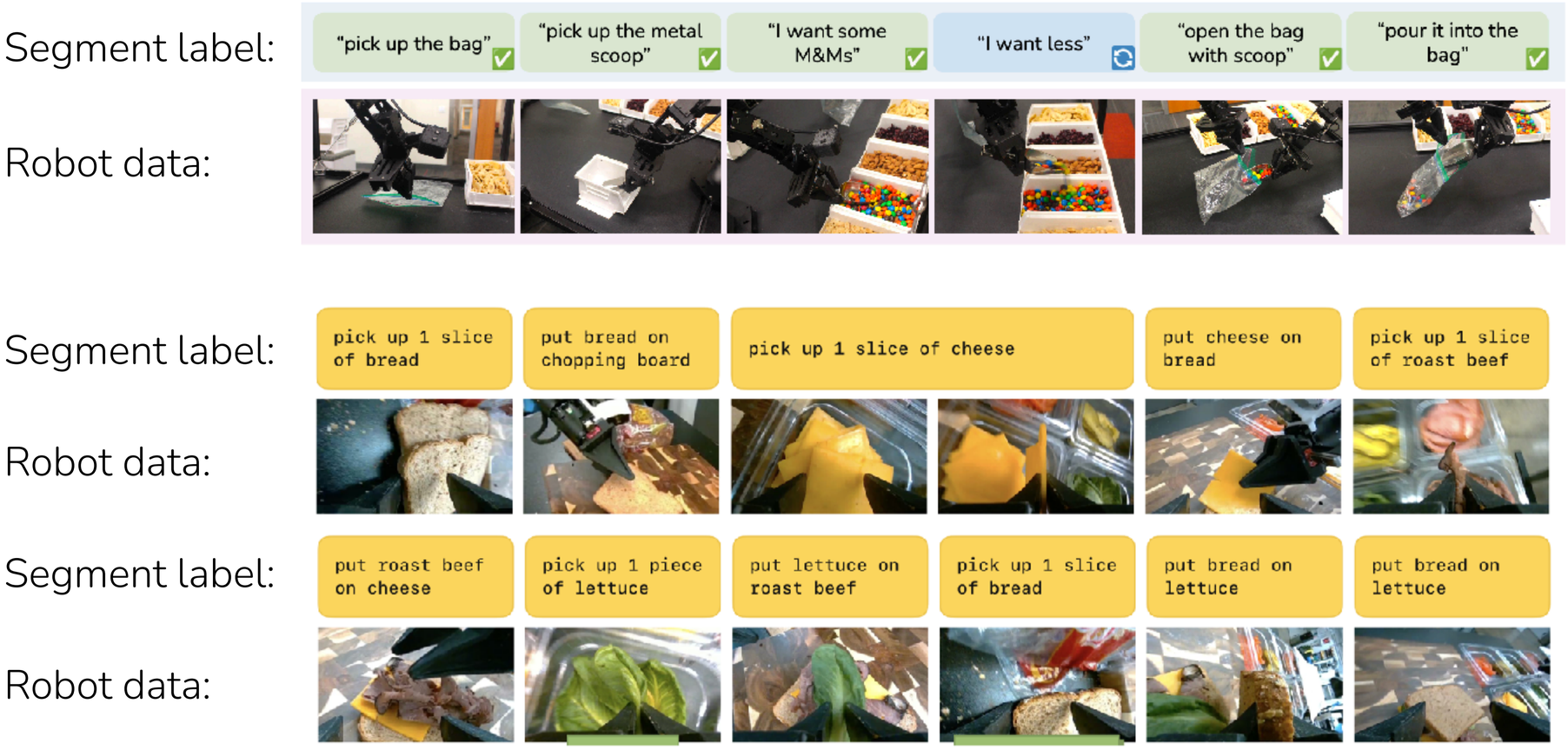
Let’s cover a few examples

1. Hierarchical imitation learning
    a. **with language subgoals**
    b. with image subgoals
2. Hierarchical reinforcement learning
    a. with language subgoals
    b. with state subgoals

Hierarchical imitation with language subgoals

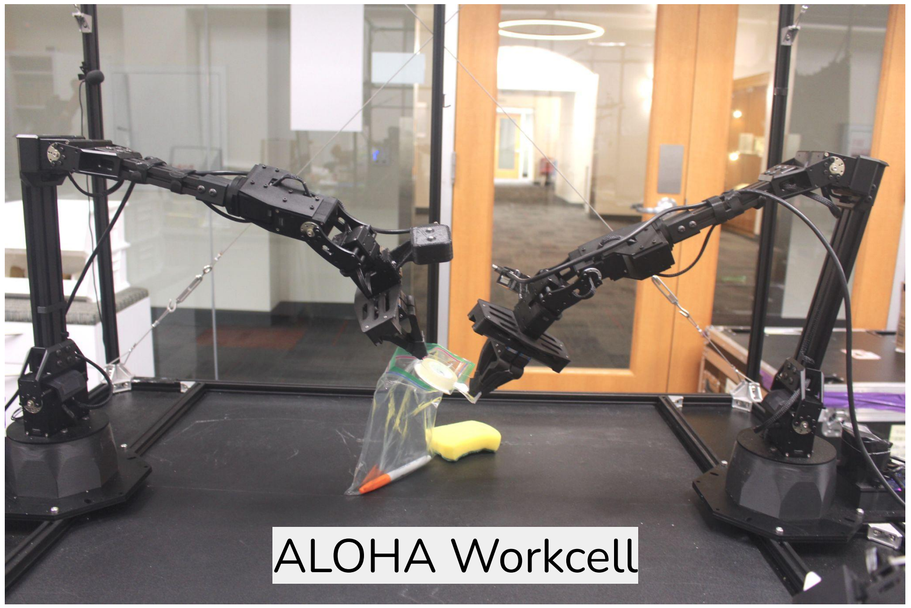
Data

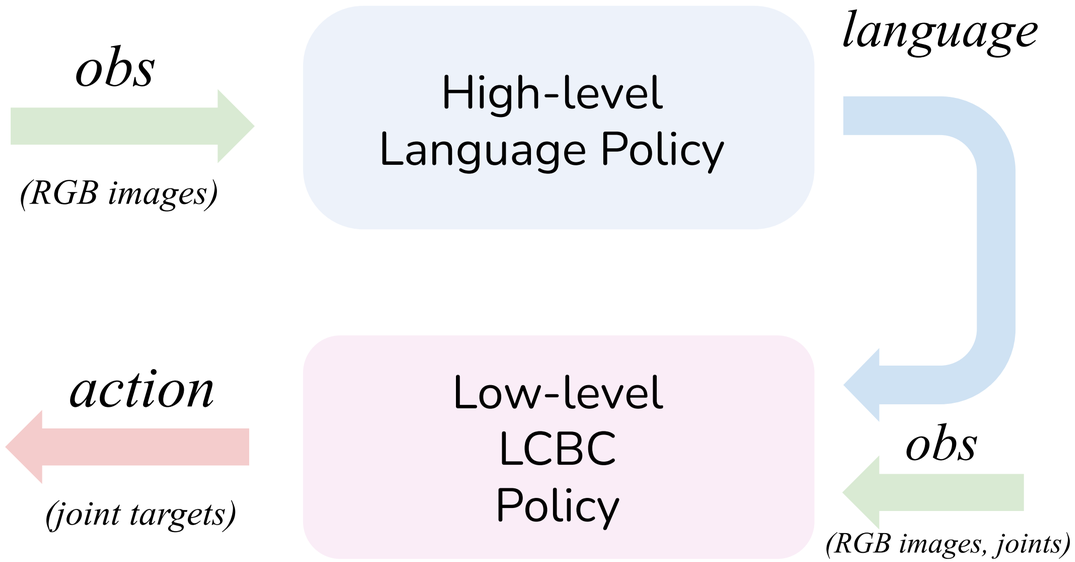
Segmented demonstration data with associated language command



Train high level and low-level policy with imitation learning.

Hierarchical imitation with language subgoals



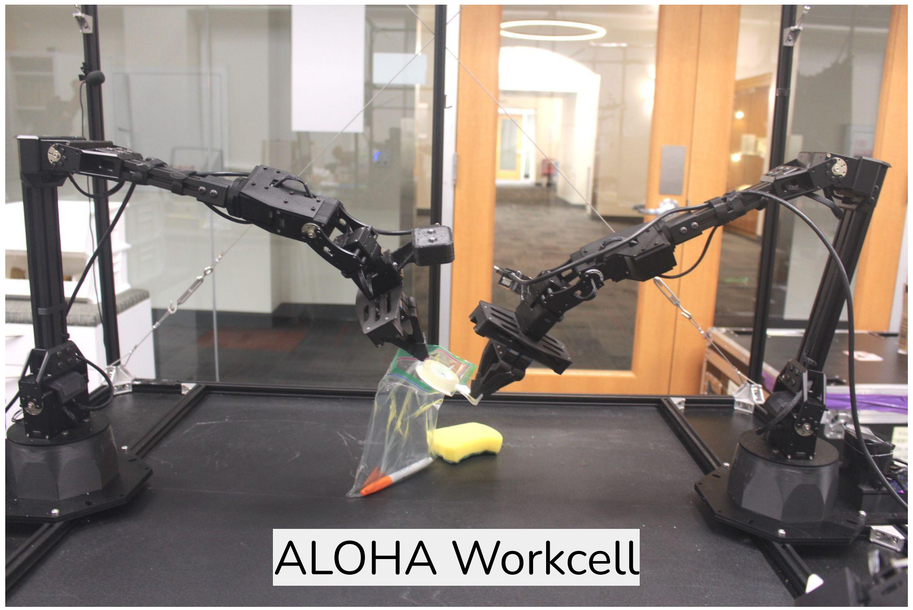


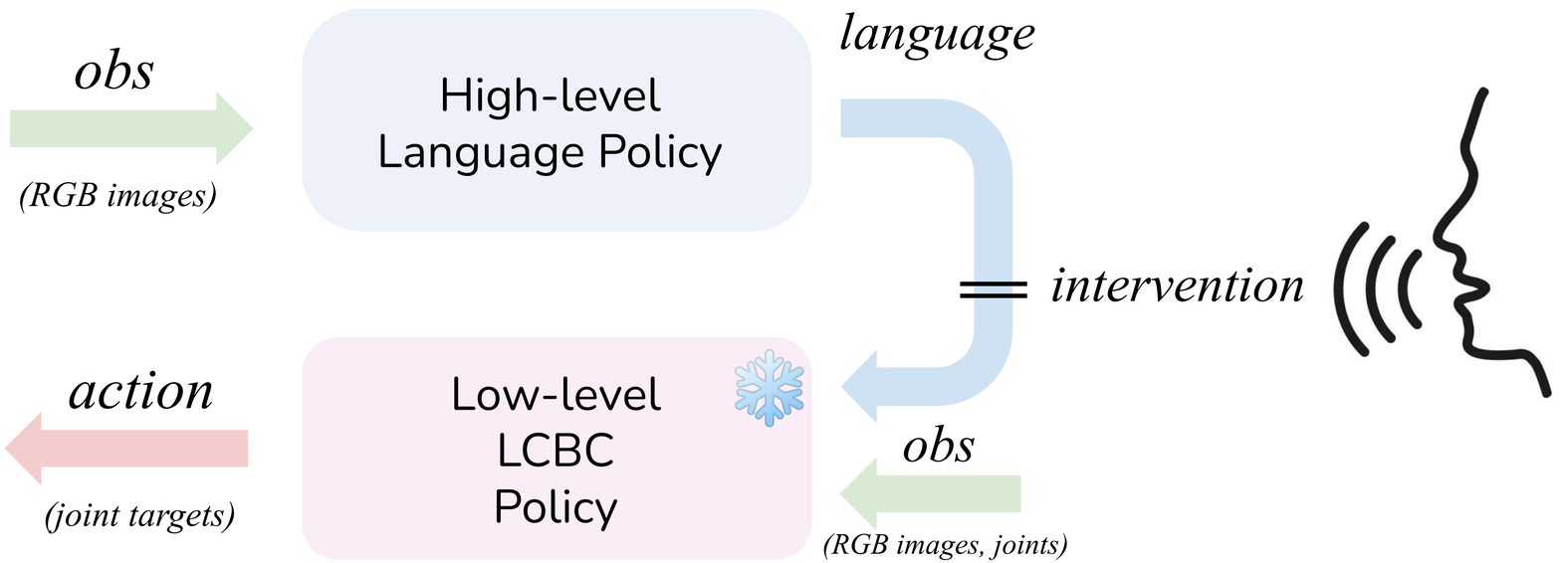
Could we also use DAgger on either policy? Yes!

Note: Can update the high-level policy with only language supervision!

Hierarchical imitation with language subgoals

Fine-tuning high-level policy with DAgger



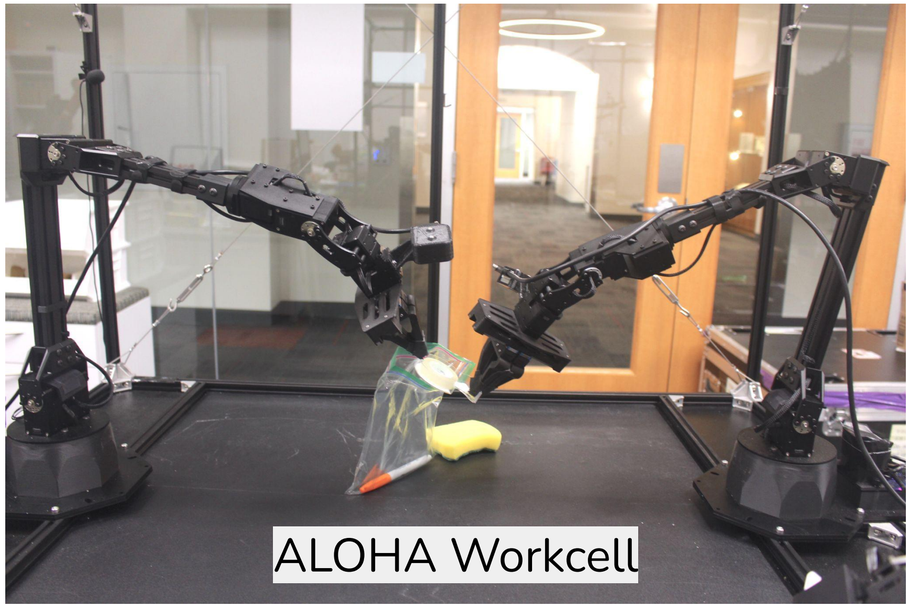


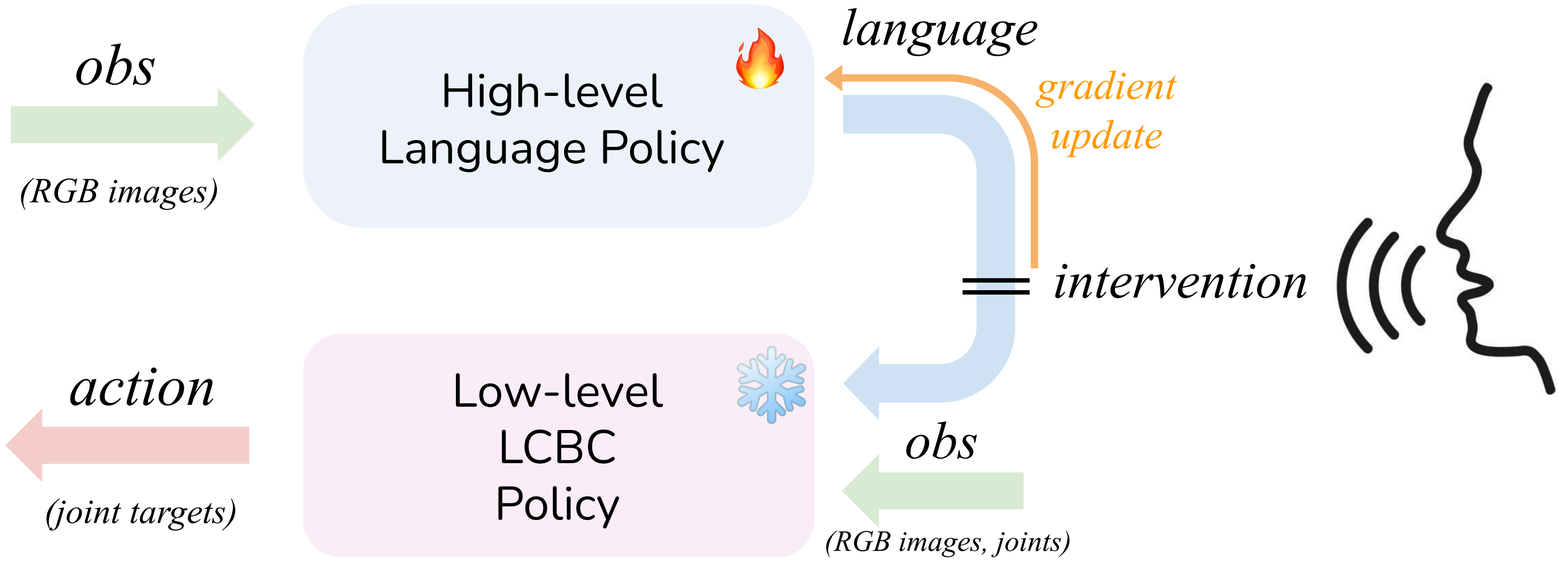
Language corrections override high-level policy prediction

Shi, Hu, Zhao, Sharma, Pertsch, Leo, Levine, Finn. Yell At Your Robot: Improving On-the-Fly from Language Corrections. RSS 2024

Hierarchical imitation with language subgoals

Fine-tuning high-level policy with DAgger





Language corrections override high-level policy prediction

HL DAgger: Freeze low-level policy. Update high-level policy by supervising on language corrections.

Shi, Hu, Zhao, Sharma, Pertsch, Leo, Levine, Finn. Yell At Your Robot: Improving On-the-Fly from Language Corrections. RSS 2024

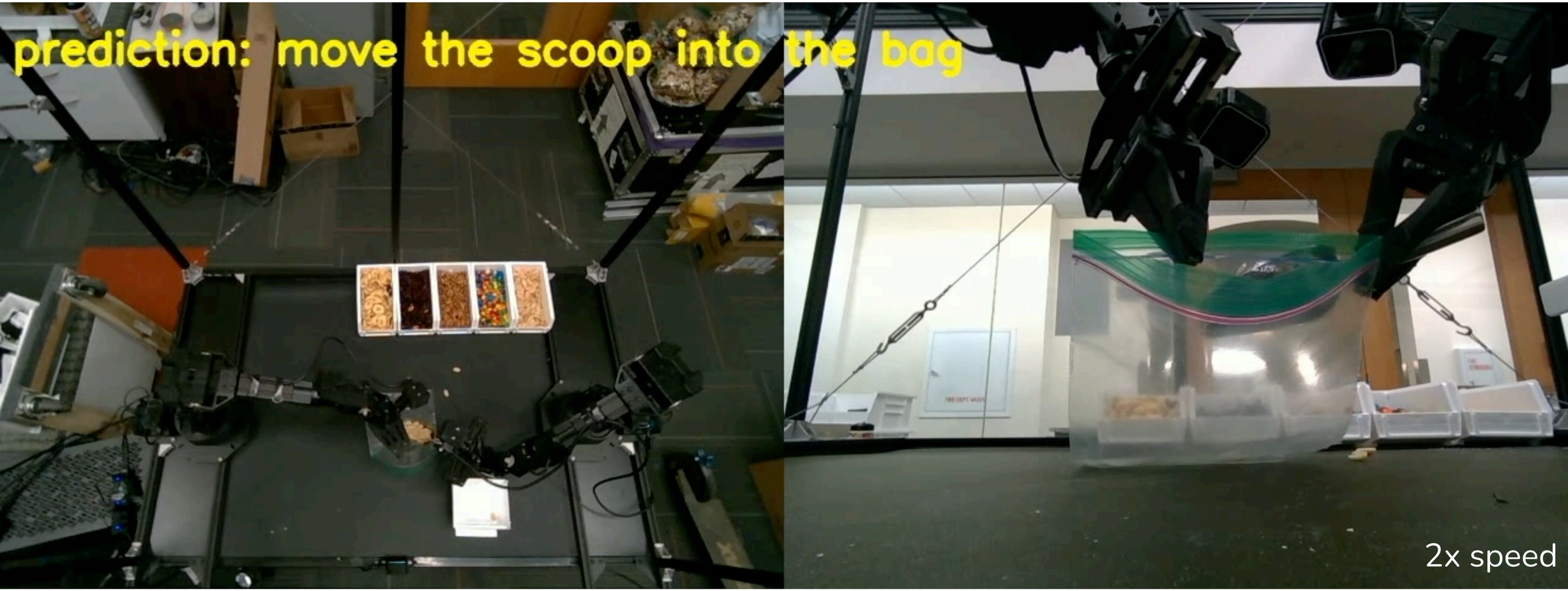
Language correction on-the-fly

(e.g. avoid pouring outside the bag)

Shi, Hu, Zhao, Sharma, Pertsch, Leo, Levine, Finn. Yell At Your Robot: Improving On-the-Fly from Language Corrections. RSS 2024

Language correction on-the-fly

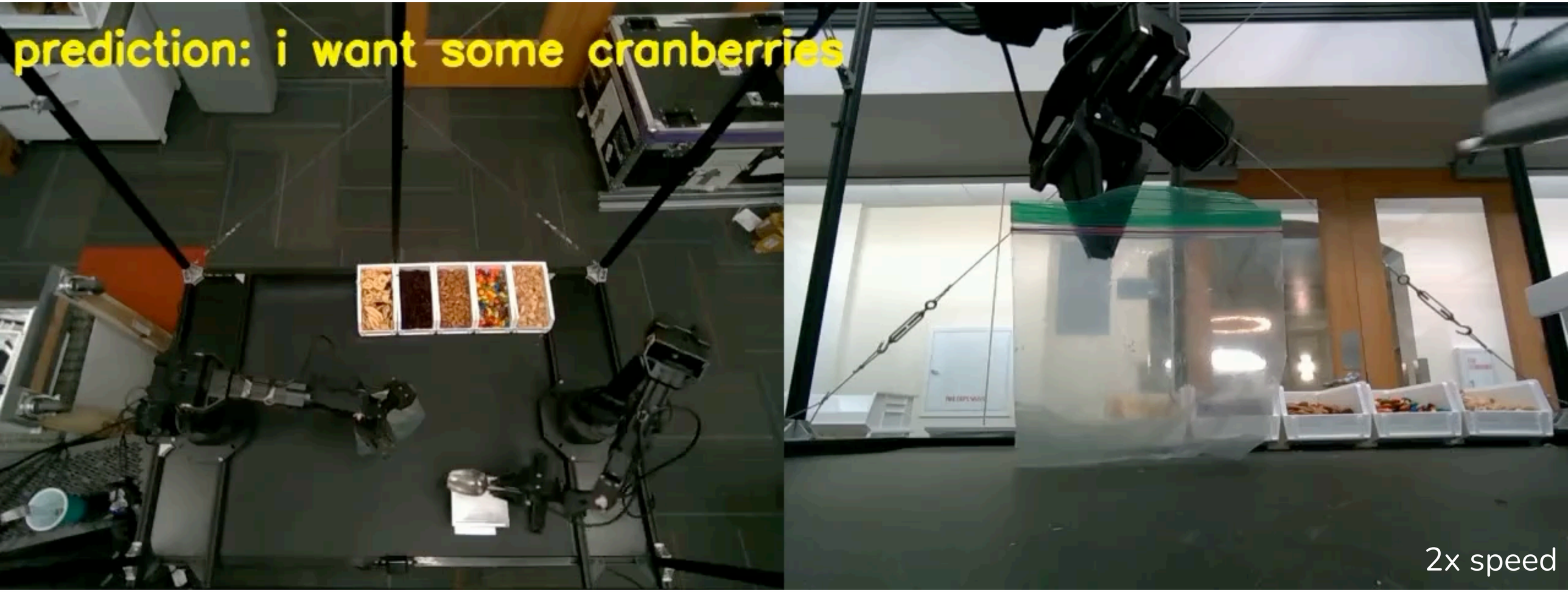
(e.g. avoid pouring outside the bag)



Shi, Hu, Zhao, Sharma, Pertsch, Leo, Levine, Finn. Yell At Your Robot: Improving On-the-Fly from Language Corrections. RSS 2024

After fine-tuning high-level policy with DAgger

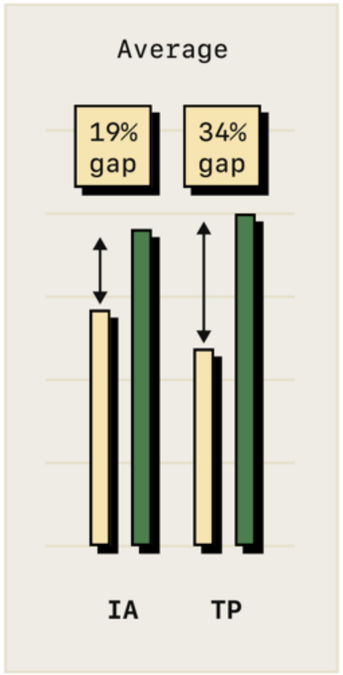
(the policy can self-correct)



Shi, Hu, Zhao, Sharma, Pertsch, Leo, Levine, Finn. Yell At Your Robot: Improving On-the-Fly from Language Corrections. RSS 2024

Hierarchical imitation with language abstraction

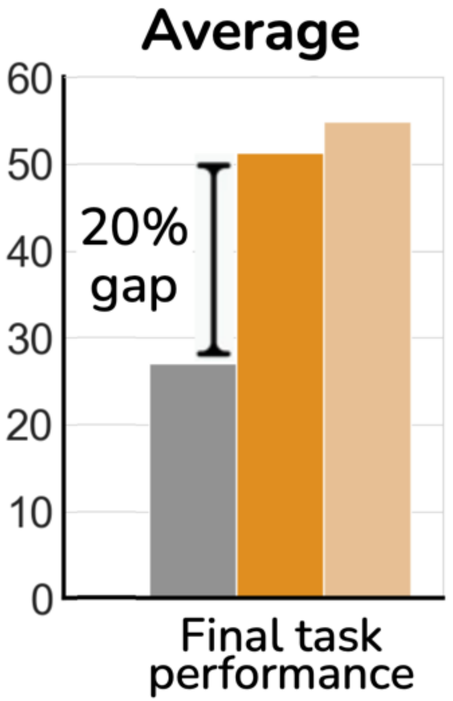
Does hierarchy help compared to flat policy?



Flat VLA policy

Hierarchical VLA policy

Does high-level DAgger help vs. vanilla imitation?



Base hierarchical policy

With high-level DAgger

With human corrections (oracle)

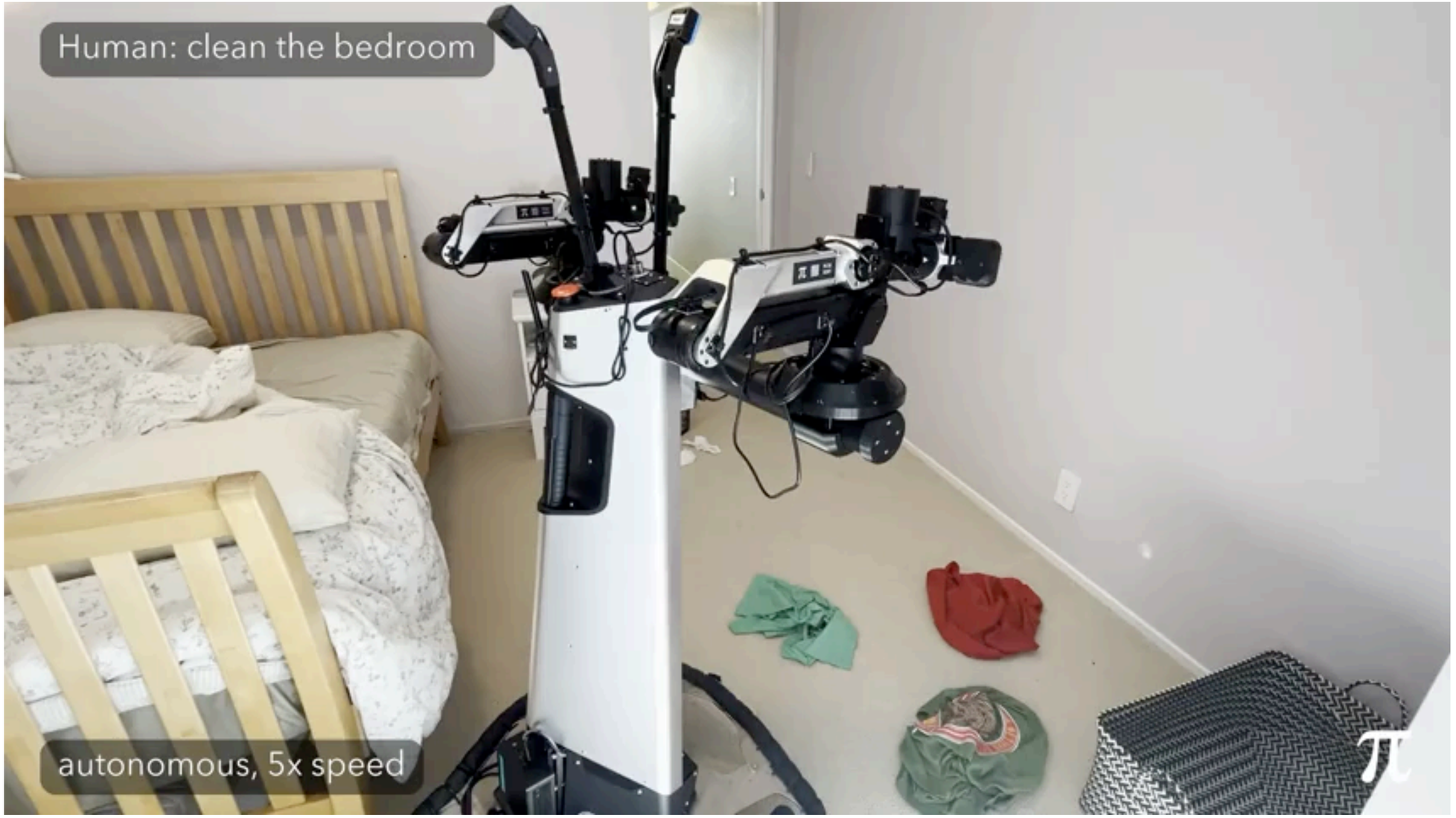
Lucy Shi et al. Hi Robot. ICML 2025

Lucy Shi, Hu, Zhao, Sharma, Pertsch, Leo, Levine, Finn. Yell At Your Robot: Improving On-the-Fly from Language Corrections. RSS 2024

With hierarchy, the robot is better at long-horizon tasks

Shi, Hu, Zhao, Sharma, Pertsch, Leo, Levine, Finn. Yell At Your Robot: Improving On-the-Fly from Language Corrections. RSS 2024

With hierarchy, the robot is better at long-horizon tasks

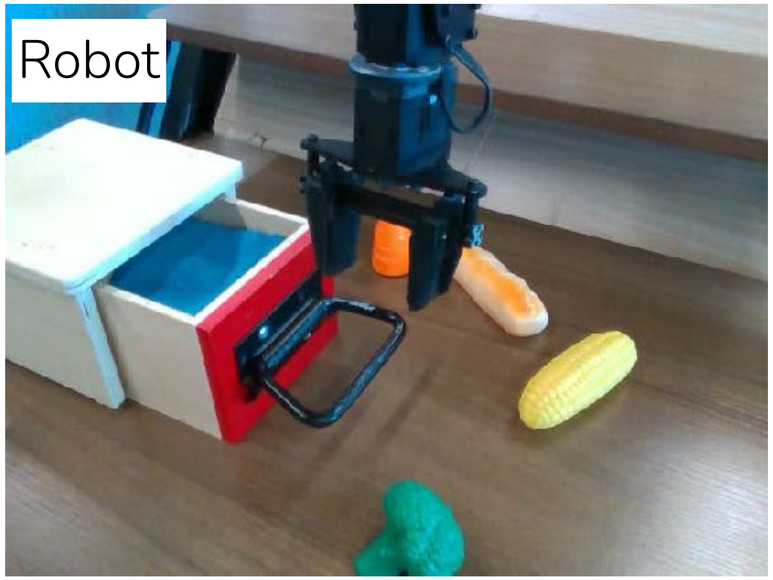


Pi team. $\pi_{0.5}$. arXiv 2025

Let’s cover a few examples

1. Hierarchical imitation learning
    a. with language subgoals
    b. with image subgoals
2. Hierarchical reinforcement learning
    a. with language subgoals
    b. with state subgoals

Hierarchical imitation with image subgoals



obs

(RGB images)

High-level policy

~~*language*~~ goal image

In practice: use an image editing model

action

(joint targets)

Low-level
~~LCBC~~
Policy

obs

(RGB images, joints)

goal image conditioned policy

Benefits: No need for segmented videos with language annotations.

High-level policy can incorporate unlabeled video data!

Hierarchical imitation with image subgoals

A few examples

“move the wooden bowl to the top of the table”

“put the toothpaste into the wooden bowl”

“open the drawer”

Black*, Nakamoto*, Atreya, Walke, Finn, Kumar, Levine. **SuSIE: Subgoal Synthesis via Image Editing.** ICLR 2024.

Hierarchical imitation with image subgoals

Is it helpful to include videos of humans in high-level policy training?

	Task	BridgeData Only	BridgeData + Something-Something
Scene B	Bell pepper in pot	0.2	**0.5**
	Bell pepper in bowl	0.4	**0.5**
	Average	0.30	**0.50**
Scene C	Toothpaste in bowl	**0.7**	0.6
	Crayon in bowl	0.9	**1.0**
	Spoon in bowl	**0.9**	**0.9**
	Bowl to top	0.7	**1.0**
	Average	0.80	**0.88**

Robot data only

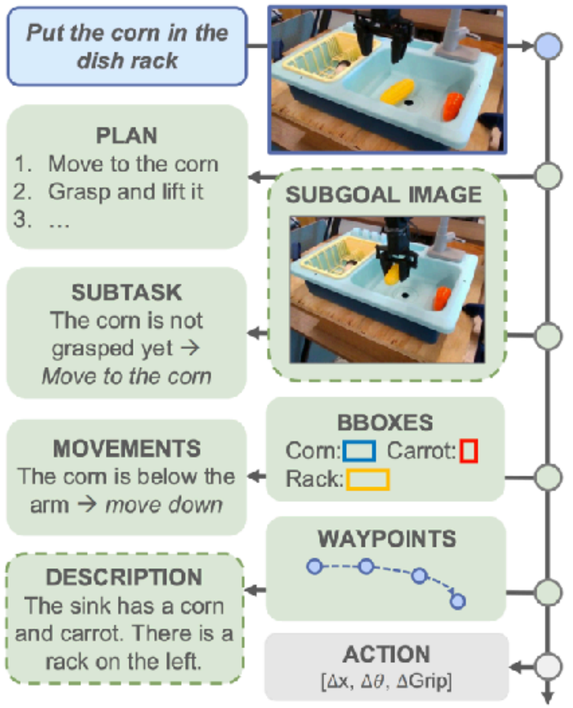
Robot data + videos of humans

Black*, Nakamoto*, Atreya, Walke, Finn, Kumar, Levine. **SuSIE: Subgoal Synthesis via Image Editing.** ICLR 2024.

Let’s cover a few examples

1. Hierarchical imitation learning
   a. with language subgoals
   b. with image subgoals
2. Hierarchical reinforcement learning
   a. with language subgoals
   b. with state subgoals

Chain of thought version:



Visual from Chen, Belkhale, et al. *Training Strategies for Efficient Embodied Reasoning*, 2025.

Note: Fine-tuning large-scale hierarchical robot learning systems with RL is an open (and important!) research direction.

Let’s cover a few examples

1. Hierarchical imitation learning
    a. with language subgoals
    b. with image subgoals
2. **Hierarchical reinforcement learning**
    a. with language subgoals
    b. with state subgoals

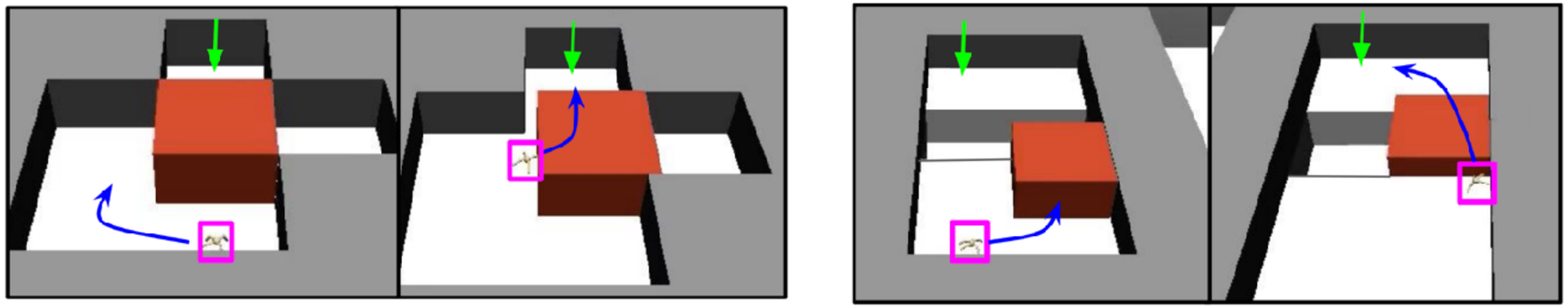
Hierarchical reinforcement learning

State reaching goals (e.g. relative target position of agent & objects)

Low-level: Goal-conditioned policy, with goal-reaching reward

High-level: Train policy to output goal states (these are the HL actions)

Use hindsight relabeling *on HL actions*, using off-policy algorithm



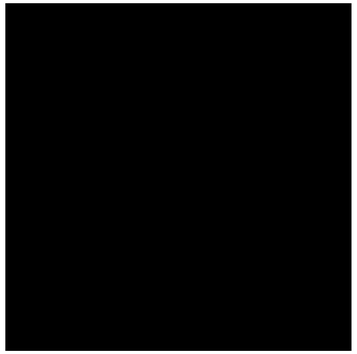
Nachum, Gu, Lee, Levine. Data-Efficient Hierarchical Reinforcement Learning. NeurIPS '18.

Language goals (for simple semantic tasks)

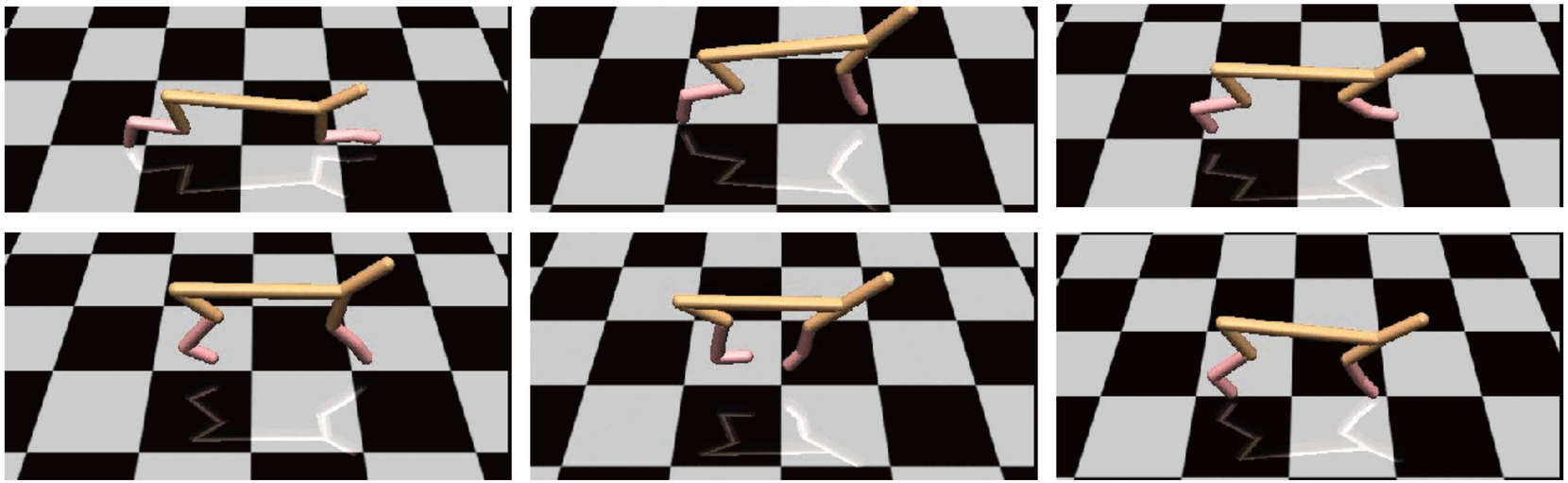
Low-level: Language-conditioned policy, with hindsight *language* relabeling

High-level: Train policy to output language

Jiang, Gu, Murphy, Finn. Language as an Abstraction for Hierarchical Deep Reinforcement Learning. NeurIPS '19.



Another active research area:
Can you *discover* a set of diverse skills without supervision?



Eysenbach, Gupta, Ibarz, Levine. Diversity is all you need: Learning diverse skills without a reward function. ICLR 2019

Hierarchy for Imitation and Reinforcement Learning

1. What's the problem
    a. Why long-horizon tasks are hard
    b. Why hierarchy may help
2. Key design choices
    a. How to represent skills/goals?
    b. Supervision for each level
    c. When to move on from one goal to the next?
3. Example systems

Next time

Specific considerations with RL for robotics:

- sim2real transfer: guest lecture from Guanya Shi! (Amazon FAR)
- RL for robot foundation models (i.e. VLAs)

Course reminders

- Project milestone due Friday.