Multi-Task and Goal-Conditioned Reinforcement Learning

CS 224R

Course reminders

- Homework 3 due tonight at 9 pm
- Midterm review session Monday at 4:30 pm
    - Reference sheet + practice problems shared on Ed

Recap of Model-Based RL

Key idea: learn a simulator of the dynamics $p_\theta(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t)$

Any method that learns this is a "model-based RL" algorithm

Use it to simulate additional data

Simulate data starting from all states seen in the data

Use it to do planning (i.e. lookahead)

Use with value functions for long-horizon planning

Mitigating model errors

- Use short synthetic roll-outs
- Ensembles of models can help average out errors

Recap of Model-Based RL

Big upsides and big downsides

+ Models are immensely useful, far more data efficient if model is easy to learn
+ Model can be trained on data without reward labels (fully self-supervised)
+ Model is somewhat task-agnostic (can sometimes be transferred across rewards)
- Models don't optimize for task performance
- Sometimes harder to learn than a policy
- Another thing to train, more hyperparameters, more compute intensive

Whether to use a model depends on how hard it is to learn!

The plan for today

1. Multi-task imitation and reinforcement learning
    a. Problem set up
    b. Task conditioning & weight sharing
    c. Data sharing with hindsight relabeling

Key learning goal: How to share weights and data across tasks for learning efficiency

What is multi-task RL?

Can we train generalist policies to do many “tasks”, not just one?

Examples:

Generalist LLM assistant: travel booking, grocery shopping, …

Legged robot: walk, run, dance, crouch, …

Mobile manipulator: hang up a towel, unload a dishwasher, wipe a spill, …

Music recommender system: personalization to many distinct users

Game player: play Flappy bird, Pokemon, Pinball, …

May have different reward functions, different dynamics, different action spaces, …!

Key idea will be to condition on the task.

Why multi-task RL?

Can we train generalist policies to do many “tasks”, not just one?

Fundamentals of CS224r so far:

- Imitation
- On-policy, off-policy and offline RL
- Model-free and model-based RL
- Reward functions

Biggest challenge so far?

Data efficiency

Can we **amortize the data complexity** across many tasks & scenarios?

Plenty of learnings can be shared across tasks!

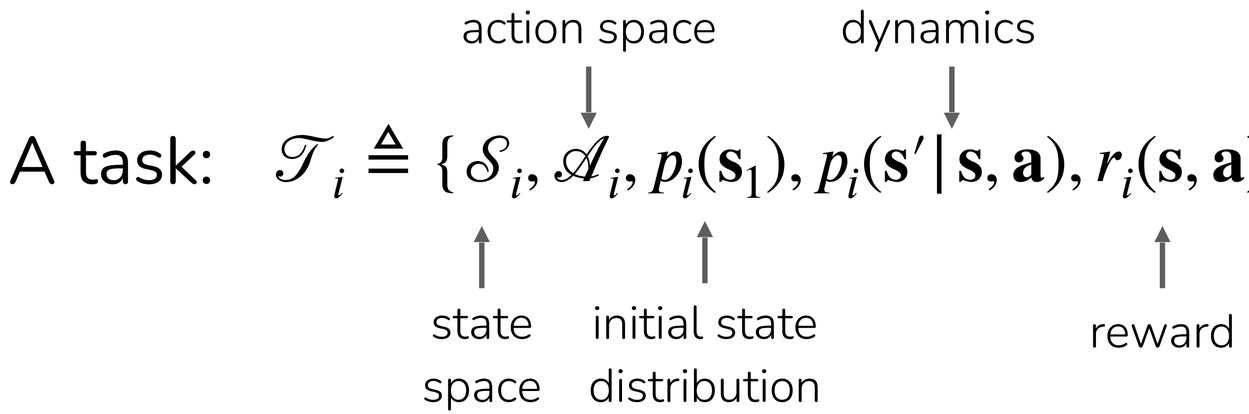
LLM assistant: grammar Legged robots: balance Personal recommenders: user similarities

There’s also a deeper motivation.

Generalist ML systems are often more reliable, performant than specialists.

How do we formalize what different “tasks” are?

Different tasks can just be different MDPs!



Note: This allows for much more variety than the semantic meaning of “task”!

A Few Example Task Distributions

A task: $\mathcal{T}_i \triangleq \{\mathcal{S}_i, \mathcal{A}_i, p_i(\mathbf{s}_1), p_i(\mathbf{s}'|\mathbf{s},\mathbf{a}), r_i(\mathbf{s},\mathbf{a})\}$

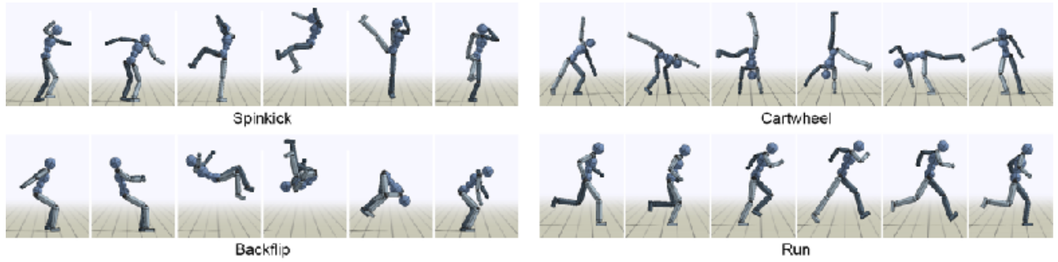
Personalized recommenders:

recommend videos
for multiple users

Figure 4: Recommending what to watch next on YouTube.

Character animation:

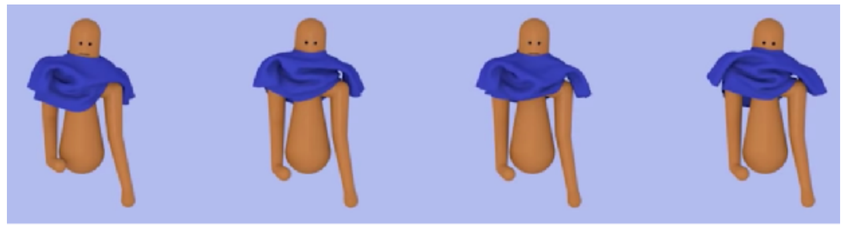
learn multiple maneuvers



Multi-robot RL:

Shirt folding for
multiple robots

learn to get dressed with
multiple garments & initial states



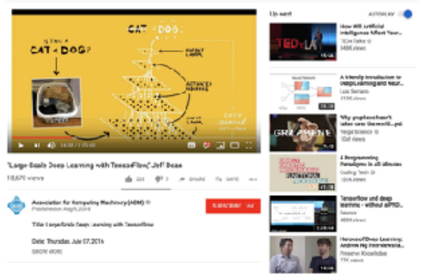
Question: What part of the MDP is varied vs. stayed fixed for these multi-task RL problems?

A Few Example Task Distributions

A task: $\mathcal{T}_i \triangleq \{\mathcal{S}_i, \mathcal{A}_i, p_i(\mathbf{s}_1), p_i(\mathbf{s}'|\mathbf{s},\mathbf{a}), r_i(\mathbf{s},\mathbf{a})\}$

Personalized recommenders:

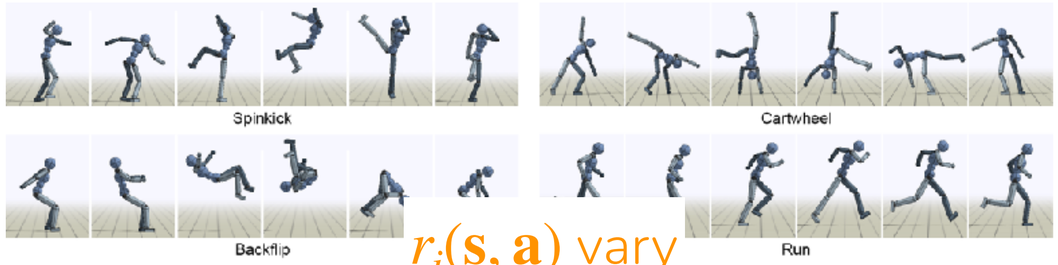
recommend videos
for multiple users



$p_i(\mathbf{s}'|\mathbf{s},\mathbf{a}), r_i(\mathbf{s},\mathbf{a})$ vary across tasks

Character animation:

learn multiple maneuvers



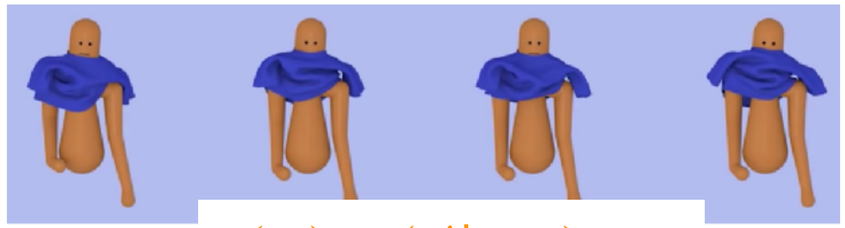
$r_i(\mathbf{s},\mathbf{a})$ vary

Multi-robot RL:

Shirt folding for
multiple robots

$\mathcal{S}_i, \mathcal{A}_i, p_i(\mathbf{s}_1), p_i(\mathbf{s}'|\mathbf{s},\mathbf{a})$ vary

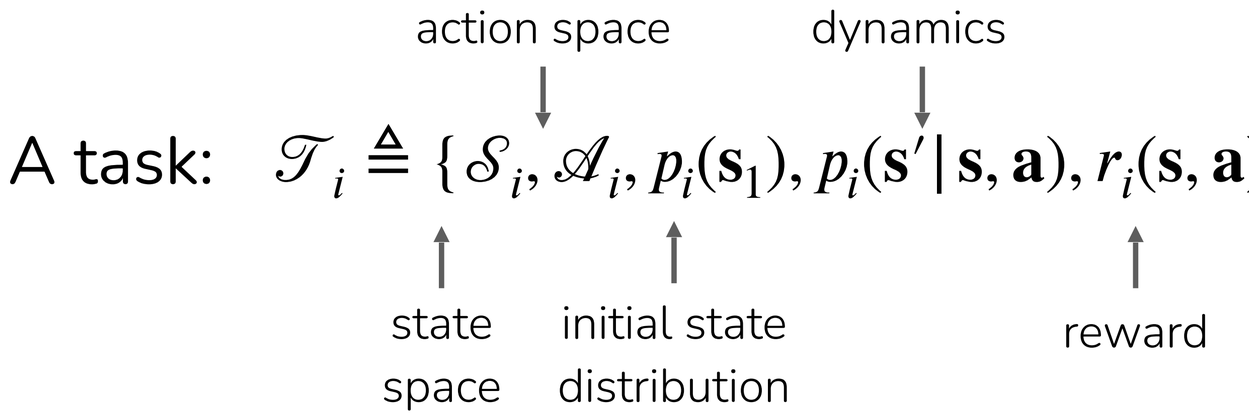
learn to get dressed with
multiple garments & initial states



$p_i(\mathbf{s}_1), p_i(\mathbf{s}'|\mathbf{s},\mathbf{a})$ vary

How do we formalize what different “tasks” are?

Different tasks can just be different MDPs!



How can do we identify one task from another?

Assign task index (e.g. task 0, task 1, …)

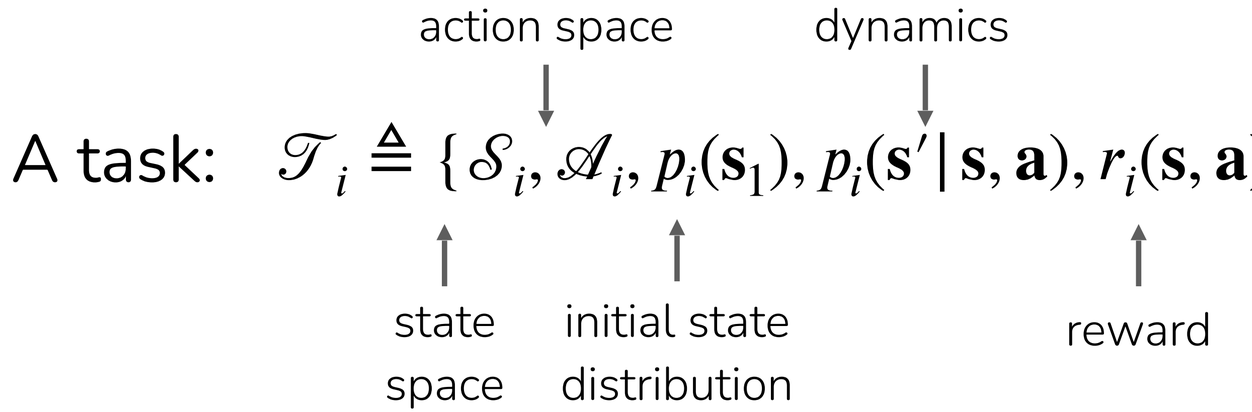
Language description

Video illustration of what to do

} These are task identifiers, $\mathbf{z}_i$

How do we formalize what different “tasks” are?

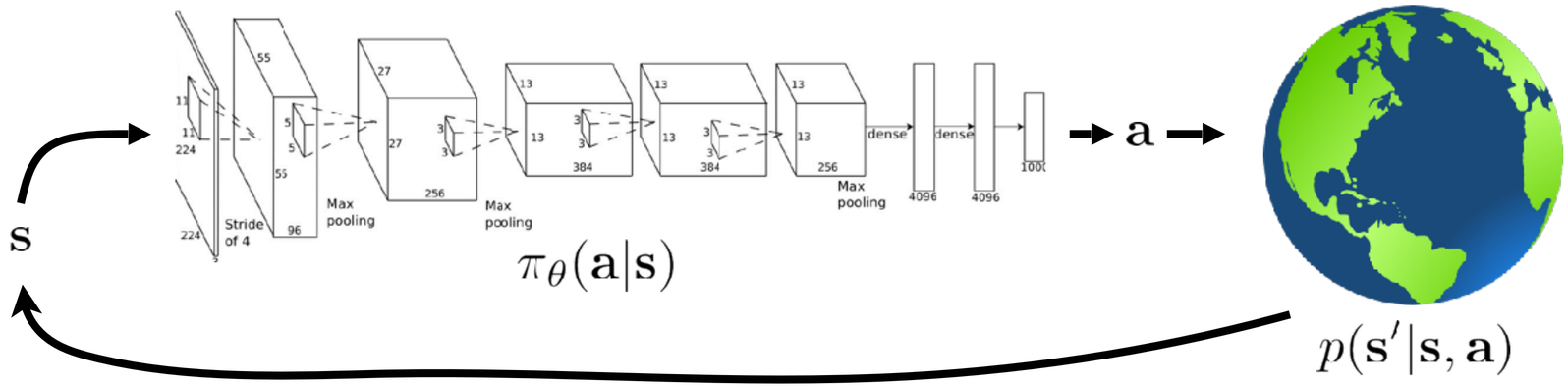
Different tasks can just be different MDPs!



An alternative view: A task identifier is part of the state: $\mathbf{s} = (\bar{\mathbf{s}}, \mathbf{z}_i)$

original state

The goal of multi-task reinforcement learning



Multi-task RL

The same as before, except:

a task identifier is part of the state: $\mathbf{s} = (\bar{\mathbf{s}}, \mathbf{z}_i)$

e.g. one-hot task ID

language description

desired goal state, $\mathbf{z}_i = \mathbf{s}_g$ ⟵ "**goal-conditioned** RL"

If it's still a standard **Markov decision process**,

then, why not apply standard **RL algorithms**? You can! You can do better in some cases.

What is the reward?

The same as before

Or, for goal-conditioned RL:

$r(\mathbf{s}) = r(\bar{\mathbf{s}}, \mathbf{s}_g) = -d(\bar{\mathbf{s}}, \mathbf{s}_g)$

Distance function d examples:

- Euclidean ℓ_2
- sparse 0/1

Let’s start with multi-task imitation learning

Single task imitation learning

$$\min_\theta -E_{(\mathbf{s},\mathbf{a})\sim\mathcal{D}} \log \pi_\theta(\mathbf{a}|\mathbf{s})$$

$$\min_\theta \mathcal{L}(\theta, \mathcal{D}) \longrightarrow \min_\theta \sum_{i=1}^{T} \mathcal{L}(\theta, \mathcal{D}_i)$$

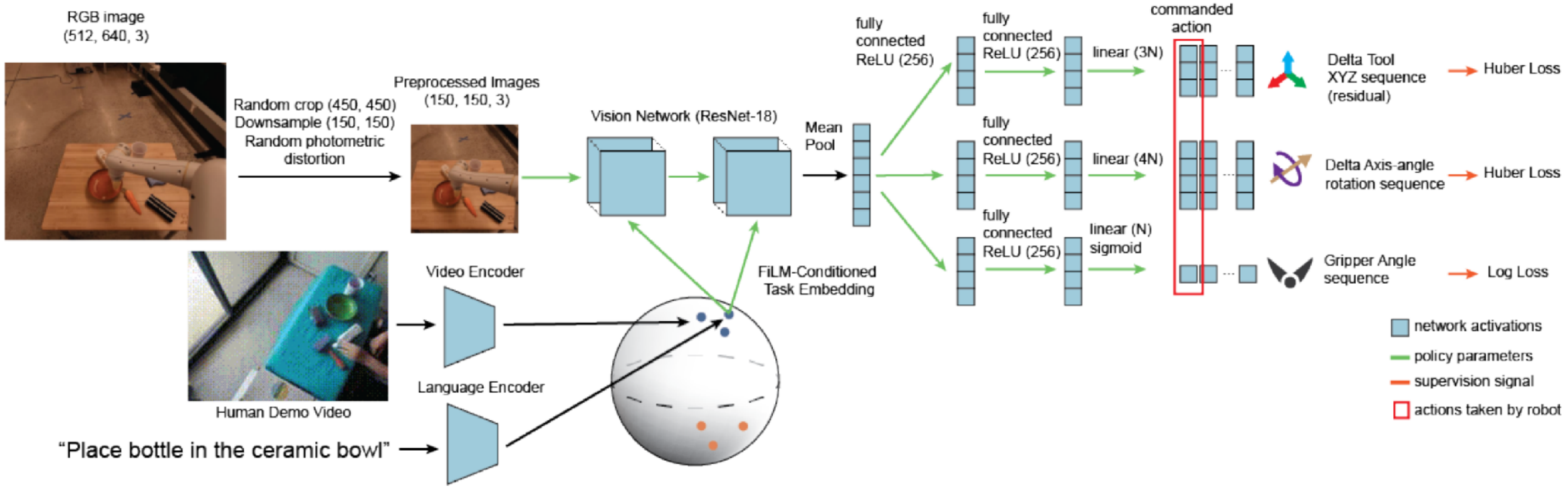
Multi-task imitation learning

One useful trick from multi-task supervised learning: **stratified sampling**

Construct each minibatch with data from each task.

-> lower variance gradients

How to condition on the task in practice?

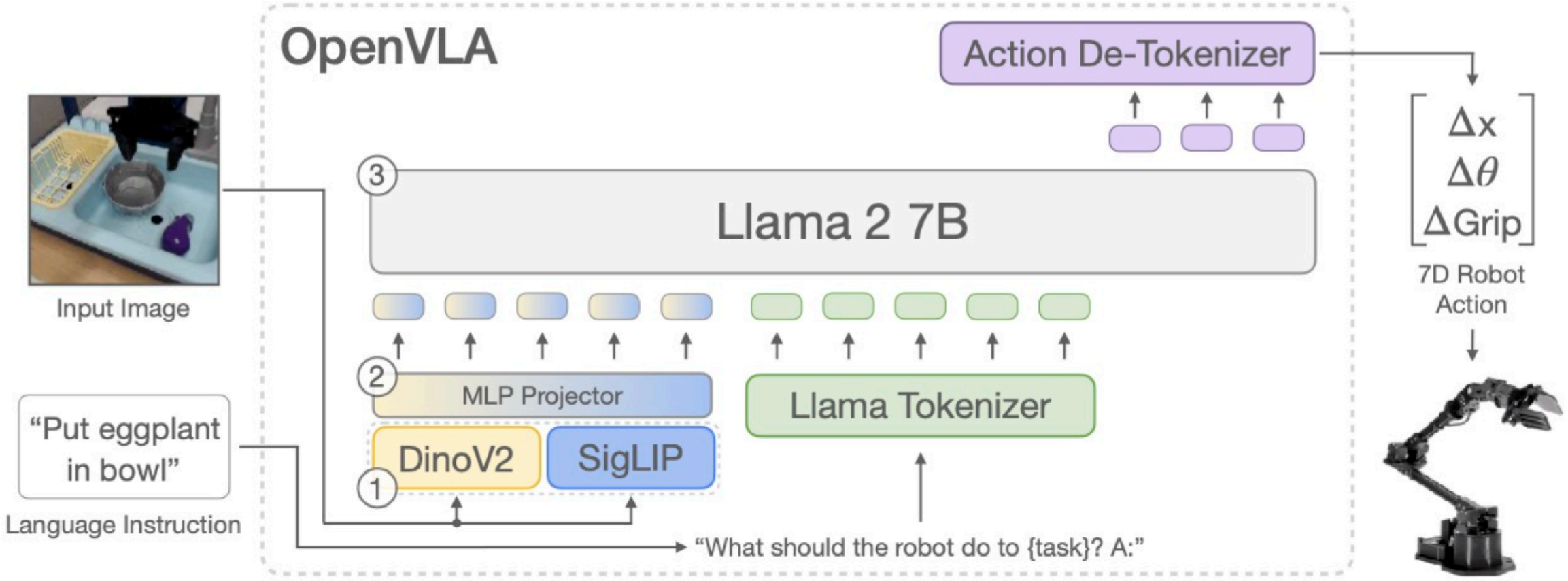


$$\min \sum_{\text{task } i} \sum_{\substack{(s,a)\sim\mathcal{D}_e^i \\ w_h \sim \mathcal{D}_h^i \bigcup \mathcal{D}_e^i}} \underbrace{-\log \pi(a|s, z^i)}_{\text{behavior cloning}} \text{, where } \underbrace{z_h^i \sim q(\cdot|w_h)}_{\text{video encoder}}, \underbrace{z_\ell^i \sim q(\cdot|w_\ell^i)}_{\text{language encoder}}$$

Jang et al. BC-Z: Zero-Shot Task Generalization with Robotic Imitation Learning CoRL 2021.

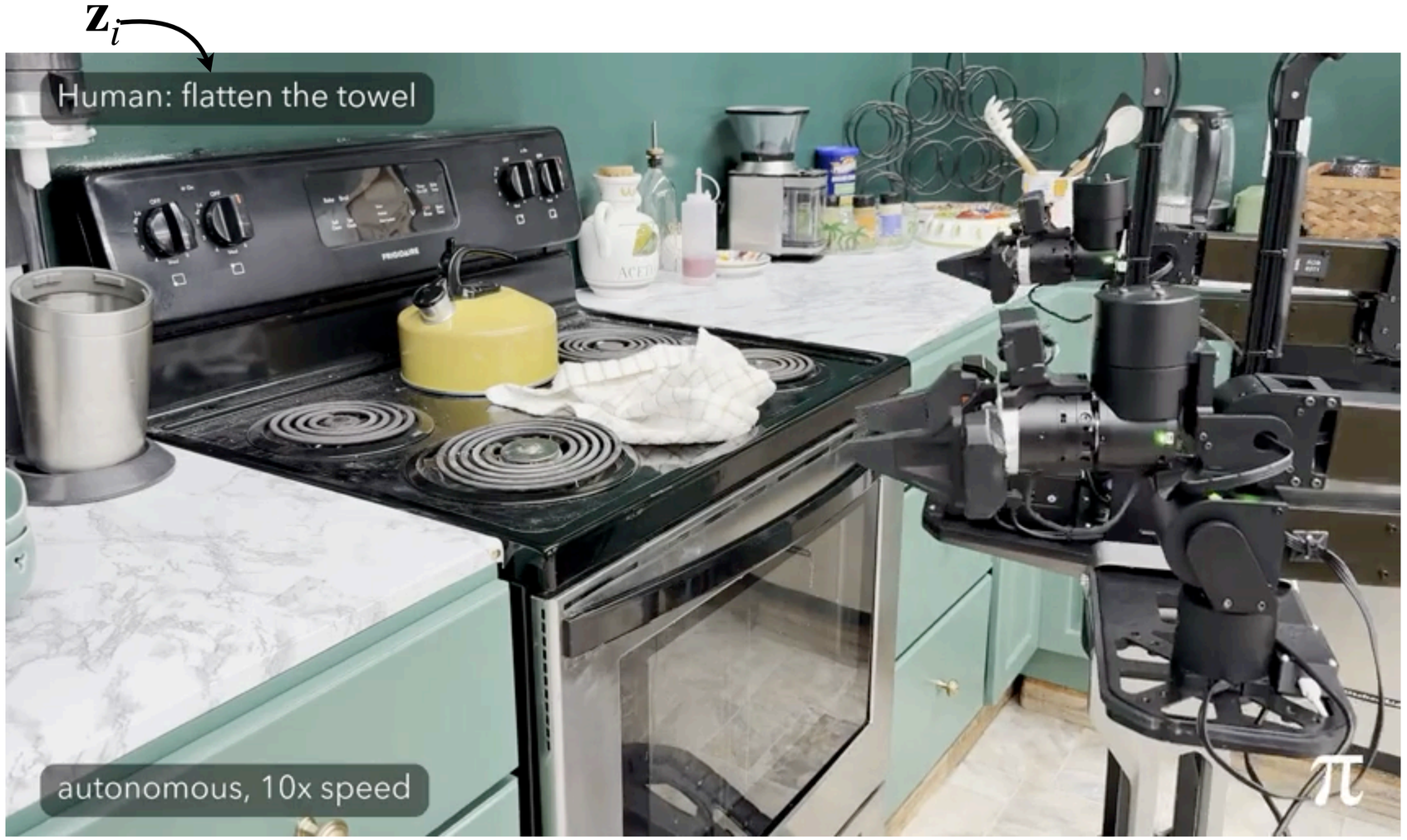
How to condition on the task in practice?

Modern policy architectures



Task identifier $\mathbf{z}_i$ passed as prompt into a (fine-tuned) LLM

Example of multi-task imitation learning



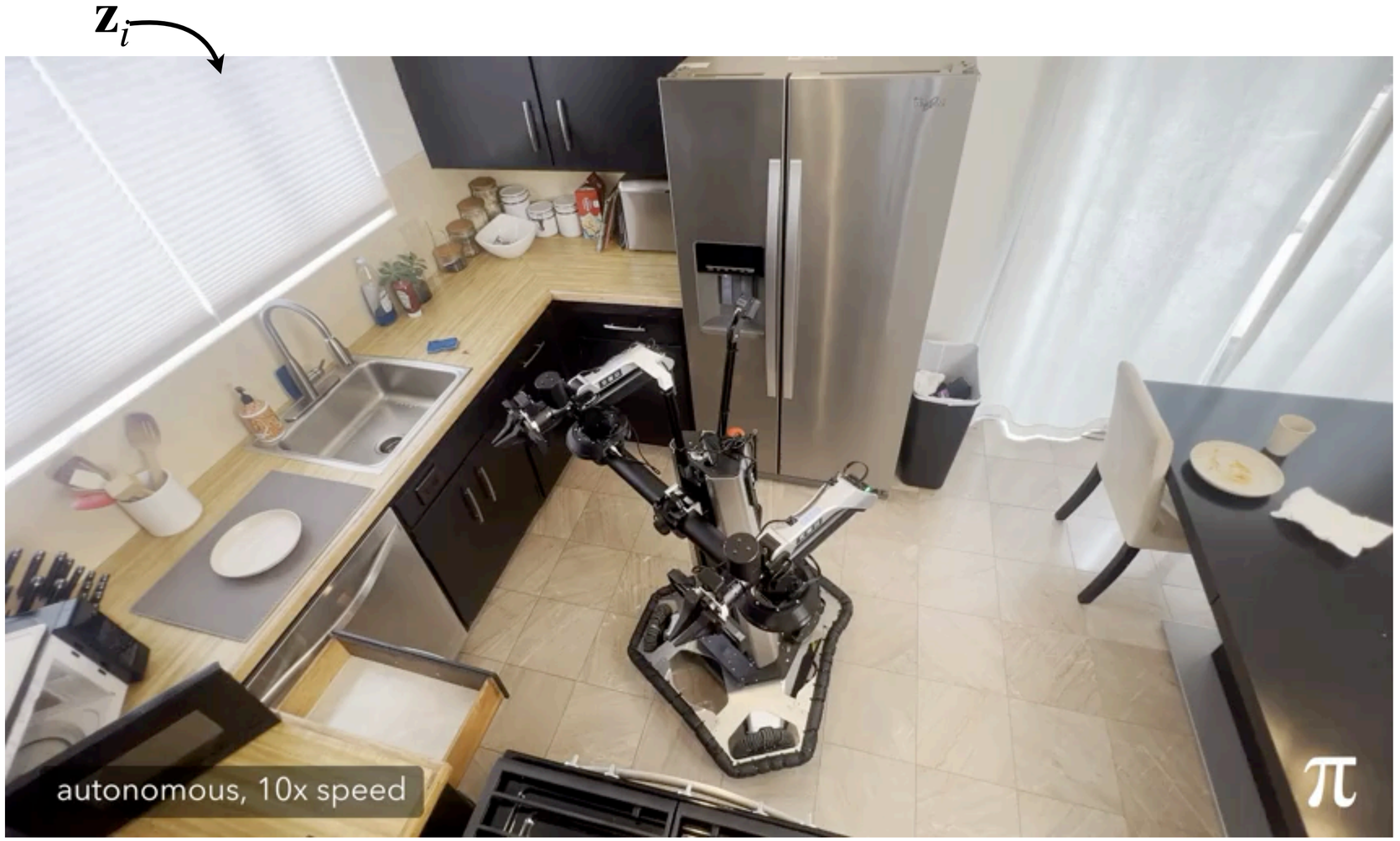
Two tasks:

$\mathbf{z}_1$ = ”flatten the towel”

$\mathbf{z}_2$ = ”hang the towel”

Physical Intelligence et al. $\pi_{0.5}$: A Vision-Language-Action Model with Open-World Generalization arXiv 2025.

Example of multi-task imitation learning



Many tasks:

$\mathbf{z}_1$ = ”put the plate in the drawer”
$\mathbf{z}_2$ = ”close the microwave”
$\mathbf{z}_3$ = ”drive base backward”
$\mathbf{z}_4$ = ”rotate base left”
$\mathbf{z}_5$ = ”throw the napkin away”
$\mathbf{z}_6$ = ”put the cup in the sink”
$\mathbf{z}_7$ = ”lift torso up”
$\mathbf{z}_8$ = ”clean the spill”

…

What about high-level, long-horizon tasks like “clean the kitchen”? -> Lecture on hierarchy

Multi-Task RL Algorithms

The basics

Policy: $\pi_\theta(\mathbf{a} \mid \bar{\mathbf{s}}) \;\; \text{—>} \;\; \pi_\theta(\mathbf{a} \mid \bar{\mathbf{s}}, \mathbf{z}_i)$

Q-function: $Q_\phi(\bar{\mathbf{s}}, \mathbf{a}) \;\; \text{—>} \;\; Q_\phi(\bar{\mathbf{s}}, \mathbf{a}, \mathbf{z}_i)$

Consider per-task replay buffer for stratified sampling.

Can we share data *across tasks*?

If we collect policy data when conditioning on $\mathbf{z}_1$, can we reuse that data to learn something for task 2?

The plan for today

1. Multi-task imitation and reinforcement learning
   a. Problem set up
   b. Task conditioning & weight sharing
   c. Data sharing with hindsight relabeling

An example

Task 1: passing

Task 2: shooting goals

What if you accidentally perform a good pass when trying to shoot a goal?

Store experience as normal. *and* Relabel experience with task 2 ID & reward and store.

“hindsight relabeling” "hindsight experience replay” (HER)

Multi-task RL with relabeling

k++

1. Collect data $\mathcal{D}_k = \{(\mathbf{s}_{1:T}, \mathbf{a}_{1:T}, \mathbf{z}_i, r_{1:T})\}$ using some policy
2. Store data in replay buffer $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_k$
3. Perform hindsight relabeling:
   a. Relabel experience in $\mathcal{D}_k$ for task $\mathcal{T}_j$:
      $\mathcal{D}'_k = \{(\mathbf{s}_{1:T}, \mathbf{a}_{1:T}, \mathbf{z}_j, r'_{1:T}\}$ where $r'_t = r_j(\mathbf{s}_t)$
   b. Store relabeled data in replay buffer $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}'_k$
4. Update policy using replay buffer $\mathcal{D}$

<— Which task $\mathcal{T}_j$ to choose?

- randomly
- task(s) in which the trajectory gets high reward

Question: In what scenarios can we apply relabeling?

- reward function form is known, evaluatable
- dynamics consistent across goals/tasks
- using an off-policy algorithm

Kaelbling. Learning to Achieve Goals. IJCAI '93
Andrychowicz et al. Hindsight Experience Replay. NeurIPS '17

Goal-conditioned RL with hindsight relabeling

k++

1. Collect data $\mathcal{D}_k = \{(\mathbf{s}_{1:T}, \mathbf{a}_{1:T}, \mathbf{s}_g, r_{1:T})\}$ using some policy
2. Store data in replay buffer $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_k$
3. Perform hindsight relabeling:
   a. Relabel experience in $\mathcal{D}_k$ using last state as goal: $\mathcal{D}'_k = \{(\mathbf{s}_{1:T}, \mathbf{a}_{1:T}, \mathbf{s}_T, r'_{1:T}\}$ where $r'_t = -d(\mathbf{s}_t, \mathbf{s}_T)$
   b. Store relabeled data in replay buffer $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}'_k$
4. Update policy using replay buffer $\mathcal{D}$

<— Other relabeling strategies?

use any future state from the trajectory

Result: exploration challenges alleviated

Kaelbling. Learning to Achieve Goals. IJCAI '93
Andrychowicz et al. Hindsight Experience Replay. NeurIPS '17

Hindsight relabeling for goal-conditioned RL

Example: goal-conditioned RL, simulated robot manipulation

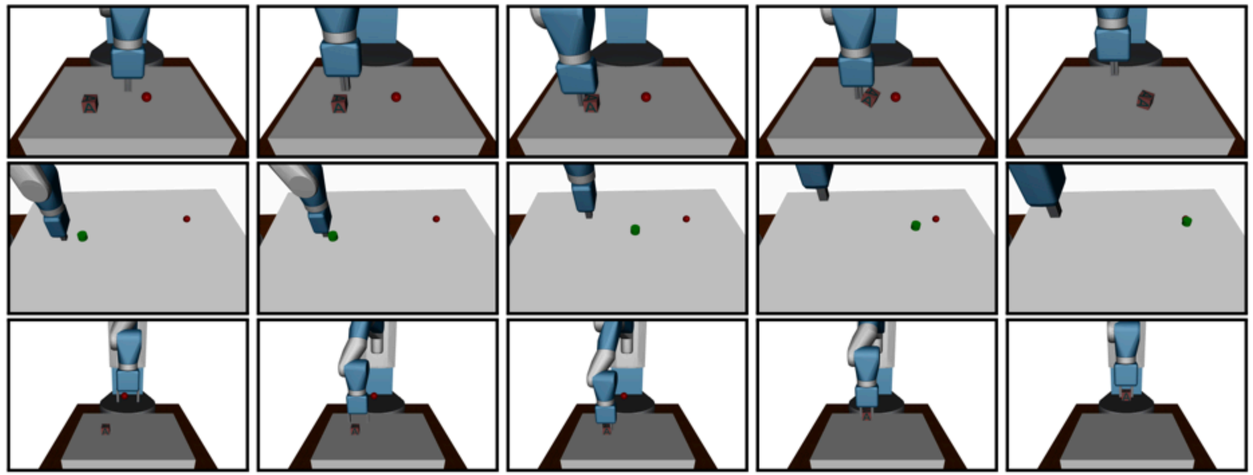
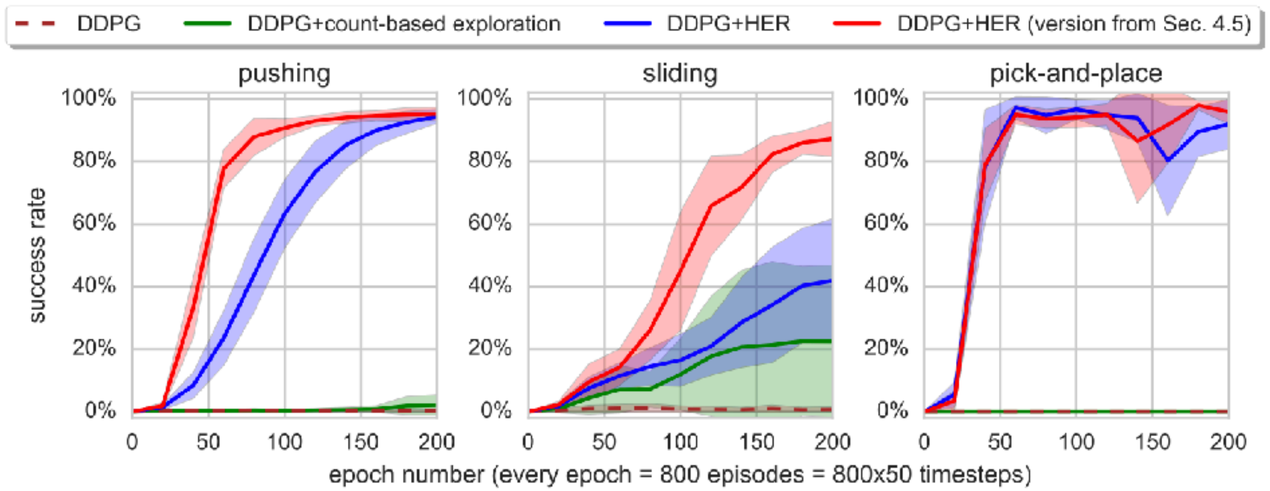


Figure 2: Different tasks: *pushing* (top row), *sliding* (middle row) and *pick-and-place* (bottom row). The red ball denotes the goal position.



Kaelbling. Learning to Achieve Goals. IJCAI '93
Andrychowicz et al. Hindsight Experience Replay. NeurIPS '17

The plan for today

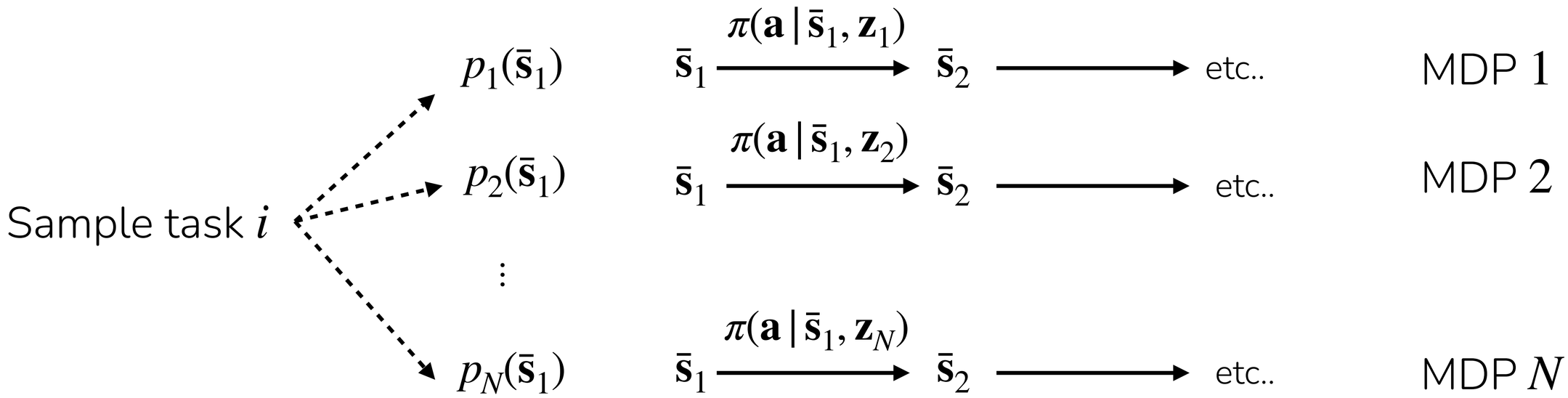
1. Multi-task imitation and reinforcement learning
   a. Problem set up
   b. Task conditioning & weight sharing
   c. Data sharing with hindsight relabeling

Key learning goal: How to share weights and data across tasks for learning efficiency

Summary: Multi-task learning

Multi-task RL as single-task RL in a joint MDP

$$\mathbf{s} = (\bar{\mathbf{s}}, \mathbf{z}_i)$$



Summary: Goal-conditioned learning

Special case of multi-task RL, where $\mathbf{z}_i = \mathbf{s}_g$

Each task aims to reach some goal state $\mathbf{s}_g$

$r(\mathbf{s}, \mathbf{a}, \mathbf{s}_g) = \delta(\mathbf{s} = \mathbf{s}_g)$ for discrete states

$r(\mathbf{s}, \mathbf{a}, \mathbf{s}_g) = \delta(\|\mathbf{s} - \mathbf{s}_g\| \leq \epsilon)$ for continuous states

+ No need to define reward (self-supervised!)

+ Many tasks can be formed as goal reaching tasks

- Can be fairly hard to train

Summary: Multi-task & goal-conditioned RL

Weight sharing

Train network to do all tasks, conditioned on $\mathbf{z}_i$

Data sharing

Add data collected for one task to buffer for another by **relabeling** the reward and task identifier

Requires:

- same dynamics across tasks
- evaluatable reward functions
- off-policy learning algorithm

Directly applicable to goal-conditioned RL setting

Course reminders

- Homework 3 due tonight at 9 pm
- Midterm review session Monday at 4:30 pm

Next Time

Can we quickly adapt to **new** tasks?

(e.g. in-context learning for RL)