Deep Reinforcement Learning
Spring 2026

Abhishek Gupta

TA: Mateo Guaman Castro

Lecture outline

Recap: MDP formalism + why should we care?

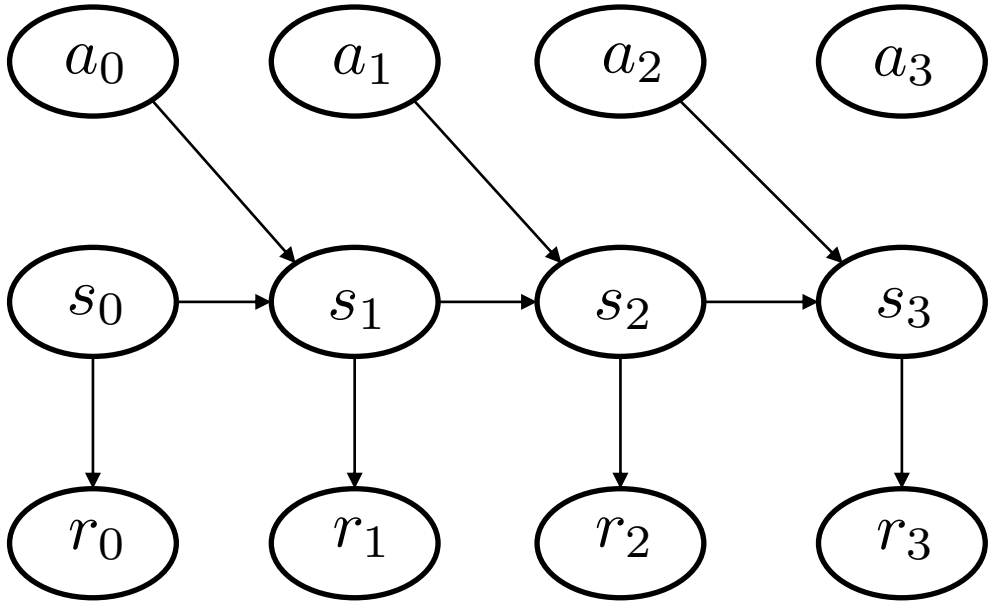
Imitation learning: preliminaries and behavior cloning

Multimodality and Underfitting in Imitation

Compounding Error in Imitation

Framework for RL - Markov Decision Process

Augment Markov chain with rewards and actions



States: $\mathcal{S}$

Initial state dist: $\rho_0(s)$

Actions: $\mathcal{A}$

Discount: γ

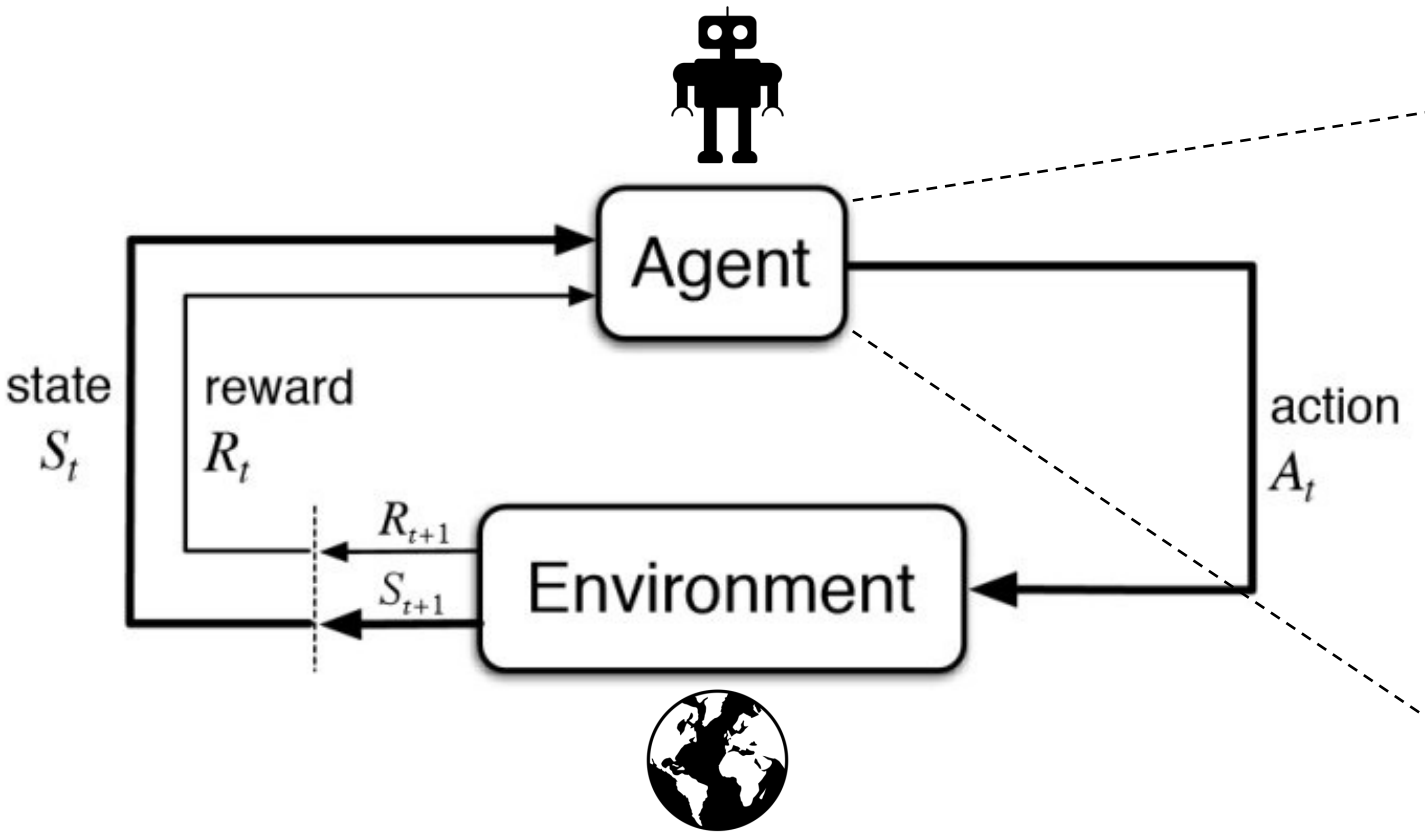
Rewards: $\mathcal{R}$

Transition Dynamics - $p(s_{t+1}|s_t, a_t)$

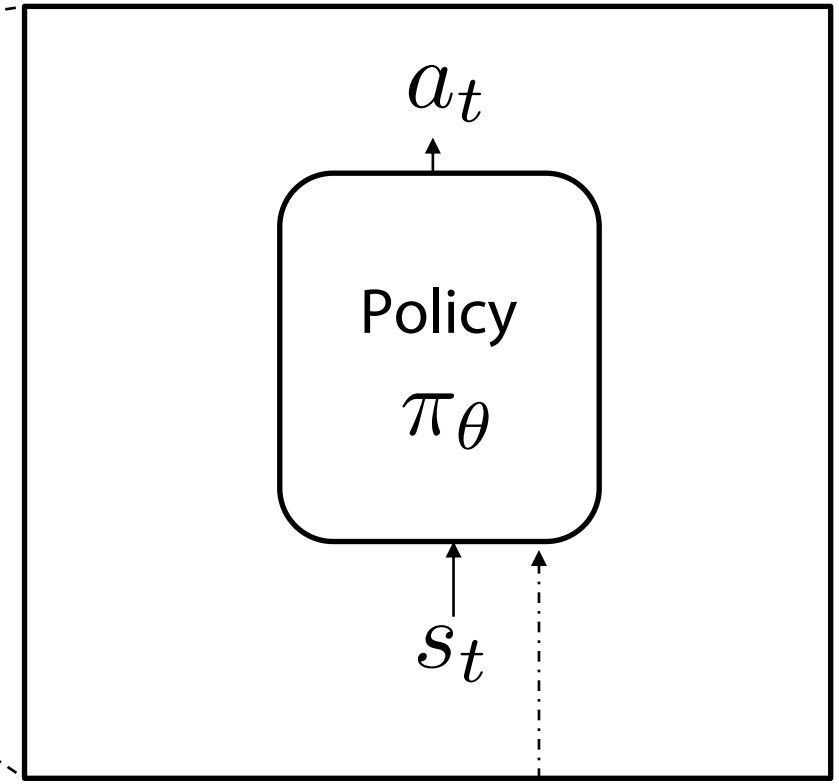
Markov property $p(s_0, s_1, s_2, a_0, a_1, a_2) = p(s_0)p(a_0|s_0)p(s_1|s_0, a_0)p(a_1|s_1)p(s_2|s_1, a_1)p(a_2|s_2)$

Trajectory $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \ldots, s_T, a_T, r_T)$

Reinforcement Learning Formalism



Rules for choosing actions



$$\max_\theta \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} r(s_t, a_t) \right]$$

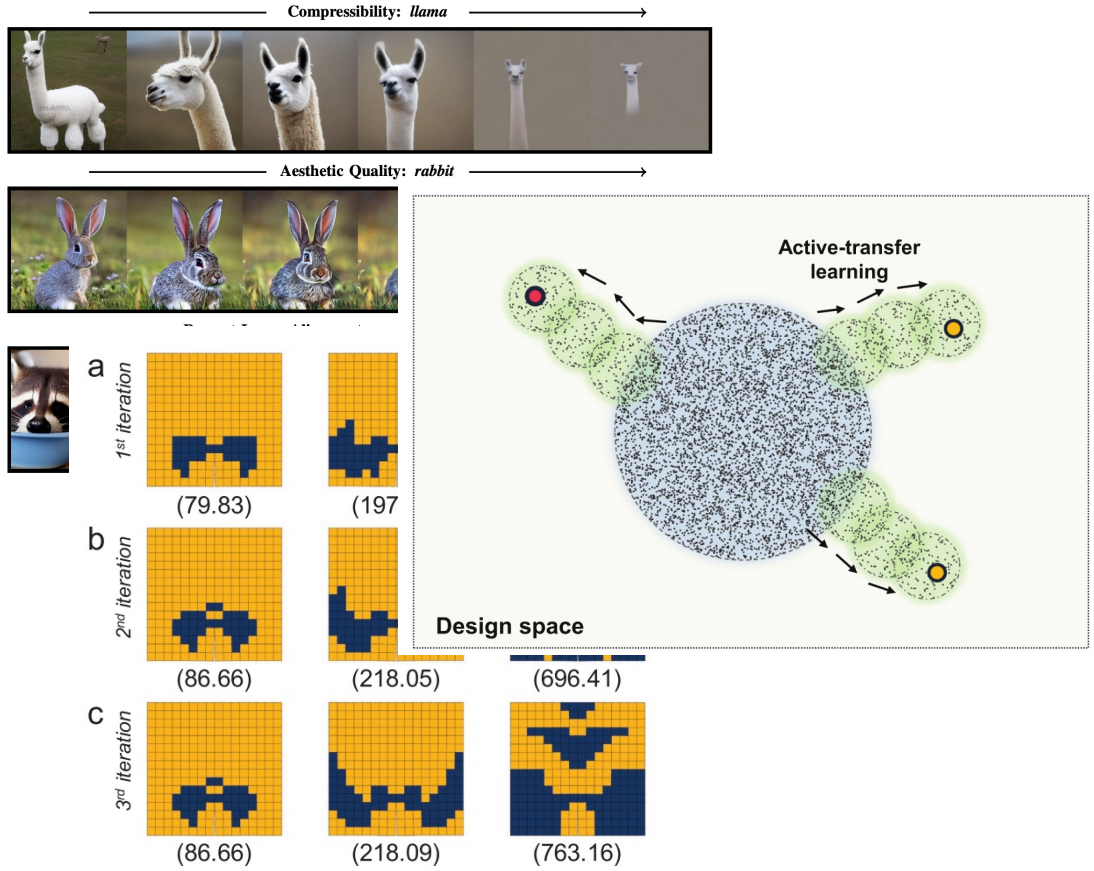
Trajectory sampled using policy

Needs to be learned

Reinforcement Learning enables going beyond the data

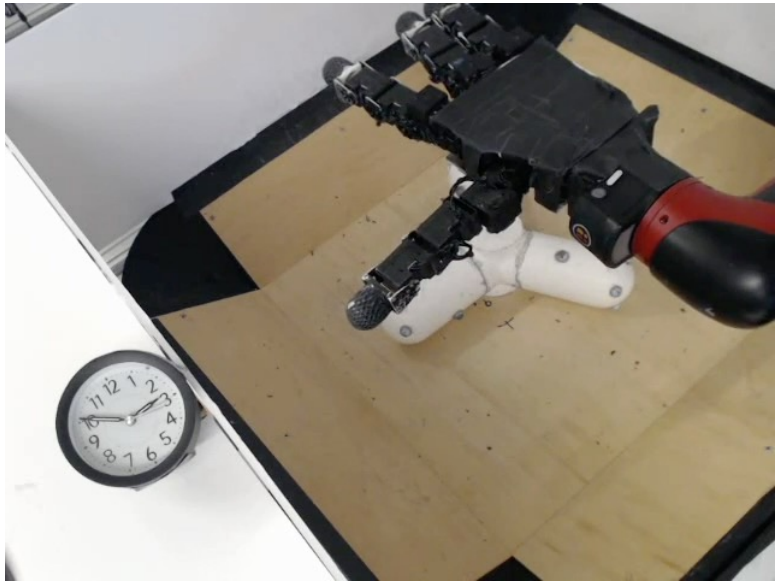
Generative AI is inherently about **replicating** the data distribution

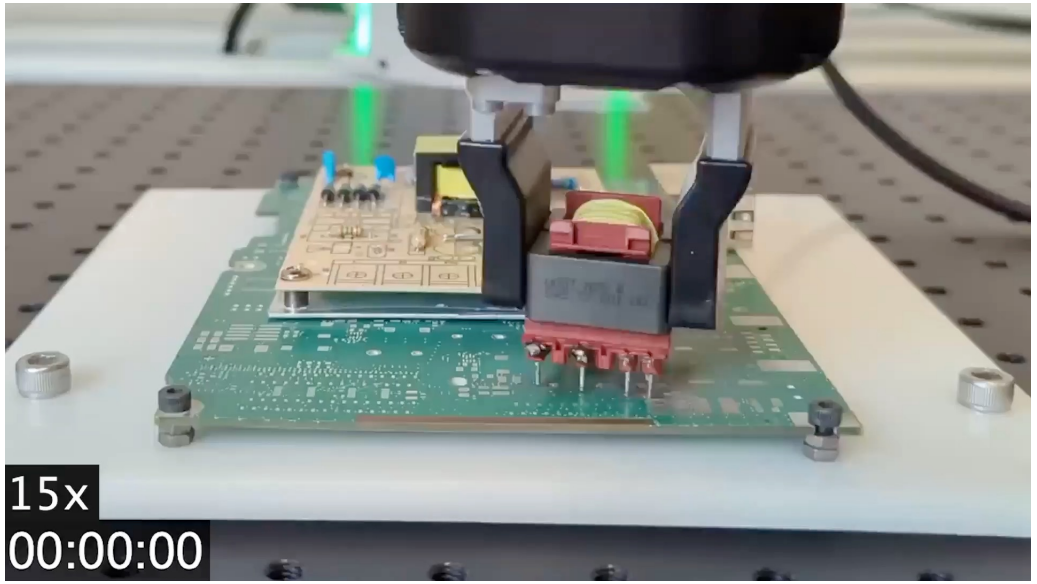
Reinforcement learning can go beyond the data



Applications of RL: Robotics

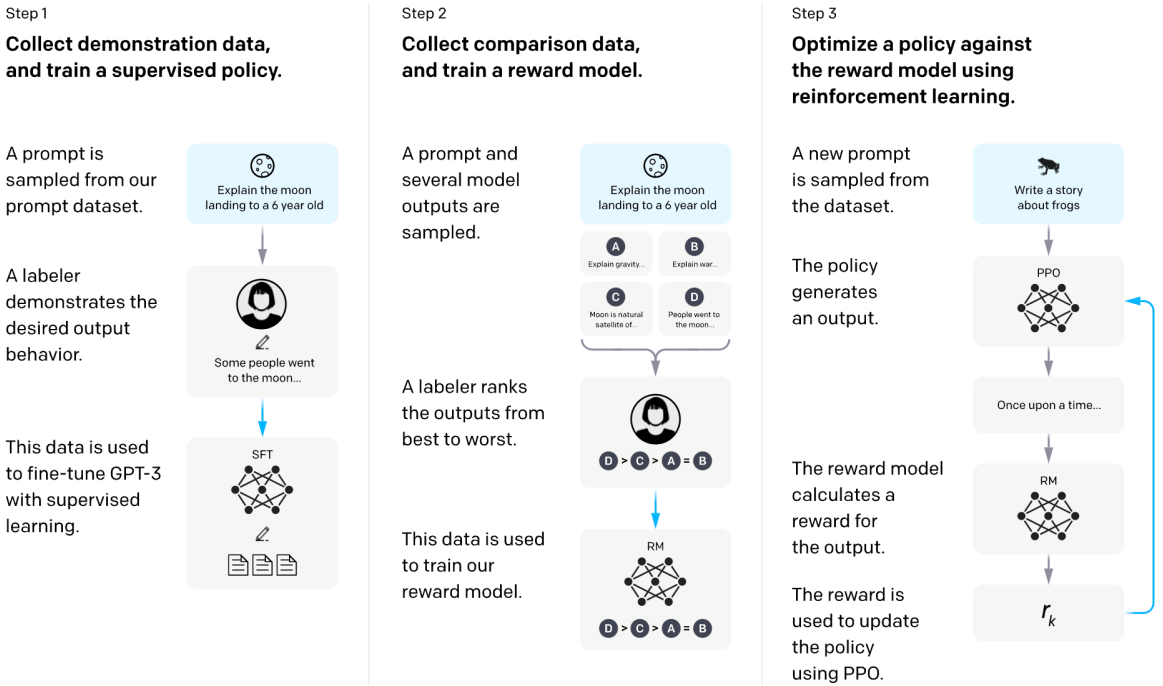
RL can enable robotic learning of hard to specify/script behaviors in the presence of contact





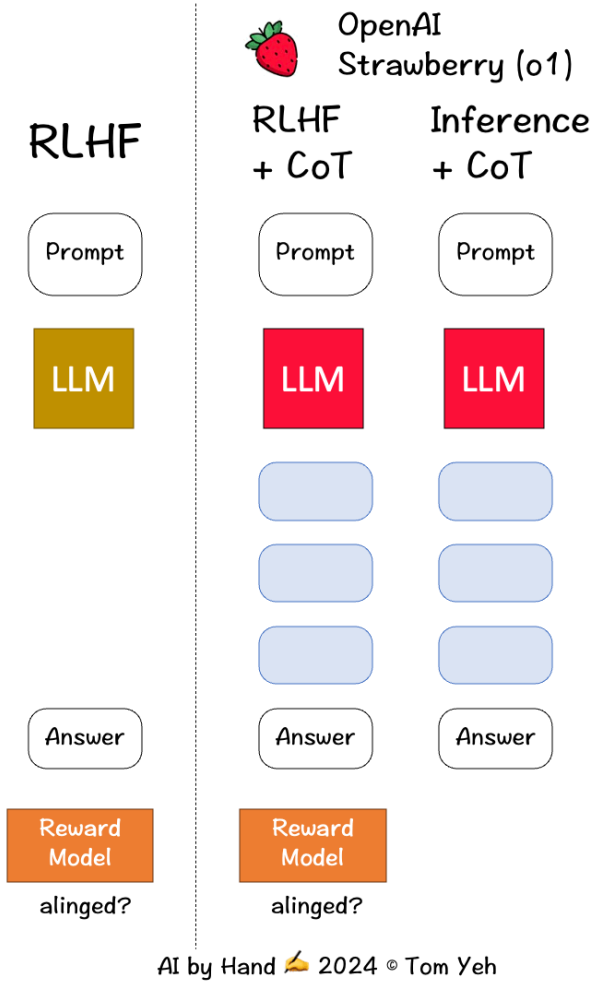
Applications of RL: Large Language Models

Systematically finds and reduces model hallucinations using RLHF

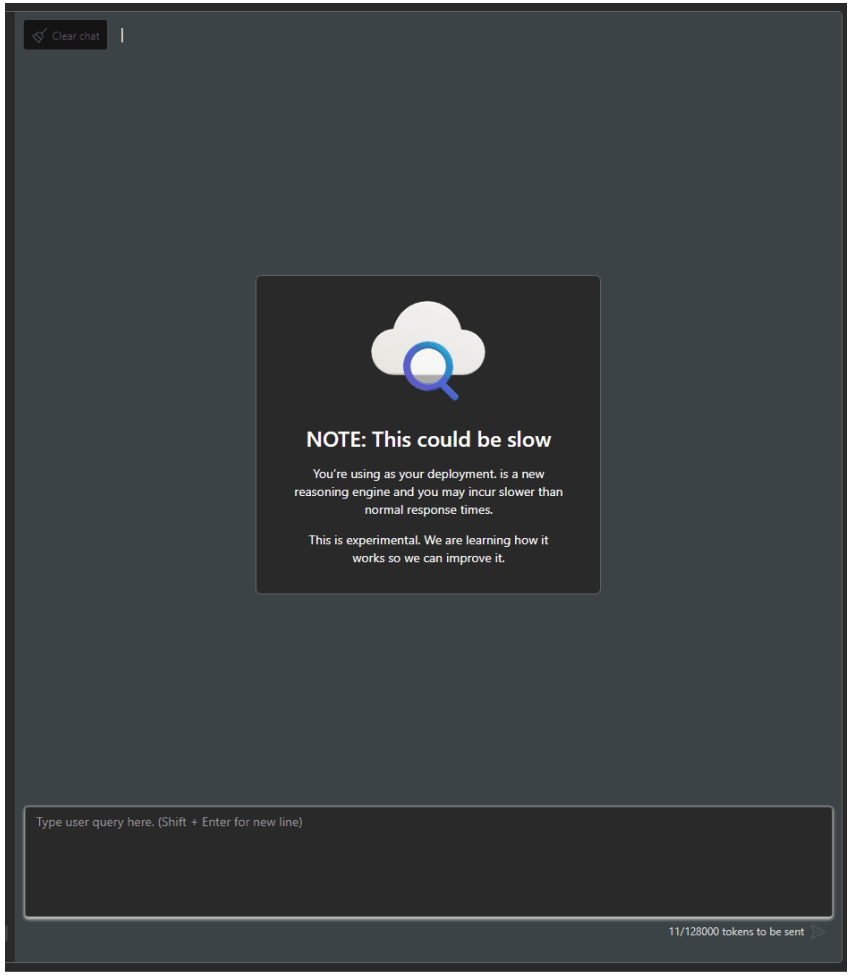


Applications of RL: Large Language Models

Recent reasoning-style LLM algorithms are typically RL based



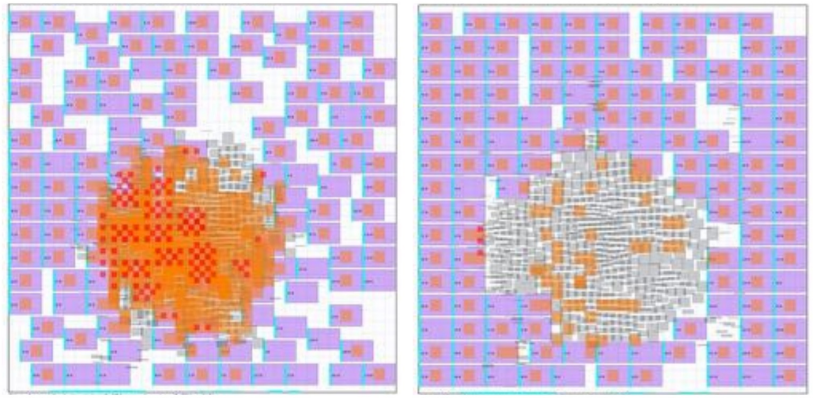
Looks a lot like model-based RL



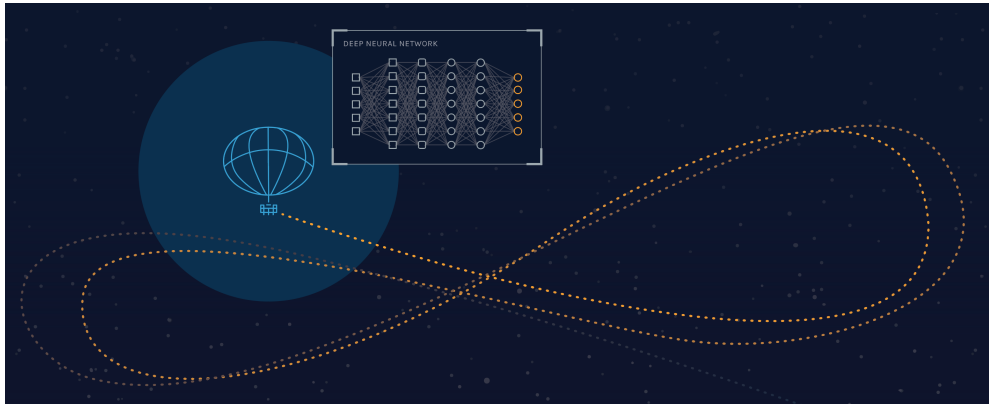
Applications of RL: Science and Engineering

RL has started to become a useful tool for engineering design

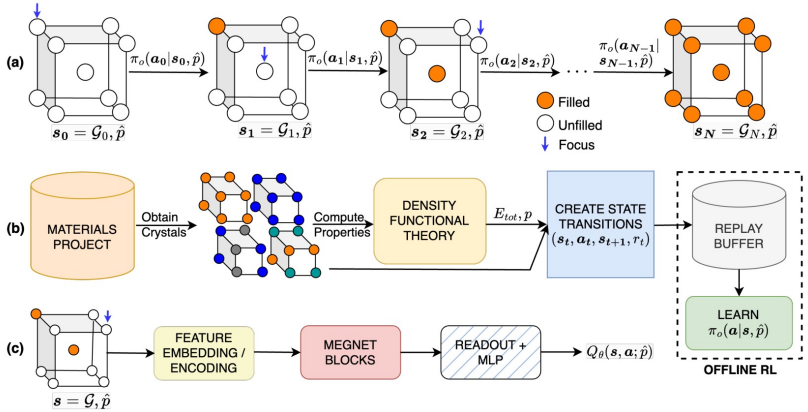
Chip Design



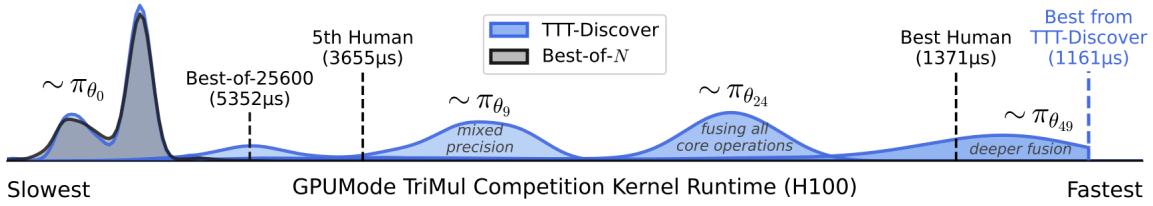
Weather balloon navigation



Crystal design

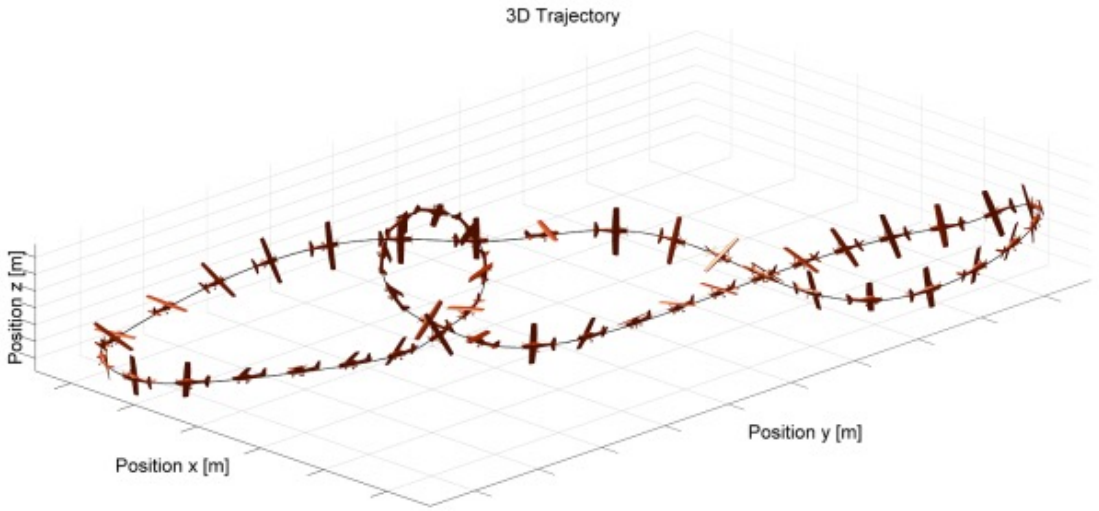


Kernel Optimization



Where is Reinforcement Learning not useful?

Not the right call for very safety-critical, repetitive applications

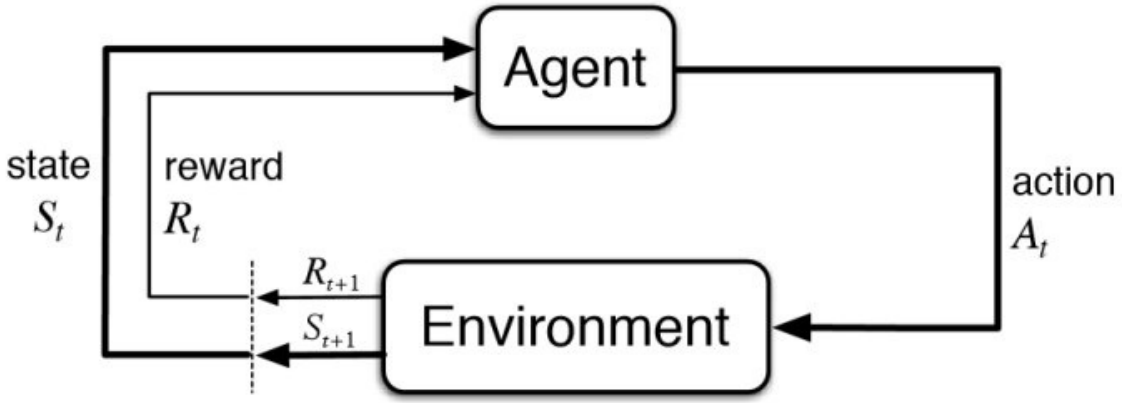


Where is Reinforcement Learning "potentially" useful?

Domains which have high diversity, yet relatively cheap autonomous data collection

But these domains are not as simple as just running RL algorithms!

So is sequential decision making = RL?

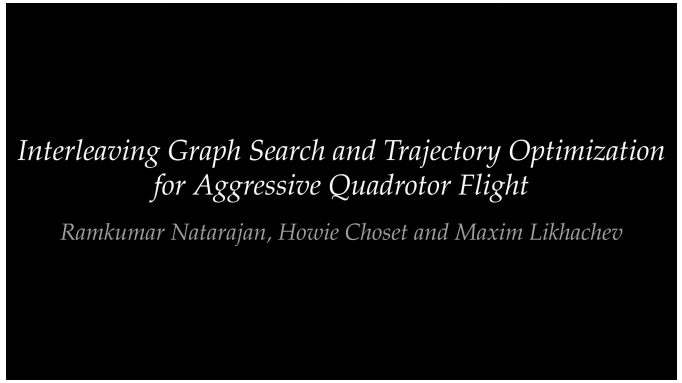


We conflated sequential decision making and RL!

RL is sequential decision making under a particular set of assumptions:

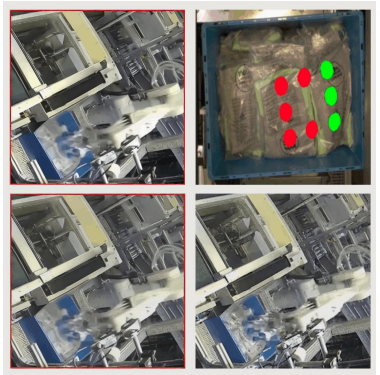
1. Sampling access to the environment
2. Access to reward
3. Goal-directed behavior

Trajectory optimization/planning

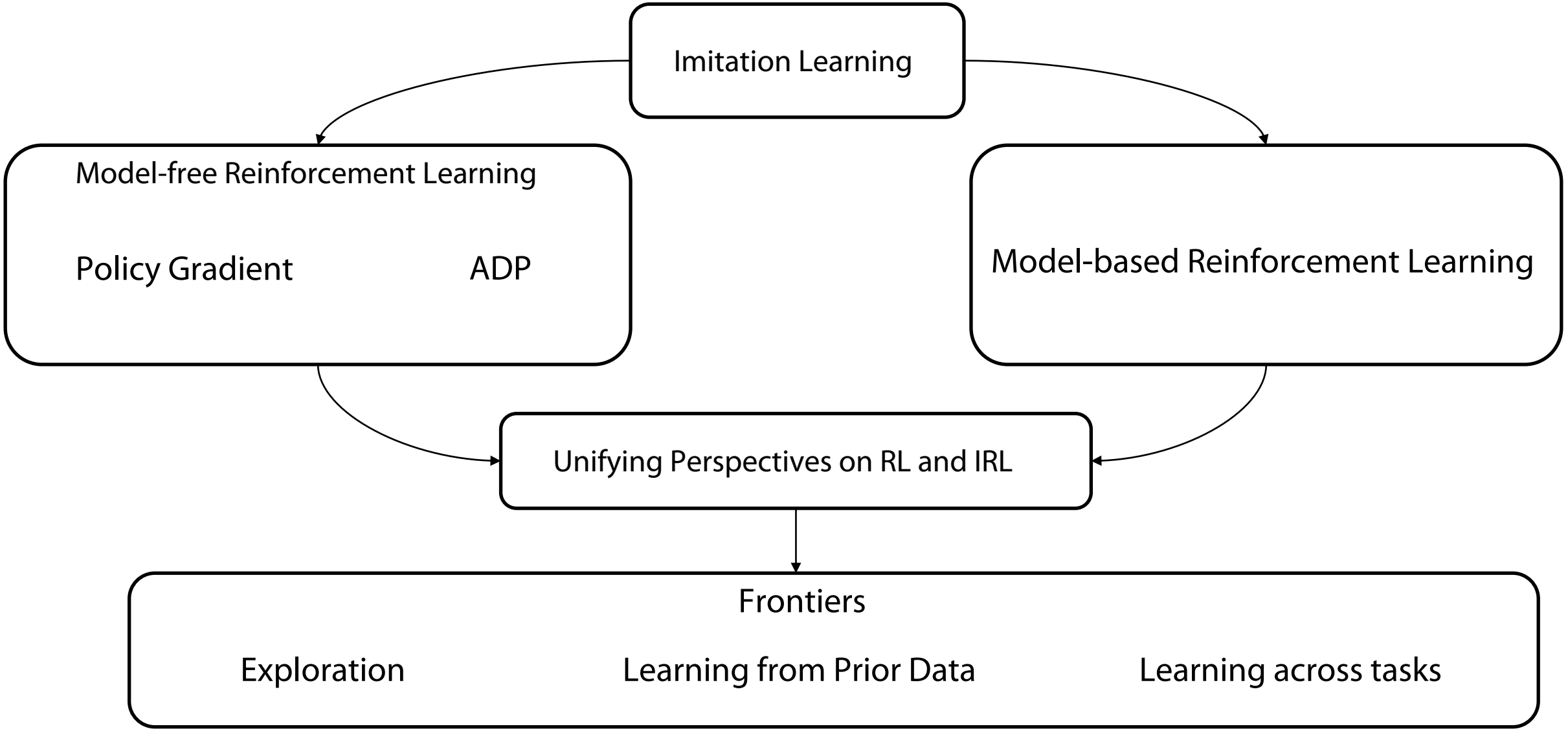


Imitation Learning

Unsupervised Decision Making



Class Structure



Lecture outline

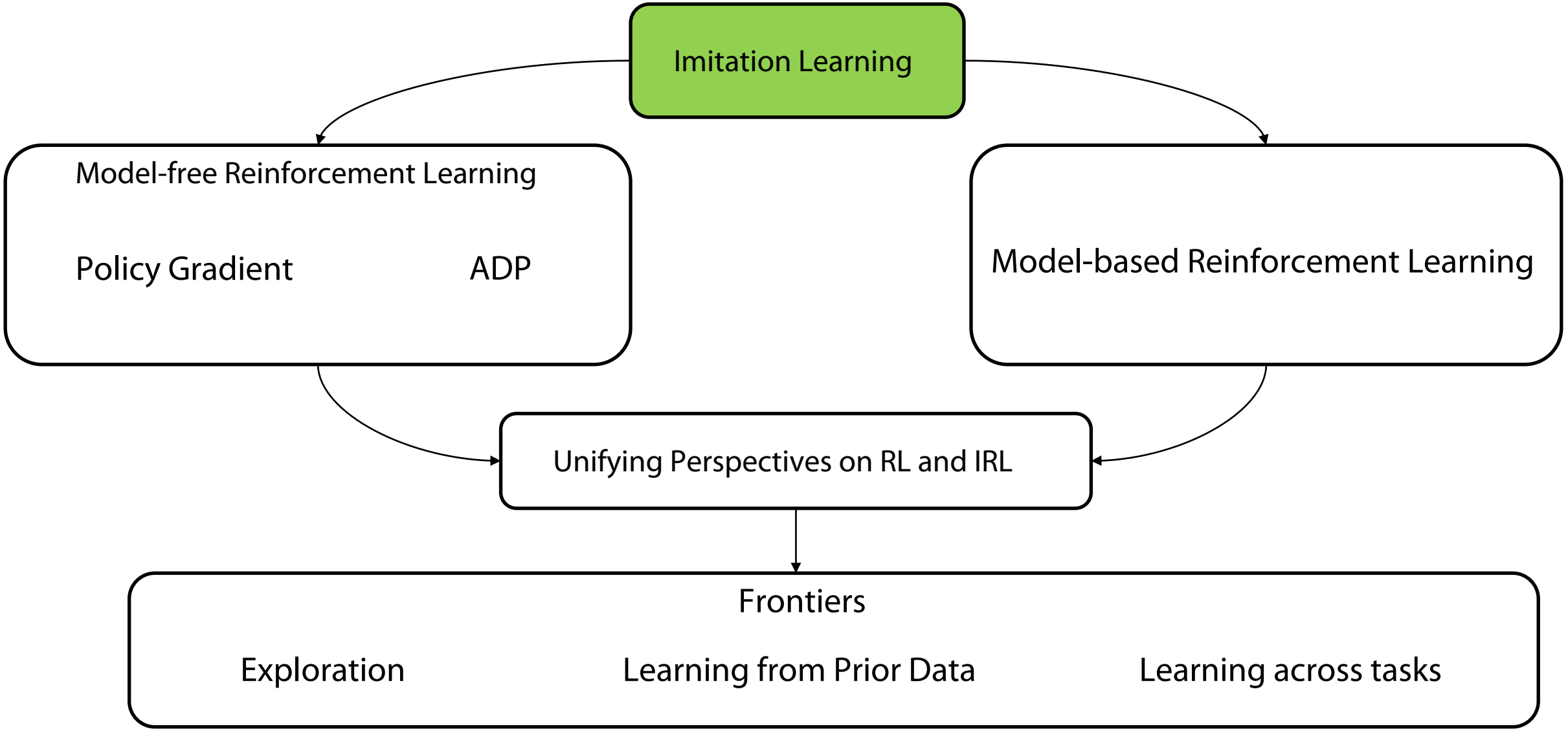
Recap: MDP formalism + why should we care?

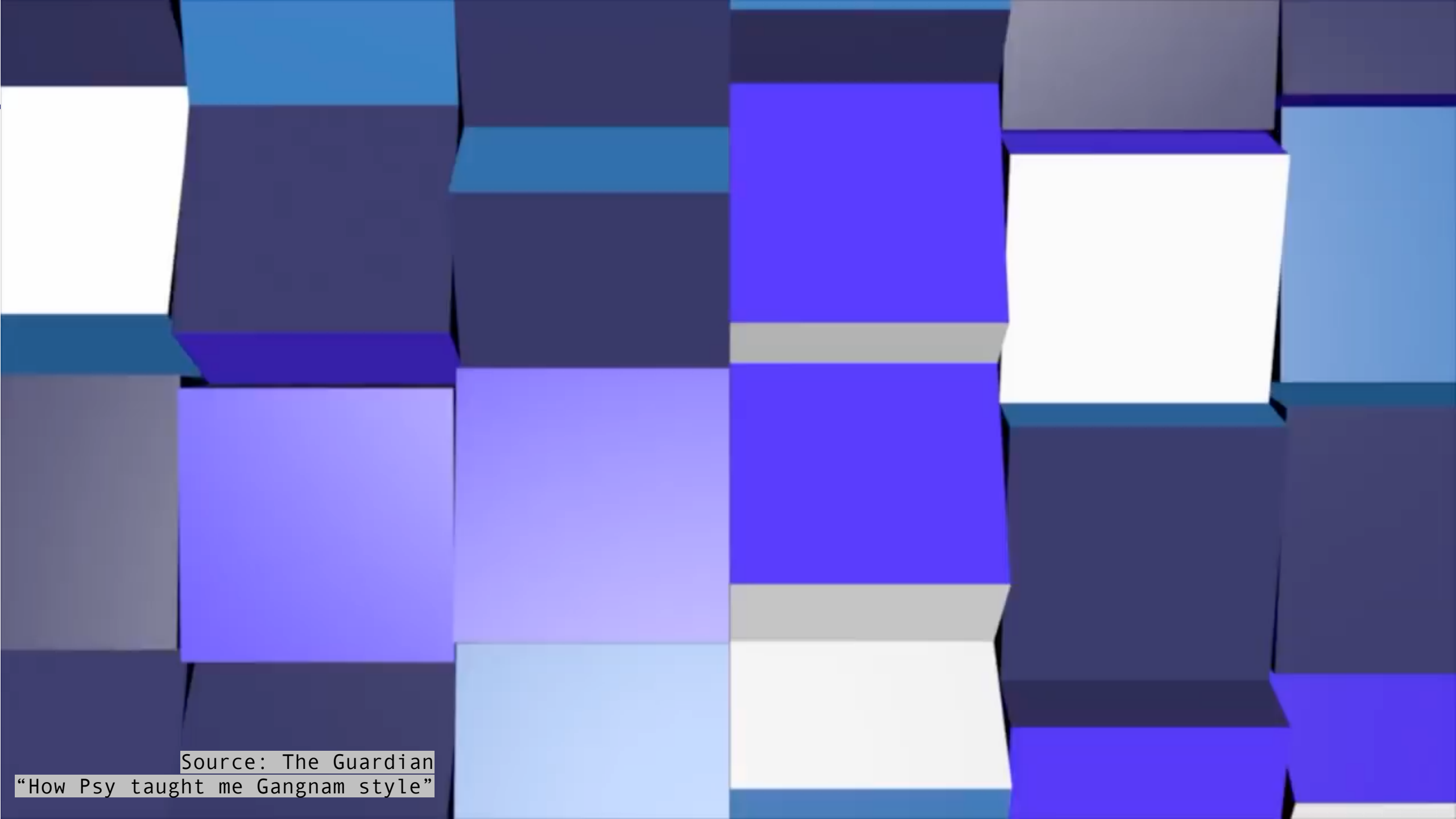
Imitation learning: preliminaries and behavior cloning

Multimodality and Underfitting in Imitation

Compounding Error in Imitation

Class Structure

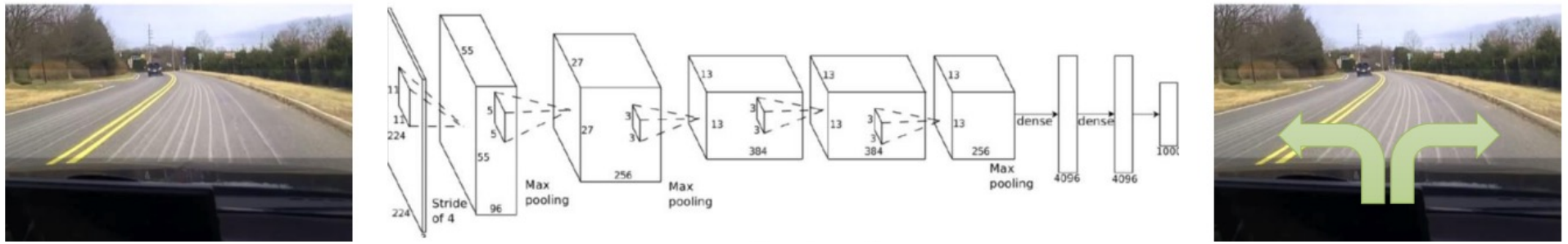



Source: The Guardian
“How Psy taught me Gangnam style”

Imitation Learning: Intuition

Given: Demonstrations of optimal behavior $\mathcal{D} = \{s_0^i, a_0^i, s_1^i, a_1^i, \ldots, s_T^i, a_T^i\}_{i=1}^N$

Goal: Train a policy to mimic the demonstrator $\pi_\theta(a|s)$

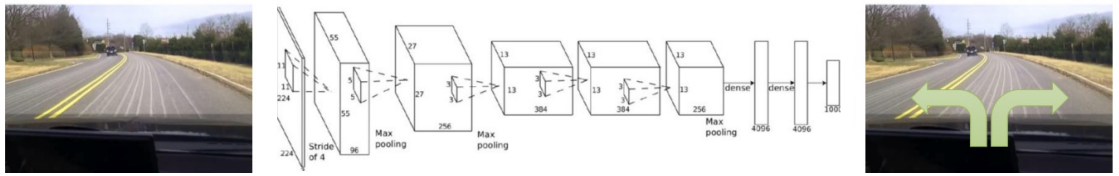


Pros: No rewards, online experience needed (?)

Why would we do this?

Given: Demonstrations of optimal behavior

Goal: Train a policy to mimic the demonstrator



Pros:

- ⊕ Avoids need for rewards, exploration
- ⊕ Natural way to do task specification
- ⊕ Can work well in practice

Cons:

- ⊖ Requires expert data, can be expensive
- ⊖ Cannot get better on deployment
- ⊖ Struggles on long horizon tasks

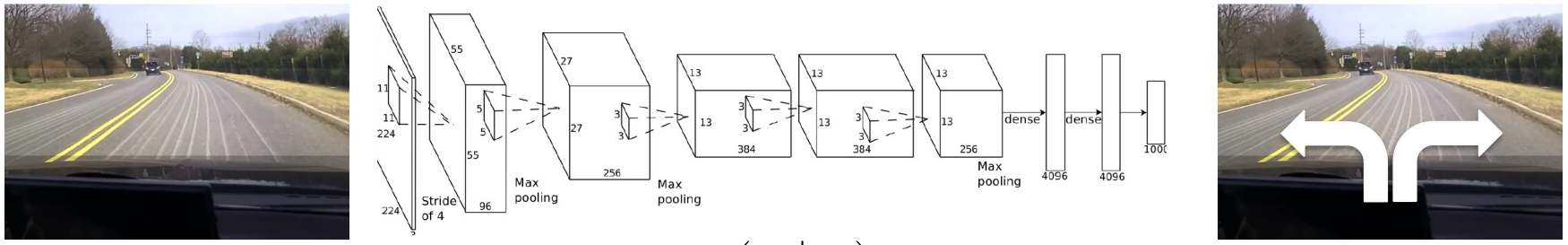
Idea 1: Imitation Learning via Supervised Learning

Given: Demonstrations of optimal behavior

Goal: Train a policy to mimic the demonstrator

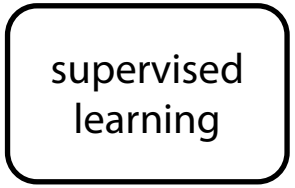
$$\arg\max_{\theta} \mathbb{E}_{(s^*,a^*)\sim\mathcal{D}} \left[\log \pi_\theta(a^*|s^*)\right]$$

Idea: Treat imitation learning as a supervised learning problem! → Behavior Cloning



$\mathbf{o}_t$ $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ $\mathbf{a}_t$

$\mathbf{o}_t$
$\mathbf{a}_t$



$\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$

Idea 1: Imitation Learning via Supervised Learning

Given: Demonstrations of optimal behavior

Goal: Train a policy to mimic the demonstrator

$$\arg\max_{\theta} \mathbb{E}_{(s^*,a^*)\sim\mathcal{D}} \left[\log \pi_\theta(a^*|s^*)\right]$$

Discrete vs continuous

Maximum likelihood

```
if isinstance(env.action_space, gym.spaces.Box):
    criterion = nn.MSELoss()
else:
    criterion = nn.CrossEntropyLoss()
# Extract initial policy
model = student.policy.to(device)
def train(model, device, train_loader, optimizer):
   model.train()
   for batch_idx, (data, target) in enumerate(train_loader):
      data, target = data.to(device), target.to(device)
      optimizer.zero_grad()
      if isinstance(env.action_space, gym.spaces.Box):
         if isinstance(student, (A2C, PPO)):
            action, _, _ = model(data)
         else:
            action = model(data)
         action_prediction = action.double()
      else:
         dist = model.get_distribution(data)
         action_prediction = dist.distribution.logits
         target = target.long()
      loss = criterion(action_prediction, target)
      loss.backward()
      optimizer.step()
```

Idea 1: Imitation Learning via Supervised Learning

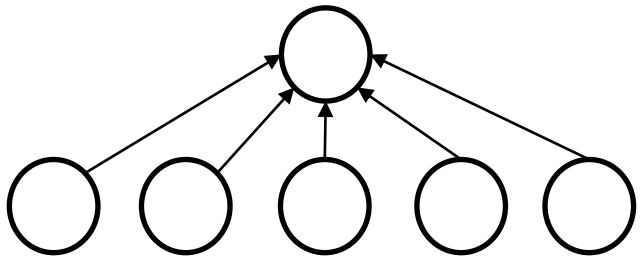
Given: Demonstrations of optimal behavior

Goal: Train a policy to mimic the demonstrator

$$\arg\max_{\theta} \mathbb{E}_{(s^*,a^*)\sim\mathcal{D}} \left[\log \pi_\theta(a^*|s^*)\right]$$

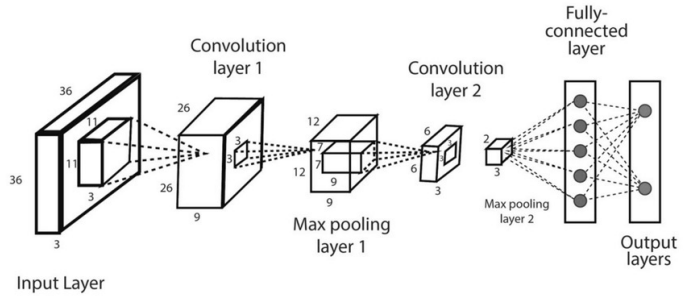
Tabular

Linear



$$\pi(a|s) = \langle \phi(s,a), w \rangle$$

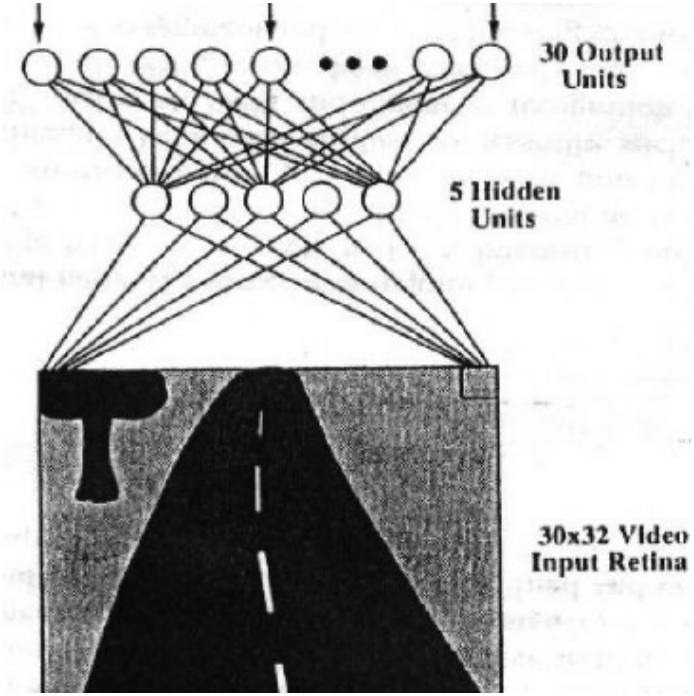
Arbitrary function approx



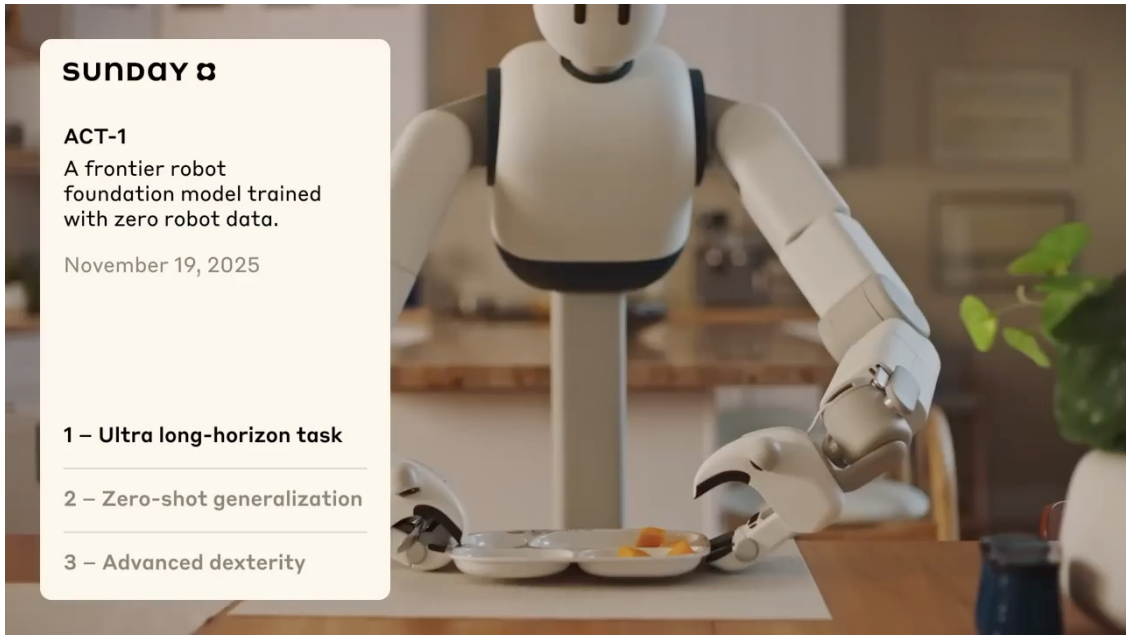
In practice, amounts to simple gradient based training with backpropagation

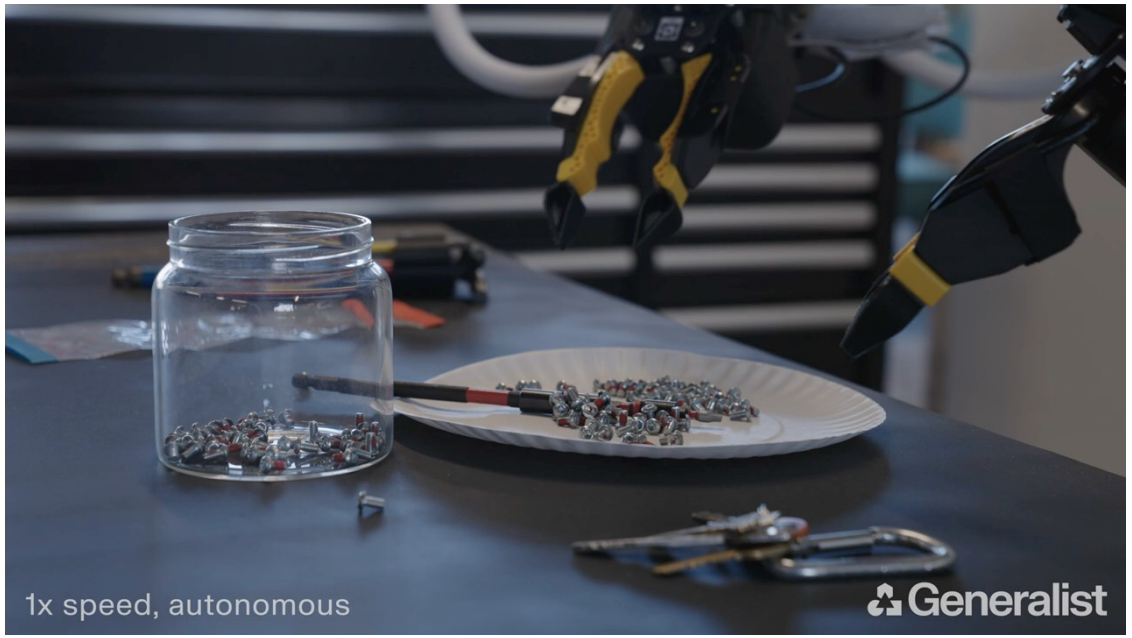
The original deep imitation learning system

ALVINN: **A**utonomous **L**and **V**ehicle **I**n a **N**eural **N**etwork
1989



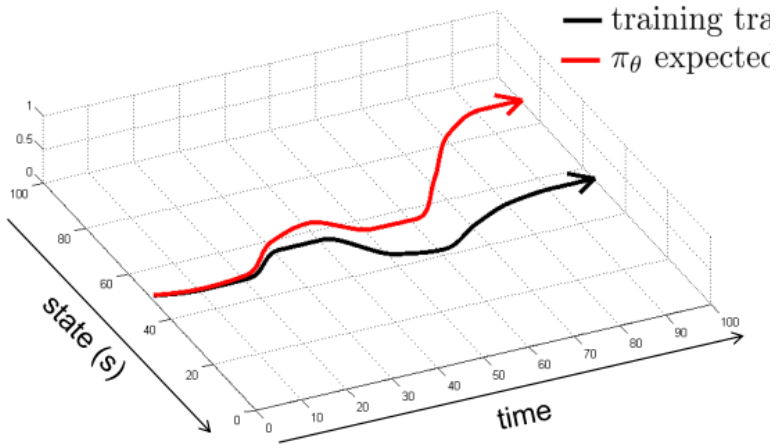
Where we are in 2026?





So does behavior cloning really work?

- Imitation Learning ≠ Supervised Learning



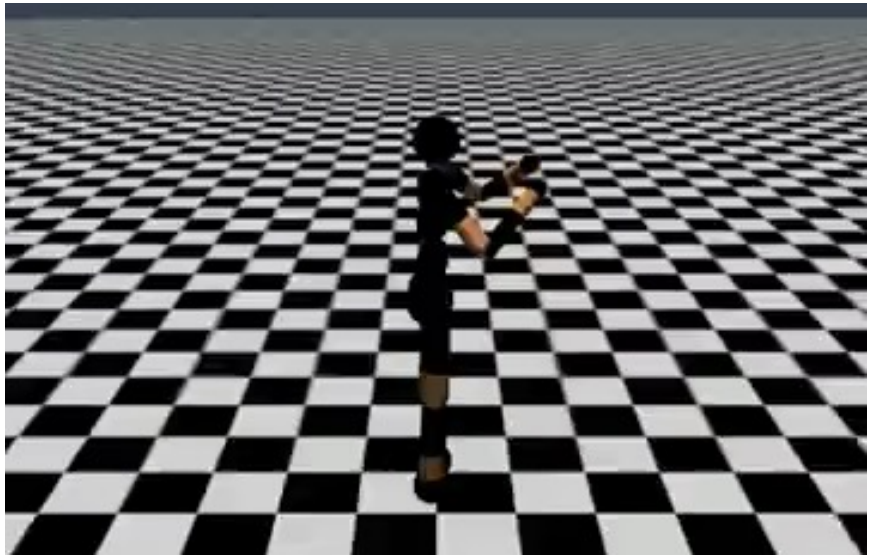
Compounding error!

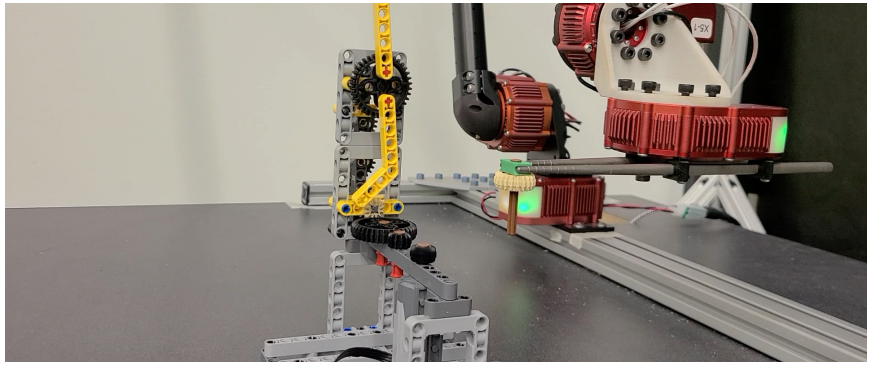
$$\arg\max_\theta \mathbb{E}_{(s^*,a^*)\sim\mathcal{D}}\left[\log \pi_\theta(a^*|s^*)\right] \qquad \mathbb{E}_{(s,a)\sim\rho(\pi)}\left[1(a=a^*)\right]$$

Not the same!

So does behavior cloning really work?

- Can fail in practice as well!





What do we actually want?

- Imitation Learning can be formalized as matching the expert

(cost for generating an action different than the expert)

$$c(s_t, a_t) = \begin{cases} 0, & \text{if } a_t = \pi^*(s_t), \\ 1, & \text{otherwise} \end{cases}$$

Measure deviation from expert actions when the policy is rolled out

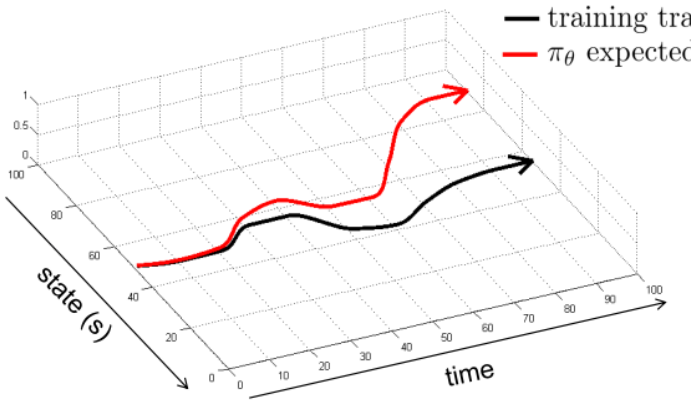
$$\mathbb{E}_{(s_t,a_t)\sim p_{\pi_\theta}(s_t,a_t)}\left[c(s_t, a_t)\right]$$

How bad is behavior cloning?

$$\arg\max_\theta \mathbb{E}_{(s^*,a^*)\sim\mathcal{D}}\left[\log \pi_\theta(a^*|s^*)\right]$$

How well does BC do?: Intuition

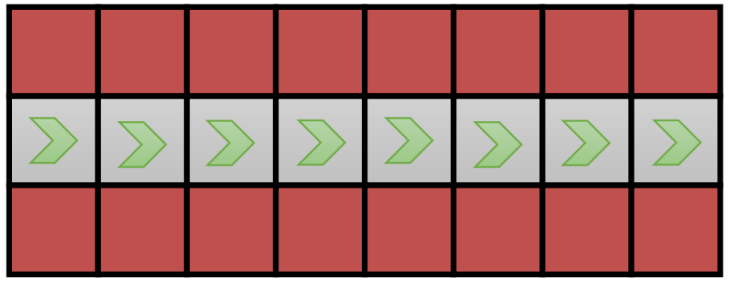
Behavior cloning has quadratically compounding error



$$\pi_\theta(a \neq \pi^*(s_t)|s_t) \leq \epsilon$$

Horizon H

If you fall off, assume the worst

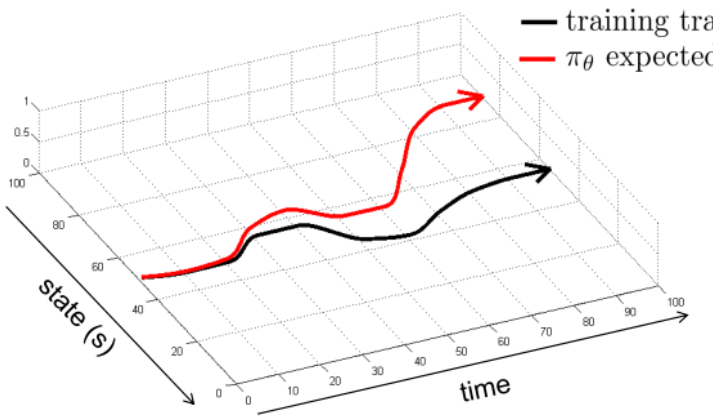


$$\underbrace{\mathbb{E}\left[\sum_t c(s_t, a_t)\right]}_{O(\epsilon H^2)} \leq \epsilon H + \cdots + \ldots$$

Let's try and understand where the problem lies?

Behavior cloning has challenges in both theory and practice

$$\sum_t \mathbb{E}_{(s_t,a_t)\sim p_{\pi_\theta}(s_t,a_t)}\left[c(s_t,a_t)\right] \leq O(\epsilon H^2)$$



Underfitting

$$\pi_\theta(a \neq \pi^*(s_t)|s_t) \leq \epsilon$$

Compounding error

$$\leq O(\epsilon H^2)$$

Lecture outline

Recap: MDP formalism + why should we care?

Imitation learning: preliminaries and behavior cloning

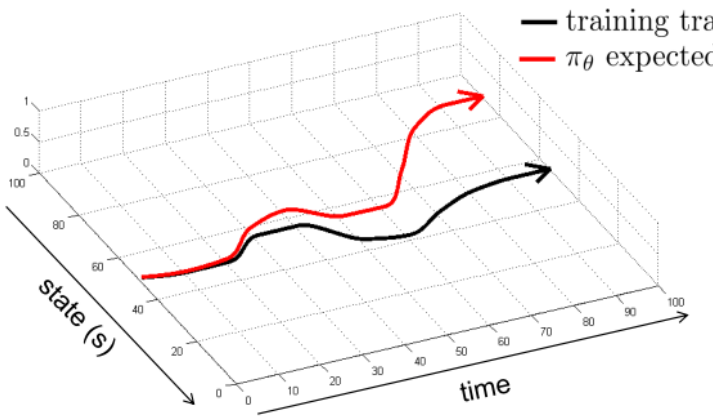
Multimodality and Underfitting in Imitation

Compounding Error in Imitation

Let's try and understand where the problem lies?

Behavior cloning has challenges in both theory and practice

$$\sum_t \mathbb{E}_{(s_t,a_t)\sim p_{\pi_\theta}(s_t,a_t)}\left[c(s_t,a_t)\right] \leq O(\epsilon H^2)$$



Underfitting

$$\pi_\theta(a \neq \pi^*(s_t)|s_t) \leq \epsilon$$

Compounding error

$$\leq O(\epsilon H^2)$$

But won't a bigger neural net just solve this?

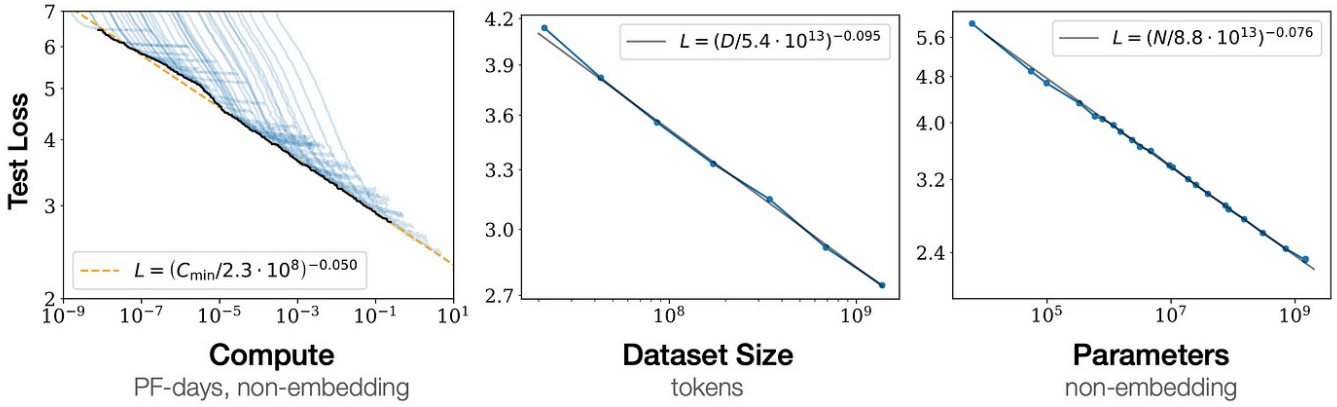
- Behavior cloning can underfit the data

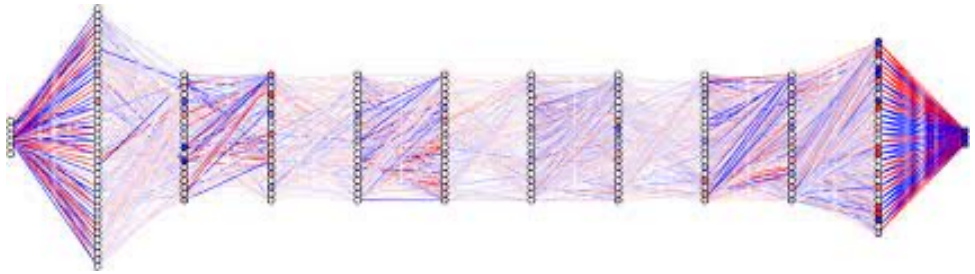
$$\sum_t \mathbb{E}_{(s_t,a_t)\sim p_{\pi_\theta}(s_t,a_t)}\left[c(s_t,a_t)\right] \leq O(\epsilon H^2)$$

$$\pi_\theta(a \neq \pi^*(s_t)|s_t) \leq \epsilon$$
$$\text{for } s_t \sim p_{\text{train}}(s_t)$$

May not be able to satisfy this

Q: won't a bigger model just solve the problem?



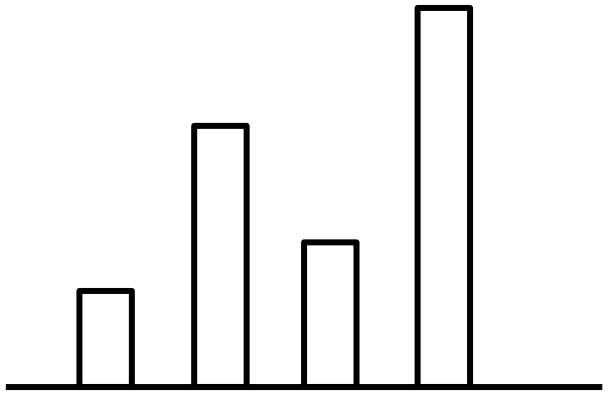


Kind of, but there's a fundamental problem!

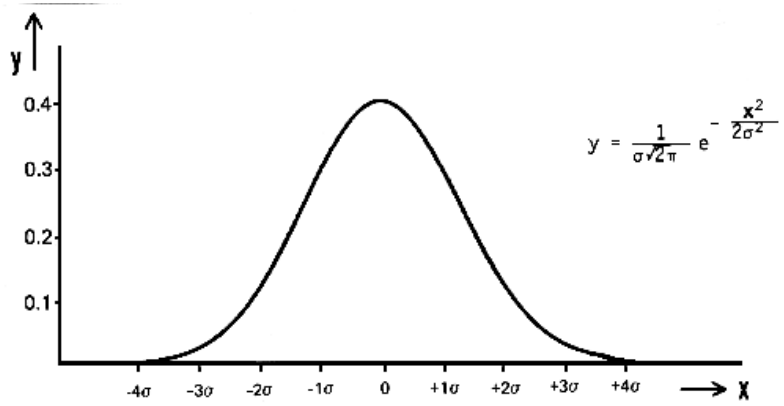
Distributional Expressivity

- Policy expressivity is a combination of expressivity of the function approximator and of the distribution family

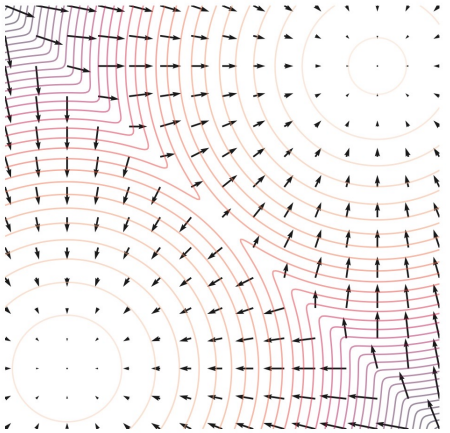
Categorical



Gaussian



Diffusion policy

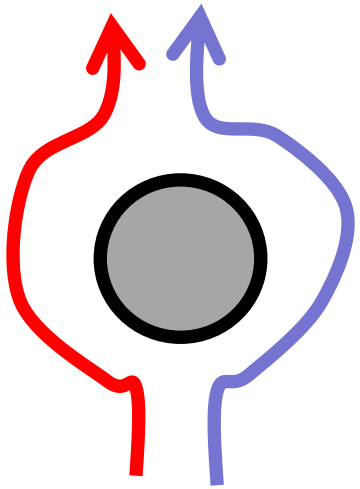


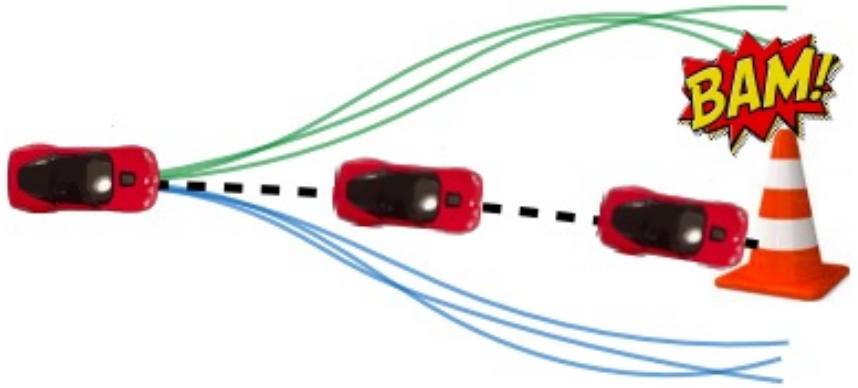
Tradeoff between expressivity and tractability

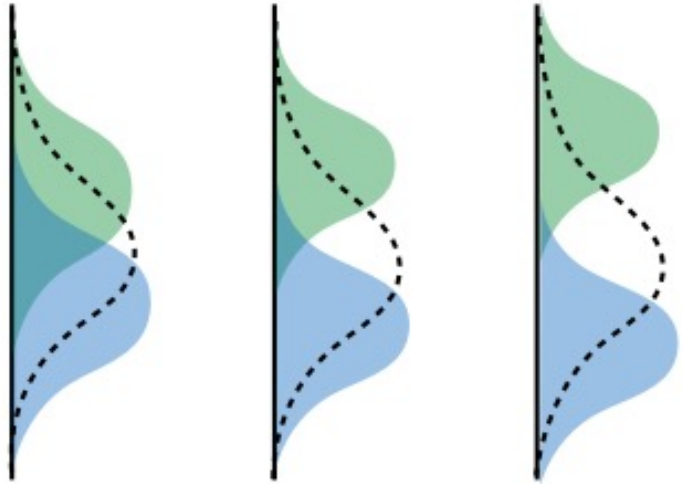
How does this reflect on imitation learning?

Let us consider a case with Gaussian policy

$$\arg\max_{\theta} \mathbb{E}_{(s^*,a^*)\sim\mathcal{D}} \left[\log \pi_\theta(a^*|s^*)\right]$$

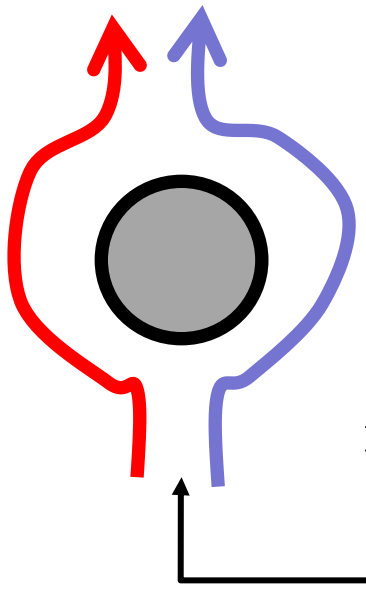






A combination of **distributional expressivity** and **objective** lead to mode averaging

Let's take a closer look at the objective



$$\max_{\theta} \mathbb{E}_{(s^*,a^*)\sim\mathcal{D}} \left[\log \pi_\theta(a^*|s^*)\right]$$

$$\max_{\theta} \mathbb{E}_{s^*\sim p_{\pi_e}(.)} \left[\mathbb{E}_{a^*\sim\pi_e(.|s^*)} \left[\log \pi_\theta(a^*|s^*) - \log \pi_e(a^*|s^*)\right]\right]$$

$$\min_{\theta} \mathbb{E}_{s^*\sim p_{\pi_e}(.)} \left[\mathbb{E}_{a^*\sim\pi_e(.|s^*)} \left[\log \frac{\pi_e(a^*|s^*)}{\pi_\theta(a^*|s^*)}\right]\right] = \mathbb{E}_{s^*\sim p_{\pi_e}(.)} \left[D_{\mathrm{KL}}(\pi_e(.|s^*)||\pi_\theta(.|s^*))\right]$$

Leads to mode averaging

Forward KL divergence

One instance of a broader class of divergences – f diverences

$$D_f(p(x), q(x)) = \mathbb{E}_{q(x)} \left[f\left(\frac{p(x)}{q(x)}\right)\right]$$

https://timvieira.github.io/blog/post/2014/10/06/kl-divergence-as-an-objective-function/

Effects of choice of f-divergence on behavior

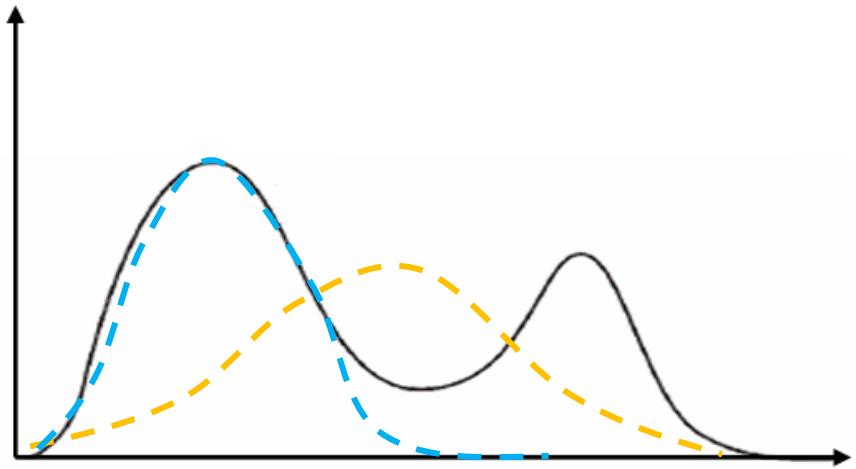
Different divergences lead to different properties

$$\mathbb{E}_{s^* \sim p_{\pi_e}(.)} \left[D_{\mathrm{KL}}(\pi_e(.|s^*) || \pi_\theta(.|s^*)) \right] \longrightarrow \mathbb{E}_{s^* \sim p_{\pi_e}(.)} \left[D_f(\pi_e(.|s^*), \pi_\theta(.|s^*)) \right]$$

Forward KL (behavior cloning)

More general class of divergences

$$D_f(p(x), q(x)) = \mathbb{E}_{q(x)} \left[f\left(\frac{p(x)}{q(x)} \right) \right]$$



Forward KL (mode covering) $f(x) = x \log(x)$

Reverse KL (mode seeking) $f(x) = -\log(x)$

So how do we fix BC?

Use a different f-divergence! (Change f)

or

Use a richer distribution class! (Change π_θ)

Using alternative f-divergences: Reverse KL

- Reverse KL helps, is mode seeking $D_{\mathrm{RKL}}(\pi_e(.|s^*), \pi^\theta(.|s^*)) = \mathbb{E}_{\pi^\theta(.|s^*)}\left[\log\left(\frac{\pi^\theta(.|s^*)}{\pi_e(.|s^*)}\right)\right]$
- Challenge – requires known expert likelihood
- We need a sample based estimate!

Imitation Learning as f-Divergence Minimization

Liyiming Ke[1], Sanjiban Choudhury[1], Matt Barnes[1], Wen Sun[2], Gilwoo Lee[1], and Siddhartha Srinivasa[1]

Go read this!

$$\min_\theta \mathbb{E}_{\pi^\theta(.|s^*)}\left[\log\left(\frac{\pi^\theta(.|s^*)}{\pi_e(.|s^*)}\right)\right] \longleftrightarrow \min_\theta \max_\phi \mathbb{E}_{a\sim\pi^\theta(.|s^*)}\left[\phi(a)\right] - \mathbb{E}_{a\sim\pi_e(.|s^*)}\left[f^*(\phi(a))\right]$$

(Intractable)

(Tractable – GAN style optimization)

Effects of choice of f-divergence on behavior

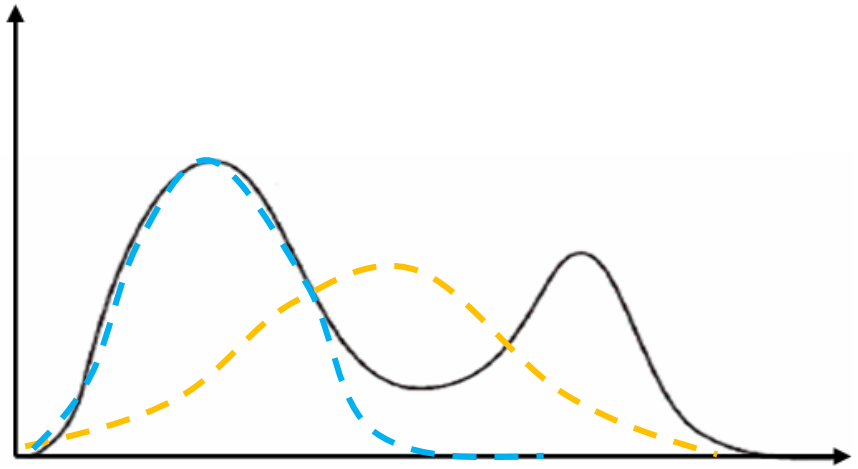
Different divergences lead to different properties

$$\mathbb{E}_{s^* \sim p_{\pi_e}(.)}\left[D_{\mathrm{KL}}(\pi_e(.|s^*)||\pi_\theta(.|s^*))\right] \longrightarrow \mathbb{E}_{s^* \sim p_{\pi_e}(.)}\left[D_f(\pi_e(.|s^*), \pi_\theta(.|s^*))\right]$$

Forward KL (behavior cloning)

More general class of divergences

$$D_f(p(x), q(x)) = \mathbb{E}_{q(x)}\left[f\left(\frac{p(x)}{q(x)}\right)\right]$$



Forward KL (mode covering) $f(x) = x\log(x)$

Reverse KL (mode seeking) $f(x) = -\log(x)$

So how do we fix BC?

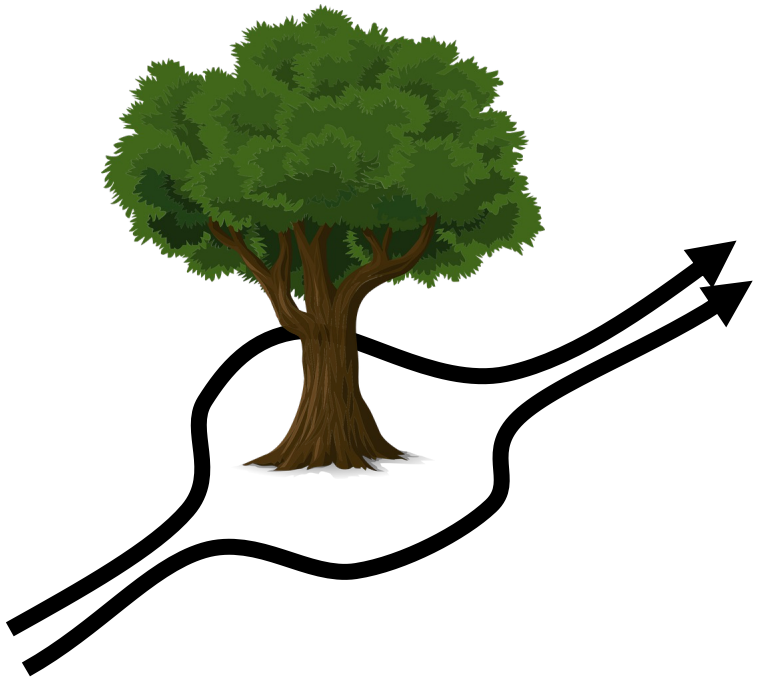
Use a different f-divergence! (Change f)

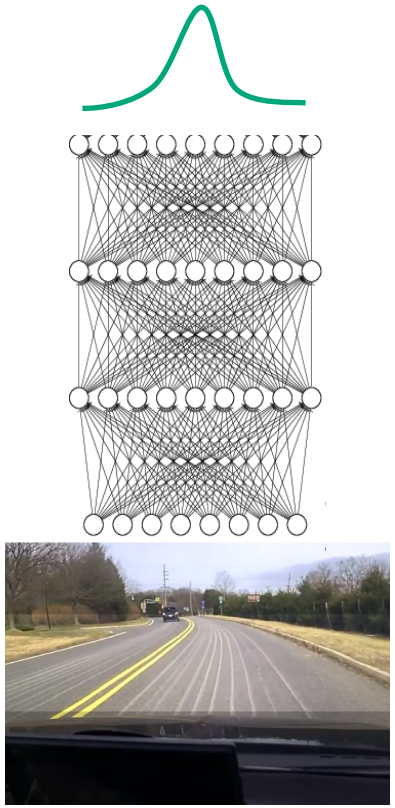
or

Use a richer distribution class! (Change π_θ)

Using Richer Policy Distribution Classes

Multimodal behavior → use more **expressive** probability distributions, no mode averaging issues



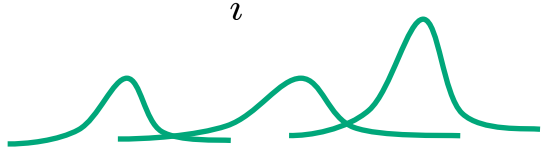


1. Output mixture of Gaussians
2. Latent variable models
3. Autoregressive discretization
4. Diffusion models
5. …

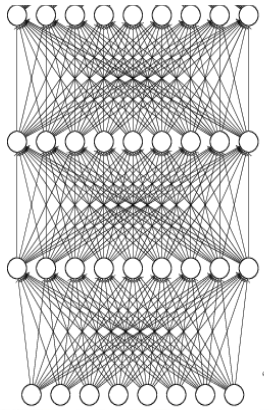
Why might we fail to fit the expert?

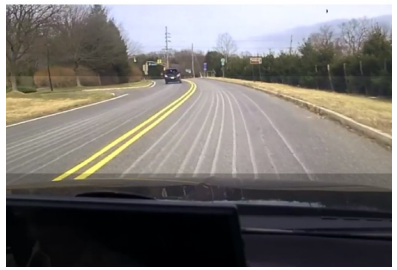
1. Output mixture of Gaussians
2. Latent variable models
3. Autoregressive discretization
4. Diffusion models
5. …

$$\pi(\mathbf{a}|\mathbf{o}) = \sum_i w_i \mathcal{N}(\mu_i, \Sigma_i)$$



$$w_1, \mu_1, \Sigma_1, \ldots, w_N, \mu_N, \sigma_N$$

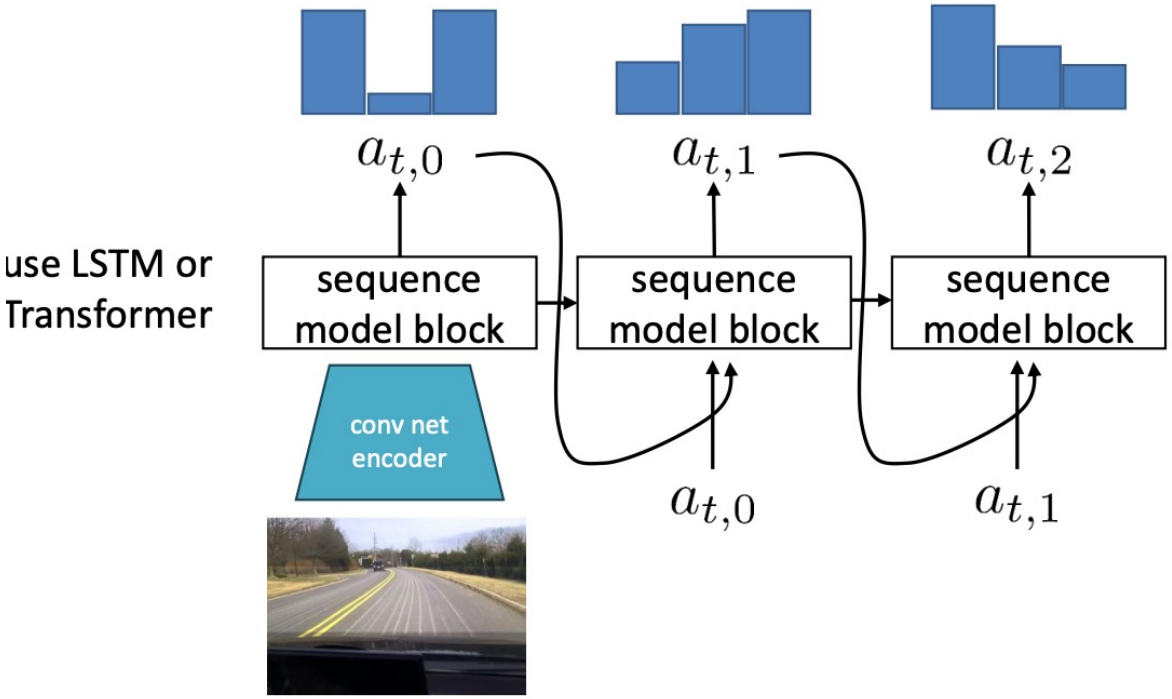




Why might we fail to fit the expert?

1. Output mixture of Gaussians
2. Latent variable models
3. Autoregressive discretization
4. Diffusion models
5. …

$$\mathbf{a}_t = \begin{pmatrix} 0.1 \\ 1.2 \\ -0.3 \end{pmatrix} \begin{matrix} a_{t,0} \\ a_{t,1} \\ a_{t,2} \end{matrix}$$



Why does this work?

first step: $p(a_{t,0}|\mathbf{s}_t)$

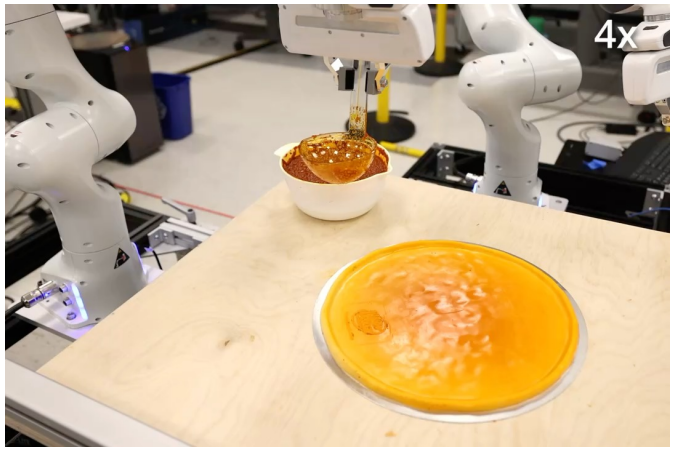
second step: $p(a_{t,1}|\mathbf{s}_t, a_{t,0})$

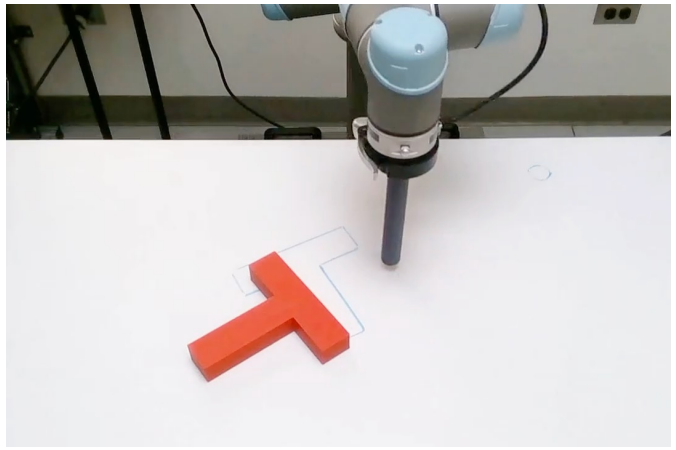
third step: $p(a_{t,2}|\mathbf{s}_t, a_{t,0}, a_{t,1})$

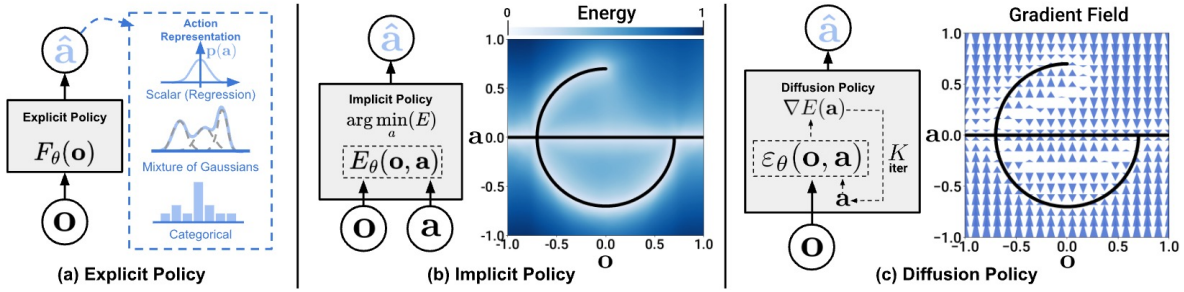
$$p(a_{t,2}|\mathbf{s}_t, a_{t,0}, a_{t,1})p(a_{t,1}|\mathbf{s}_t, a_{t,0})p(a_{t,0}|\mathbf{s}_t)$$
$$= p(a_{t,0}, a_{t,1}, a_{t,2}|\mathbf{s}_t)$$
$$= p(\mathbf{a}_t|\mathbf{s}_t)$$

Why might we fail to fit the expert?

1. Output mixture of Gaussians
2. Latent variable models
3. Autoregressive discretization
4. Diffusion models
5. …

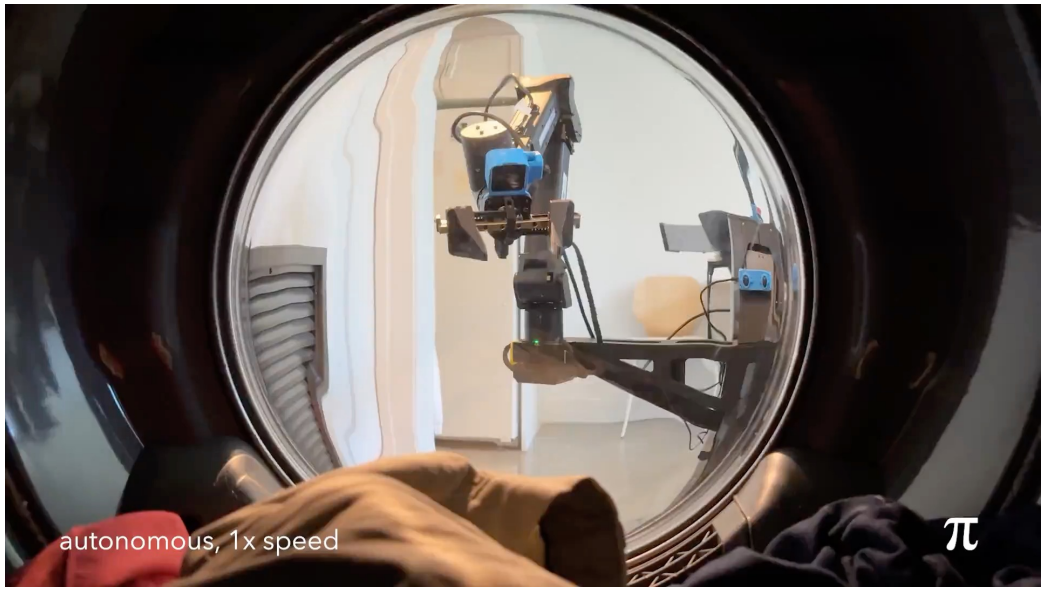


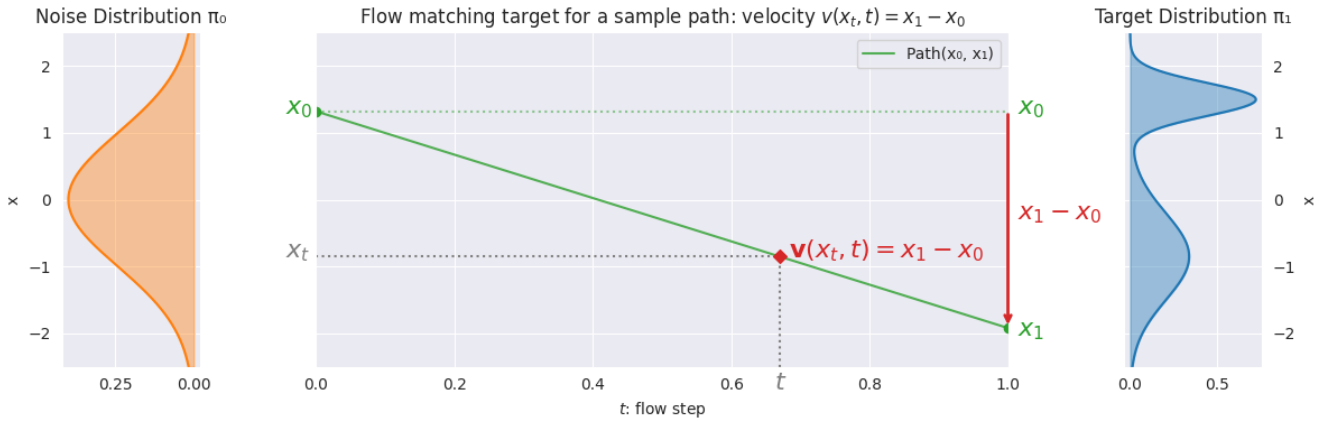




Why might we fail to fit the expert?

1. Output mixture of Gaussians
2. Latent variable models
3. Autoregressive discretization
4. Flow models
5. …

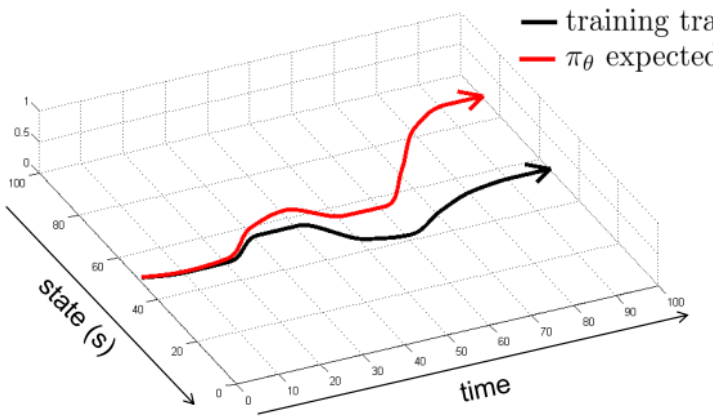




Let's try and understand where the problem lies?

Behavior cloning has challenges in both theory and practice

$$\sum_t \mathbb{E}_{(s_t,a_t)\sim p_{\pi_\theta}(s_t,a_t)}\left[c(s_t,a_t)\right] \leq O(\epsilon H^2)$$



Underfitting

$$\pi_\theta(a \neq \pi^*(s_t)|s_t) \leq \epsilon$$

Richer policy class

Alternative Divergence

Compounding error

$$\leq O(\epsilon H^2)$$

Lecture outline

Recap: MDP formalism + why should we care?

Imitation learning: preliminaries and behavior cloning

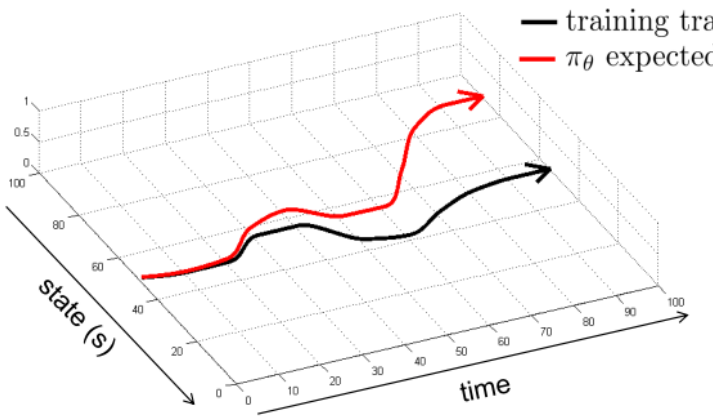
Multimodality and Underfitting in Imitation

Compounding Error in Imitation

Let's try and understand where the problem lies?

Behavior cloning has challenges in both theory and practice

$$\sum_t \mathbb{E}_{(s_t,a_t)\sim p_{\pi_\theta}(s_t,a_t)}\left[c(s_t,a_t)\right] \leq O(\epsilon H^2)$$



Underfitting

$$\pi_\theta(a \neq \pi^*(s_t)|s_t) \leq \epsilon$$

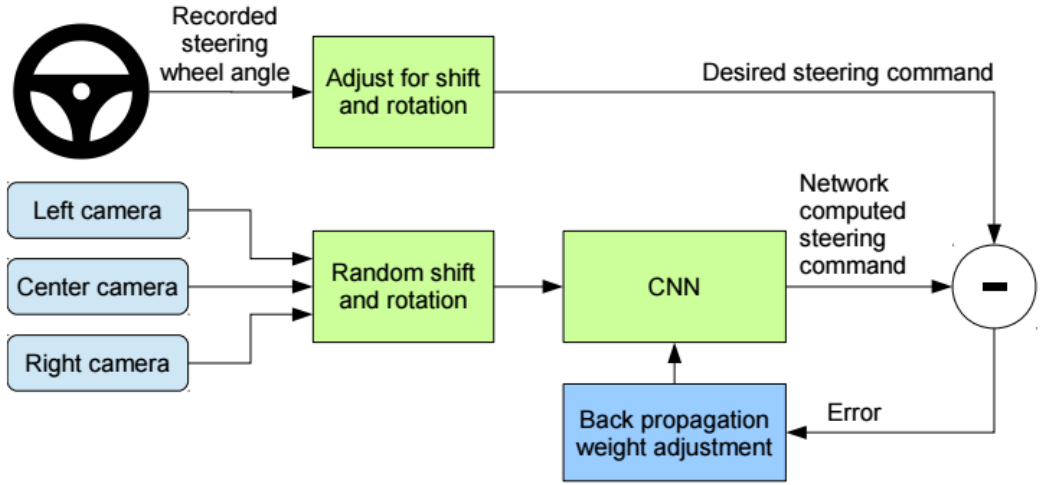
Compounding error

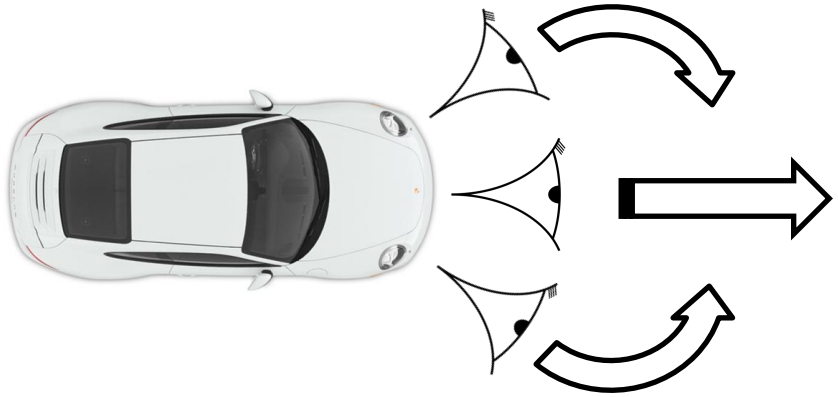
$$\leq O(\epsilon H^2)$$

Can we avoid compounding error in special cases?

Video: Bojarski et al. ‘16, NVIDIA

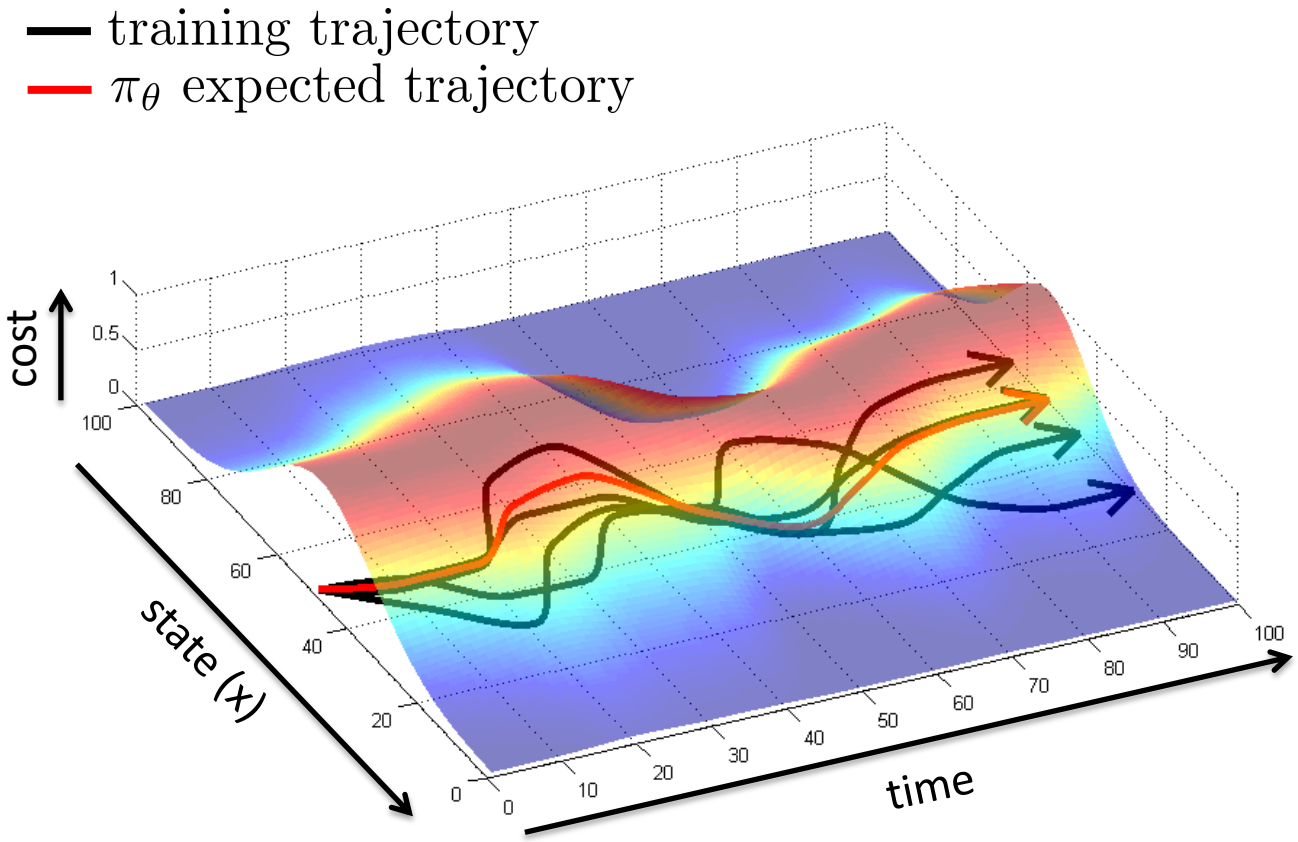
Why did that work?





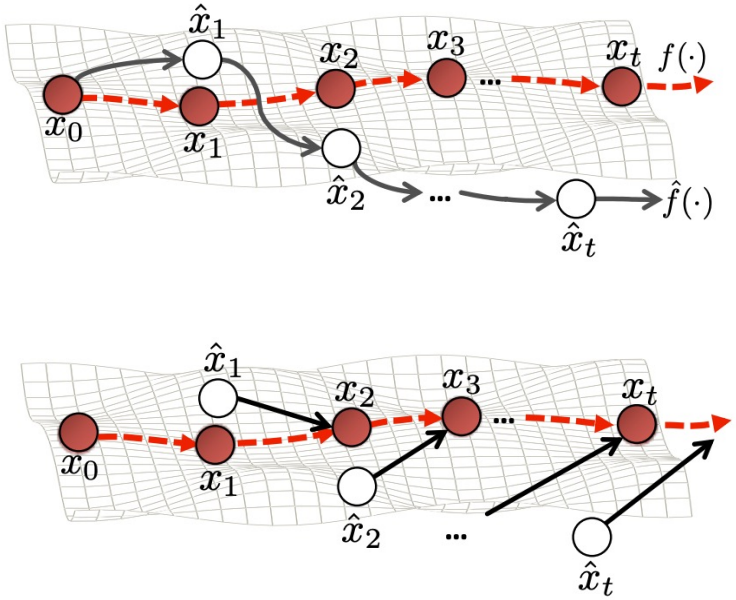
Bojarski et al. '16, NVIDIA

What is the general principle?

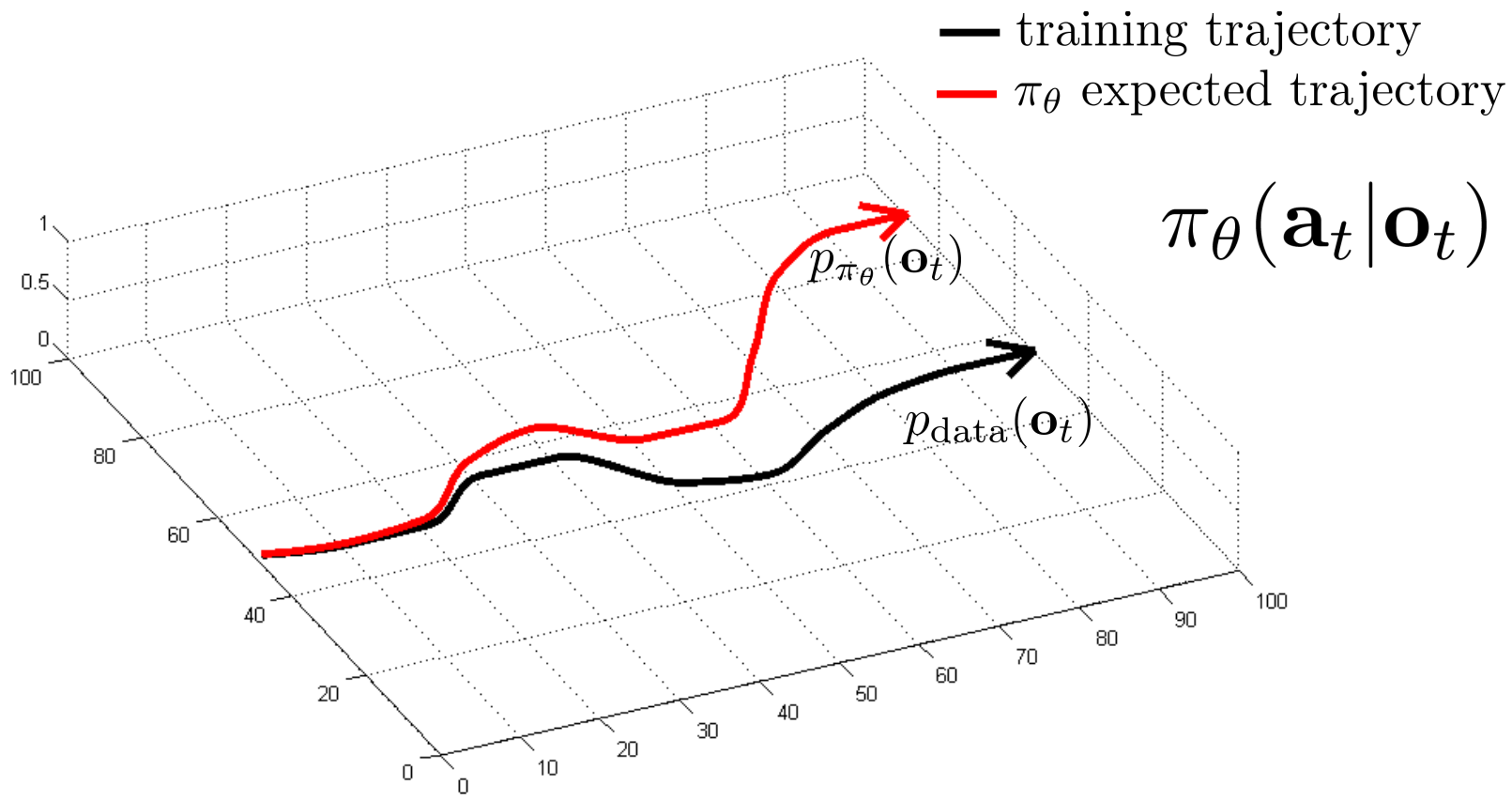


stability

Corrective labels that bring you back to the data



What might this mean mathematically?



can we make $p_{\text{data}}(\mathbf{o}_t) = p_{\pi_\theta}(\mathbf{o}_t)$?

Concrete Instantation: DAgger

can we make $p_{\text{data}}(\mathbf{o}_t) = p_{\pi_\theta}(\mathbf{o}_t)$?

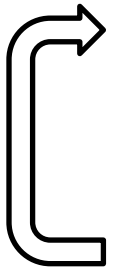
idea: instead of being clever about $p_{\pi_\theta}(\mathbf{o}_t)$, be clever about $p_{\text{data}}(\mathbf{o}_t)$!

DAgger: Dataset Aggregation

goal: collect training data from $p_{\pi_\theta}(\mathbf{o}_t)$ instead of $p_{\text{data}}(\mathbf{o}_t)$

how? just run $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$

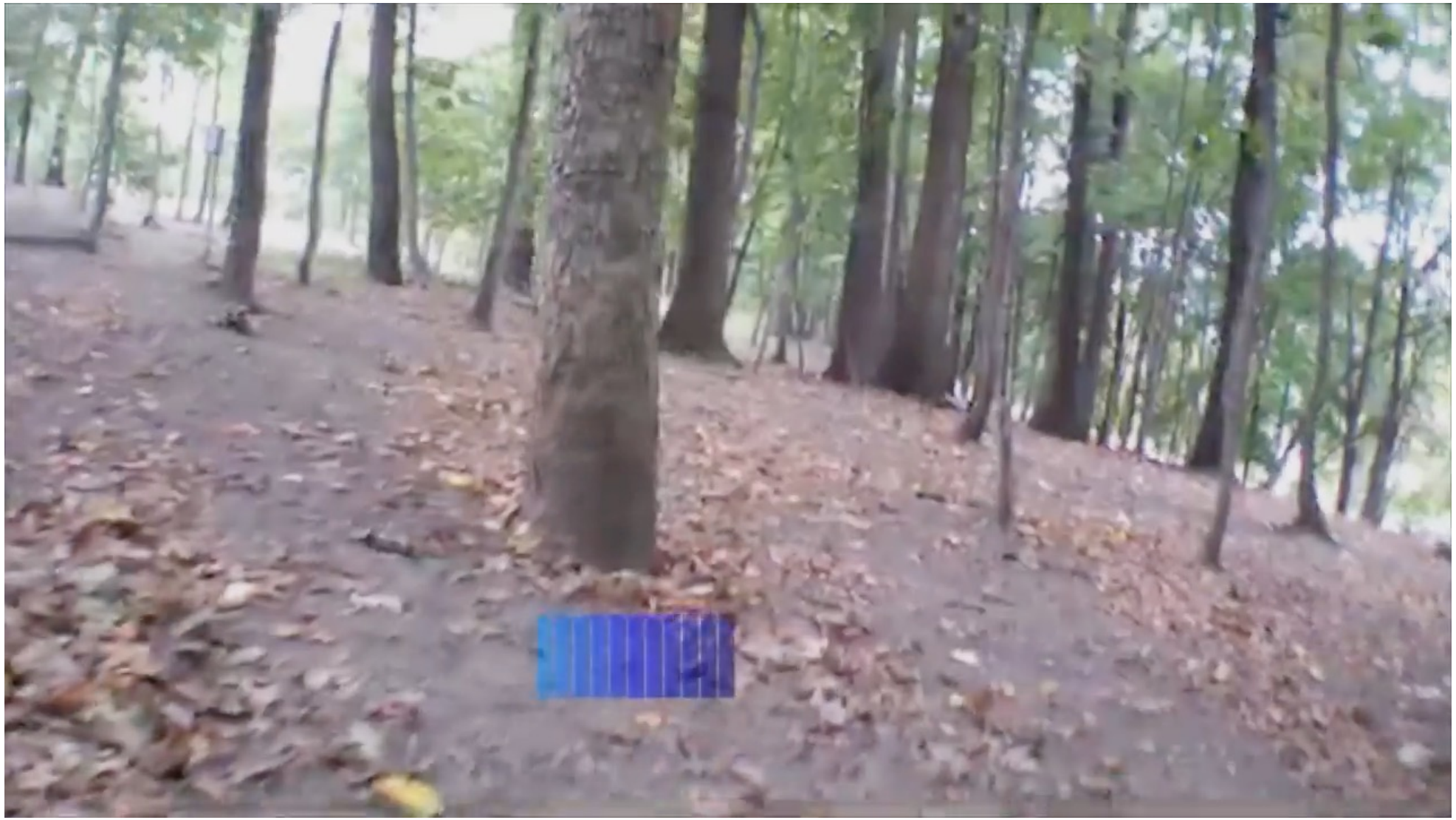
but need labels $\mathbf{a}_t$!



1. train $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ from human data $\mathcal{D} = \{\mathbf{o}_1, \mathbf{a}_1, \ldots, \mathbf{o}_N, \mathbf{a}_N\}$
2. run $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ to get dataset $\mathcal{D}_\pi = \{\mathbf{o}_1, \ldots, \mathbf{o}_M\}$
3. Ask human to label $\mathcal{D}_\pi$ with actions $\mathbf{a}_t$
4. Aggregate: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$

Ross et al. '11

DAgger Example

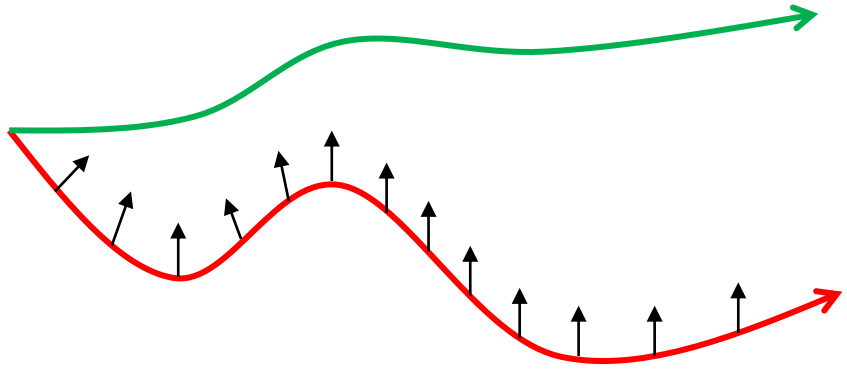


Ross et al. '13

DAgger + reverse KL: On-Policy Distillation

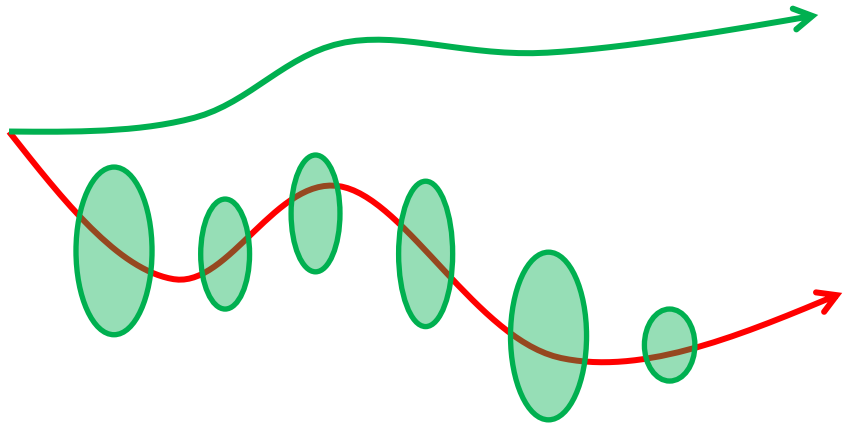
- Reverse KL objectives have made a return in LLMs through distillation

DAgger



$$\max_{\theta} \mathbb{E}_{\substack{s\sim d^{\pi_{\bar{\theta}}} \\ a\sim\pi^*(.|s)}} [\log \pi_\theta(a|s)]$$

On-Policy Distillation



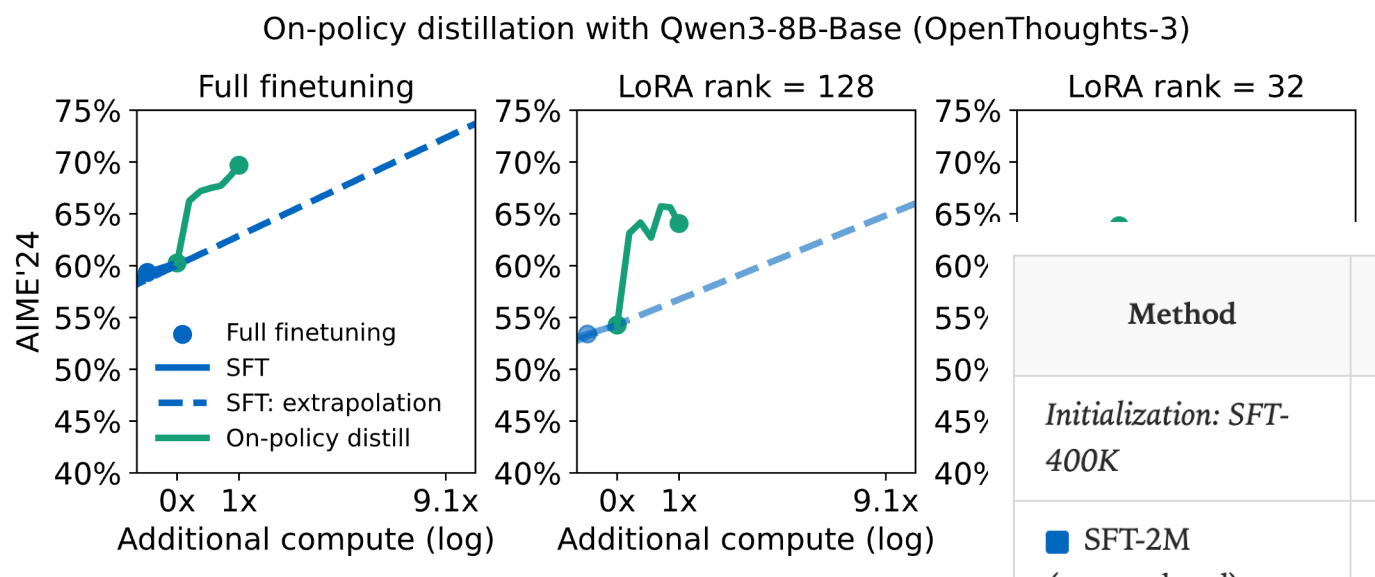
$$\max_{\theta} \mathbb{E}_{s\sim d^{\pi_{\bar{\theta}}}} [D_f(\pi_\theta(\cdot|s), \pi^*(\cdot|s)]$$

↓

Can include reverse KL

On-Policy Distillation: What can it do?

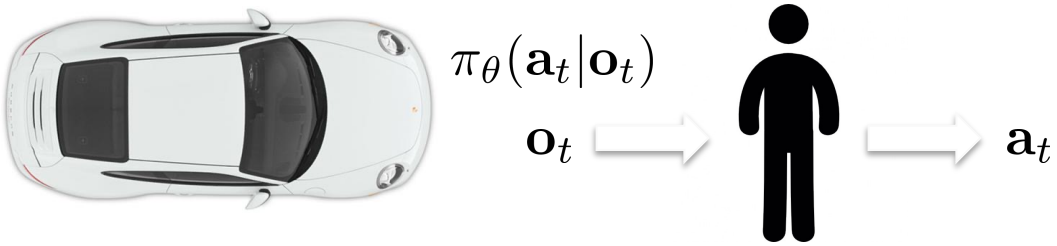
- Provides gains in performance with much smaller models



Method	AIME'24	Teacher FLOPs	Student FLOPs	CE vs SFT-2M
Initialization: SFT-400K	60%	8.5×10^{20}	3.8×10^{20}	–
SFT-2M (extrapolated)	~70% (extrapolated)	3.4×10^{21}	1.5×10^{21}	1×
Reinforcement learning	68%	-	-	≈1×
On-policy distillation	70%	8.4×10^{19}	8.2×10^{19}	**9-30×**

What's the problem?

1. train $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ from human data $\mathcal{D} = \{\mathbf{o}_1, \mathbf{a}_1, \ldots, \mathbf{o}_N, \mathbf{a}_N\}$
2. run $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ to get dataset $\mathcal{D}_\pi = \{\mathbf{o}_1, \ldots, \mathbf{o}_M\}$
3. Ask human to label $\mathcal{D}_\pi$ with actions $\mathbf{a}_t$
4. Aggregate: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$



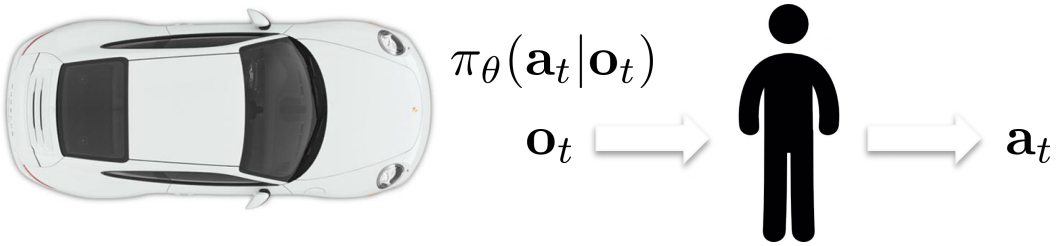
Ross et al. '11

How might we fix this?

Can we reduce human cost?

1. train $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ from human data $\mathcal{D} = \{\mathbf{o}_1, \mathbf{a}_1, \ldots, \mathbf{o}_N, \mathbf{a}_N\}$
2. run $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ to get dataset $\mathcal{D}_\pi = \{\mathbf{o}_1, \ldots, \mathbf{o}_M\}$
3. Ask human to label $\mathcal{D}_\pi$ with actions $\mathbf{a}_t$
4. Aggregate: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$

Can we avoid labeling post-hoc?

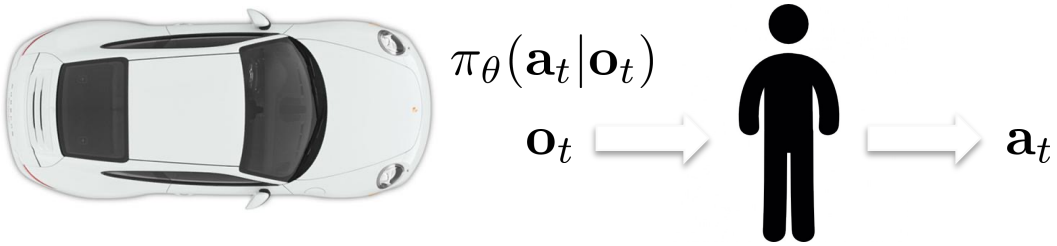


Ross et al. '11

How might we fix this?

1. train $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ from human data $\mathcal{D} = \{\mathbf{o}_1, \mathbf{a}_1, \ldots, \mathbf{o}_N, \mathbf{a}_N\}$
2. run $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ to get dataset $\mathcal{D}_\pi = \{\mathbf{o}_1, \ldots, \mathbf{o}_M\}$
3. Ask human to label $\mathcal{D}_\pi$ with actions $\mathbf{a}_t$
4. Aggregate: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$

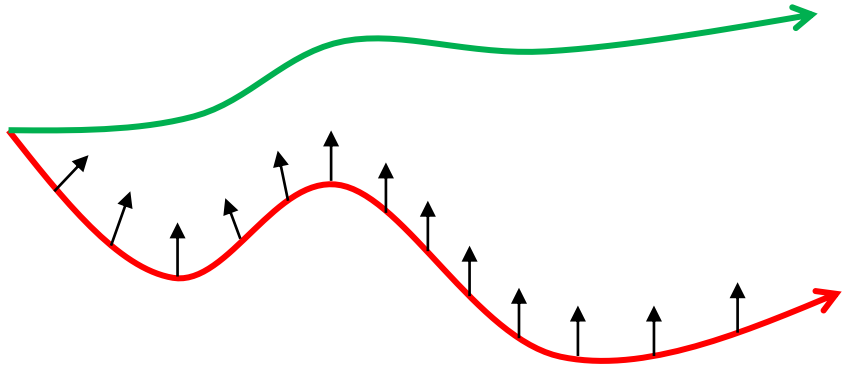
Can we avoid labeling post-hoc?



Ross et al. '11

DAgger is a bit impractical –> HG-DAgger

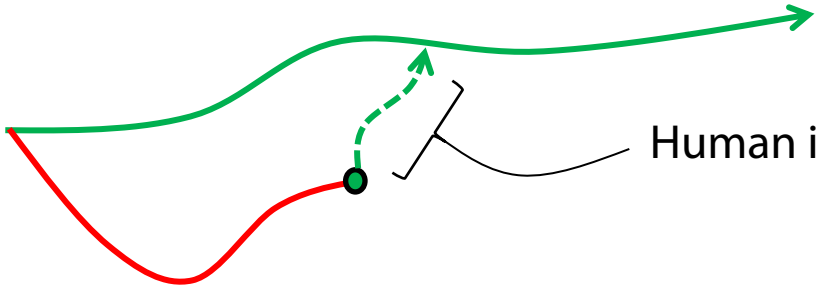
Dagger requires revisiting states and labeling post-hoc



$$s_t \sim d^{\pi_\theta}$$
$$a \sim \pi_h(a|s)$$

Can do "blended" control, but becomes hard to get precision

What if we allowed people to take control for longer?



$s_t \sim d^{\pi_\theta}$ (point of intervention)

$s_{t+1:t+k} \sim d^{\pi_h}|s_t$ (human takeover)

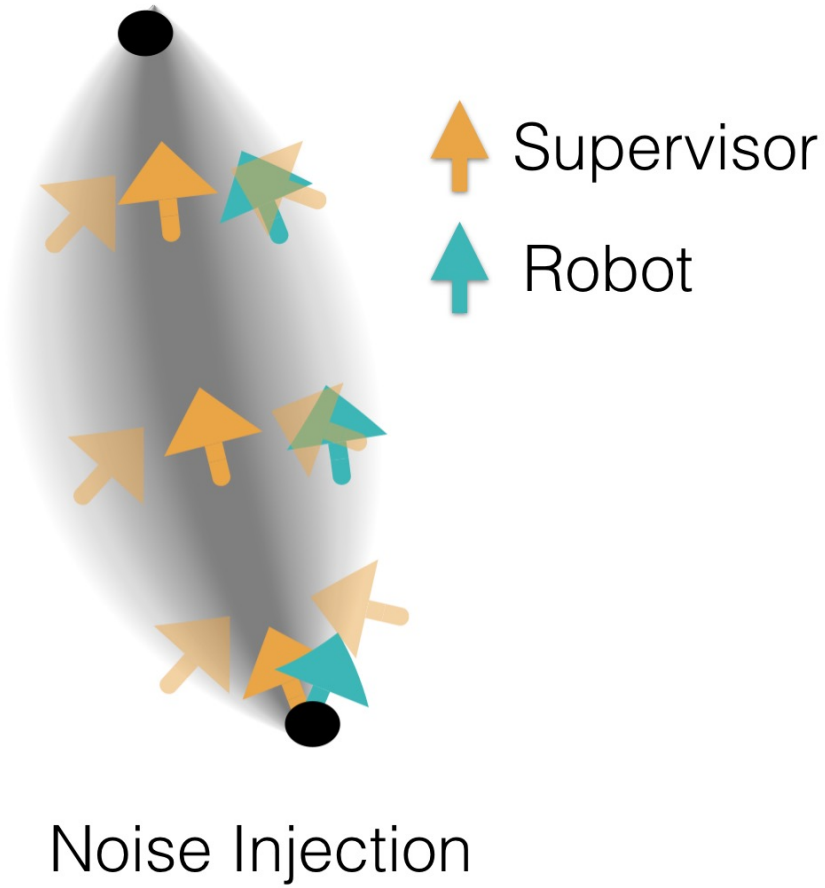
$a \sim \pi_h(a|s)$ (human actions)

Can we do this pre-emptively?

Kelly et al, HG-Dagger, 2018

Noising the Data Collection Process

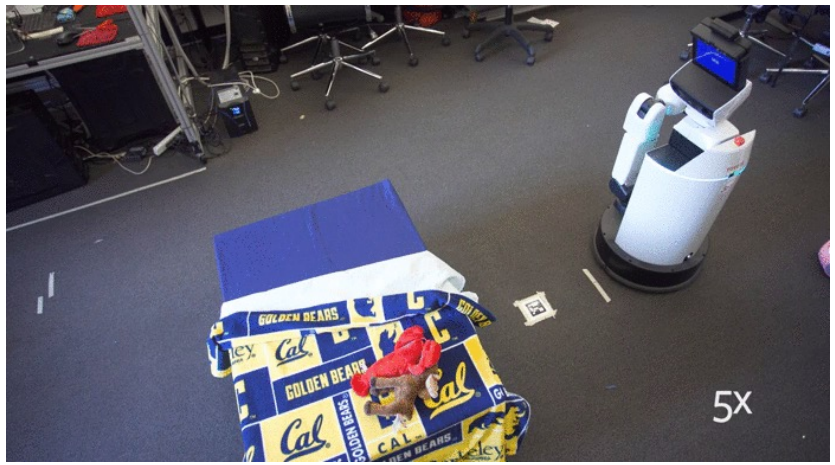
Key idea: force the human to correct for noise **during** training



Under noise during data collection

Maximize likelihood

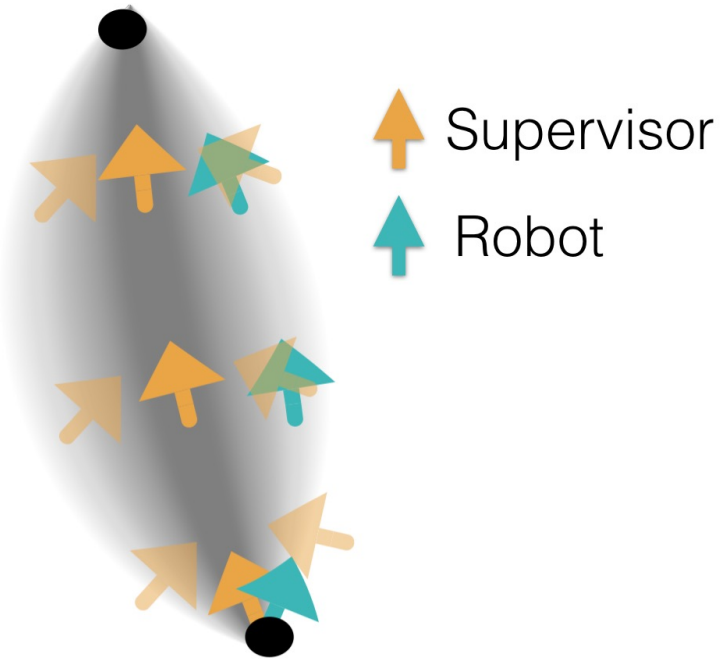
$$\hat{\psi}_{k+1} = \underset{\psi}{\operatorname{argmin}} E_{p(\xi|\pi_{\theta^*},\psi_k)} - \sum_{t=0}^{T-1} \log\left[\pi_{\theta^*}(\pi_{\hat{\theta}}(\mathbf{x_t})|\mathbf{x_t},\psi)\right]$$



DART: Noise Injection for Robust Imitation Learning, Laskey et al CoRL ‘17

Why might this not be enough?

Key idea: force the human to correct for noise **during** training



Noise Injection

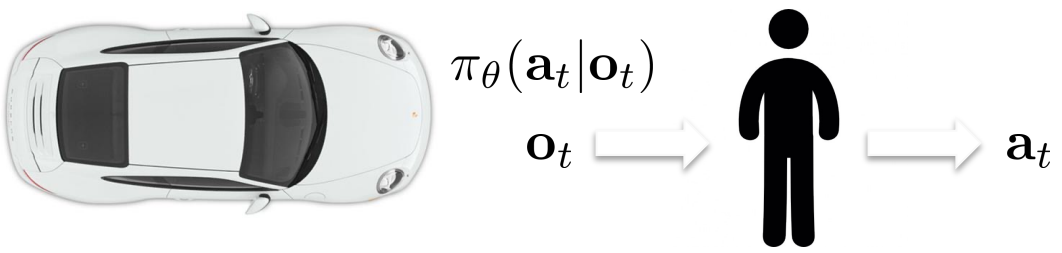
Assumes that the expert **can** actually perform behaviors under noise
→ Not always possible!

DART: Noise Injection for Robust Imitation Learning, Laskey et al CoRL ‘17

How might we fix this?

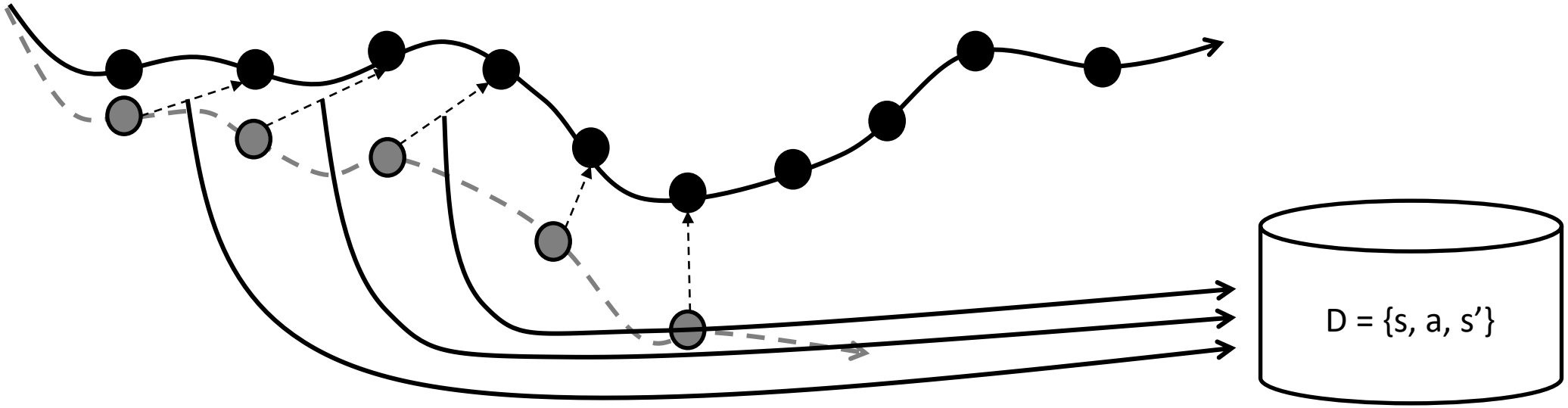
Can we reduce human cost?

1. train $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ from human data $\mathcal{D} = \{\mathbf{o}_1, \mathbf{a}_1, \ldots, \mathbf{o}_N, \mathbf{a}_N\}$
2. run $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ to get dataset $\mathcal{D}_\pi = \{\mathbf{o}_1, \ldots, \mathbf{o}_M\}$
3. Ask human to label $\mathcal{D}_\pi$ with actions $\mathbf{a}_t$
4. Aggregate: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$



Ross et al. '11

Can we avoid expensive online data collection/labeling?



Generate corrective labels to dataset for imitation

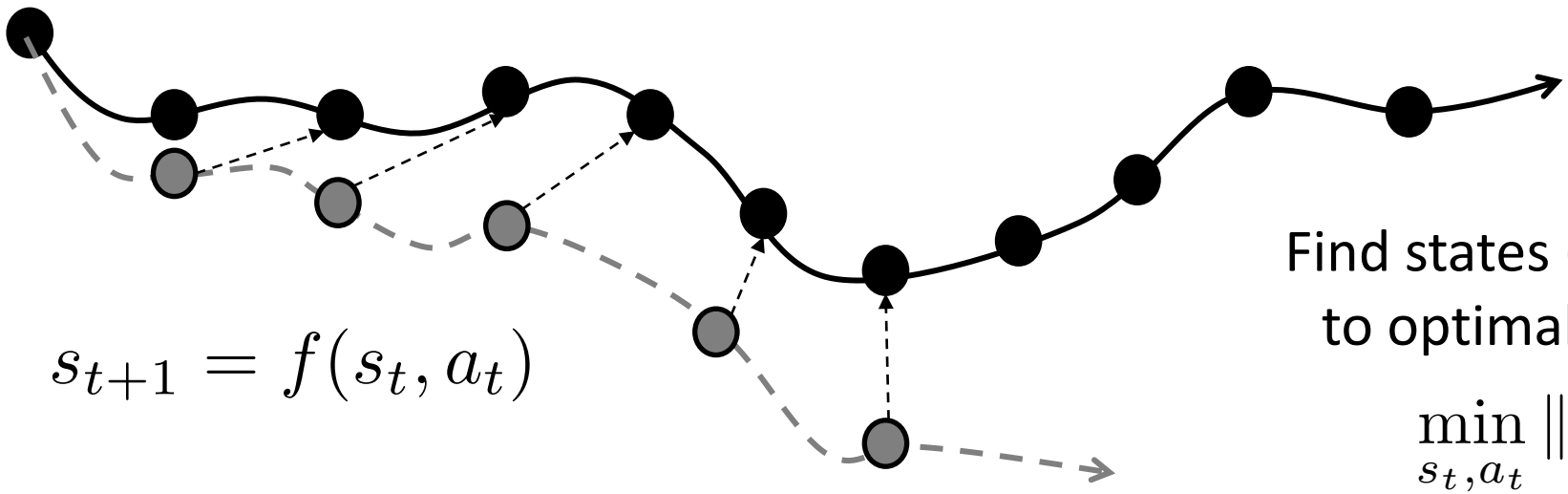
Abhay Deshpande

Yunchu Zhang

Liyiming Ke

How can we find corrective labels without an expensive human in the loop and online data collection?

Generating Corrective Labels From True Dynamics



$$s_{t+1} = f(s_t, a_t)$$

Find states (s_t), actions (a_t) that lead back to optimal states under true dynamics

$$\min_{s_t, a_t} \|s^*_{t+1} - f(s_t, a_t)\| \leq \epsilon$$

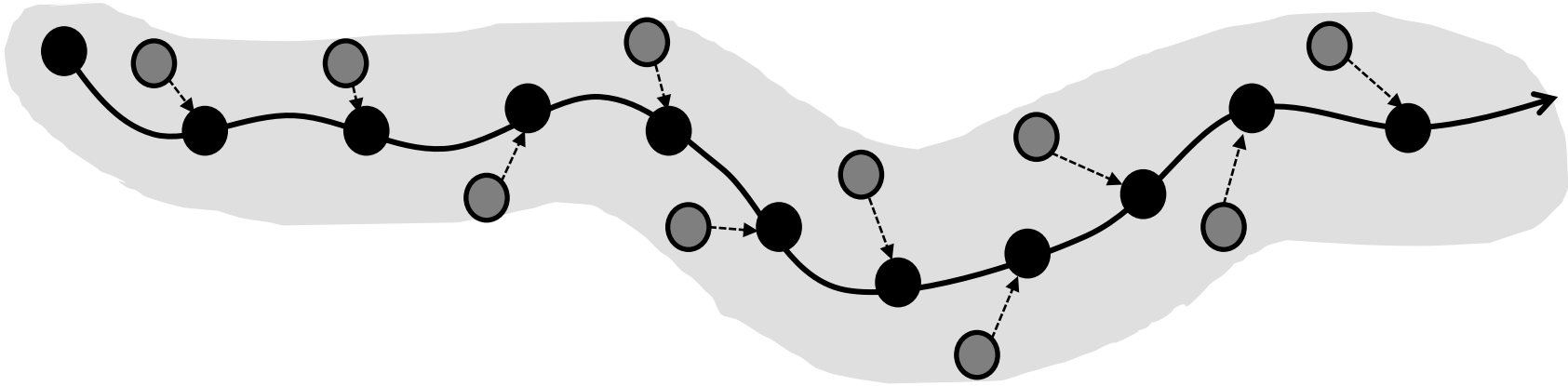
Easy with known dynamics

Intuition: find labels to bring OOD states back in distribution

But models are unknown! ☹

CCIL: Continuity-based Data Augmentation for Corrective Imitation Learning, Ke et al '24
Data Efficient Behavior Cloning for Fine Manipulation via Continuity-based Corrective Labels, Deshpande et al '24

Generating Corrective Labels with **Learned** Dynamics



Ok models are unknown, let's learn them!

$$\min_{\hat{f}} \mathbb{E}_{(s_t,a_t,s_{t+1})\sim\mathcal{D}} \left[\|\hat{f}(s_t,a_t) - s_{t+1}\|_2 \right]$$

But learned dynamics $\hat{f}_\phi$ are not globally accurate?

$$\|s^*_{t+1} - \hat{f}_\phi(s_t,a_t)\| \leq \epsilon$$

Under approximately Lipschitz smooth models, trust models around training data

Find states (s_t), actions (a_t) that lead back to optimal states under ~~true~~ learned dynamics, **where learned dynamics can be trusted**

$$\min_{s_t,a_t} \|s^*_{t+1} - \hat{f}_\phi(s_t,a_t)\| \leq \epsilon \longleftarrow \text{Corrective label}$$

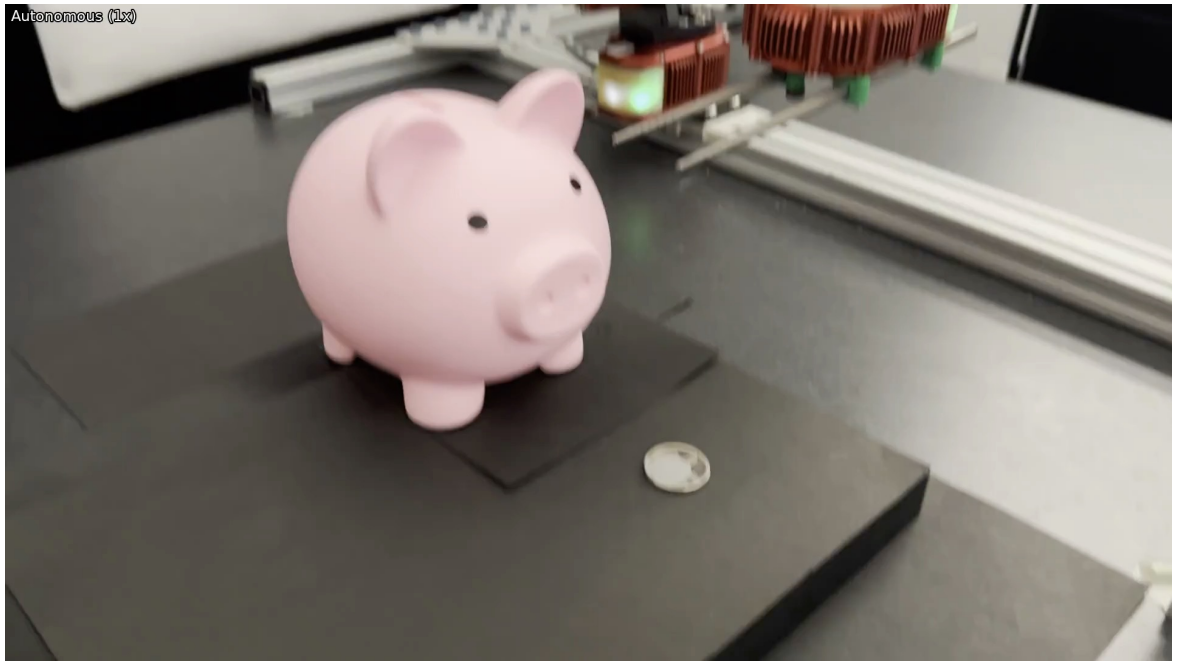
$$\text{s.t} \quad \|s^*_t - s_t\| \leq \epsilon_1, \|a^*_t - a_t\| \leq \epsilon_2 \longleftarrow \text{Close to data}$$

CCIL: Continuity-based Data Augmentation for Corrective Imitation Learning, Ke et al '24
Data Efficient Behavior Cloning for Fine Manipulation via Continuity-based Corrective Labels, Deshpande et al '24

How well does generating corrective labels work?

With corrective labels

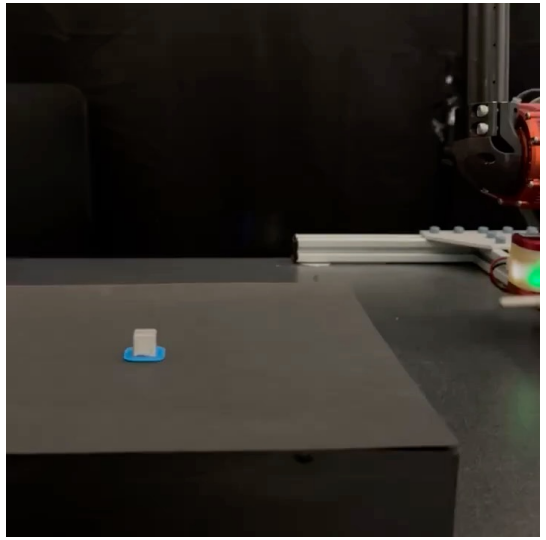
Without corrective labels

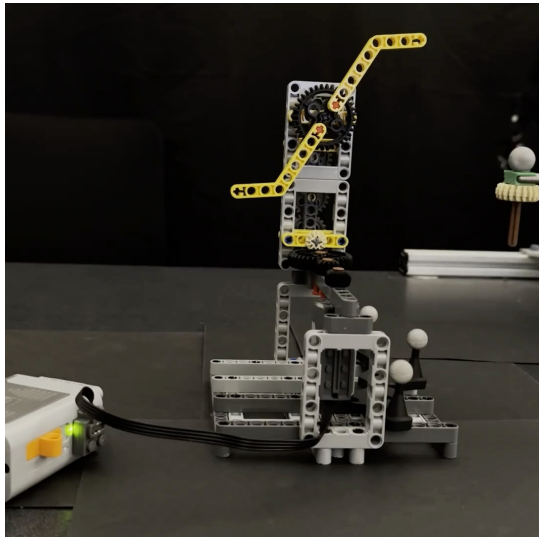


CCIL: Continuity-based Data Augmentation for Corrective Imitation Learning, Ke et al '24
Data Efficient Behavior Cloning for Fine Manipulation via Continuity-based Corrective Labels, Deshpande et al '24

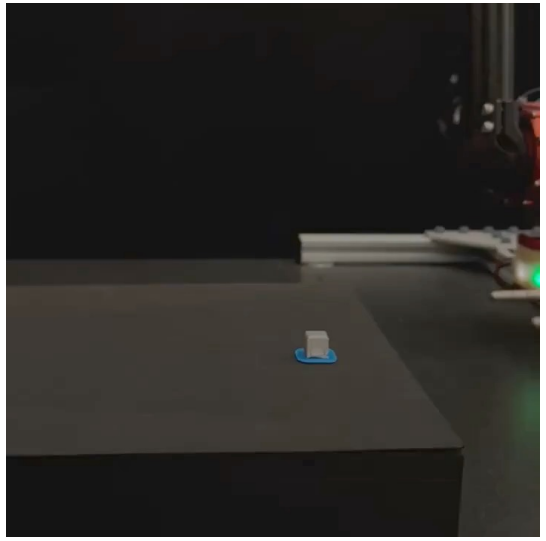
How well does generating corrective labels work?

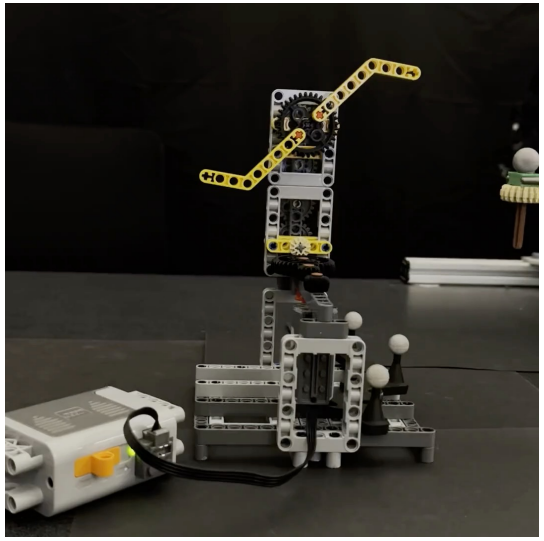
With corrective labels

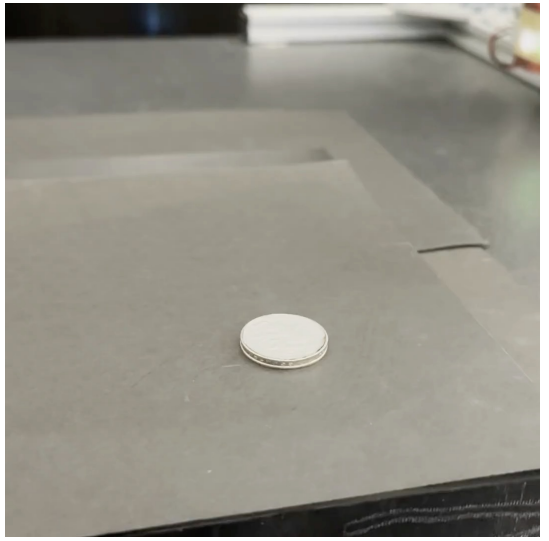




Without corrective labels







Lecture outline

Recap: MDP formalism + why should we care?

Imitation learning: preliminaries and behavior cloning

Multimodality and Underfitting in Imitation

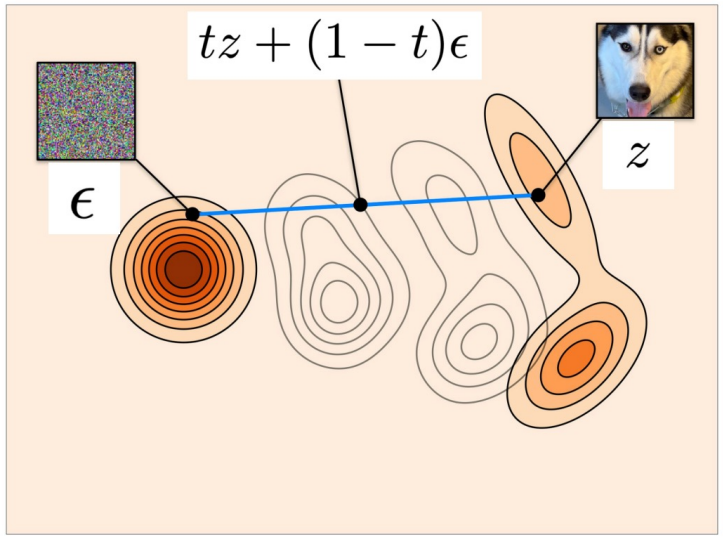
Compounding Error in Imitation

Modern Takes on Imitation Learning

But BC works unreasonably well!

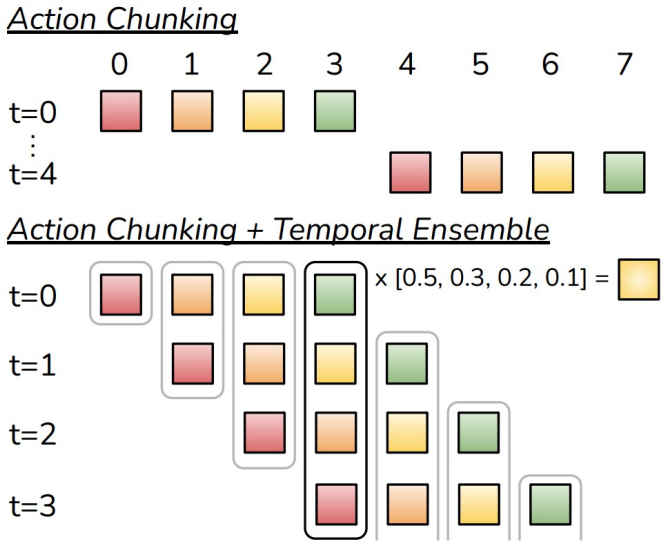
Is this by accident?

Generative Models



Flow matching, diffusion, etc

Action chunking

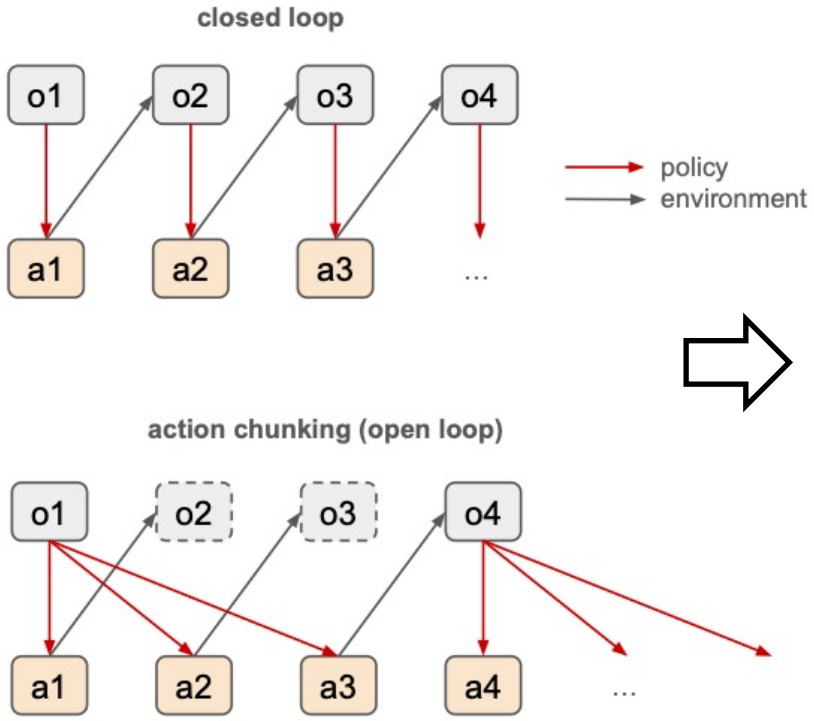


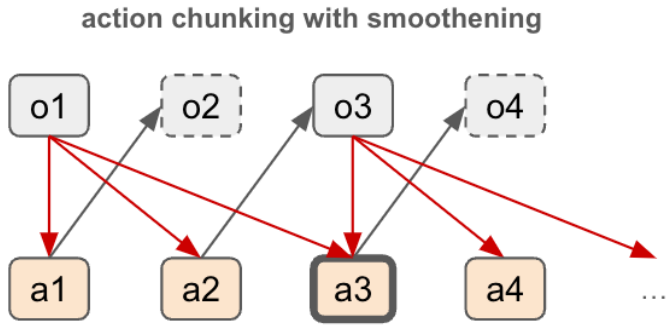
Short-horizon open-loop prediction

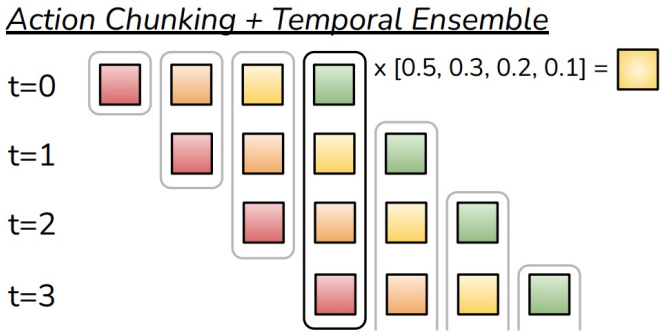
Both are non-trivially important!

Action Chunking

Action chunking = predicting small “chunk” of open loop actions



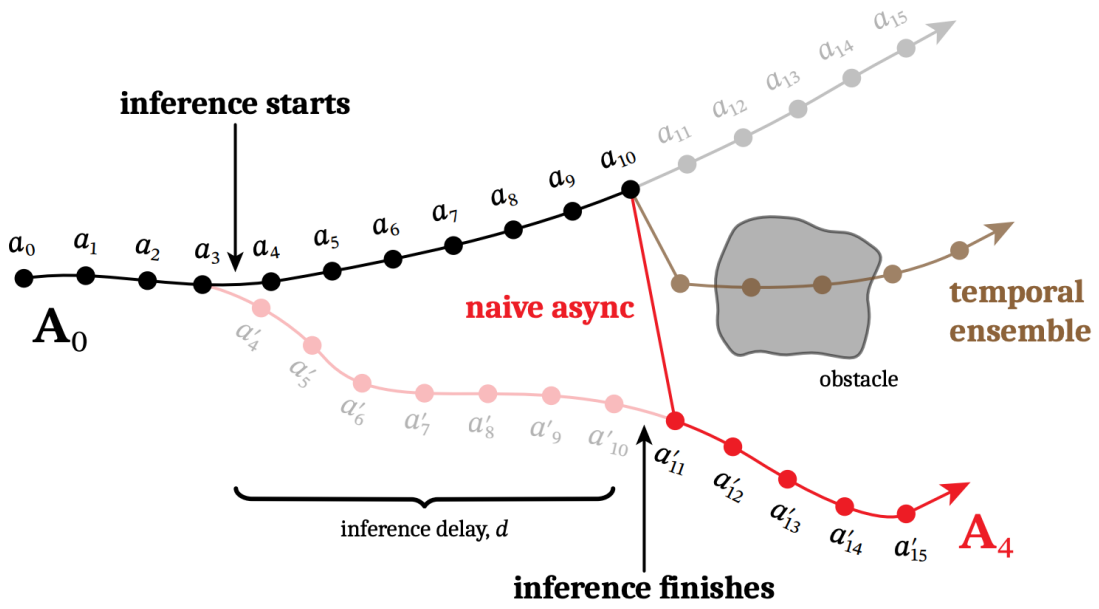




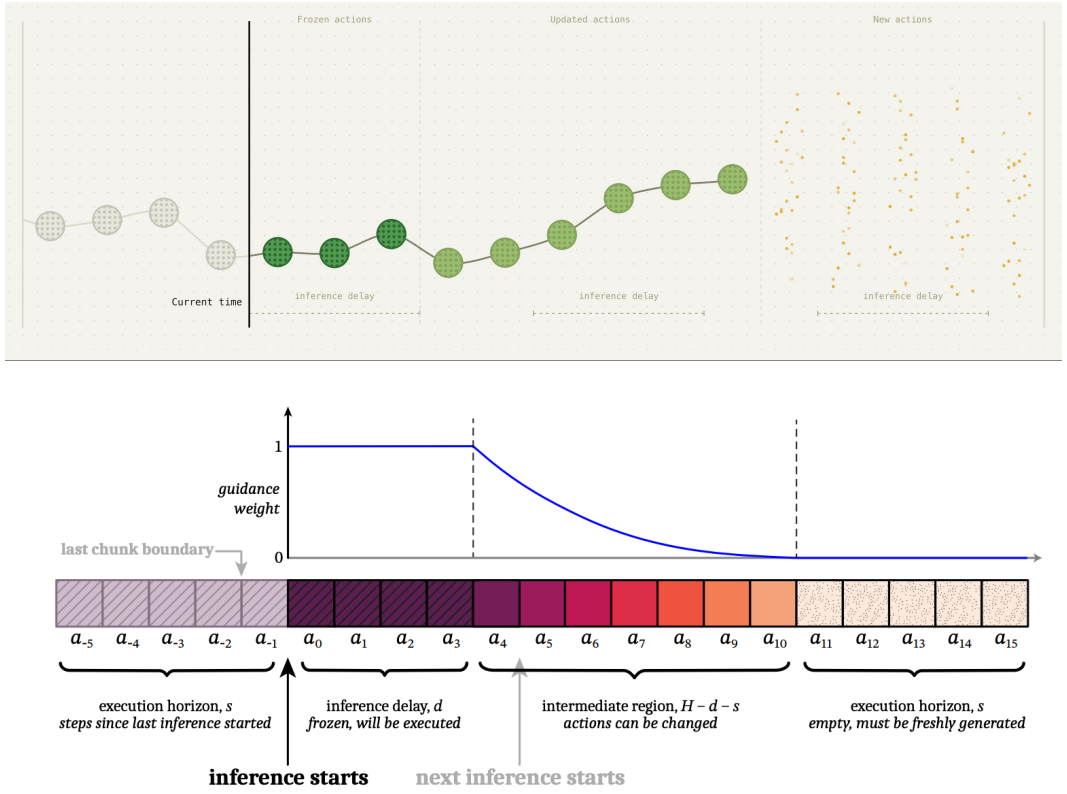
https://www.haonanyu.blog/post/action_chunking/
https://arxiv.org/abs/2304.13705

“Real-Time” Action Chunking

Real challenge when trying to operate robotic policies quickly



Leads to jerky/suboptimal motion



Freeze k steps and inpaint the rest

Real-Time Execution of Action Chunking Flow Policies, Black et al, ‘25

Generative Modeling – why does it help?

Do generative models help because human data is multimodal?

Much Ado About Noising: Dispelling the Myths of Generative Robotic Control

Chaoyi Pan[a,$] Giri Anantharaman[a] Nai-Chieh Huang[a] Claire Jin[a] Daniel Pfrommer[b]
Chenyang Yuan[c] Frank Permenter[c] Guannan Qu[a,†] Nicholas Boffi[a,†] Guanya Shi[a,†]
Max Simchowitz[a,†]

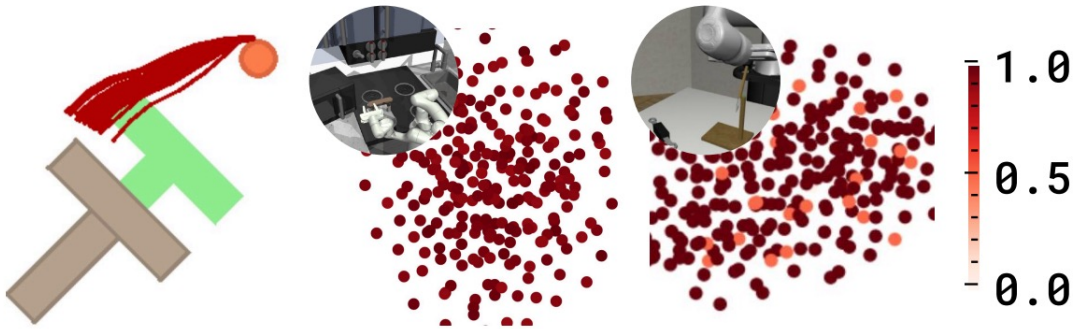
Shows that the benefits of generative models is not only in stochastic settings

Shows that the benefits come from stochasticity injection + iteration

Generative Modeling – why does it help?

Do generative models help because human data is multimodal?

Not much multimodality in existing policies



(a)Push T (b)Kitchen (c)Tool Hang

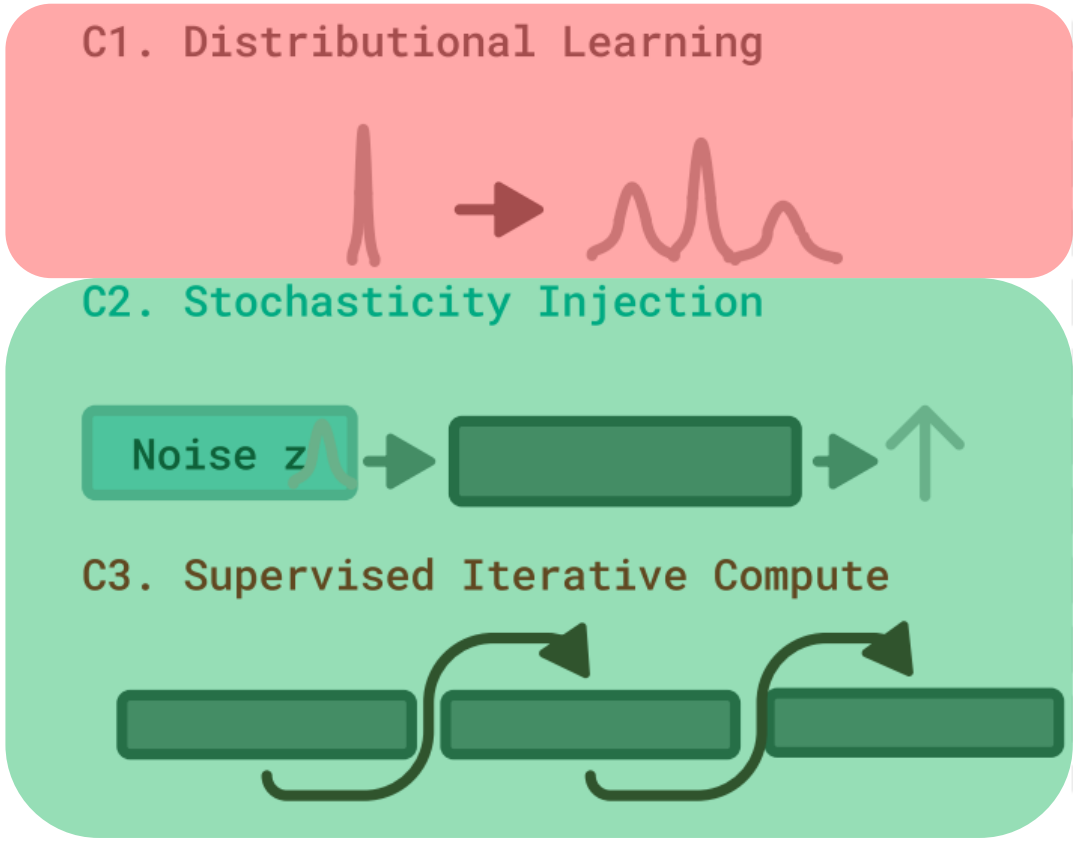
Task	$z \equiv 0$	$N(0, I)$	Mean z
Push-T	0.97	0.97	0.95
Kitchen	0.99	0.99	0.97
Tool-Hang	0.78	0.80	0.76

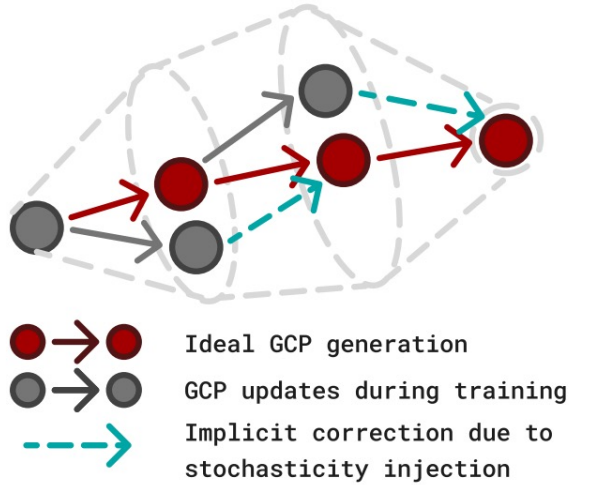
Better even with deterministic experts

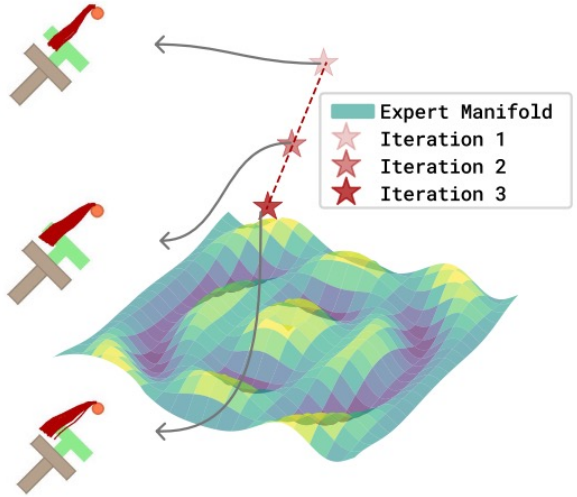
Dataset	Flow	Reg.
Original	0.78	0.58
Deterministic	0.72	0.64

Ok – so why do these flow models help?

Generative Modeling – why does it help?

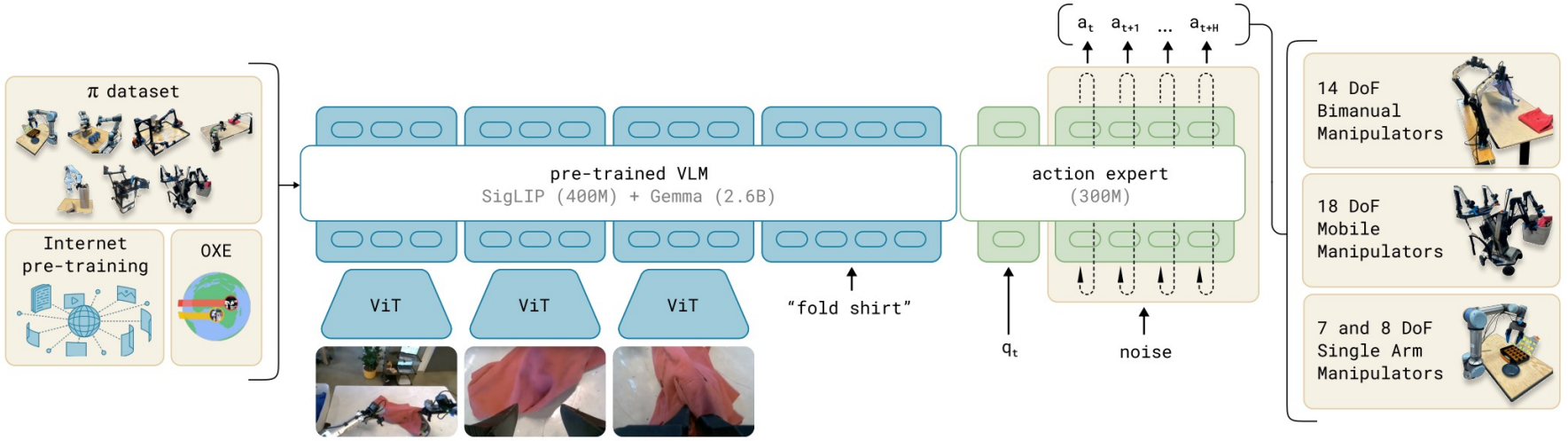




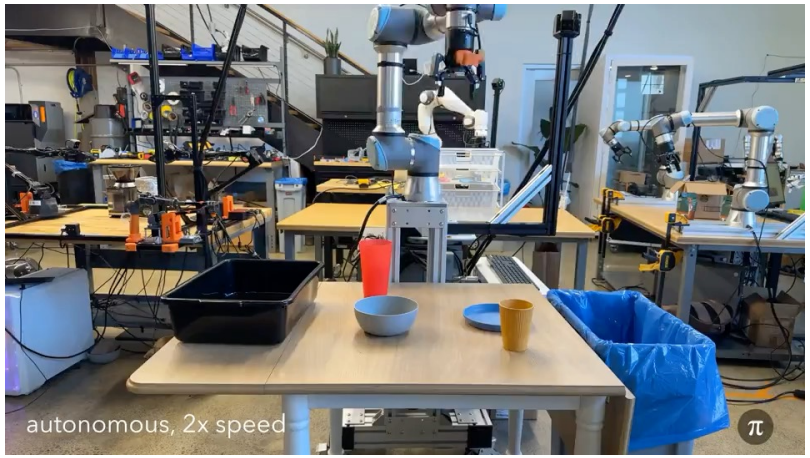


Train with stochasticity injected to "correct for errors" in generation

How is this being actualized today?



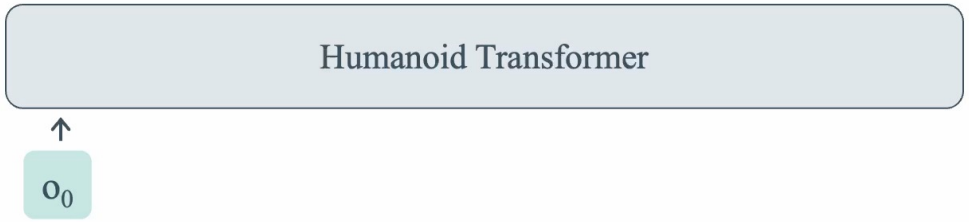
Vision-language-action policies



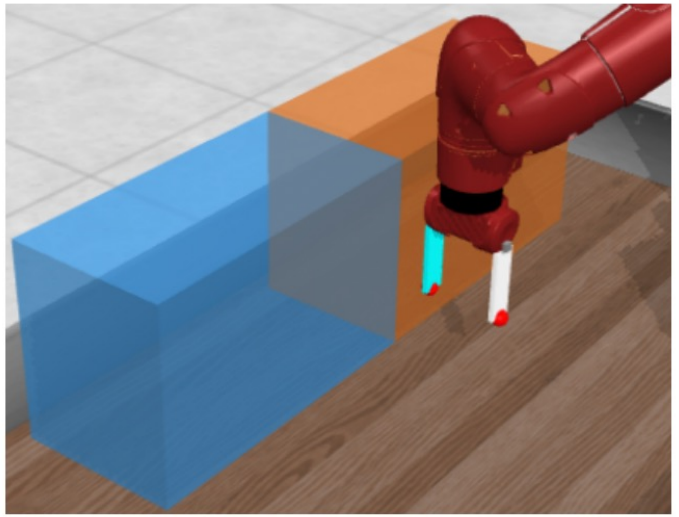
So does this solve all the issues in imitation?

Frontiers in Imitation Learning

Non-Markovian Demonstrators



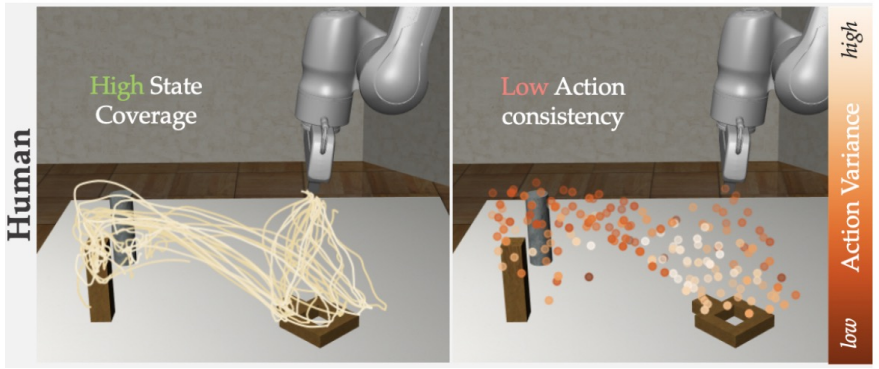
Characterizing generalization



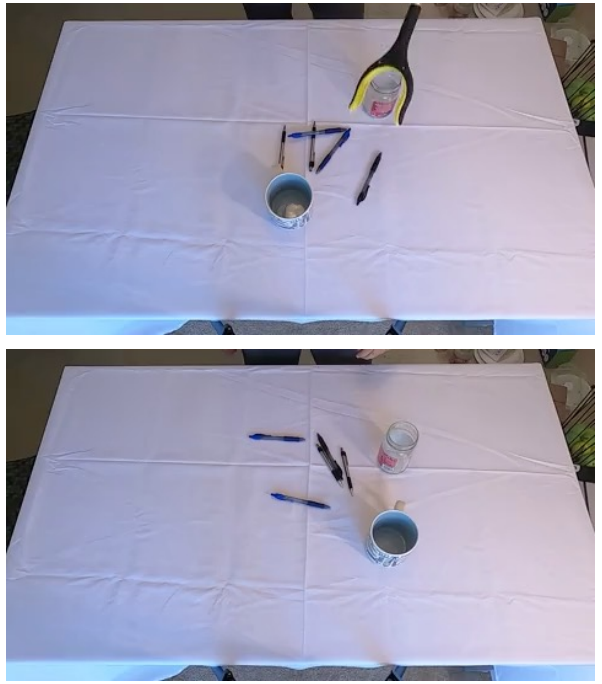
Action-Free Data

Frontiers in Imitation Learning

Data Curation and Quality



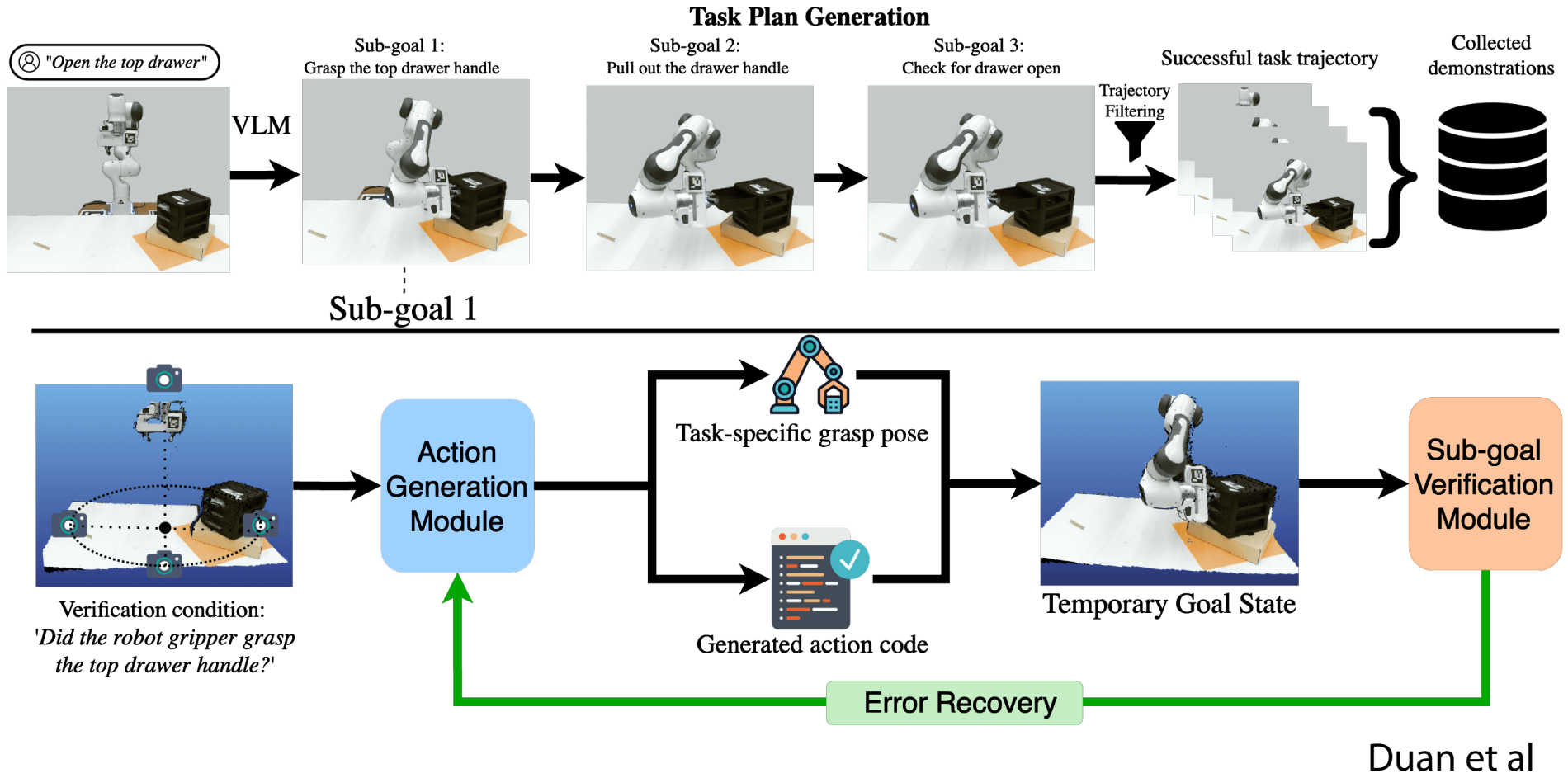
Embodiment Shift



Teleoperation Interfaces

Frontiers in Imitation Learning

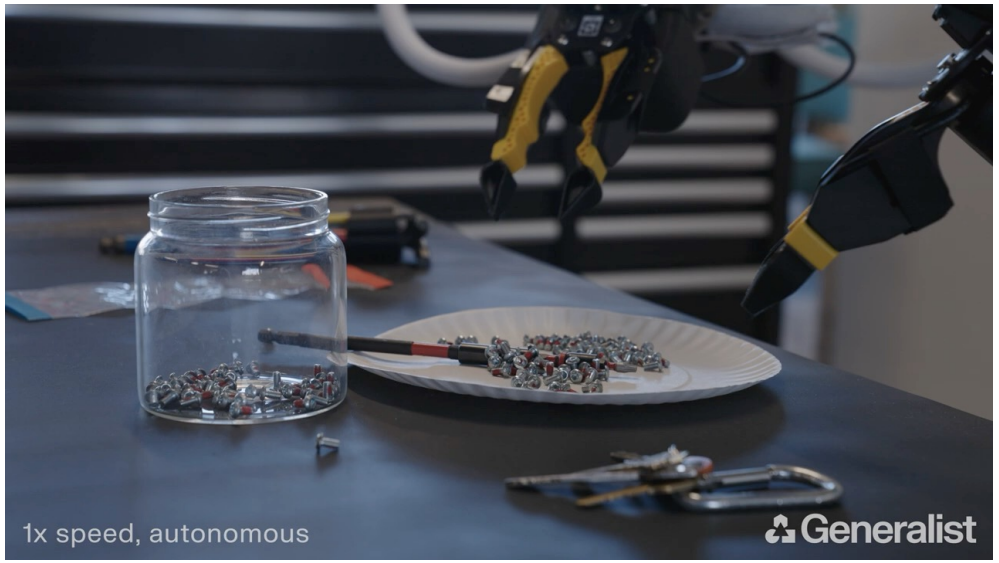
Learning how to retry and improve

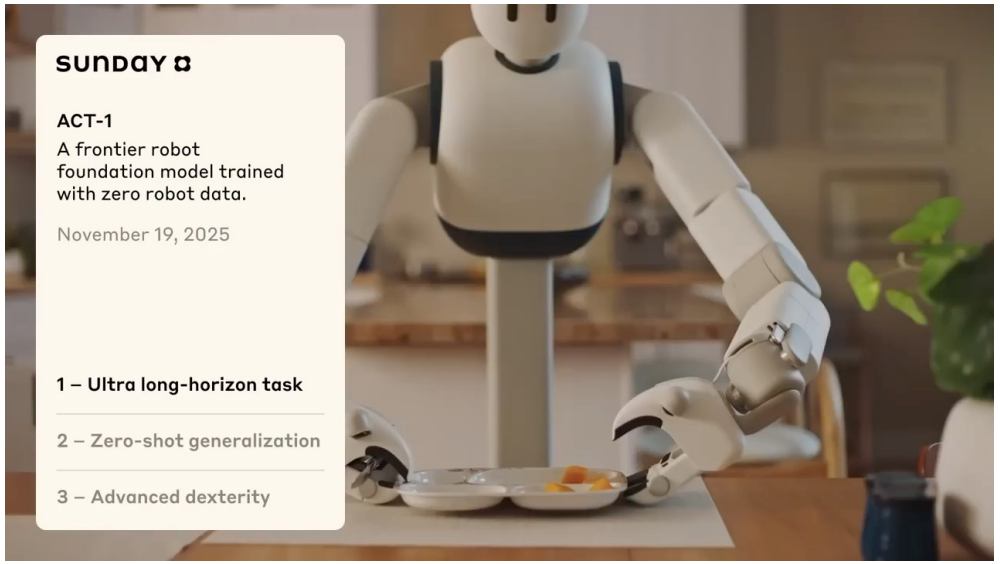


Duan et al

Some cool imitation videos

Generalist/Sunday Robots



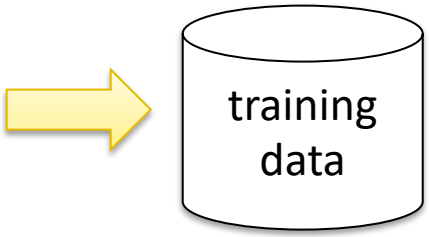


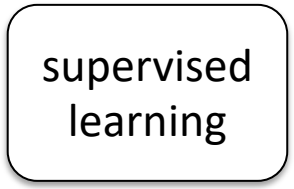
Sensorized people improve data throughput

Inheriting capabilities of vision-language models

Perspectives on Imitation

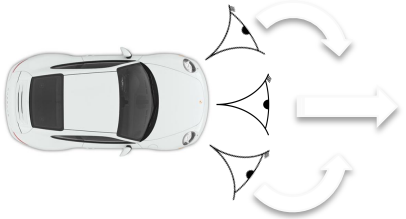
$\mathbf{o}_t$
$\mathbf{a}_t$



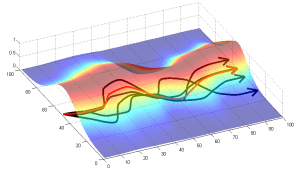


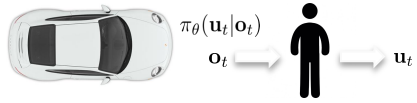
$\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$

- Pros:
  - Easy to use, no additional infra
  - Can sometimes be unreasonably effective



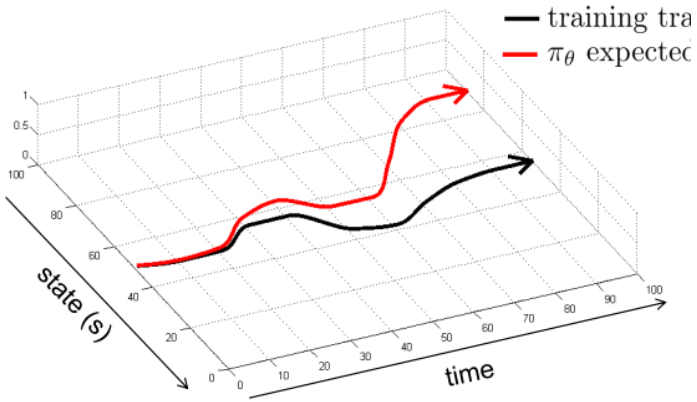
- Cons:
  - Challenges of compounding error, multimodality
  - Doesn't really generalize
  - Very expensive in terms of data collection!





Let's do a little analysis: Start with BC

Behavior cloning has quadratically compounding error



Assumption:

$$\mathbb{E}_{s\sim d_t^{\pi^*}}\left[\mathbf{1}\{\hat{\pi}(s)\neq\pi^*(s)\}\right]\leq\epsilon.$$

(bounded SL error on training distribution)

To prove:

$$\sum_{t=1}^{H}\mathbb{E}_{s\sim d_t^{\hat{\pi}}}\left[\mathbf{1}\{\hat{\pi}(s)\neq\pi^*(s)\}\right]\leq\epsilon\frac{H(H+1)}{2}=O(\epsilon H^2).$$

(quadratically bad error on rollouts)

How well does BC do?: Proof

Definitions

$$\ell(s) := \mathbf{1}\{\hat{\pi}(s) \neq \pi^*(s)\}.$$

$$\sum_{t=1}^{H} \mathbb{E}_{s \sim d_t^{\hat{\pi}}}\left[\mathbf{1}\{\hat{\pi}(s) \neq \pi^*(s)\}\right] = \sum_{t=1}^{H} \mathbb{E}_{s \sim d_t^{\hat{\pi}}}[\ell(s)].$$

$$E_t := \{\text{no mistakes were made in the first } t-1 \text{ steps}\}.$$

Proof Sketch

1) Define per step losses conditional on E_t, E_t^c
2) Bound $p(E_t^c)$
3) Bound per step loss
4) Bound cumulative loss

How well does BC do?: Proof

$$\sum_{t=1}^{H} \mathbb{E}_{s \sim d_t^{\hat{\pi}}} \left[\mathbf{1}\{\hat{\pi}(s) \neq \pi^*(s)\} \right] = \sum_{t=1}^{H} \mathbb{E}_{s \sim d_t^{\hat{\pi}}} [\ell(s)].$$

Bounding losses per event

$$\mathbb{E}[\ell(s_t) \mid E_t] = \mathbb{E}_{s \sim d_t^{\pi^*}} [\ell(s)] \leq \epsilon.$$

$$\mathbb{E}[\ell(s_t) \mid E_t^c] \leq 1.$$

Bounding per step losses overall

$$\mathbb{E}_{s \sim d_t^{\hat{\pi}}} [\ell(s)] = \mathbb{E}[\ell(s_t) \mid E_t] \Pr(E_t) + \mathbb{E}[\ell(s_t) \mid E_t^c] \Pr(E_t^c).$$

$$\mathbb{E}_{s \sim d_t^{\hat{\pi}}} [\ell(s)] \leq \epsilon \Pr(E_t) + \Pr(E_t^c) \leq \epsilon + \Pr(E_t^c).$$

How well does BC do?: Proof

Known

$$\mathbb{E}_{s\sim d_t^{\hat{\pi}}}[\ell(s)] \leq \epsilon \Pr(E_t) + \Pr(E_t^c) \leq \epsilon + \Pr(E_t^c).$$

Bounding E_t^c

$$\Pr(E_t^c) \leq \sum_{i=1}^{t-1} \Pr(\text{mistake at time } i \text{ and no earlier mistake}).$$

$$\Pr(E_t^c) \leq (t-1)\epsilon.$$

Bounding cumulative loss

$$\mathbb{E}_{s\sim d_t^{\hat{\pi}}}[\ell(s)] \leq \epsilon + (t-1)\epsilon = t\epsilon.$$

$$\sum_{t=1}^{H} \mathbb{E}_{s\sim d_t^{\hat{\pi}}}[\ell(s)] \leq \sum_{t=1}^{H} t\epsilon = \epsilon \frac{H(H+1)}{2} = O(\epsilon H^2).$$

Why is Dagger good?

Too long to prove in class - but here's the intuition

$$J(\pi) \leq J(\pi^\star) + O(T^2\epsilon). \quad \Longrightarrow \quad J(\hat{\pi}) - J(\pi^\star) \leq uT(\epsilon_N + \gamma_N).$$

Step 1: Express performance difference in terms of $Q_t^{\pi'}(s,a)$

$$J(\pi) - J(\pi') = \sum_{t=1}^{T} \mathbb{E}_{s\sim d_\pi^t}\left[Q_t^{\pi'}(s,\pi(s)) - Q_t^{\pi'}(s,\pi'(s))\right].$$

Step 2: Use bounded Q difference assumption to get a linear bound

$$Q_t^{\pi^\star}(s,\pi(s)) - Q_t^{\pi^\star}(s,\pi^\star(s)) \leq u\,\ell(s,\pi). \quad \Longrightarrow \quad J(\pi) - J(\pi^\star) \leq uT\,\epsilon_{\text{on}}(\pi).$$

Step 3: Use no-regret online learning to show that on-policy error is small

$$\frac{1}{N}\sum_{i=1}^{N} L_i(\pi_i) \leq \min_{\pi\in\Pi} \frac{1}{N}\sum_{i=1}^{N} L_i(\pi) + \gamma_N, \quad \Longrightarrow \quad J(\hat{\pi}) - J(\pi^\star) \leq uT(\epsilon_N + \gamma_N).$$

Fin.