PPO & Importance Sampling

Lucas Janson

CS/Stat 184(0): Introduction to Reinforcement Learning

Fall 2024

Today

- Feedback from last lecture
- Recap
- Importance Sampling (for PPO)
- PG review
- Exploration?

Feedback from feedback forms

Feedback from feedback forms

1. Thank you to everyone who filled out the forms!

Today

- Feedback from last lecture
- Recap
- Importance Sampling (for PPO)
- PG review
- Exploration?

PG with a Learned Baseline:

Let $g'(\theta, \tau, b()) := \sum_{h=0}^{H-1} \nabla_\theta \ln \pi_\theta(a_h | s_h) \left(R_h(\tau) - b(s_h, h) \right)$

1. Initialize θ^0, parameters: $\eta^1, \eta^2, \ldots$
2. For $k = 0, \ldots$:
    1. Supervised Learning: Using N trajectories sampled under π_{θ^k}, estimate a baseline $\widetilde{b}$
       $\widetilde{b}(s, h) \approx V_h^{\theta^k}(s)$
    2. Obtain a trajectory $\tau \sim \rho_{\theta^k}$
       Compute $g'(\theta^k, \tau, \widetilde{b}())$
    3. Update: $\theta^{k+1} = \theta^k + \eta^k g'(\theta^k, \tau, \widetilde{b}())$

Note that regardless of our choice of $\widetilde{b}$, we still get unbiased gradient estimates.

The Performance Difference Lemma (PDL)

- Let $\rho_{\widetilde{\pi},s}$ be the distribution of trajectories from starting state s acting under $\widetilde{\pi}$. (we are making the starting distribution explicit now).
- For any two policies π and $\widetilde{\pi}$ and any state s,

$$V^{\widetilde{\pi}}(s) - V^{\pi}(s) = \mathbb{E}_{\tau \sim \rho_{\widetilde{\pi},s}} \left[\sum_{h=0}^{H-1} A^{\pi}(s_h, a_h, h) \right]$$

Comments:

- Helps us think about error analysis, instabilities of fitted PI, and sub-optimality.
- Helps to understand algorithm design (TRPO, NPG, PPO)
- This also motivates the use of “local” methods (e.g. policy gradient descent)

Trust Region Policy Optimization (TRPO)

1. Initialize θ^0
2. For $k = 0,\ldots,K$:

   try to approximately solve:

$$\theta^{k+1} = \arg\max_{\theta} \mathbb{E}_{s_0,\ldots,s_{H-1}\sim\rho_{\pi_{\theta^k}}} \left[\sum_{h=0}^{H-1} \mathbb{E}_{a_h\sim\pi_\theta(\cdot|s_h)} \left[A^{\pi_{\theta^k}}(s_h, a_h, h)\right] \right]$$

$$\text{s.t. } KL\left(\rho_{\pi_{\theta^k}} | \rho_{\pi_\theta}\right) \leq \delta$$

3. Return π_{θ^K}

NPG has a closed form update!

1. Initialize θ^0
2. For $k = 0, \ldots, K$:

$$\theta^{k+1} = \arg\max_{\theta} \nabla_\theta J(\theta^k)^\top (\theta - \theta^k)$$

$$\text{s.t. } (\theta - \theta^k)^\top F_{\theta^k} (\theta - \theta^k) \le \delta$$

3. Return π_{θ^K}

Linear objective and quadratic convex constraint: we can solve it optimally!

Indeed this gives us:

$$\theta^{k+1} = \theta^k + \eta F_{\theta^k}^{-1} \nabla_\theta J(\theta^k)$$

Where $\eta = \sqrt{\dfrac{\delta}{\nabla_\theta J(\theta^k)^\top F_{\theta^k}^{-1} \nabla_\theta J(\theta^k)}}$

Proximal Policy Optimization (PPO)

1. Initialize θ^0
2. For $k = 0,\ldots,K$:
   use SGD to approximately solve:
   $\theta^{k+1} = \arg\max_{\theta} \ell^k(\theta)$
   where:
   $$\ell^k(\theta) := \mathbb{E}_{s_0,\ldots,s_{H-1}\sim\rho_{\pi_{\theta^k}}}\left[\sum_{h=0}^{H-1}\mathbb{E}_{a_h\sim\pi_\theta(\cdot|s_h)}\left[A^{\pi_{\theta^k}}(s_h, a_h, h)\right]\right] - \lambda\mathbb{E}_{\tau\sim\rho_{\pi_{\theta^k}}}\left[\sum_{h=0}^{H-1}\ln\frac{1}{\pi_\theta(a_h\,|\,s_h)}\right]$$
3. Return π_{θ^K}

How do we estimate this objective?

Today

- Feedback from last lecture
- Recap
- Importance Sampling (for PPO)
- PG review
- Exploration?

SGD and Importance Sampling

- Recall that SGD requires an unbiased estimate of the objective function's gradient
- This was easy when the objective function was an expectation, and the only θ-dependence appears inside the expectation
  - This was true for supervised learning / ERM
  - Not true for RL, and was part of why we needed likelihood ratio method in REINFORCE
- When not true (as in PPO), we want to make it so, if possible
- Enter: importance sampling
  - rewrites expectations by changing the distribution the expectation is over
  - we will use this to move that distribution's θ-dependence inside the expectation
- **Key point**: once all θ-dependence inside objective's expectation,
  - Can estimate objective unbiasedly via sample average
  - Can estimate objective's gradient unbiasedly via gradient of sample average

Importance Sampling

Importance Sampling

- Suppose we seek to estimate $\mathbb{E}_{x \sim \tilde{p}}[f(x)]$.

Importance Sampling

- Suppose we seek to estimate $\mathbb{E}_{x\sim\widetilde{p}}[f(x)]$.
- Assume: we have an (i.i.d.) dataset $x_1, \dots x_N$, where $x_i \sim p$, where p is known, and
    - f and $\widetilde{p}$ are known.
    - we are not able to collect values of $f(x)$ for $x \sim \widetilde{p}$.
      (e.g. we have already collected our data from some costly experiment).

Importance Sampling

- Suppose we seek to estimate $\mathbb{E}_{x \sim \widetilde{p}}[f(x)]$.
- Assume: we have an (i.i.d.) dataset $x_1, \ldots x_N$, where $x_i \sim p$, where p is known, and
    - f and $\widetilde{p}$ are known.
    - we are not able to collect values of $f(x)$ for $x \sim \widetilde{p}$.
      (e.g. we have already collected our data from some costly experiment).
- Note: $\mathbb{E}_{x \sim \widetilde{p}}\left[f(x)\right] =$

Importance Sampling

- Suppose we seek to estimate $\mathbb{E}_{x \sim \widetilde{p}}[f(x)]$.
- Assume: we have an (i.i.d.) dataset $x_1, \ldots x_N$, where $x_i \sim p$, where p is known, and
  - f and $\widetilde{p}$ are known.
  - we are not able to collect values of $f(x)$ for $x \sim \widetilde{p}$.
    (e.g. we have already collected our data from some costly experiment).
- Note: $\mathbb{E}_{x \sim \widetilde{p}}\left[f(x)\right] = \mathbb{E}_{x \sim p}\left[\frac{\widetilde{p}(x)}{p(x)} f(x)\right]$

Importance Sampling

- Suppose we seek to estimate $\mathbb{E}_{x \sim \tilde{p}}[f(x)]$.
- Assume: we have an (i.i.d.) dataset $x_1, \ldots x_N$, where $x_i \sim p$, where p is known, and
  - f and $\tilde{p}$ are known.
  - we are not able to collect values of $f(x)$ for $x \sim \tilde{p}$.
    (e.g. we have already collected our data from some costly experiment).
- Note: $\mathbb{E}_{x \sim \tilde{p}}\left[f(x)\right] = \mathbb{E}_{x \sim p}\left[\frac{\tilde{p}(x)}{p(x)} f(x)\right]$
- So an unbiased estimate of $\mathbb{E}_{x \sim \tilde{p}}[f(x)]$ is given by $\frac{1}{N}\sum_{i=1}^{N} \frac{\tilde{p}(x_i)}{p(x_i)} f(x_i)$

Importance Sampling

- Suppose we seek to estimate $\mathbb{E}_{x \sim \widetilde{p}}[f(x)]$.
- Assume: we have an (i.i.d.) dataset $x_1, \ldots x_N$, where $x_i \sim p$, where p is known, and
  - f and $\widetilde{p}$ are known.
  - we are not able to collect values of $f(x)$ for $x \sim \widetilde{p}$.
    (e.g. we have already collected our data from some costly experiment).
- Note: $\mathbb{E}_{x \sim \widetilde{p}}\left[f(x)\right] = \mathbb{E}_{x \sim p}\left[\frac{\widetilde{p}(x)}{p(x)} f(x)\right]$
- So an unbiased estimate of $\mathbb{E}_{x \sim \widetilde{p}}[f(x)]$ is given by $\frac{1}{N}\sum_{i=1}^{N}\frac{\widetilde{p}(x_i)}{p(x_i)} f(x_i)$
- Terminology:
  - $\widetilde{p}(x)$ is the target distribution
  - $p(x)$ is the proposal distribution
  - $\widetilde{p}(x)/p(x)$ is the likelihood ratio or importance weight

Importance Sampling

- Suppose we seek to estimate $\mathbb{E}_{x \sim \widetilde{p}}[f(x)]$.
- Assume: we have an (i.i.d.) dataset $x_1, \ldots x_N$, where $x_i \sim p$, where p is known, and
  - f and $\widetilde{p}$ are known.
  - we are not able to collect values of $f(x)$ for $x \sim \widetilde{p}$.
    (e.g. we have already collected our data from some costly experiment).
- Note: $\mathbb{E}_{x \sim \widetilde{p}}\left[f(x)\right] = \mathbb{E}_{x \sim p}\left[\frac{\widetilde{p}(x)}{p(x)} f(x)\right]$
- So an unbiased estimate of $\mathbb{E}_{x \sim \widetilde{p}}[f(x)]$ is given by $\frac{1}{N}\sum_{i=1}^{N} \frac{\widetilde{p}(x_i)}{p(x_i)} f(x_i)$
- Terminology:
  - $\widetilde{p}(x)$ is the target distribution
  - $p(x)$ is the proposal distribution
  - $\widetilde{p}(x)/p(x)$ is the likelihood ratio or importance weight
- What about the variance of this estimator?

Back to Estimating $\ell^k(\theta)$

Back to Estimating $\ell^k(\theta)$

- To estimate

$$\ell^k(\theta) := \mathbb{E}_{s_0,\ldots,s_{H-1}\sim\rho_{\pi_{\theta^k}}}\left[\sum_{h=0}^{H-1}\mathbb{E}_{a_h\sim\pi_\theta(\cdot|s_h)}\left[A^{\pi_{\theta^k}}(s_h,a_h,h)\right]\right] - \lambda\mathbb{E}_{\tau\sim\rho_{\pi_{\theta^k}}}\left[\sum_{h=0}^{H-1}\ln\frac{1}{\pi_\theta(a_h\,|\,s_h)}\right]$$

Back to Estimating $\ell^k(\theta)$

- To estimate

$$\ell^k(\theta) := \mathbb{E}_{s_0,\ldots,s_{H-1}\sim\rho_{\pi_{\theta^k}}}\left[\sum_{h=0}^{H-1}\mathbb{E}_{a_h\sim\pi_\theta(\cdot|s_h)}\left[A^{\pi_{\theta^k}}(s_h,a_h,h)\right]\right] - \lambda\mathbb{E}_{\tau\sim\rho_{\pi_{\theta^k}}}\left[\sum_{h=0}^{H-1}\ln\frac{1}{\pi_\theta(a_h\,|\,s_h)}\right]$$

- we will use importance sampling:

$$= \mathbb{E}_{s_0,\ldots,s_{H-1}\sim\rho_{\pi_{\theta^k}}}\left[\sum_{h=0}^{H-1}\mathbb{E}_{a_h\sim\pi_{\theta^k}(\cdot|s_h)}\left[\frac{\pi_\theta(a_h\,|\,s_h)}{\pi_{\theta^k}(a_h\,|\,s_h)}A^{\pi_{\theta^k}}(s_h,a_h,h)\right]\right] - \lambda\mathbb{E}_{\tau\sim\rho_{\pi_{\theta^k}}}\left[\sum_{h=0}^{H-1}\ln\frac{1}{\pi_\theta(a_h\,|\,s_h)}\right]$$

Back to Estimating $\ell^k(\theta)$

- To estimate

$$\ell^k(\theta) := \mathbb{E}_{s_0,\ldots,s_{H-1}\sim\rho_{\pi_{\theta^k}}}\left[\sum_{h=0}^{H-1}\mathbb{E}_{a_h\sim\pi_\theta(\cdot|s_h)}\left[A^{\pi_{\theta^k}}(s_h,a_h,h)\right]\right] - \lambda\mathbb{E}_{\tau\sim\rho_{\pi_{\theta^k}}}\left[\sum_{h=0}^{H-1}\ln\frac{1}{\pi_\theta(a_h|s_h)}\right]$$

- we will use importance sampling:

$$= \mathbb{E}_{s_0,\ldots,s_{H-1}\sim\rho_{\pi_{\theta^k}}}\left[\sum_{h=0}^{H-1}\mathbb{E}_{a_h\sim\pi_{\theta^k}(\cdot|s_h)}\left[\frac{\pi_\theta(a_h|s_h)}{\pi_{\theta^k}(a_h|s_h)}A^{\pi_{\theta^k}}(s_h,a_h,h)\right]\right] - \lambda\mathbb{E}_{\tau\sim\rho_{\pi_{\theta^k}}}\left[\sum_{h=0}^{H-1}\ln\frac{1}{\pi_\theta(a_h|s_h)}\right]$$

$$= \mathbb{E}_{\tau\sim\rho_{\pi_{\theta^k}}}\left[\sum_{h=0}^{H-1}\left(\frac{\pi_\theta(a_h|s_h)}{\pi_{\theta^k}(a_h|s_h)}A^{\pi_{\theta^k}}(s_h,a_h,h) - \lambda\ln\frac{1}{\pi_\theta(a_h|s_h)}\right)\right]$$

Estimating $\ell^k(\theta)$ and its gradient

Estimating $\ell^k(\theta)$ and its gradient

1. Using N trajectories sampled under $\rho_{\pi_{\theta^k}}$ to learn a $\widetilde{b}_h$

   $\widetilde{b}(s, h) \approx V_h^{\pi_{\theta^k}}(s)$

Estimating $\ell^k(\theta)$ and its gradient

1. Using N trajectories sampled under $\rho_{\pi_{\theta^k}}$ to learn a $\widetilde{b}_h$

   $\widetilde{b}(s,h) \approx V_h^{\pi_{\theta^k}}(s)$

2. Obtain M NEW trajectories $\tau_1, \ldots \tau_M \sim \rho_{\pi_{\theta^k}}$

   Set $\widehat{\ell}^k(\theta) = \dfrac{1}{M} \sum_{m=1}^{M} \sum_{h=0}^{H-1} \left(\dfrac{\pi_\theta(a_h^m \mid s_h^m)}{\pi_{\theta^k}(a_h^m \mid s_h^m)} \left(R_h(\tau_m) - \widetilde{b}(s_h^m, h) \right) - \lambda \ln \dfrac{1}{\pi_\theta(a_h^m \mid s_h^m)} \right)$

   for SGD, use gradient: $g(\theta) := \nabla_\theta \widehat{\ell}^k(\theta)$

Estimating $\ell^k(\theta)$ and its gradient

1. Using N trajectories sampled under $\rho_{\pi_{\theta^k}}$ to learn a $\widetilde{b}_h$

   $\widetilde{b}(s,h) \approx V_h^{\pi_{\theta^k}}(s)$

2. Obtain M NEW trajectories $\tau_1, \ldots \tau_M \sim \rho_{\pi_{\theta^k}}$

   Set $\widehat{\ell}^k(\theta) = \frac{1}{M}\sum_{m=1}^{M}\sum_{h=0}^{H-1}\left(\frac{\pi_\theta(a_h^m \mid s_h^m)}{\pi_{\theta^k}(a_h^m \mid s_h^m)}\left(R_h(\tau_m) - \widetilde{b}(s_h^m, h)\right) - \lambda \ln \frac{1}{\pi_\theta(a_h^m \mid s_h^m)}\right)$

   for SGD, use gradient: $g(\theta) := \nabla_\theta \widehat{\ell}^k(\theta)$

$g(\theta^k)$ is unbiased for $\nabla_\theta \ell^k(\theta)\Big|_{\theta=\theta^k}$

Importance Sampling & Variance

Importance Sampling & Variance

- If we can do importance sampling, why do we need our objective function to keep updating?

Importance Sampling & Variance

- If we can do importance sampling, why do we need our objective function to keep updating?
- I.e., why not just optimize $\mathbb{E}_{\tau \sim \rho_{\pi_{\theta^0}}} \left[\sum_{h=0}^{H-1} \frac{\pi_{\theta}(a_h | s_h)}{\pi_{\theta^0}(a_h | s_h)} A^{\pi_{\theta^0}}(s_h, a_h, h) \right]$?

Importance Sampling & Variance

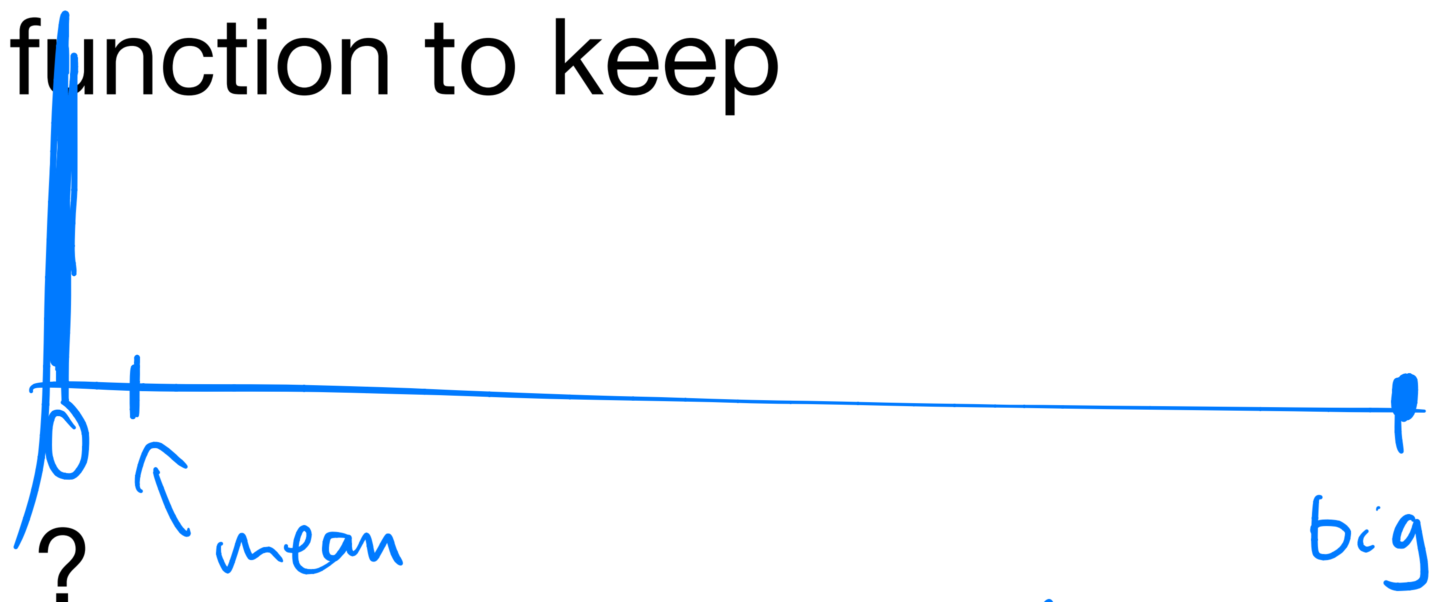
- If we can do importance sampling, why do we need our objective function to keep updating?
- I.e., why not just optimize $\mathbb{E}_{\tau \sim \rho_{\pi_{\theta^0}}} \left[\sum_{h=0}^{H-1} \frac{\pi_{\theta}(a_h \mid s_h)}{\pi_{\theta^0}(a_h \mid s_h)} A^{\pi_{\theta^0}}(s_h, a_h, h) \right]$?
- Or in PG, why do we sample online, when the likelihood ratio method still gives unbiasedness for trajectories sampled from π_{θ^0}?

Importance Sampling & Variance

- If we can do importance sampling, why do we need our objective function to keep updating?
- I.e., why not just optimize $\mathbb{E}_{\tau \sim \rho_{\pi_{\theta^0}}} \left[\sum_{h=0}^{H-1} \frac{\pi_{\theta}(a_h \mid s_h)}{\pi_{\theta^0}(a_h \mid s_h)} A^{\pi_{\theta^0}}(s_h, a_h, h) \right]$?
- Or in PG, why do we sample online, when the likelihood ratio method still gives unbiasedness for trajectories sampled from π_{θ^0}?
- **Variance**: Importance sampling incurs big variance when the target and proposal distributions are far apart

Importance Sampling & Variance

- If we can do importance sampling, why do we need our objective function to keep updating?



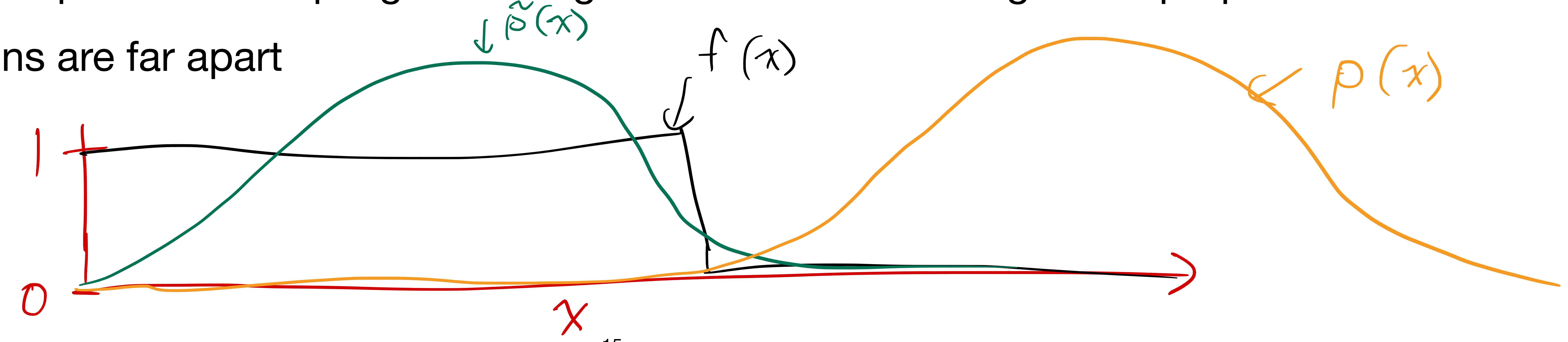
- I.e., why not just optimize $\mathbb{E}_{\tau \sim \rho_{\pi_{\theta^0}}} \left[\sum_{h=0}^{H-1} \frac{\pi_{\theta}(a_h \mid s_h)}{\pi_{\theta^0}(a_h \mid s_h)} A^{\pi_{\theta^0}}(s_h, a_h, h) \right]$?

$\mathbb{E}_{x\sim p(x)}\left[\overbrace{\frac{\tilde{p}(x)}{p(x)} f(x)}^{Y}\right]$

- Or in PG, why do we sample online, when the likelihood ratio method still gives unbiasedness for trajectories sampled from π_{θ^0}?

- **Variance**: Importance sampling incurs big variance when the target and proposal distributions are far apart
- Picture:

$\mathbb{E}_{x\sim \tilde{p}(x)}\left[f(x)\right] = 1$



Today

- Feedback from last lecture
- Recap
- Importance Sampling (for PPO)
- PG review
- Exploration?

Meta-Approach: TRPO/NPG/PPO are all pretty similar

Meta-Approach: TRPO/NPG/PPO are all pretty similar

1. Initialize θ^0
2. For $k = 0, \ldots, K$:

$$\theta^{k+1} \approx \arg\max_{\theta} \Delta_k(\pi_\theta), \quad \text{where } \Delta_k(\pi_\theta) := \mathbb{E}_{s_0,\ldots,s_{H-1}\sim\rho_{\pi_{\theta^k}}}\left[\sum_{h=0}^{H-1} \mathbb{E}_{a_h\sim\pi_\theta(\cdot|s_h)}\left[A^{\pi_{\theta^k}}(s_h, a_h, h)\right]\right]$$

such that ρ_{π_θ} is "close" to $\rho_{\pi_{\theta^k}}$

Meta-Approach: TRPO/NPG/PPO are all pretty similar

1. Initialize θ^0
2. For $k = 0, \ldots, K$:

$$\theta^{k+1} \approx \arg\max_{\theta} \Delta_k(\pi_\theta), \qquad \text{where } \Delta_k(\pi_\theta) := \mathbb{E}_{s_0,\ldots,s_{H-1} \sim \rho_{\pi_{\theta^k}}}\left[\sum_{h=0}^{H-1} \mathbb{E}_{a_h \sim \pi_\theta(\cdot|s_h)}\left[A^{\pi_{\theta^k}}(s_h, a_h, h)\right]\right]$$

such that ρ_{π_θ} is "close" to $\rho_{\pi_{\theta^k}}$

- TRPO: use KL to enforce closeness.

Meta-Approach: TRPO/NPG/PPO are all pretty similar

1. Initialize θ^0
2. For $k = 0, \ldots, K$:

$$\theta^{k+1} \approx \arg\max_{\theta} \Delta_k(\pi_\theta), \quad \text{where } \Delta_k(\pi_\theta) := \mathbb{E}_{s_0,\ldots,s_{H-1} \sim \rho_{\pi_{\theta^k}}} \left[\sum_{h=0}^{H-1} \mathbb{E}_{a_h \sim \pi_\theta(\cdot|s_h)} \left[A^{\pi_{\theta^k}}(s_h, a_h, h) \right] \right]$$

such that ρ_{π_θ} is "close" to $\rho_{\pi_{\theta^k}}$

- TRPO: use KL to enforce closeness.
- NPG: is TRPO up to "leading order" (via Taylor's theorem).

Meta-Approach: TRPO/NPG/PPO are all pretty similar

1. Initialize θ^0
2. For $k = 0, \ldots, K$:

$$\theta^{k+1} \approx \arg\max_{\theta} \Delta_k(\pi_\theta), \qquad \text{where } \Delta_k(\pi_\theta) := \mathbb{E}_{s_0,\ldots,s_{H-1} \sim \rho_{\pi_{\theta^k}}} \left[\sum_{h=0}^{H-1} \mathbb{E}_{a_h \sim \pi_\theta(\cdot|s_h)} \left[A^{\pi_{\theta^k}}(s_h, a_h, h) \right] \right]$$

such that ρ_{π_θ} is "close" to $\rho_{\pi_{\theta^k}}$

- TRPO: use KL to enforce closeness.
- NPG: is TRPO up to "leading order" (via Taylor's theorem).
- PPO: uses a Lagrangian relaxation (i.e. regularization)

Meta-Approach: TRPO/NPG/PPO are all pretty similar

1. Initialize θ^0
2. For $k = 0, \ldots, K$:

$$\theta^{k+1} \approx \arg\max_{\theta} \Delta_k(\pi_\theta), \qquad \text{where } \Delta_k(\pi_\theta) := \mathbb{E}_{s_0,\ldots,s_{H-1} \sim \rho_{\pi_{\theta^k}}} \left[\sum_{h=0}^{H-1} \mathbb{E}_{a_h \sim \pi_\theta(\cdot|s_h)} \left[A^{\pi_{\theta^k}}(s_h, a_h, h) \right] \right]$$

such that ρ_{π_θ} is "close" to $\rho_{\pi_{\theta^k}}$

- TRPO: use KL to enforce closeness.
- NPG: is TRPO up to "leading order" (via Taylor's theorem).
- PPO: uses a Lagrangian relaxation (i.e. regularization)

3. Return π_{θ^K}

One algorithmic difference between TRPO/PPO and PG

One algorithmic difference between TRPO/PPO and PG

- PG just said: define our objective function $J(\theta)$, and then run SGD on it

One algorithmic difference between TRPO/PPO and PG

- PG just said: define our objective function $J(\theta)$, and then run SGD on it
    - Had to use likelihood ratio method to get stochastic gradients

One algorithmic difference between TRPO/PPO and PG

- PG just said: define our objective function $J(\theta)$, and then run SGD on it
  - Had to use likelihood ratio method to get stochastic gradients
  - Had to use variance reduction / baseline functions / batching to reduce variance

One algorithmic difference between TRPO/PPO and PG

- PG just said: define our objective function $J(\theta)$, and then run SGD on it
  - Had to use likelihood ratio method to get stochastic gradients
  - Had to use variance reduction / baseline functions / batching to reduce variance
- Each step of the algorithm takes a stochastic gradient step for the same objective

One algorithmic difference between TRPO/PPO and PG

- PG just said: define our objective function $J(\theta)$, and then run SGD on it
  - Had to use likelihood ratio method to get stochastic gradients
  - Had to use variance reduction / baseline functions / batching to reduce variance
- Each step of the algorithm takes a stochastic gradient step for the same objective

- TRPO and PPO said: compute an approximate objective function and try to optimize it, but make sure not to move too far since then the approximation of the objective breaks down

One algorithmic difference between TRPO/PPO and PG

- PG just said: define our objective function $J(\theta)$, and then run SGD on it
  - Had to use likelihood ratio method to get stochastic gradients
  - Had to use variance reduction / baseline functions / batching to reduce variance
- Each step of the algorithm takes a stochastic gradient step for the same objective

- TRPO and PPO said: compute an approximate objective function and try to optimize it, but make sure not to move too far since then the approximation of the objective breaks down
- So they constructed a new objective at each step, and then within those steps, performed SGD on that step's objective

One algorithmic difference between TRPO/PPO and PG

- PG just said: define our objective function $J(\theta)$, and then run SGD on it
  - Had to use likelihood ratio method to get stochastic gradients
  - Had to use variance reduction / baseline functions / batching to reduce variance
- Each step of the algorithm takes a stochastic gradient step for the same objective

- TRPO and PPO said: compute an approximate objective function and try to optimize it, but make sure not to move too far since then the approximation of the objective breaks down
- So they constructed a new objective at each step, and then within those steps, performed SGD on that step's objective

- Really not so different, and NPG provides a unifying perspective: TRPO/PPO essentially doing PG with a 2nd-order correction to the gradient

All Policy Gradient Algorithms in One Slide

All Policy Gradient Algorithms in One Slide

Parameterize policy and optimize directly while sampling from MDP

All Policy Gradient Algorithms in One Slide

Parameterize policy and optimize directly while sampling from MDP

<u>Policy Gradient (PG)</u>

All Policy Gradient Algorithms in One Slide

Parameterize policy and optimize directly while sampling from MDP

<u>Policy Gradient (PG)</u>

$\Downarrow$ Variance too high

Variance reduction techniques
like mini-batches and baselines

All Policy Gradient Algorithms in One Slide

Parameterize policy and optimize directly while sampling from MDP

Fitted Policy Iteration

↓ Big steps unstable

<u>**Policy Gradient (PG)**</u>

↓ Variance too high

Variance reduction techniques like mini-batches and baselines

All Policy Gradient Algorithms in One Slide

Parameterize policy and optimize directly while sampling from MDP

Fitted Policy Iteration

Big steps
unstable

Policy Gradient (PG)

Variance
too high

Variance reduction techniques
like mini-batches and baselines

**Trust Region Policy
Optimization (TRPO)**

All Policy Gradient Algorithms in One Slide

Parameterize policy and optimize directly while sampling from MDP

Fitted Policy Iteration

Big steps
unstable

Policy Gradient (PG)

Variance
too high

Variance reduction techniques
like mini-batches and baselines

Trust Region Policy Optimization (TRPO)

Approximation
in closed form

Natural Policy Gradient (NPG)

All Policy Gradient Algorithms in One Slide

Parameterize policy and optimize directly while sampling from MDP

Policy Gradient (PG)

Variance too high

Variance reduction techniques like mini-batches and baselines

Fitted Policy Iteration

Big steps unstable

Trust Region Policy Optimization (TRPO)

Approximation in closed form

Natural Policy Gradient (NPG)

2nd-order updates expensive

Proximal Policy Optimization (PPO)

All Policy Gradient Algorithms in One Slide

Parameterize policy and optimize directly while sampling from MDP

Fitted Policy Iteration

Big steps unstable

Policy Gradient (PG)

Variance too high

Variance reduction techniques like mini-batches and baselines

Trust Region Policy Optimization (TRPO)

Approximation in closed form

Natural Policy Gradient (NPG)

2nd-order updates expensive

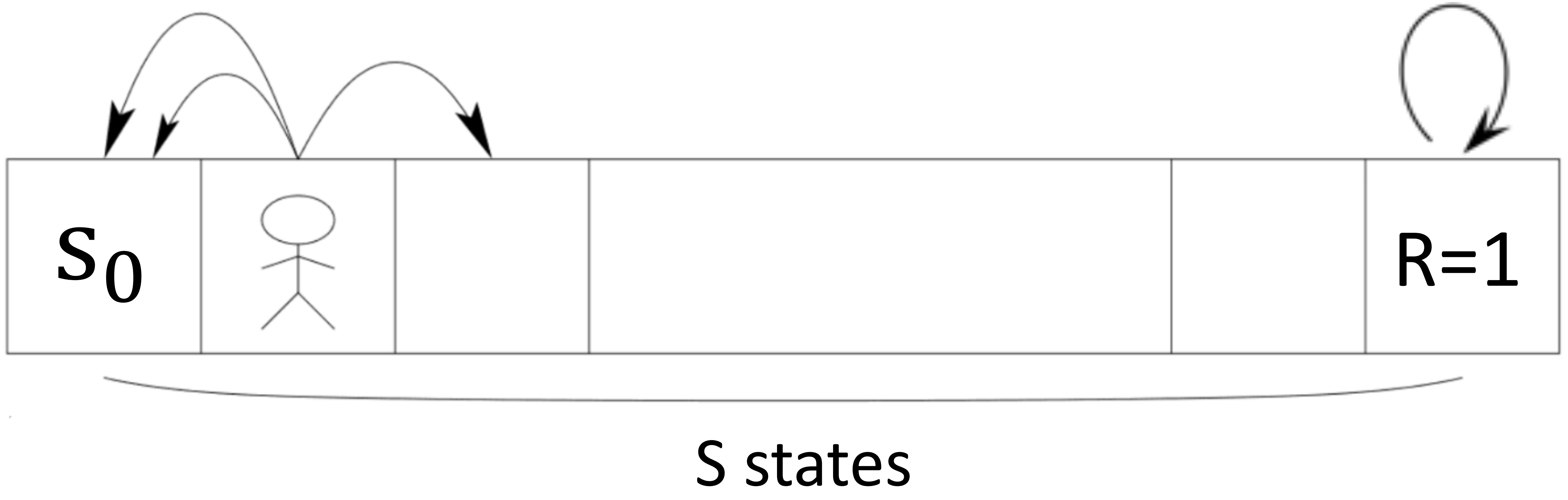
Proximal Policy Optimization (PPO)

PPO gets 2nd-order optimization benefits over PG and 1st-order computation benefits over TRPO/NPG

Today

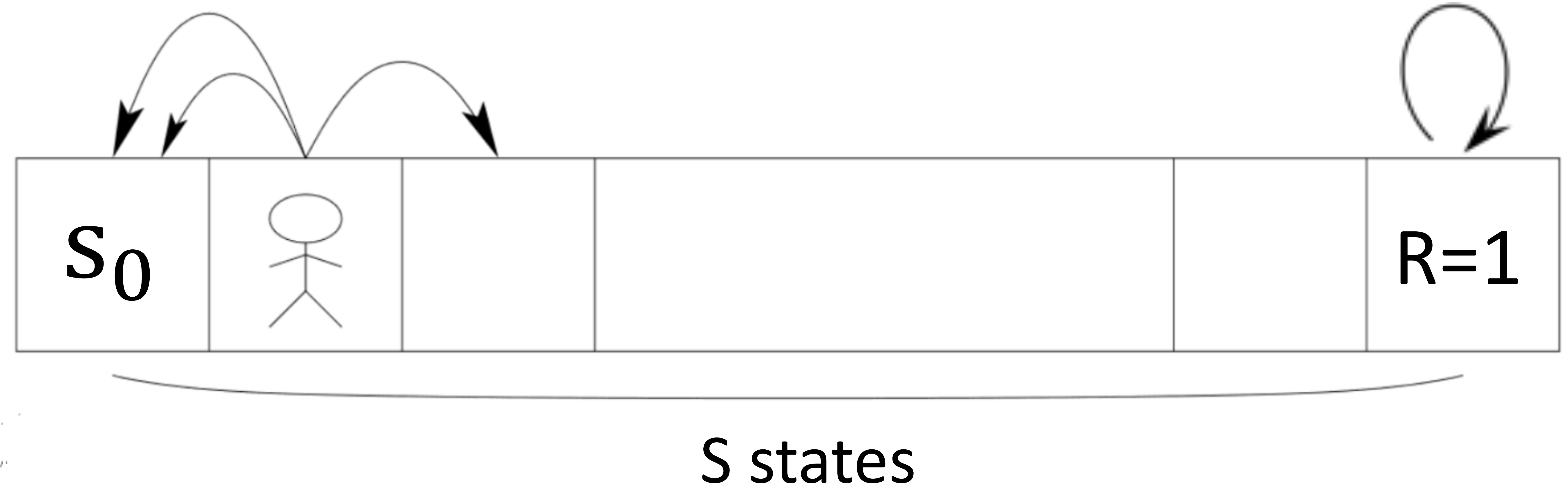
- Feedback from last lecture
- Recap
- Importance Sampling (for PPO)
- PG review
- Exploration?

“Lack of Exploration” leads to Optimization and Statistical Challenges



Thrun ‘92

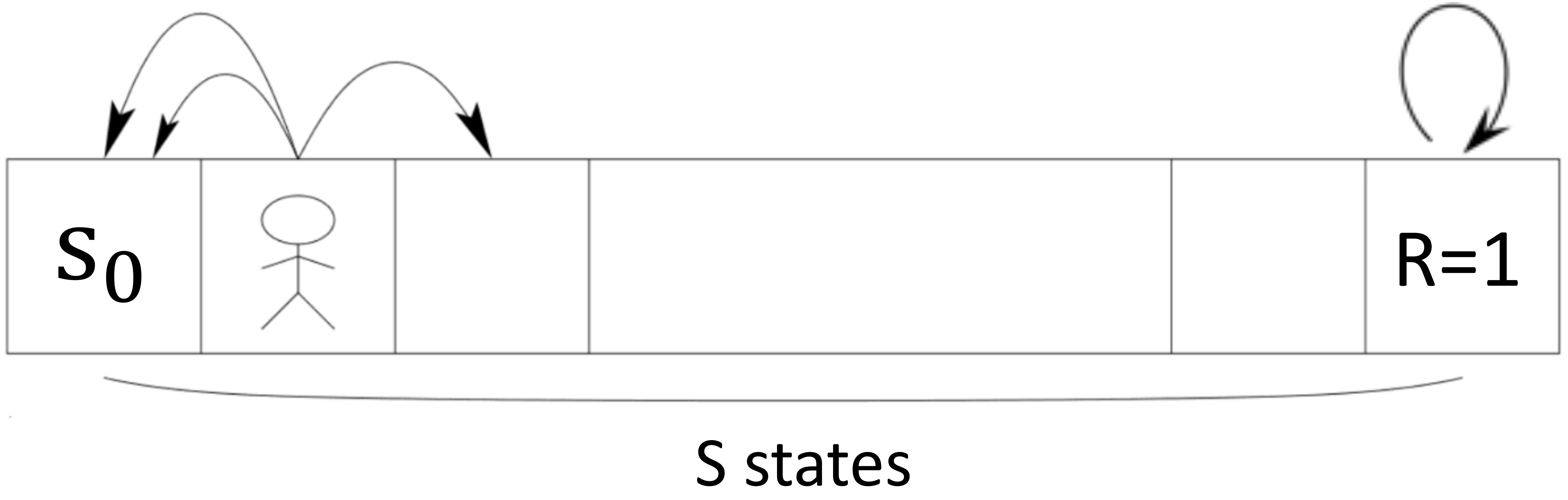
"Lack of Exploration" leads to Optimization and Statistical Challenges



Thrun '92

- Suppose $H \approx \text{poly}(|S|)$ & $\mu(s_0) = 1$ (i.e. we start at s_0).

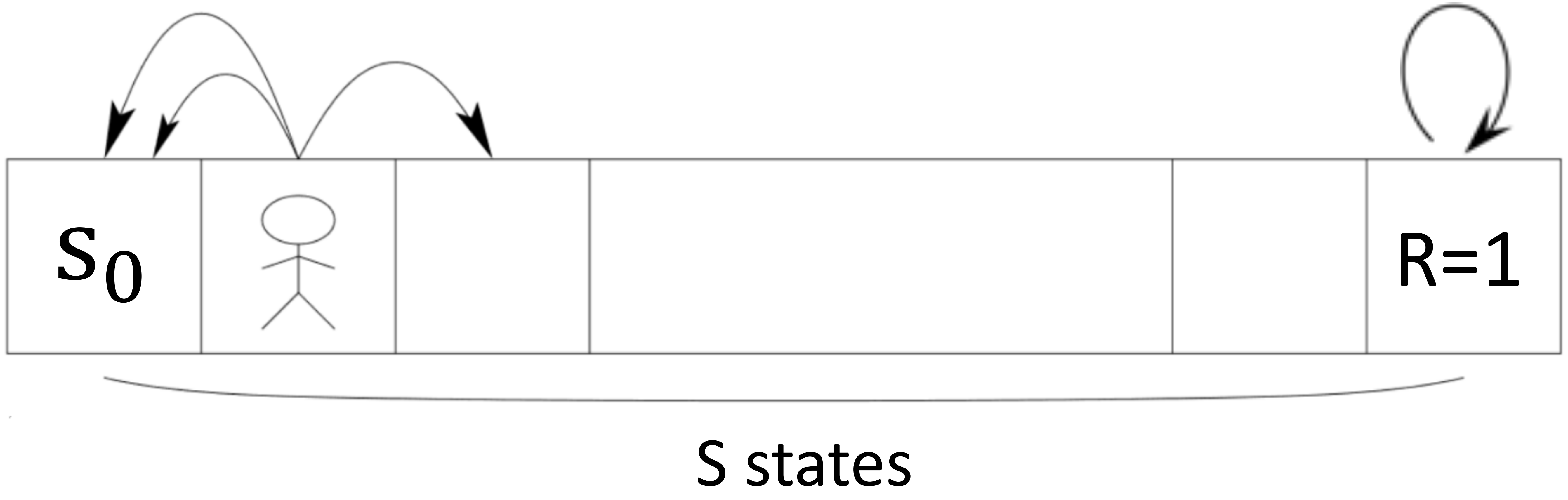
"Lack of Exploration" leads to Optimization and Statistical Challenges



Thrun '92

- Suppose $H \approx \text{poly}(|S|)$ & $\mu(s_0) = 1$ (i.e. we start at s_0).
- A randomly initialized policy π^0 has prob. $O(1/3^{|S|})$ of hitting the goal state in a trajectory.

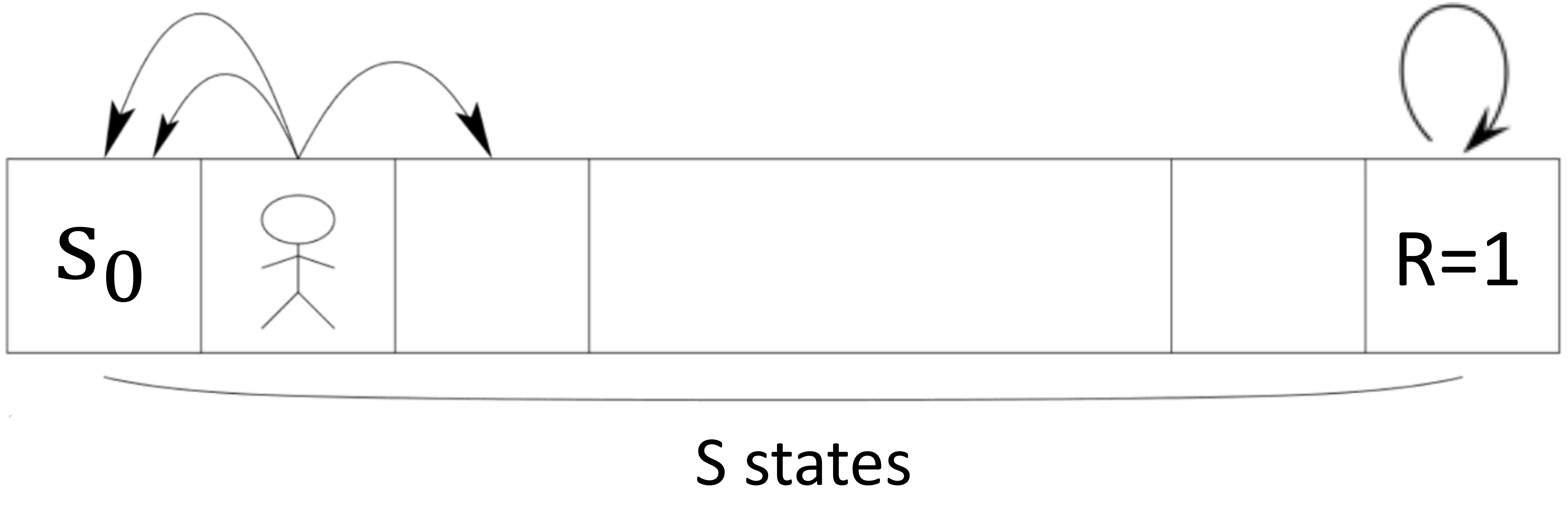
“Lack of Exploration” leads to Optimization and Statistical Challenges



Thrun ‘92

- Suppose $H \approx \text{poly}(|S|)$ & $\mu(s_0) = 1$ (i.e. we start at s_0).
- A randomly initialized policy π^0 has prob. $O(1/3^{|S|})$ of hitting the goal state in a trajectory.
- Thus a sample-based approach, with $\mu(s_0) = 1$, require $O(3^{|S|})$ trajectories.
  - Holds for (sample based) Fitted DP
  - Holds for (sample based) PG/TRPO/NPG/PPO

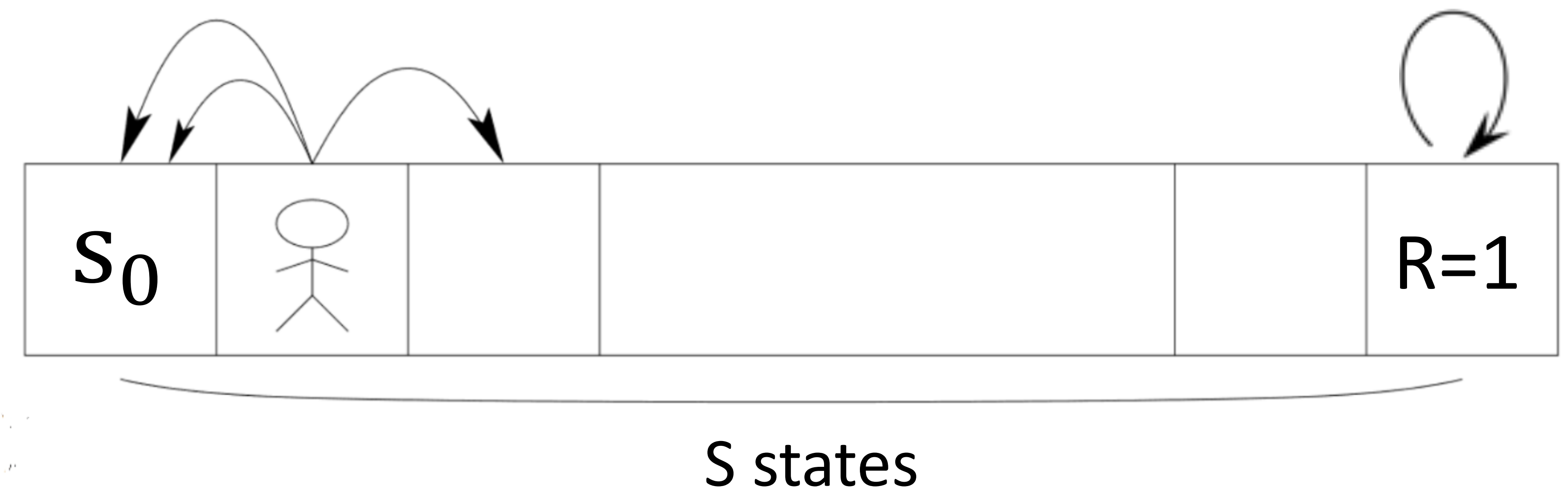
“Lack of Exploration” leads to Optimization and Statistical Challenges



Thrun ‘92

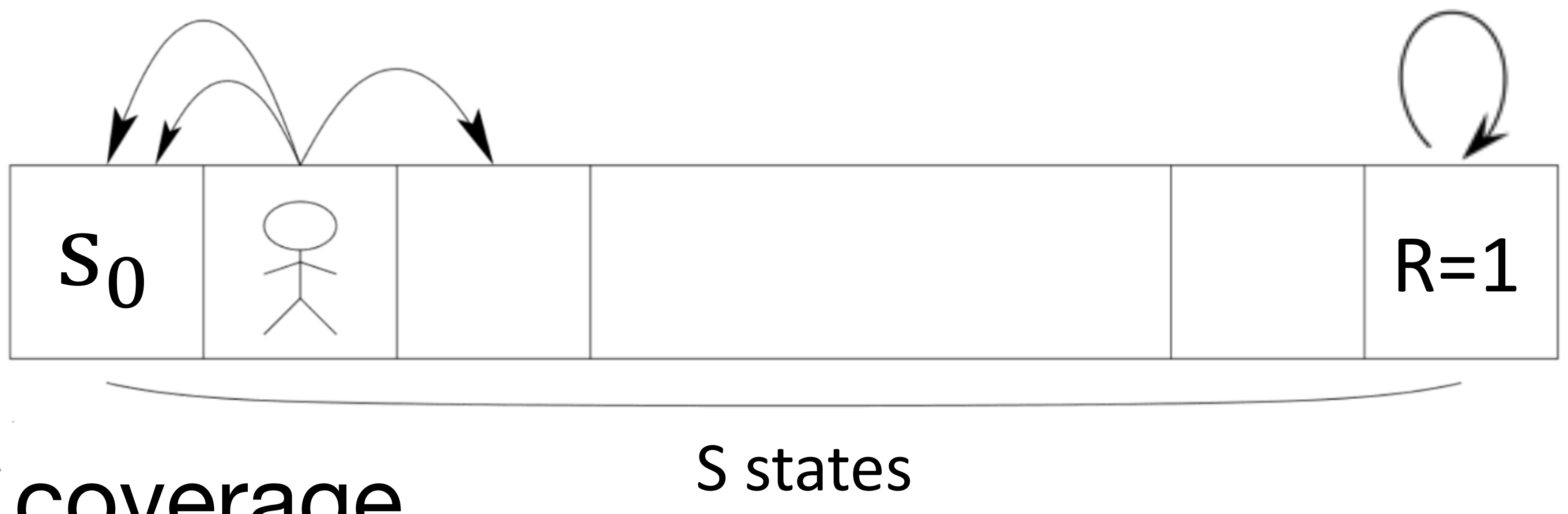
- Suppose $H \approx \text{poly}(|S|)$ & $\mu(s_0) = 1$ (i.e. we start at s_0).
- A randomly initialized policy π^0 has prob. $O(1/3^{|S|})$ of hitting the goal state in a trajectory.
- Thus a sample-based approach, with $\mu(s_0) = 1$, require $O(3^{|S|})$ trajectories.
    - Holds for (sample based) Fitted DP
    - Holds for (sample based) PG/TRPO/NPG/PPO
- Basically, for these approaches, there is no hope of learning the optimal policy if $\mu(s_0) = 1$.

Let's examine the role of μ



Thrun '92

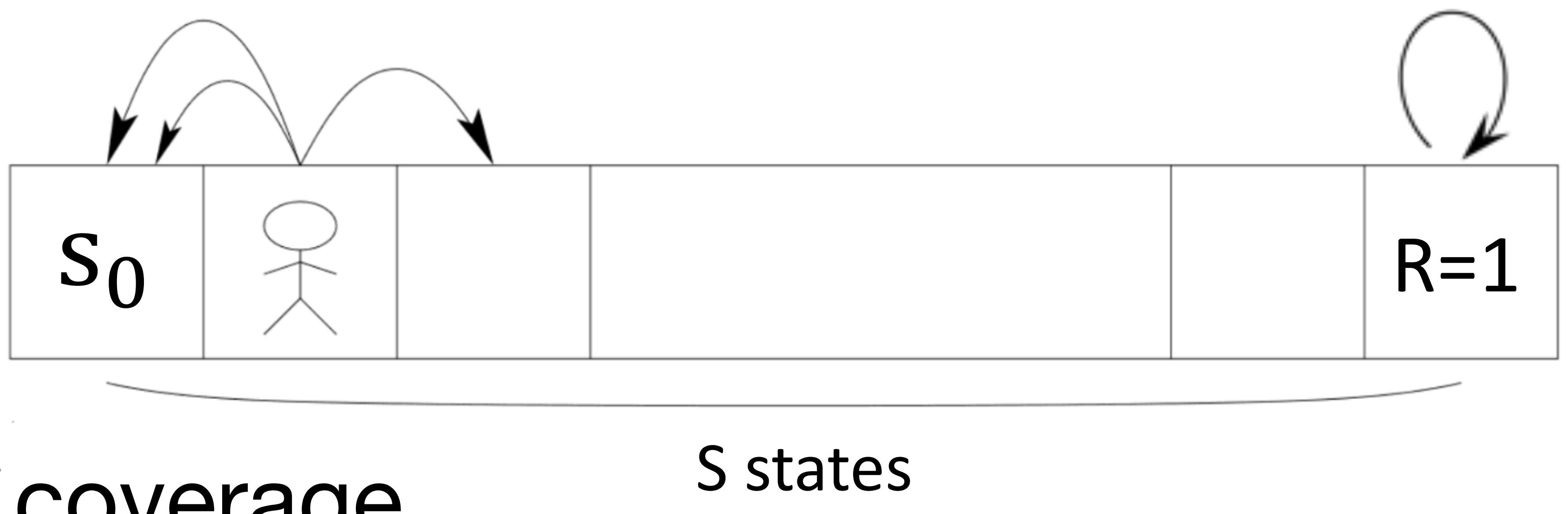
Let's examine the role of μ



Thrun '92

- Suppose that somehow the distribution μ had better coverage.
  - e.g, if μ was uniform overall states in our toy problem, then all approaches we covered would work (with mild assumptions)
  - Theory: TRPO/NPG/PPO have better guarantees than fitted DP methods (assuming some "coverage")

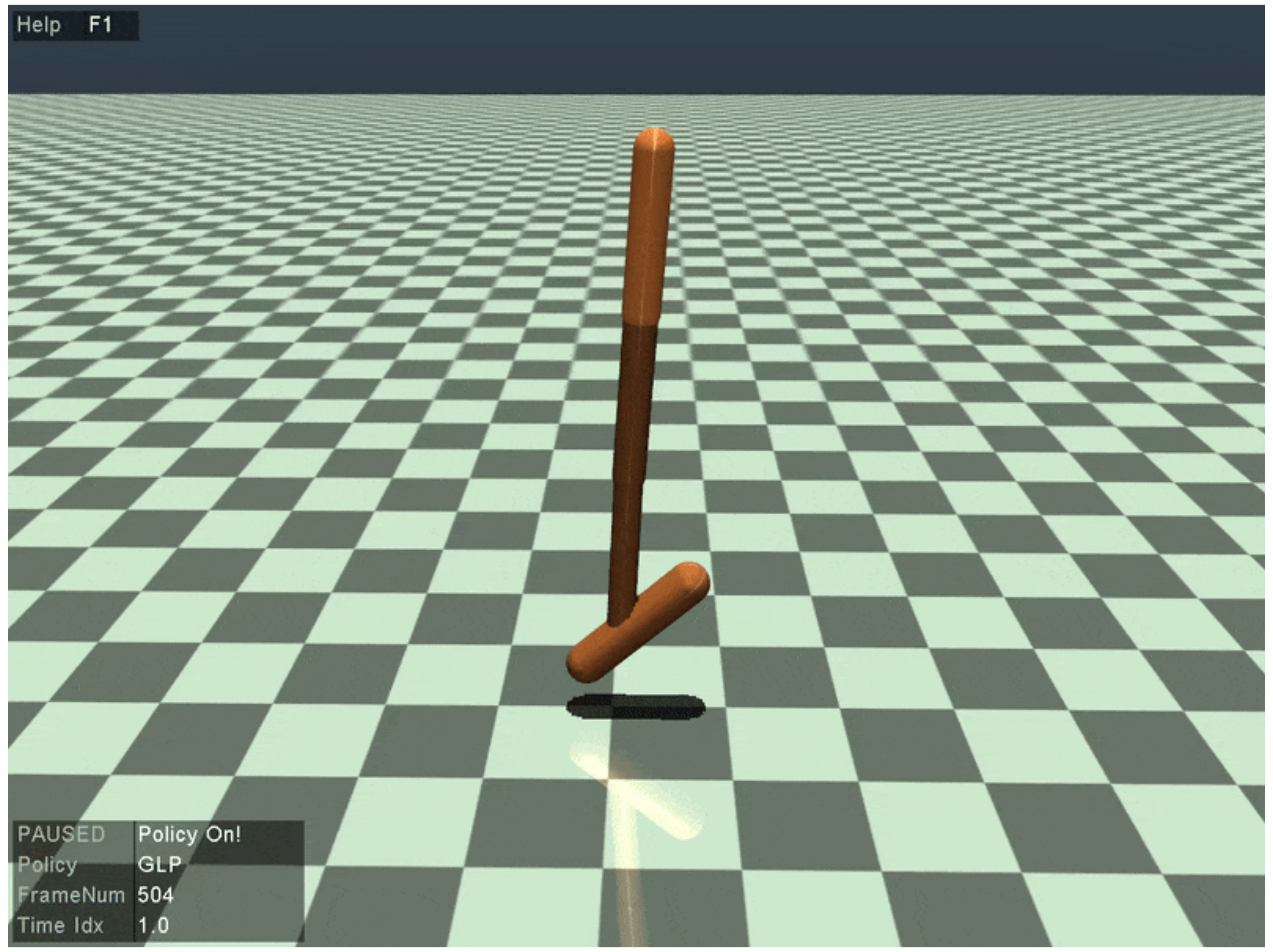
Let's examine the role of μ

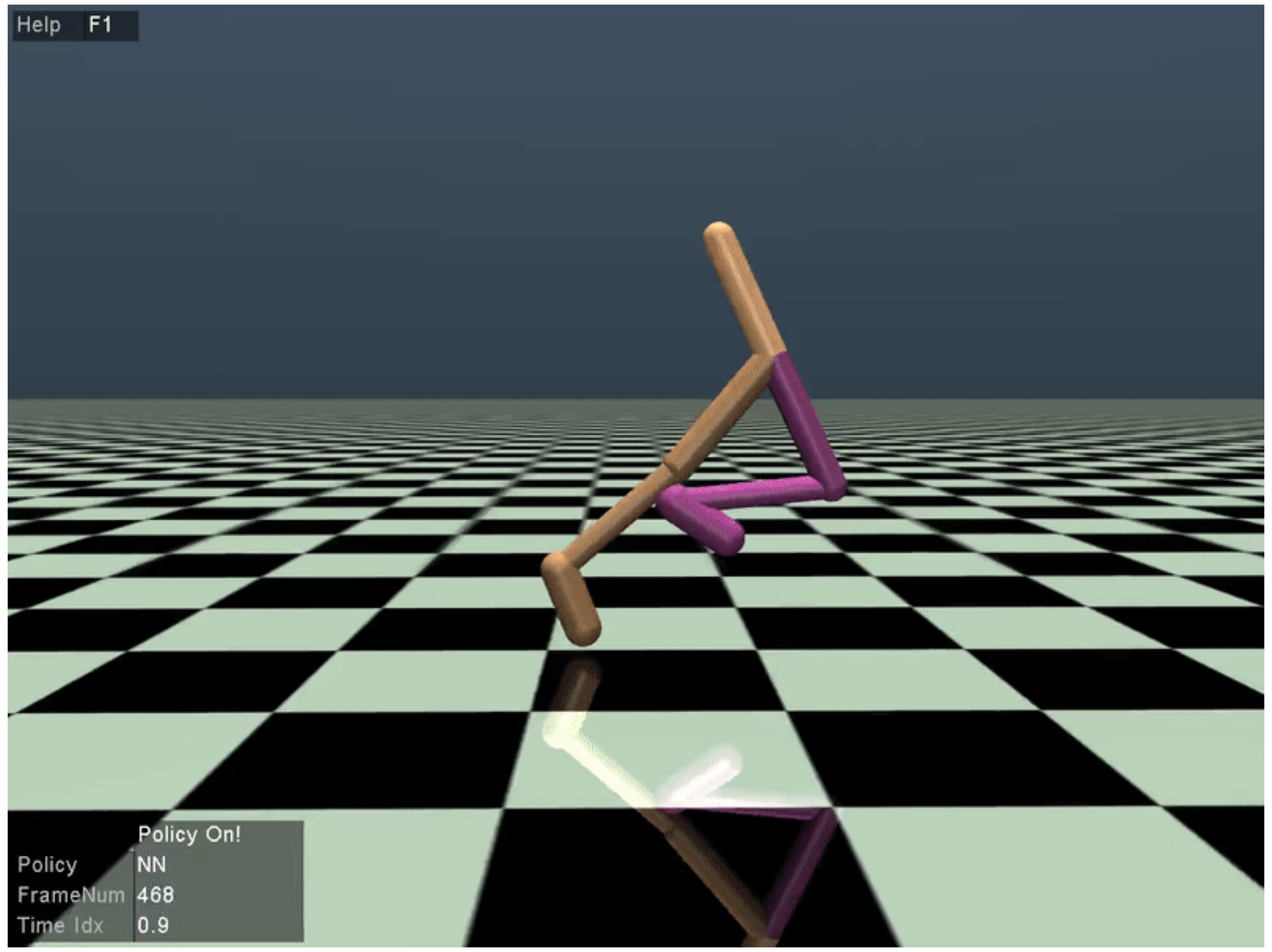


Thrun '92

- Suppose that somehow the distribution μ had better coverage.
  - e.g, if μ was uniform overall states in our toy problem, then all approaches we covered would work (with mild assumptions)
  - Theory: TRPO/NPG/PPO have better guarantees than fitted DP methods (assuming some "coverage")
- Strategies without coverage:
  - If we have a simulator, sometimes we can design μ to have better coverage.
    - this is helpful for robustness as well.
  - Imitation learning (next time).
    - An expert gives us samples from a "good" μ.
  - Explicit exploration:
    - UCB-VI: we'll merge two good ideas!
    - Encourage exploration in PG methods.
  - Try with reward shaping

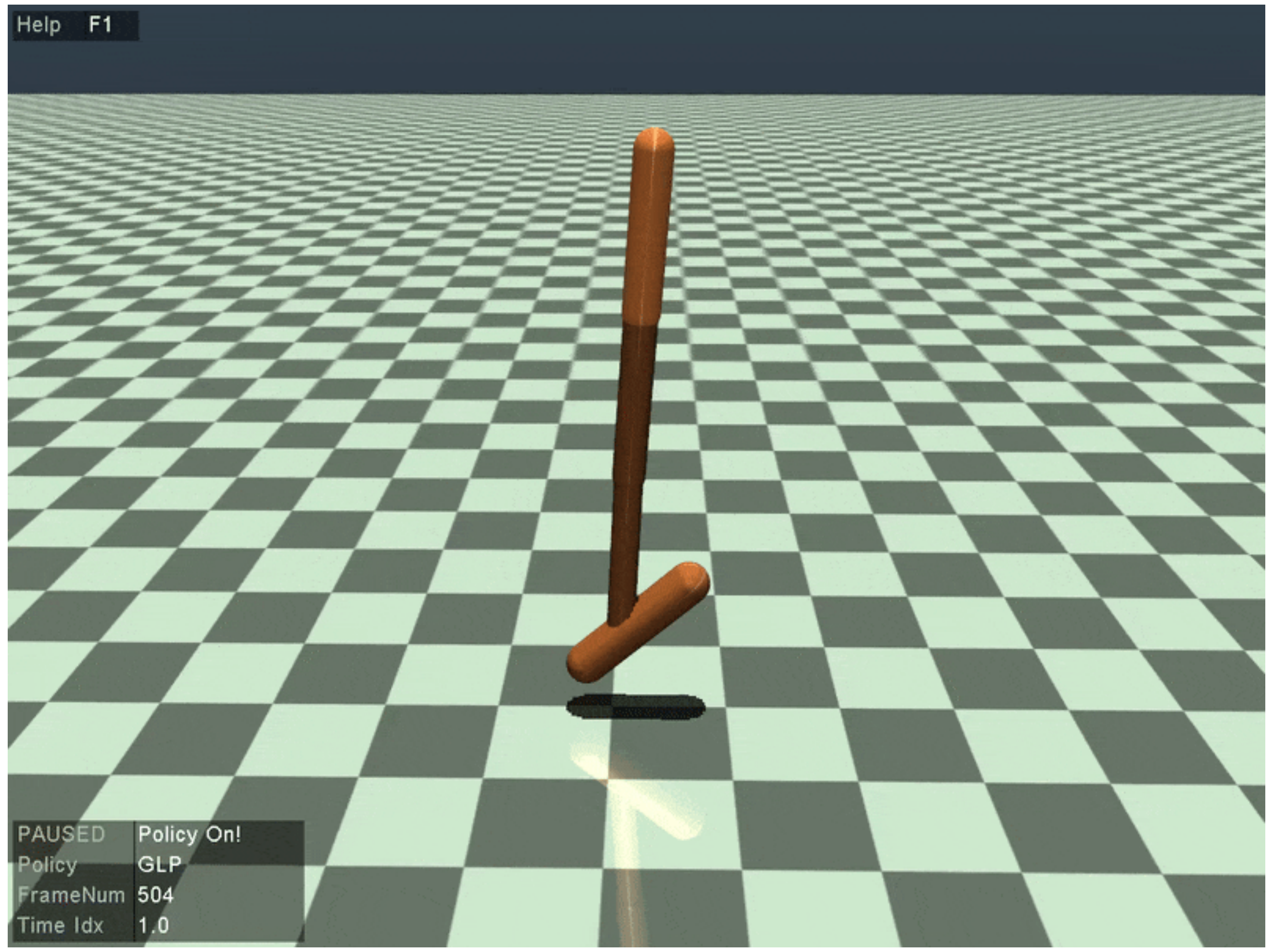
Aside: Brittle policies if we train starting from only from one configuration!

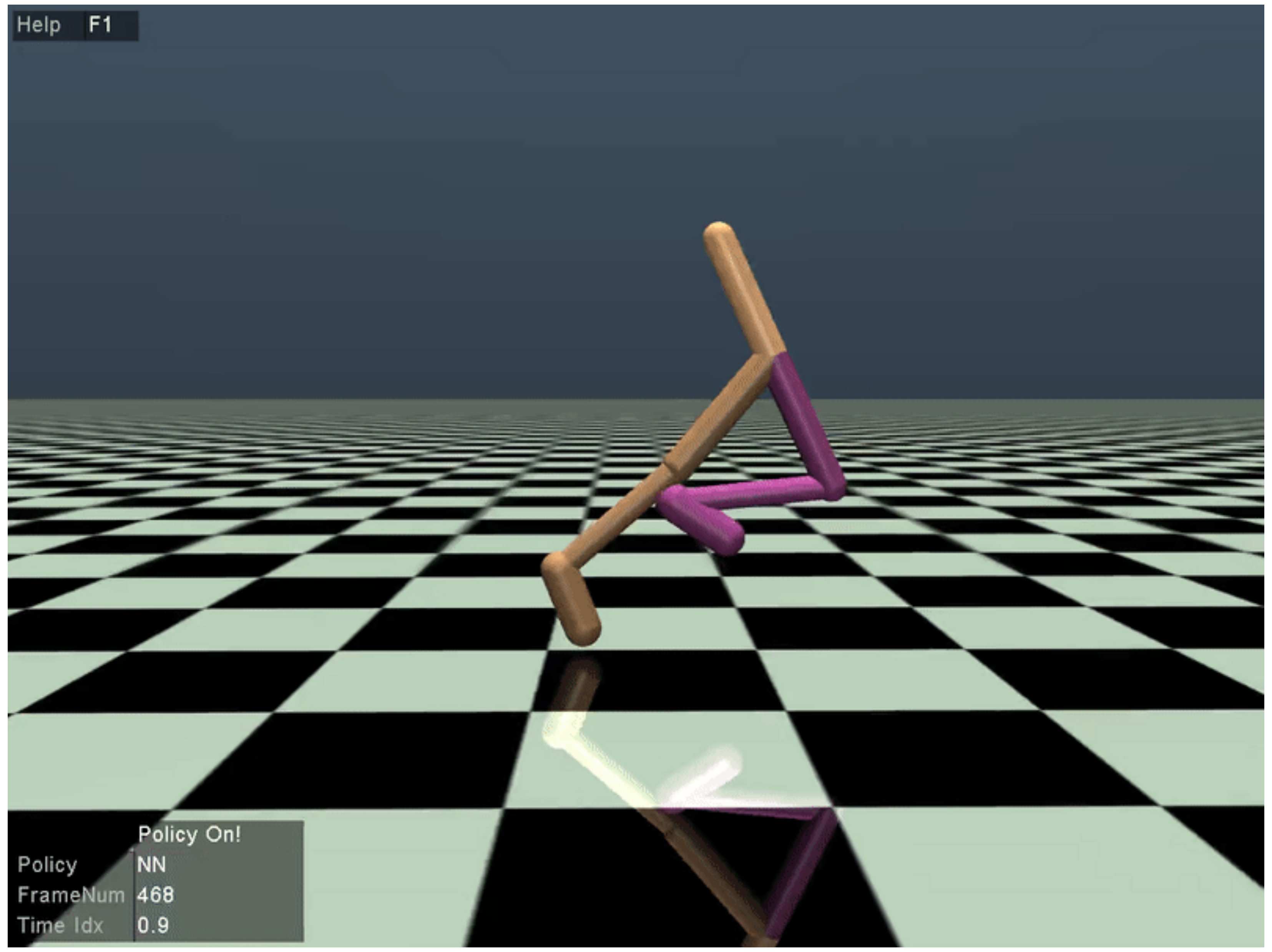




- [Rajeswaran, Lowrey, Todorov, K. 2017]: showed policies optimized for a single starting configuration s_0 are not robust!

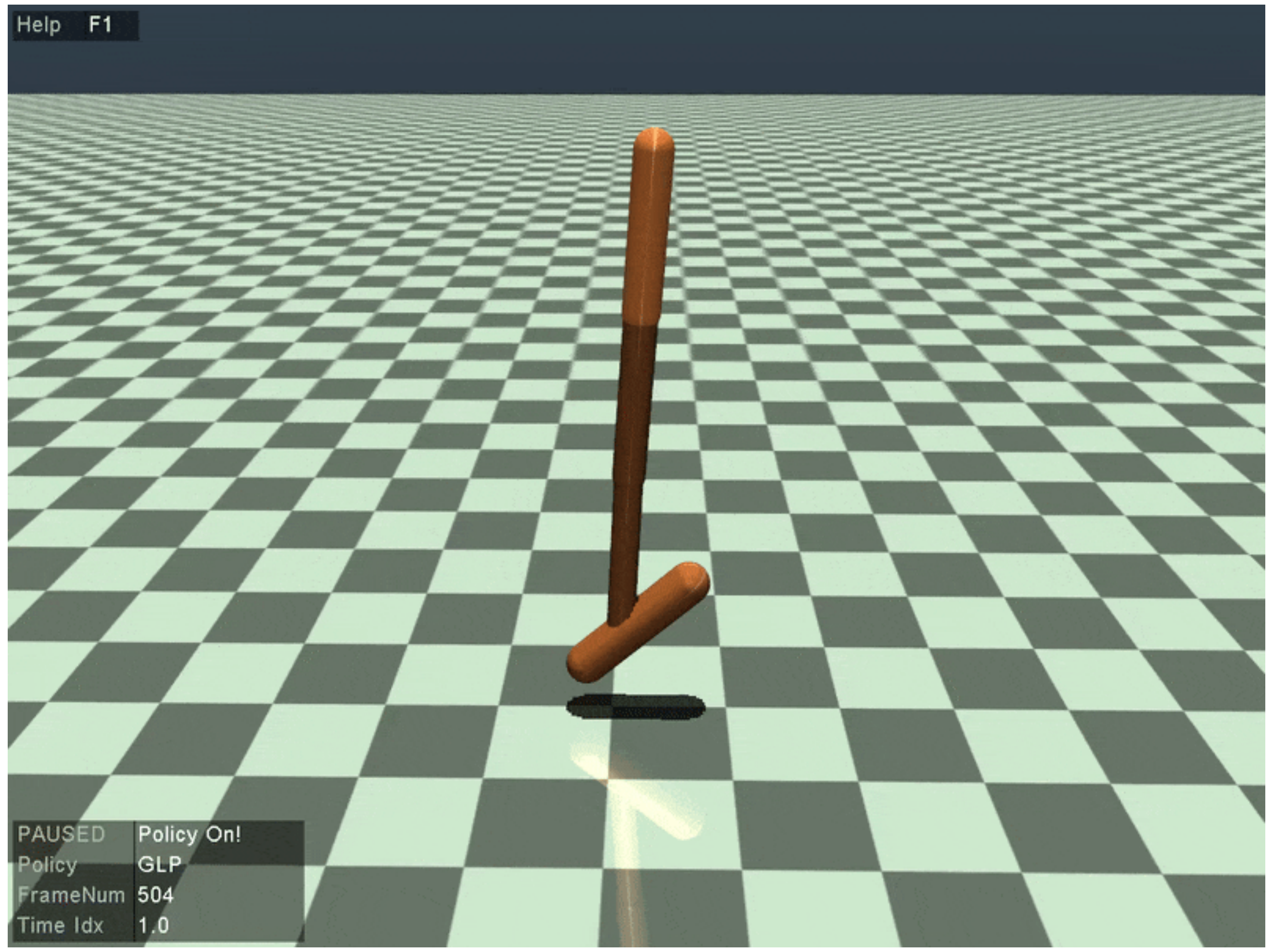
Aside: Brittle policies if we train starting from only from one configuration!

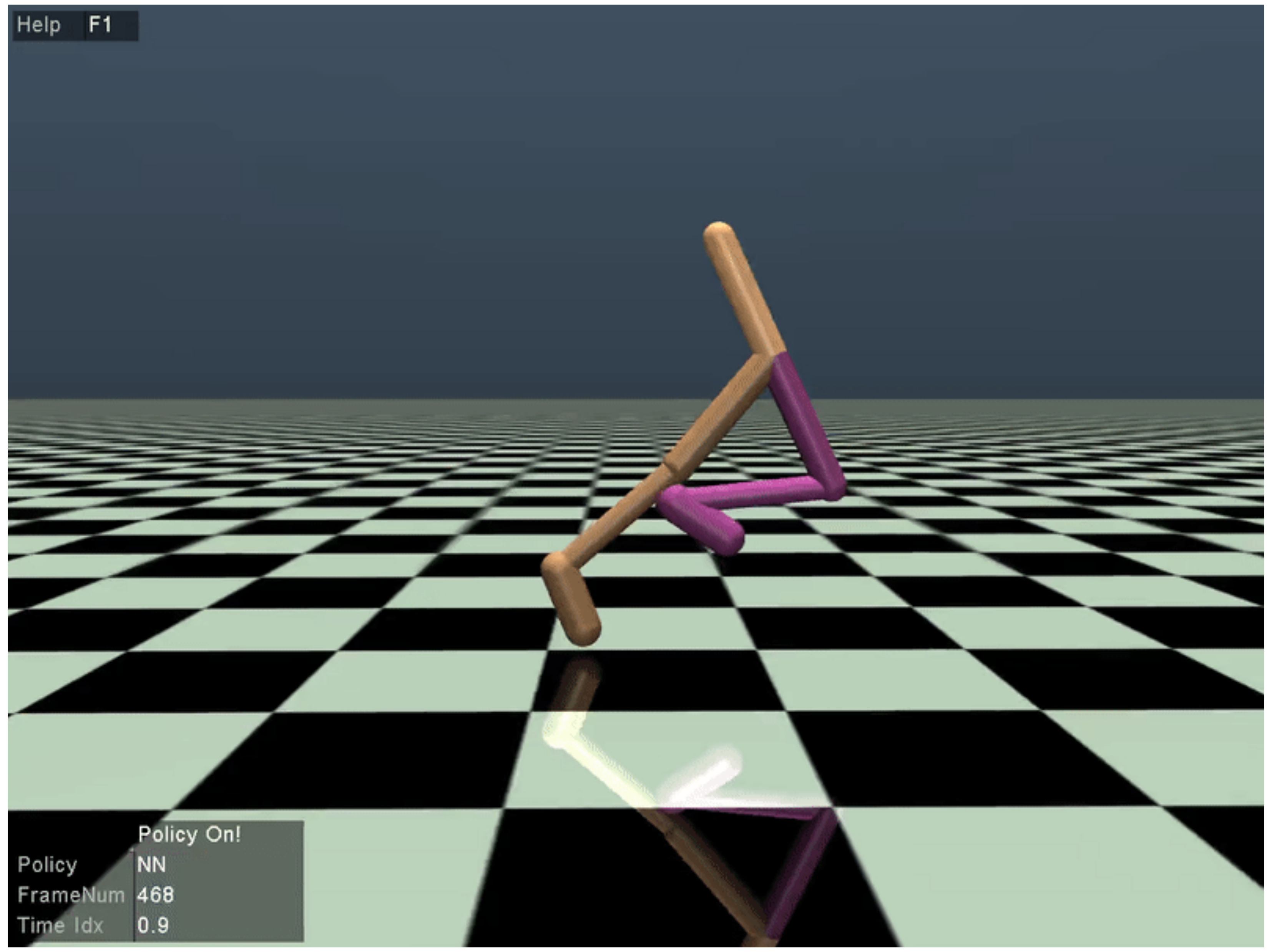




- [Rajeswaran, Lowrey, Todorov, K. 2017]: showed policies optimized for a single starting configuration s_0 are not robust!

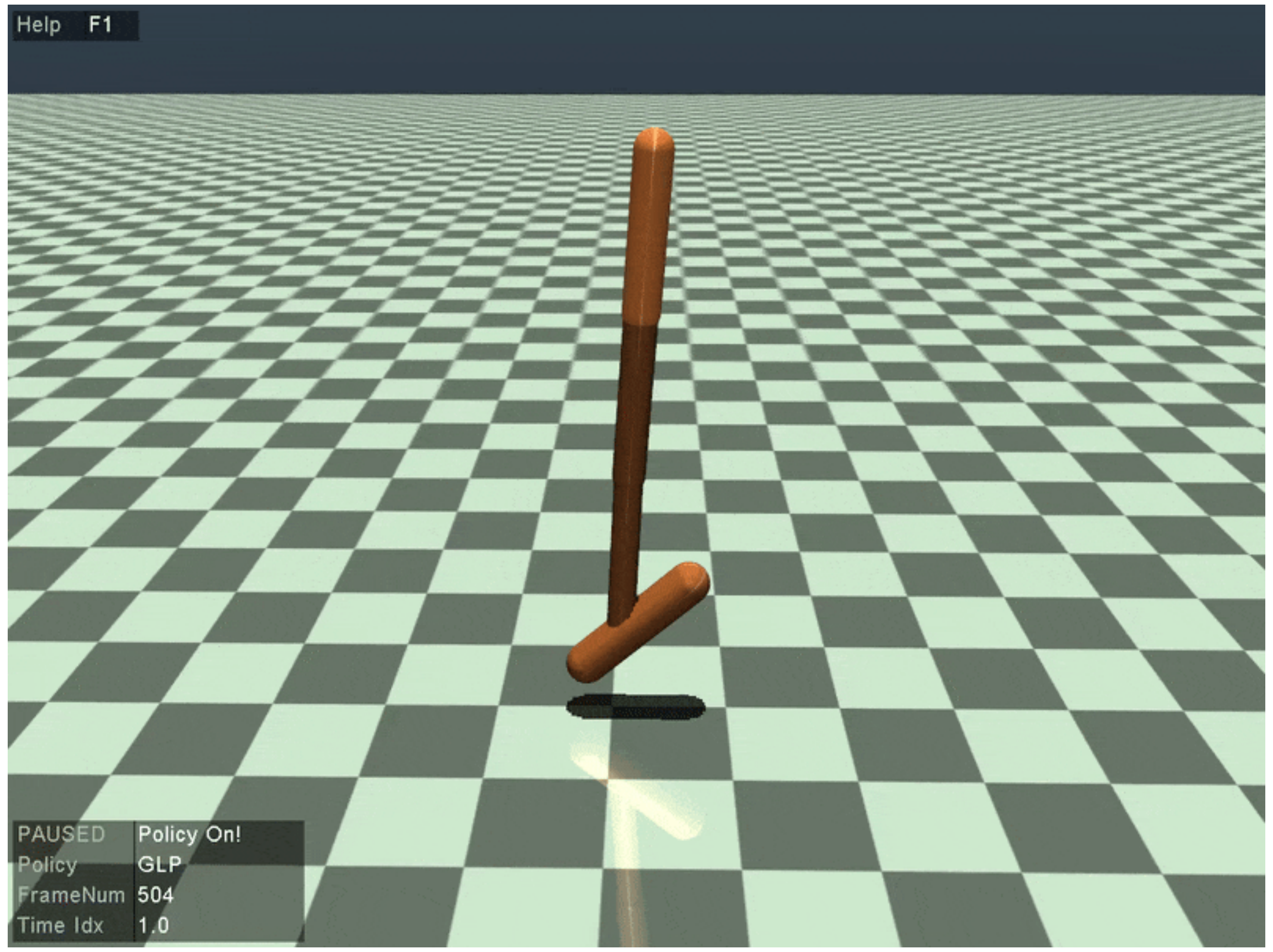
Aside: Brittle policies if we train starting from only from one configuration!

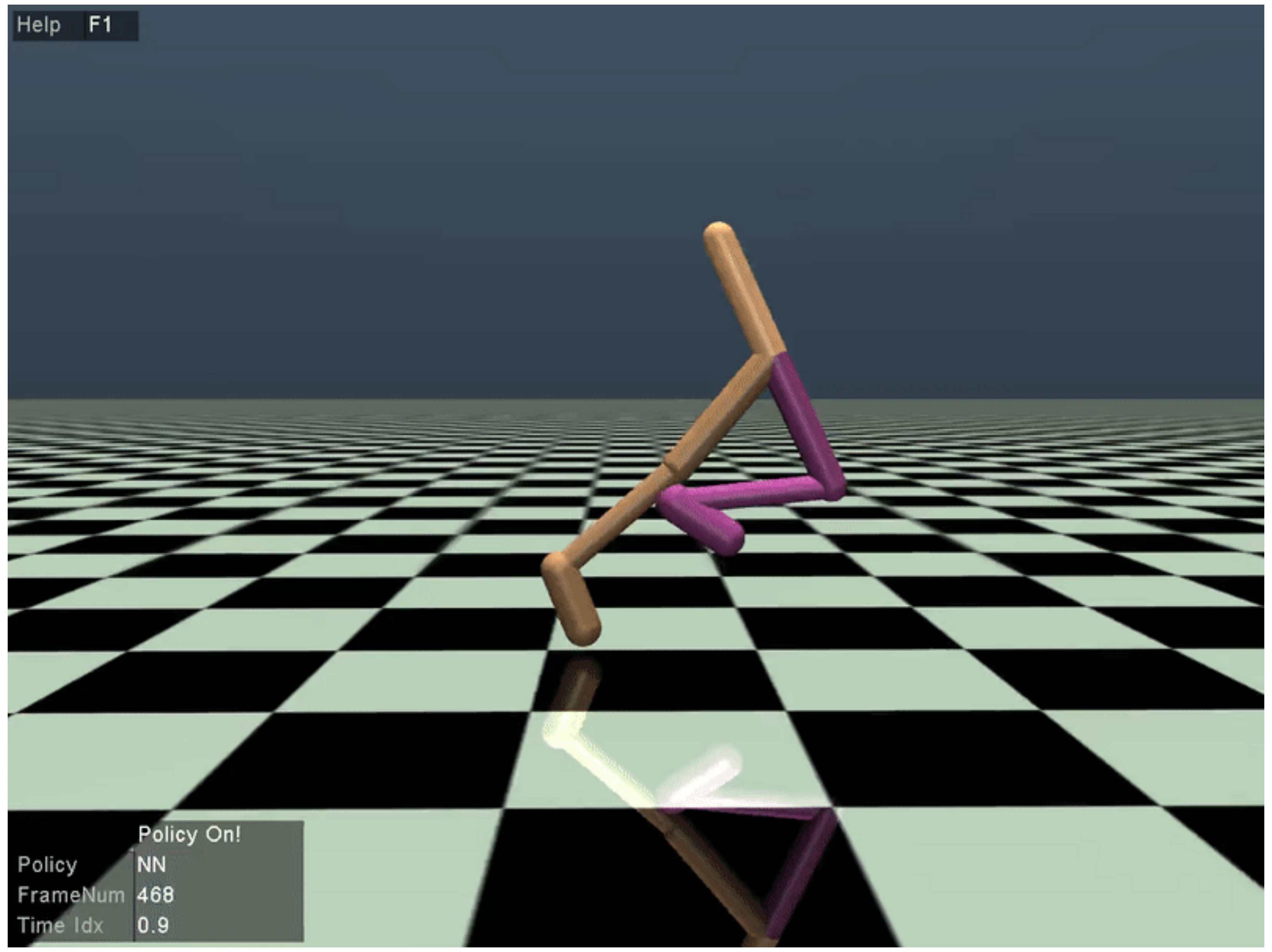




- [Rajeswaran, Lowrey, Todorov, K. 2017]: showed policies optimized for a single starting configuration s_0 are not robust!

Aside: Brittle policies if we train starting from only from one configuration!





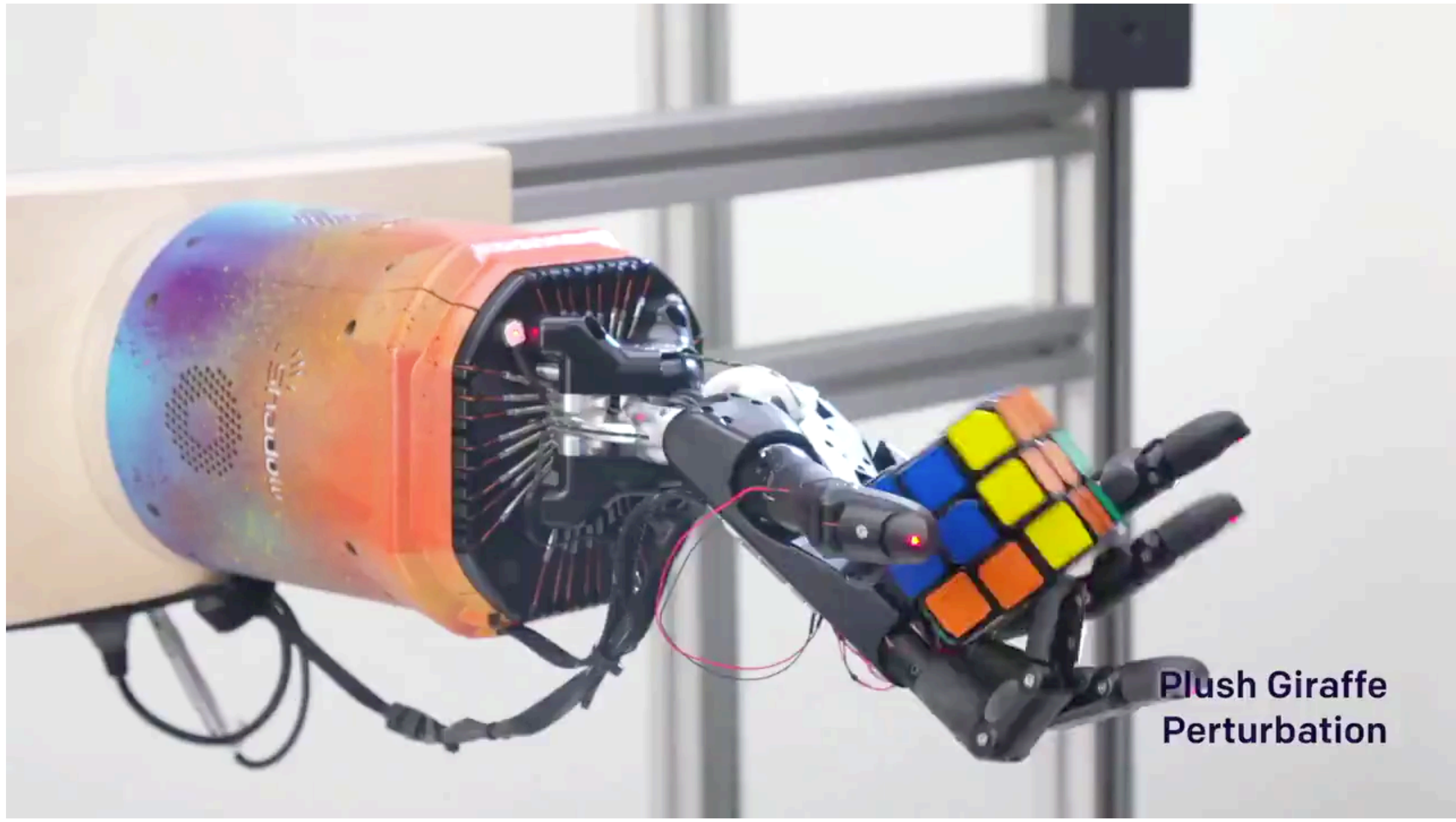
- [Rajeswaran, Lowrey, Todorov, K. 2017]: showed policies optimized for a single starting configuration s_0 are not robust!
- How to fix this?
  - Training from different starting configurations sampled from $s_0 \sim \mu$ fixes this:

$$\max_\theta \mathbb{E}_{s_0 \sim \mu}[V^\theta(s_0)]$$

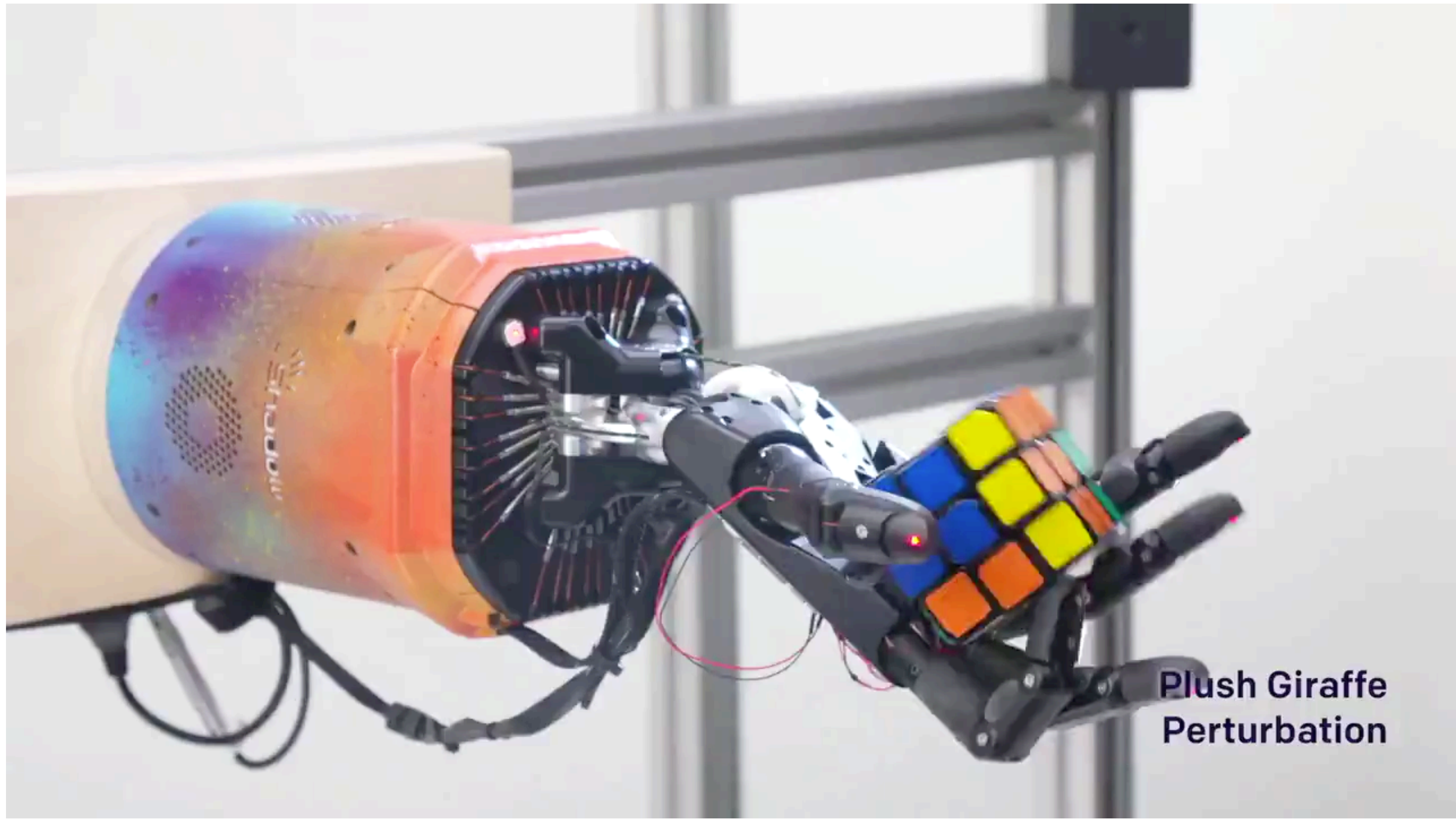
Even if starting position concentrated at just one point—good for robustness!

OpenAI: progress on dexterous hand manipulation

OpenAI: progress on dexterous hand manipulation



OpenAI: progress on dexterous hand manipulation



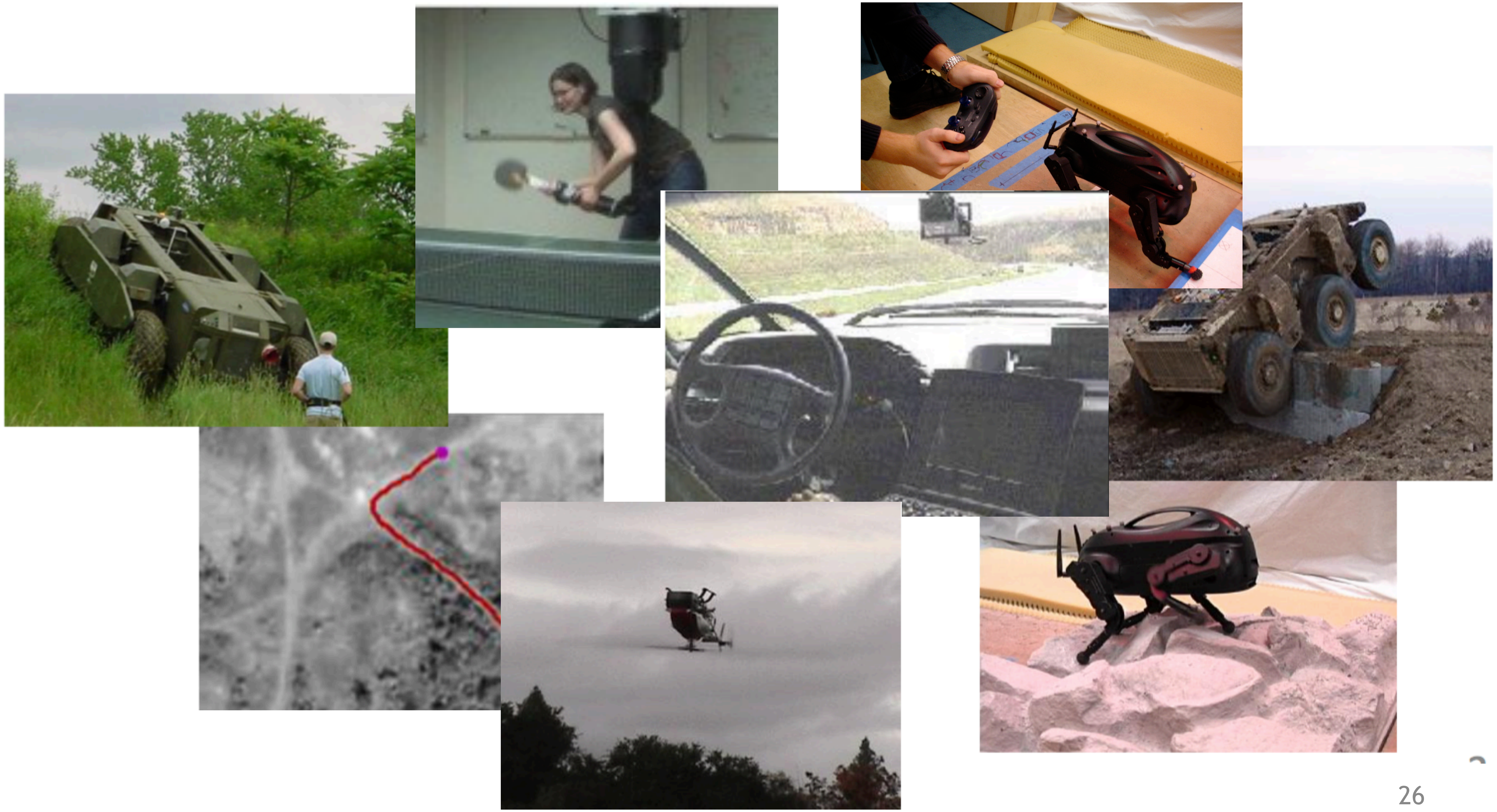
Trained with "domain randomization"

Basically, the measure $s_0 \sim \mu$ was diverse.

Today

- Feedback from last lecture
- Recap
- Importance Sampling (for PPO)
- PG review
- Exploration?

Imitation Learning



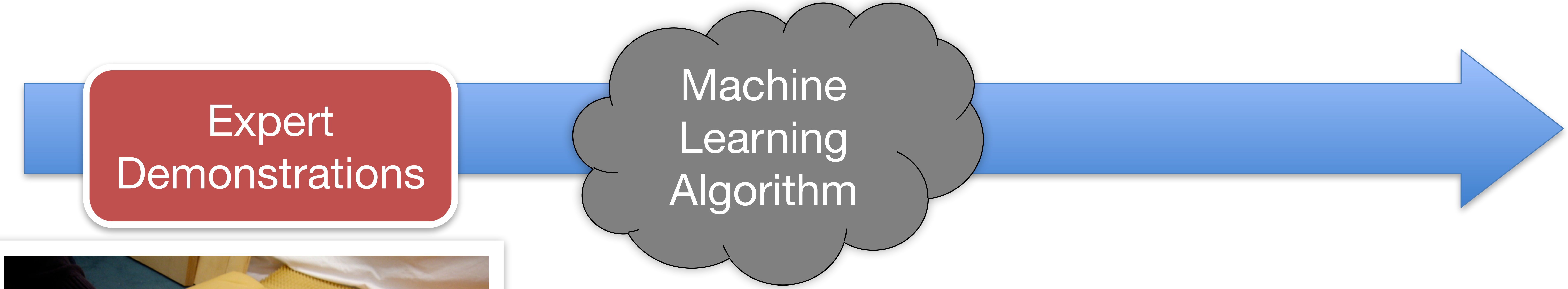
Imitation Learning

Imitation Learning

Imitation Learning

Expert Demonstrations

Imitation Learning



- SVM
- Gaussian Process
- Kernel Estimator
- Deep Networks
- Random Forests
- LWR
- …

Imitation Learning

Expert Demonstrations → Machine Learning Algorithm → Policy π

- SVM
- Gaussian Process
- Kernel Estimator
- Deep Networks
- Random Forests
- LWR
- …

Maps *states* to <u>actions</u>

Learning to Drive by Imitation

[Pomerleau89, Saxena05, Ross11a]

Input:

Camera Image

Policy

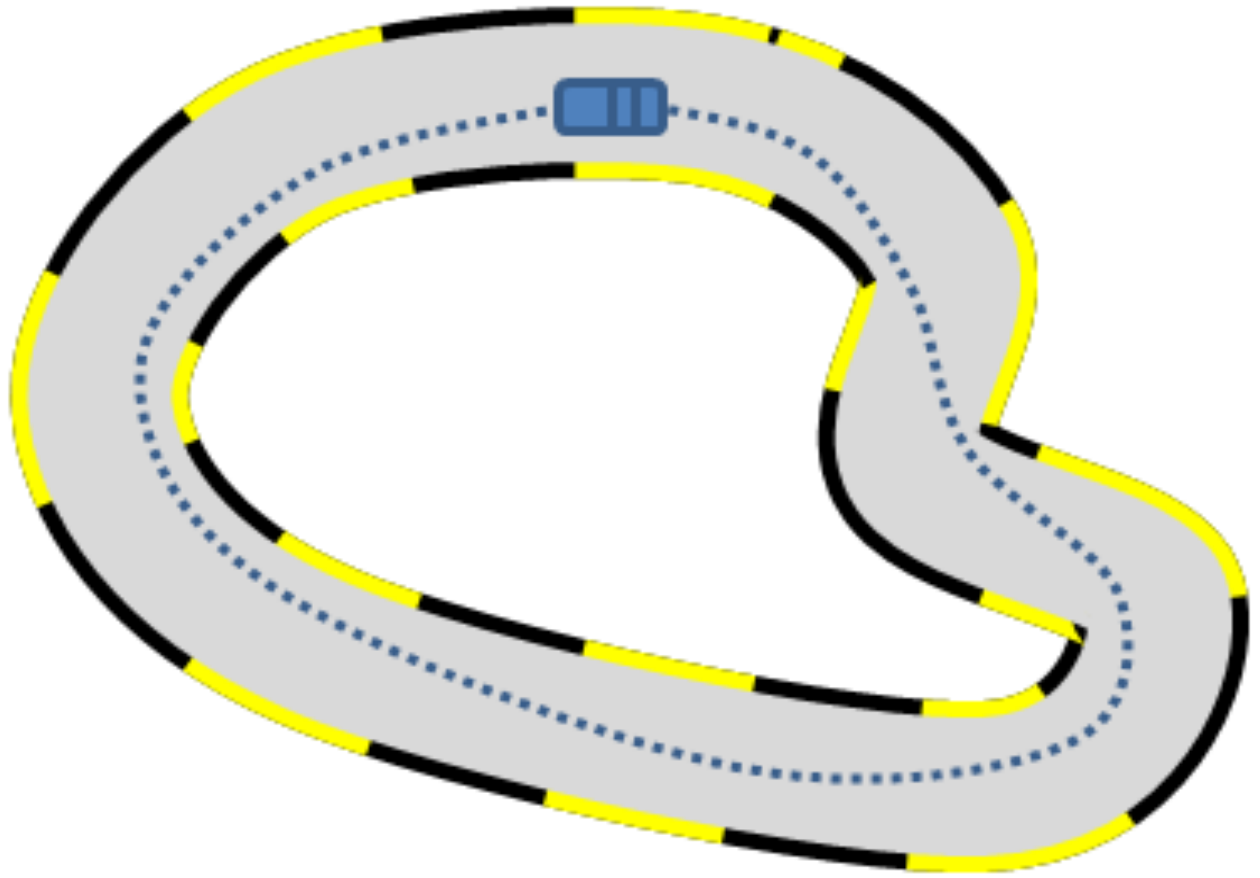
Output:

Steering Angle
in [-1, 1]

[Widrow64,Pomerleau89]

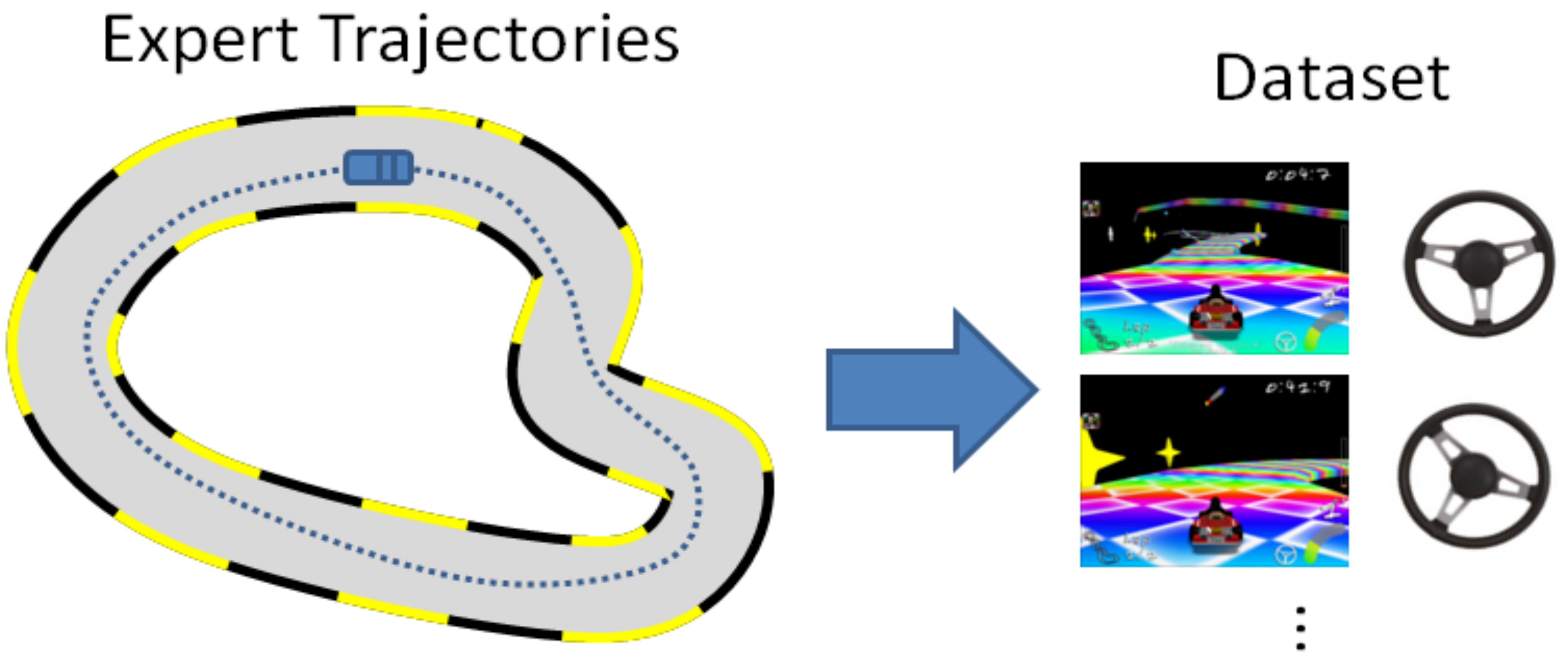
Supervised Learning Approach: Behavior Cloning

Expert Trajectories



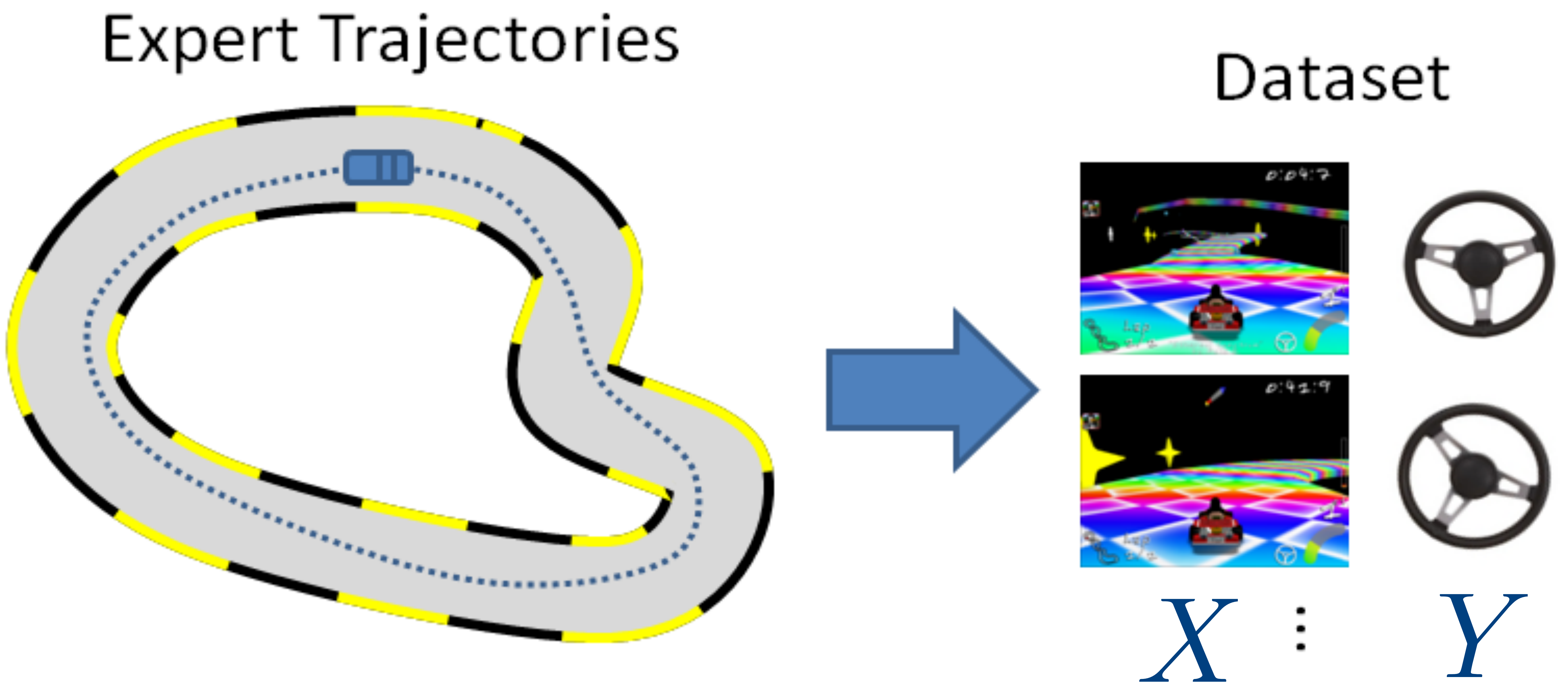
[Widrow64,Pomerleau89]

Supervised Learning Approach: Behavior Cloning



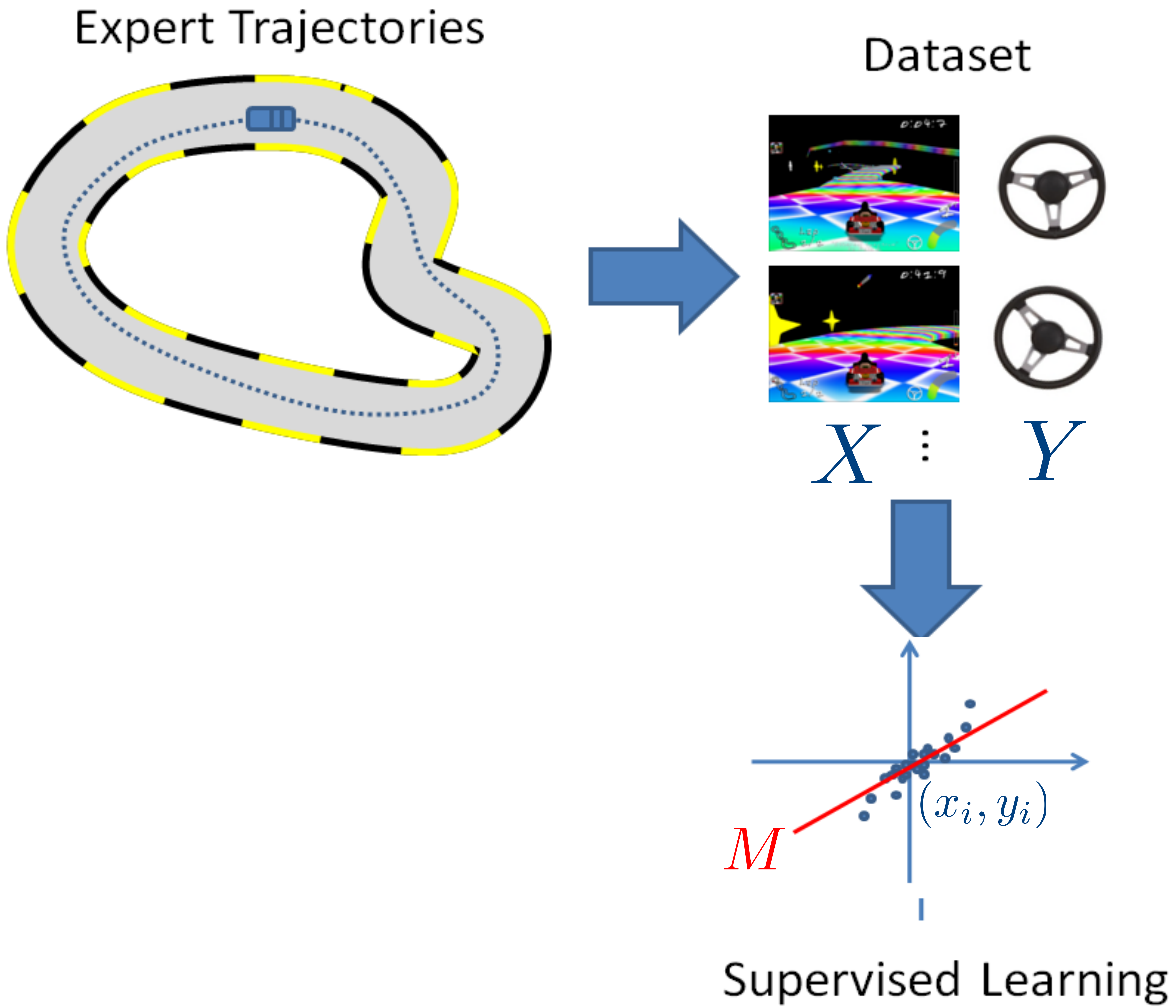
[Widrow64,Pomerleau89]

Supervised Learning Approach: Behavior Cloning



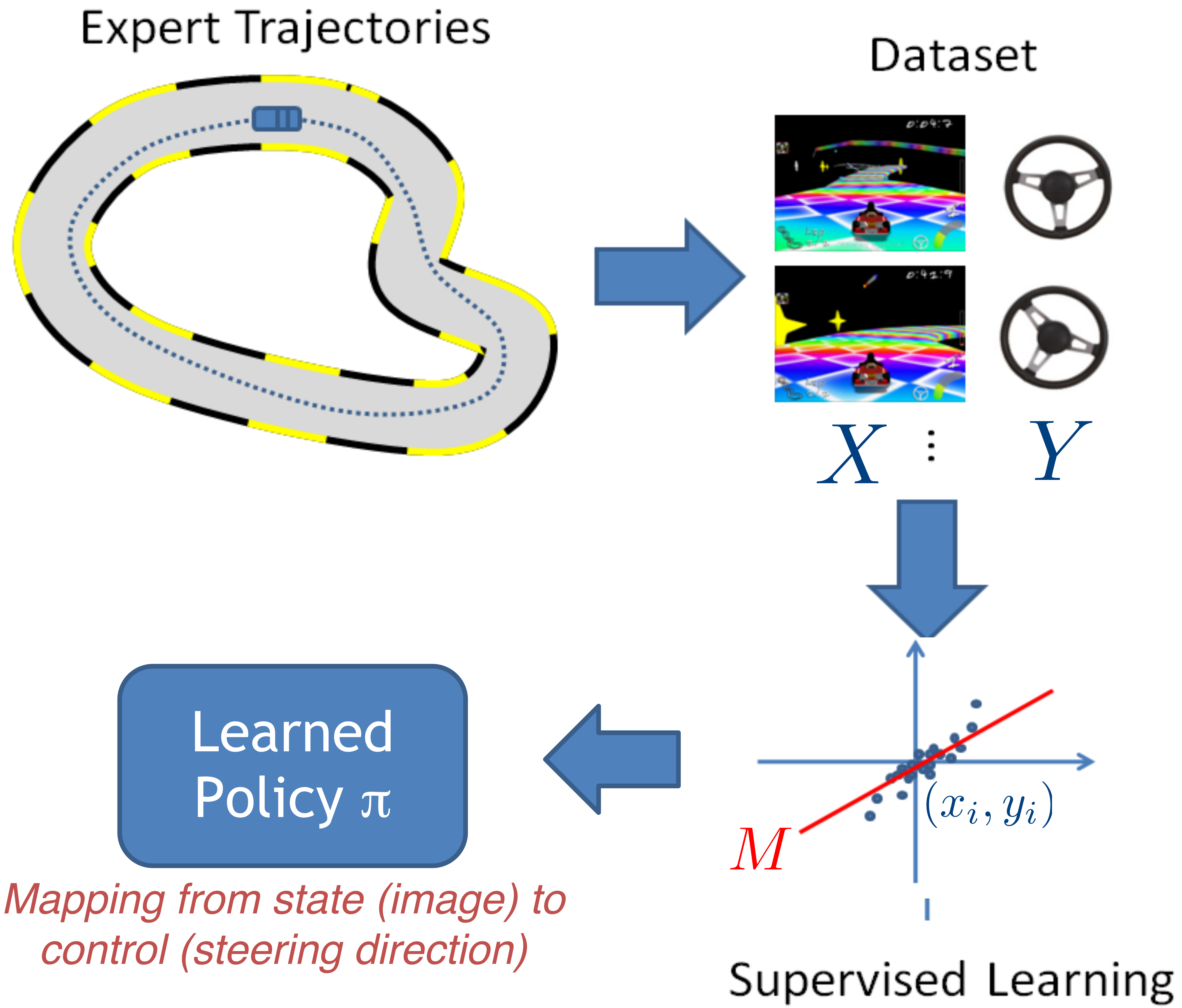
[Widrow64,Pomerleau89]

Supervised Learning Approach: Behavior Cloning



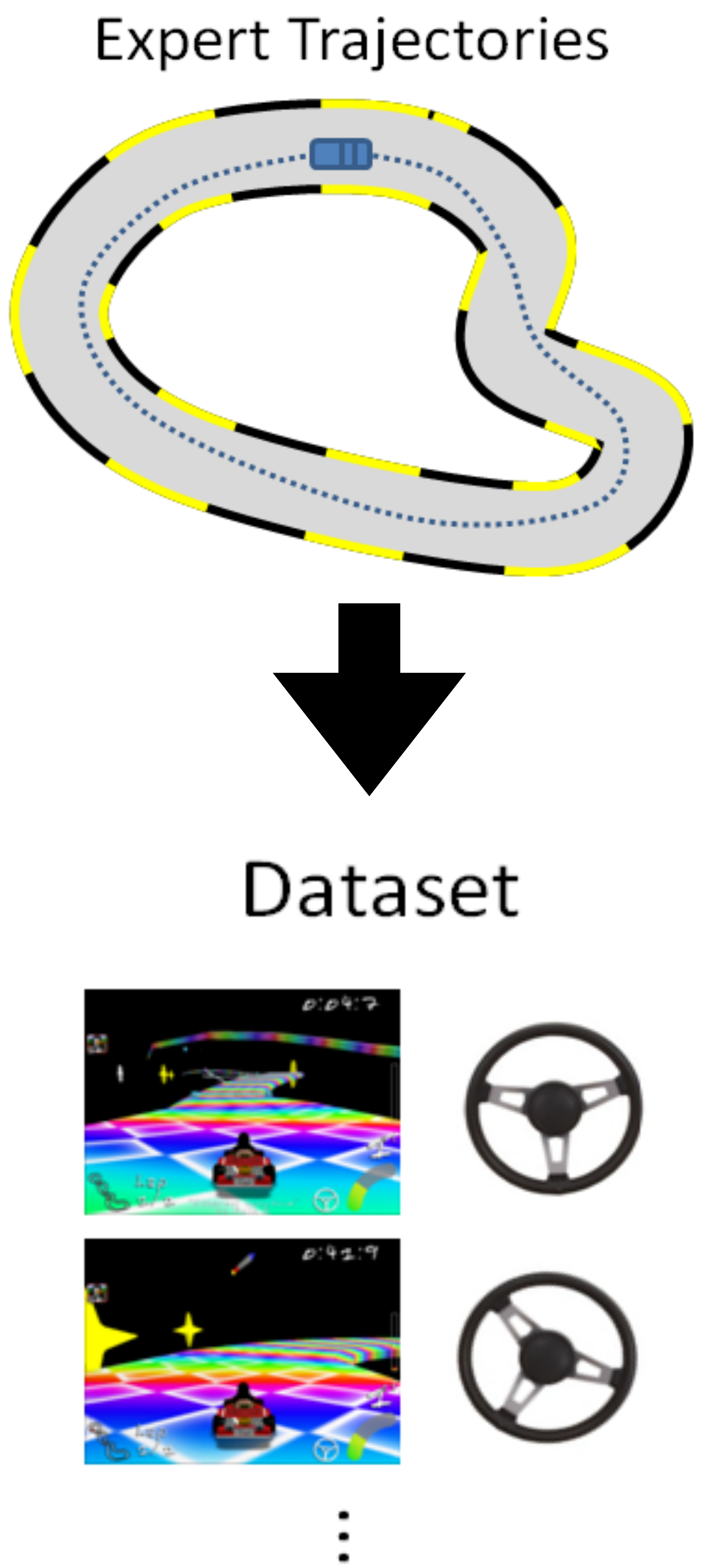
[Widrow64,Pomerleau89]

Supervised Learning Approach: Behavior Cloning



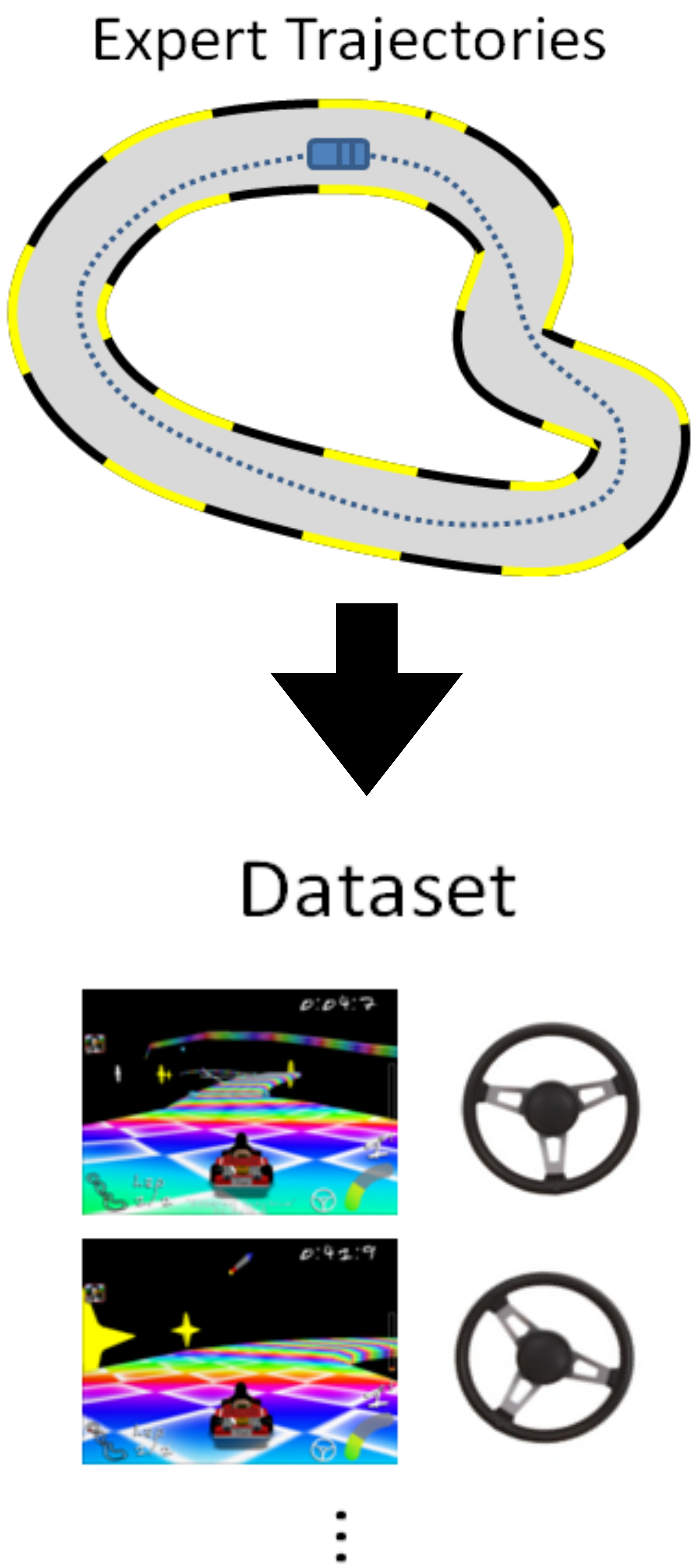
Mapping from state (image) to control (steering direction)

Let’s formalize the offline IL Setting and the Behavior Cloning algorithm



Finite horizon MDP $\mathscr{M}$

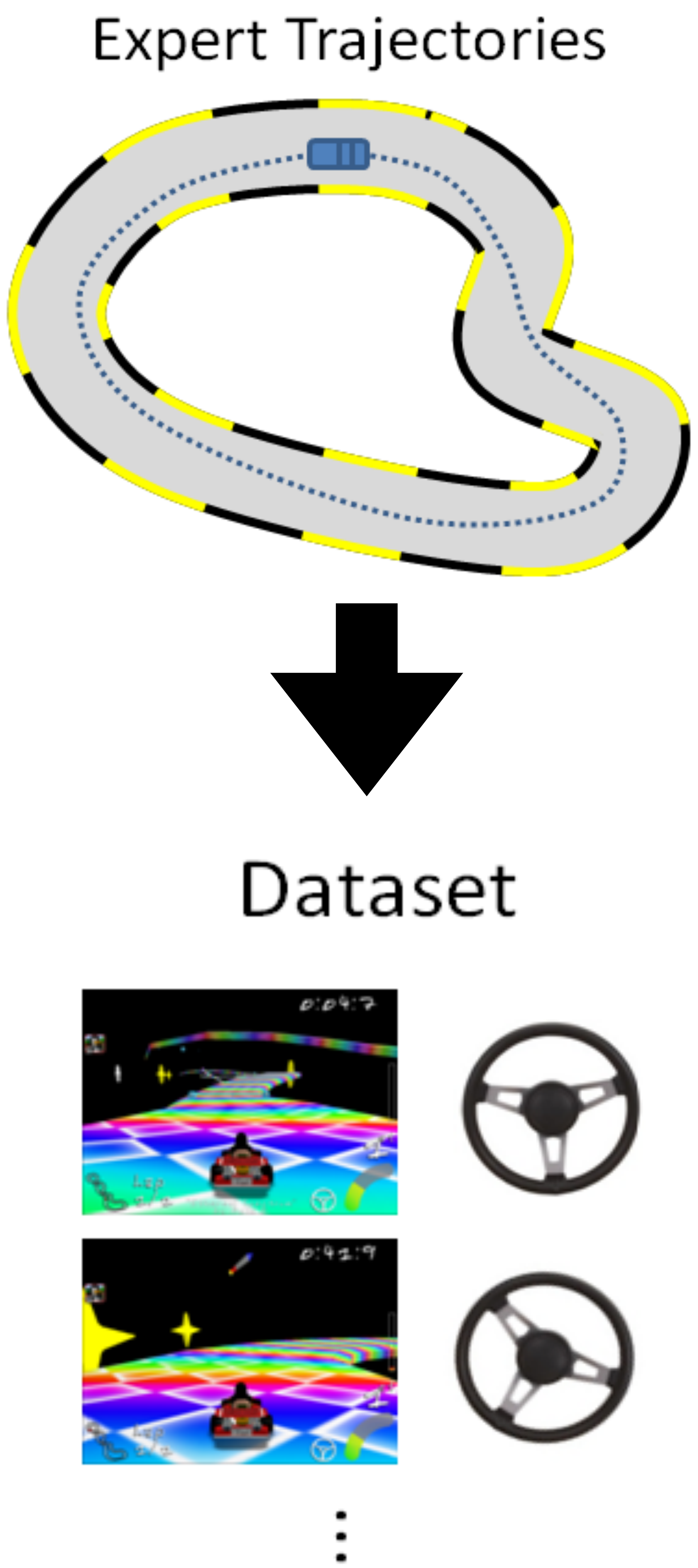
Let's formalize the offline IL Setting and the Behavior Cloning algorithm



Finite horizon MDP $\mathscr{M}$

Ground truth reward $r(s, a) \in [0,1]$ is unknown;
Assume the expert has a good policy $\pi^\star$ (not necessarily opt)

Let's formalize the offline IL Setting and the Behavior Cloning algorithm

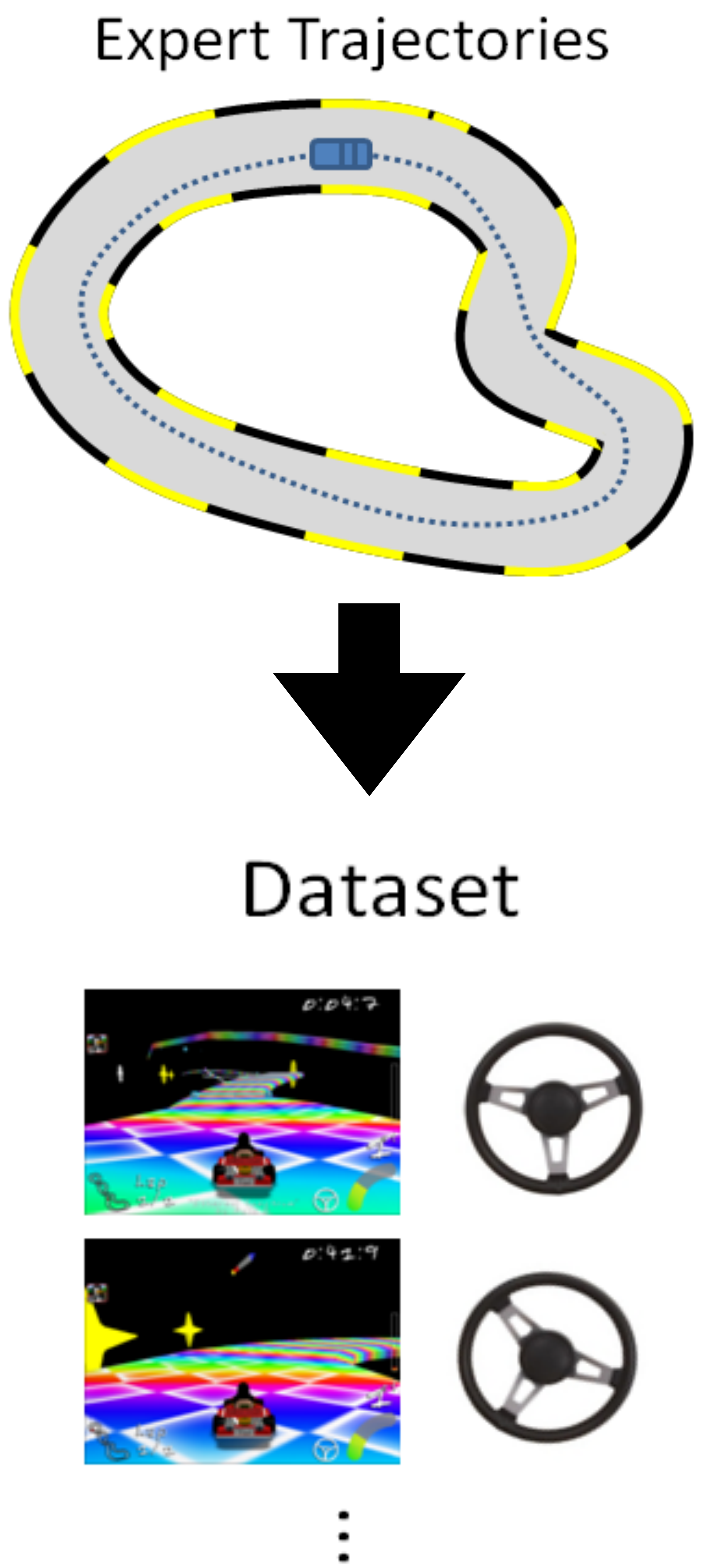


Finite horizon MDP $\mathscr{M}$

Ground truth reward $r(s, a) \in [0,1]$ is unknown;
Assume the expert has a good policy $\pi^\star$ (not necessarily opt)

We have a dataset of M trajectories: $\mathscr{D} = \{\tau_1, \ldots \tau_M\}$,
where $\tau_i = (s_h^i, a_h^i)_{h=0}^{H-1} \sim \rho_{\pi^\star}$

Let's formalize the offline IL Setting and the Behavior Cloning algorithm



Finite horizon MDP $\mathscr{M}$

Ground truth reward $r(s, a) \in [0,1]$ is unknown;
Assume the expert has a good policy $\pi^\star$ (not necessarily opt)

We have a dataset of M trajectories: $\mathscr{D} = \{\tau_1, \ldots \tau_M\}$,
where $\tau_i = (s_h^i, a_h^i)_{h=0}^{H-1} \sim \rho_{\pi^\star}$

Goal: learn a policy from $\mathscr{D}$ that is as good as the expert $\pi^\star$

Let's formalize the Behavior Cloning algorithm

BC Algorithm input: a restricted policy class $\Pi = \{\pi : S \mapsto \Delta(A)\}$

Let's formalize the Behavior Cloning algorithm

BC Algorithm input: a restricted policy class $\Pi = \{\pi : S \mapsto \Delta(A)\}$

BC is a Reduction to Supervised Learning:

Let's formalize the Behavior Cloning algorithm

BC Algorithm input: a restricted policy class $\Pi = \{\pi : S \mapsto \Delta(A)\}$

BC is a Reduction to Supervised Learning:

$$\widehat{\pi} = \arg\min_{\pi \in \Pi} \sum_{i=1}^{M} \sum_{h=0}^{H-1} \ell\left(\pi, s_h^i, a_h^i\right)$$

Let's formalize the Behavior Cloning algorithm

BC Algorithm input: a restricted policy class $\Pi = \{\pi : S \mapsto \Delta(A)\}$

BC is a Reduction to Supervised Learning:

$$\widehat{\pi} = \arg\min_{\pi \in \Pi} \sum_{i=1}^{M} \sum_{h=0}^{H-1} \ell\left(\pi, s_h^i, a_h^i\right)$$

Many choices of loss functions:

Let's formalize the Behavior Cloning algorithm

BC Algorithm input: a restricted policy class $\Pi = \{\pi : S \mapsto \Delta(A)\}$

BC is a Reduction to Supervised Learning:

$$\widehat{\pi} = \arg\min_{\pi \in \Pi} \sum_{i=1}^{M} \sum_{h=0}^{H-1} \ell\left(\pi, s_h^i, a_h^i\right)$$

Many choices of loss functions:

1. Classification (0/1) loss: $\mathbf{1}[\pi(s) \neq a]$

Let's formalize the Behavior Cloning algorithm

BC Algorithm input: a restricted policy class $\Pi = \{\pi : S \mapsto \Delta(A)\}$

BC is a Reduction to Supervised Learning:

$$\widehat{\pi} = \arg\min_{\pi \in \Pi} \sum_{i=1}^{M} \sum_{h=0}^{H-1} \ell\left(\pi, s_h^i, a_h^i\right)$$

Many choices of loss functions:

1. Classification (0/1) loss: $\mathbf{1}[\pi(s) \neq a]$
2. Negative log-likelihood (NLL): $\ell(\pi, s, a) = -\ln \pi(a \mid s)$

Let's formalize the Behavior Cloning algorithm

BC Algorithm input: a restricted policy class $\Pi = \{\pi : S \mapsto \Delta(A)\}$

BC is a Reduction to Supervised Learning:

$$\widehat{\pi} = \arg\min_{\pi\in\Pi} \sum_{i=1}^{M} \sum_{h=0}^{H-1} \ell\left(\pi, s_h^i, a_h^i\right)$$

Many choices of loss functions:

1. Classification (0/1) loss: $\mathbf{1}[\pi(s) \neq a]$
2. Negative log-likelihood (NLL): $\ell(\pi, s, a) = -\ln \pi(a \mid s)$
3. square loss (i.e., regression for continuous action): $\ell(\pi, s, a) = \|\pi(s) - a\|_2^2$

Summary:

1. Importance sampling enables sample-based optimization in RL
2. Policy gradient methods are great and work well in practice, but can suffer from lack of exploration

Attendance:
bit.ly/3RcTC9T

Feedback:
bit.ly/3RHtlxy