Scaling Offline RL via Efficient and Expressive Shortcut Models

Nicolas Espinosa Dice

Joint work with

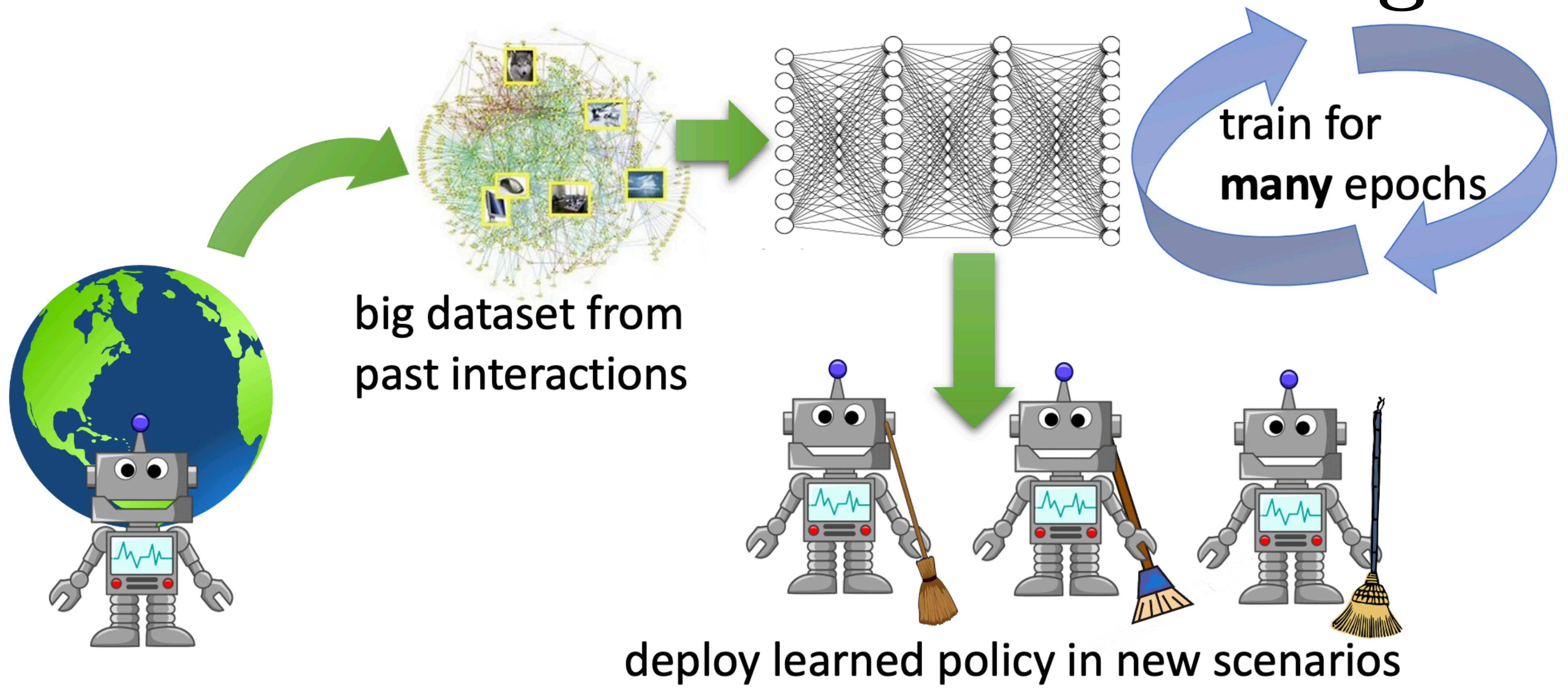
Yiyi Zhang, Yiding Chen, Bradley Guo, Owen Oertell, Gokul Swamy, Kianté Brantley, Wen Sun

Cornell Bowers C·IS
College of Computing and Information Science

How do we scale offline reinforcement learning?

Offline reinforcement learning

Axes of scale

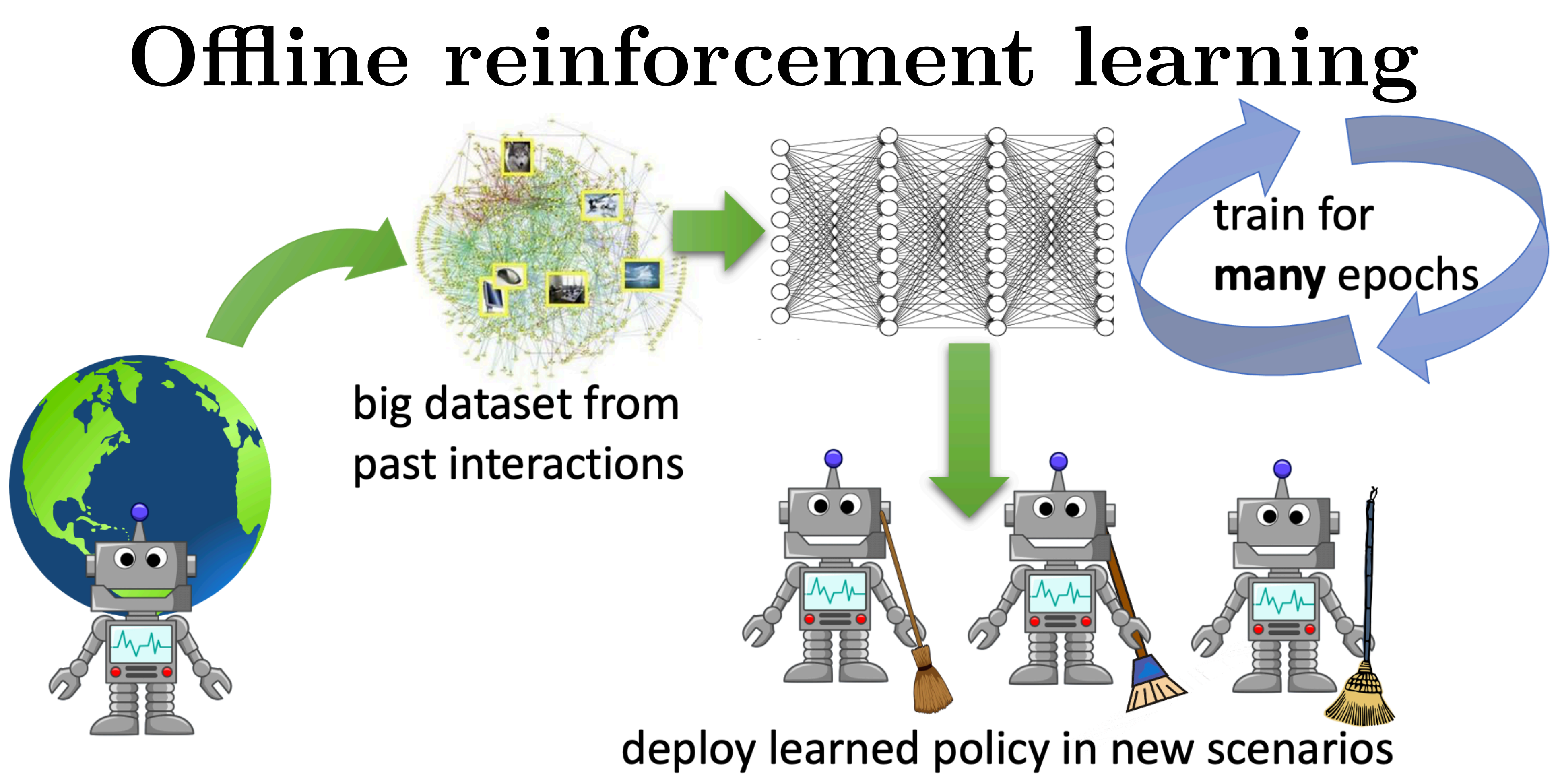


PC: bair.berkeley.edu/blog/2020/12/07/offline/

How do we scale offline reinforcement learning?

Axes of scale

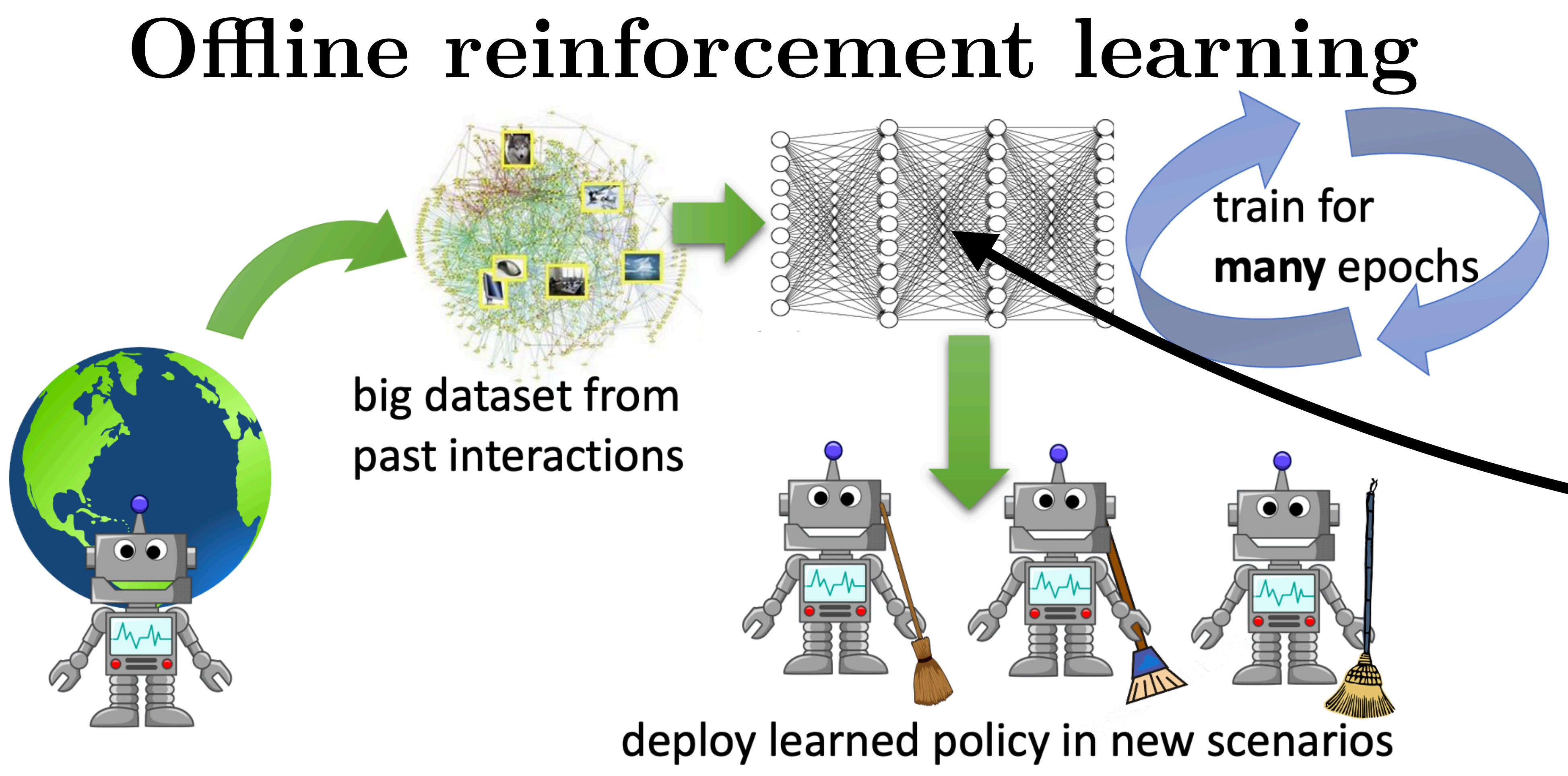
1. **Data**



How do we scale offline reinforcement learning?

Axes of scale

1. Data
2. **Models**



Represent policy via modern generative models (e.g. diffusion & flow models)

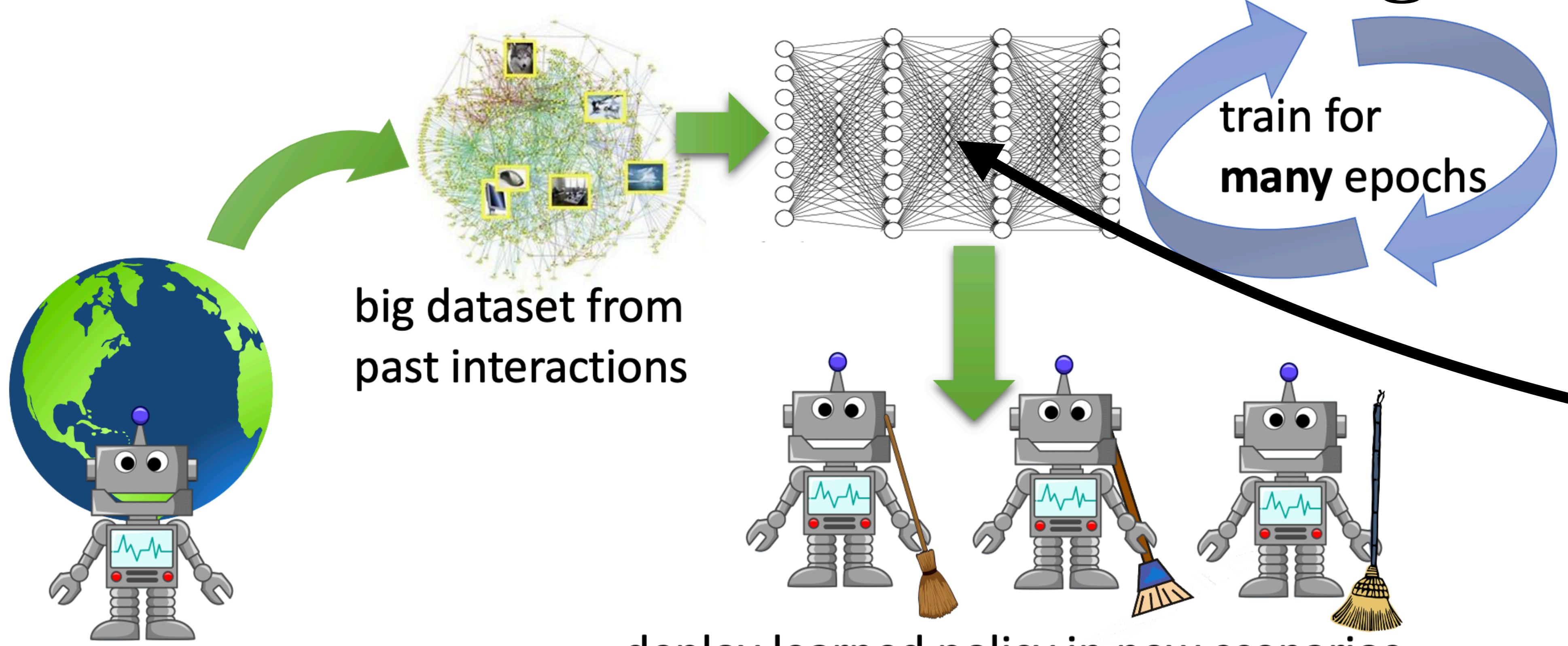
PC: bair.berkeley.edu/blog/2020/12/07/offline/

How do we scale offline reinforcement learning?

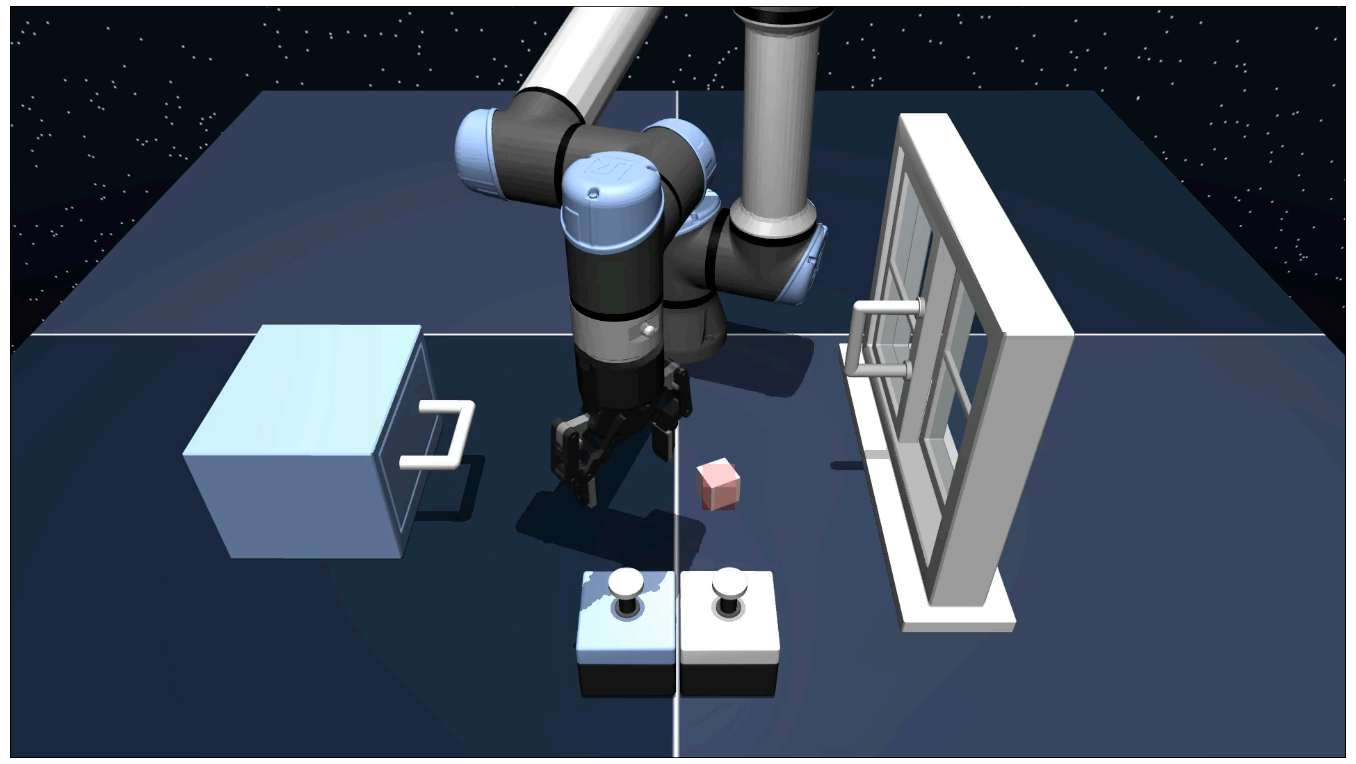
Axes of scale

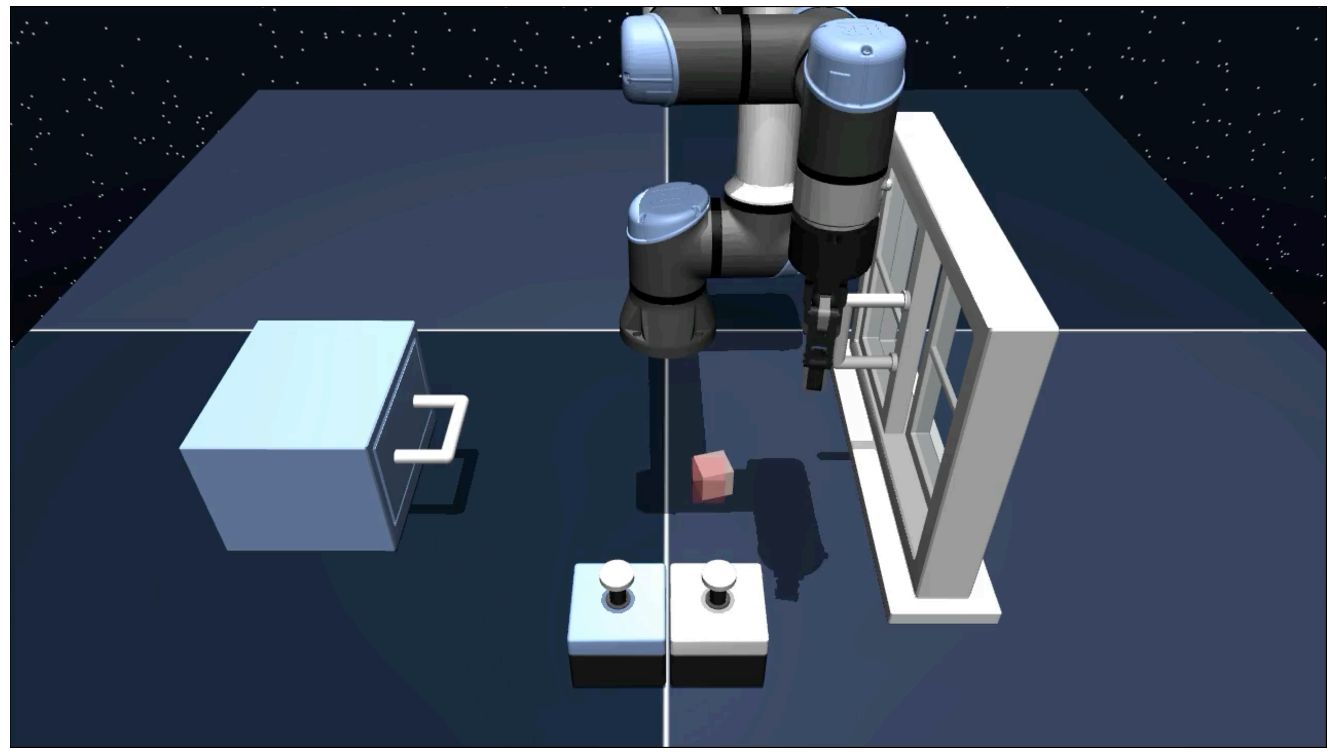
1. Data
2. **Models**

Offline reinforcement learning



Represent policy via modern generative models (e.g. diffusion & flow models)





Advantages: model multi-modal data

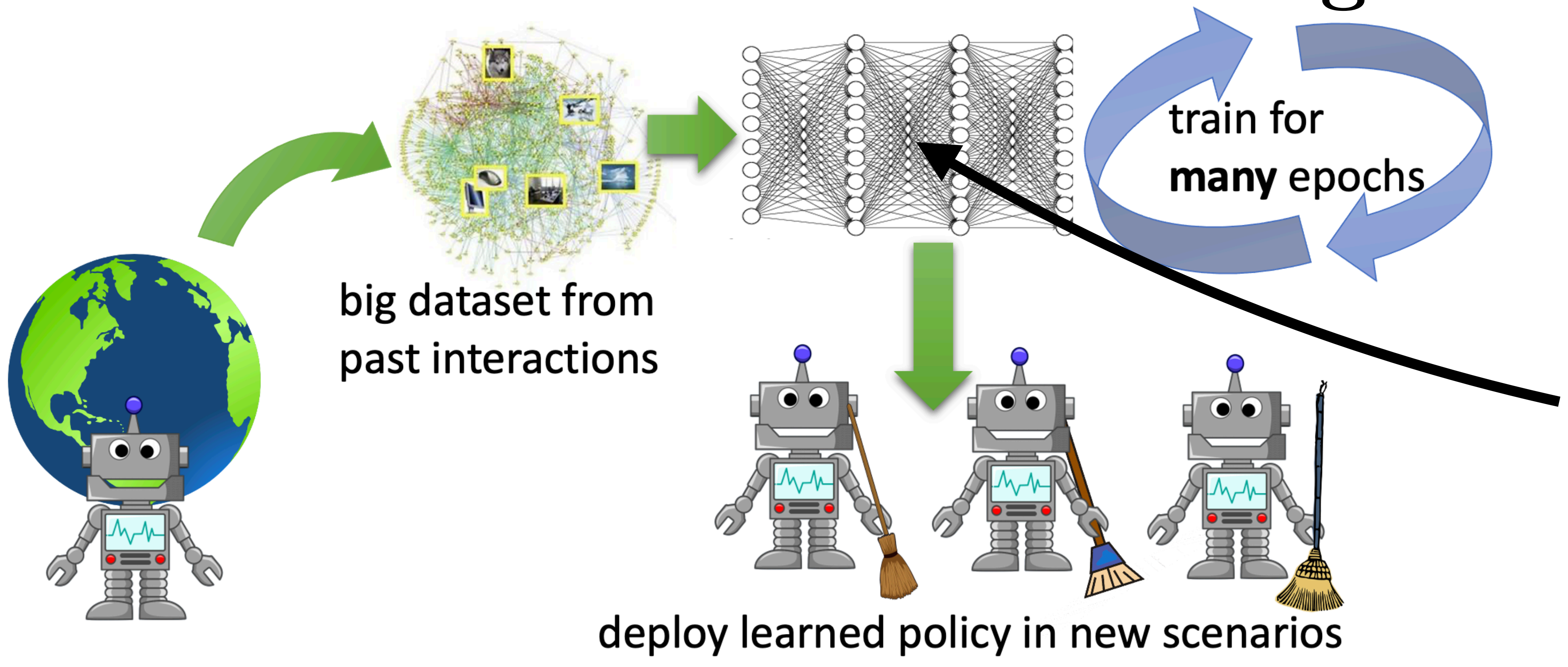
PC: bair.berkeley.edu/blog/2020/12/07/offline/

How do we scale offline reinforcement learning?

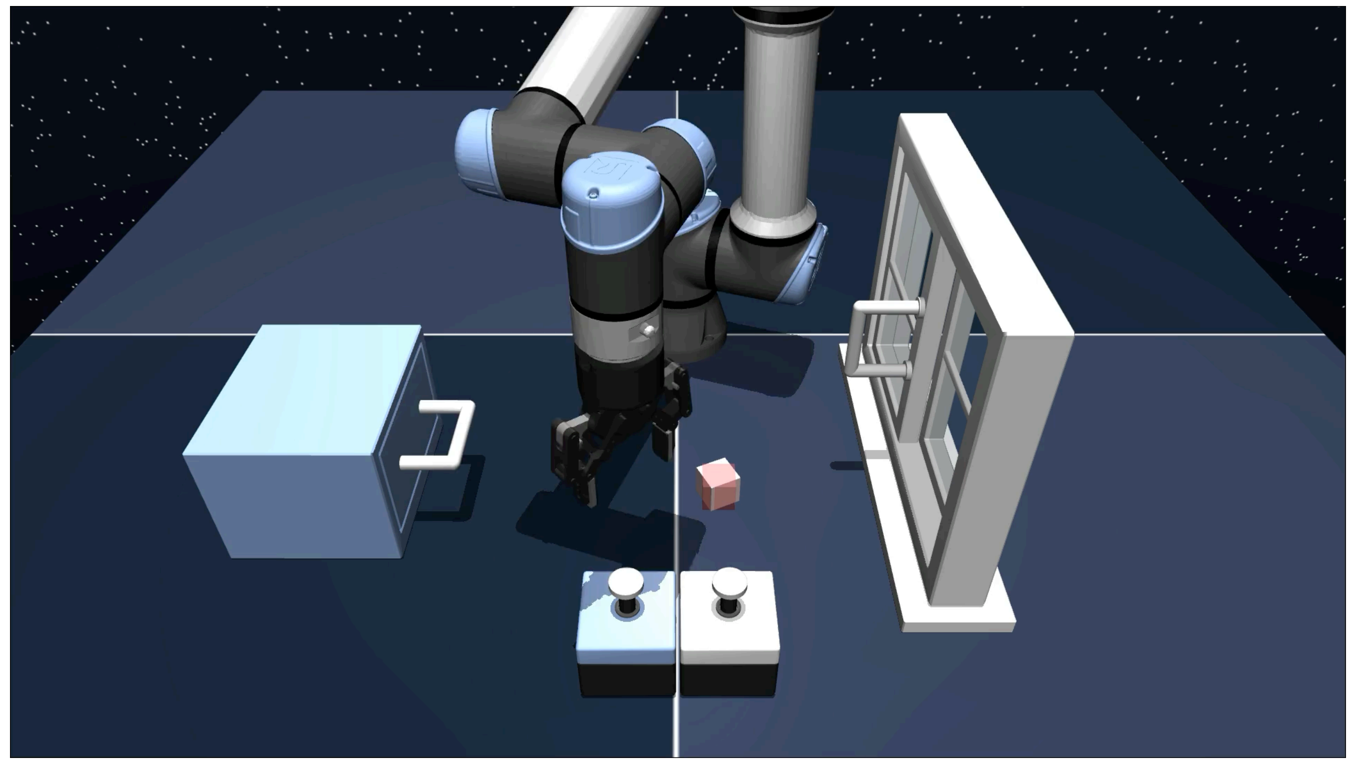
Axes of scale

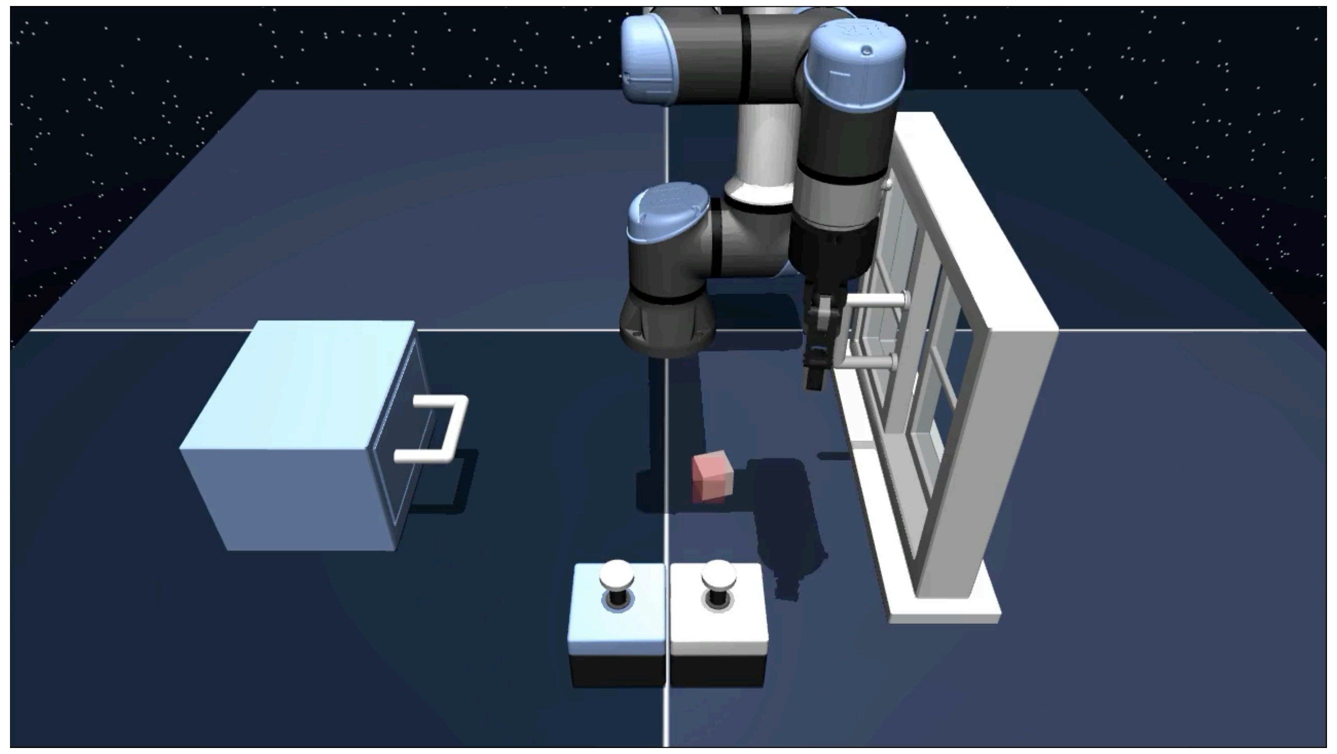
1. Data
2. Models
3. **Compute**

Offline reinforcement learning



Represent policy via modern generative models (e.g. diffusion & flow models)



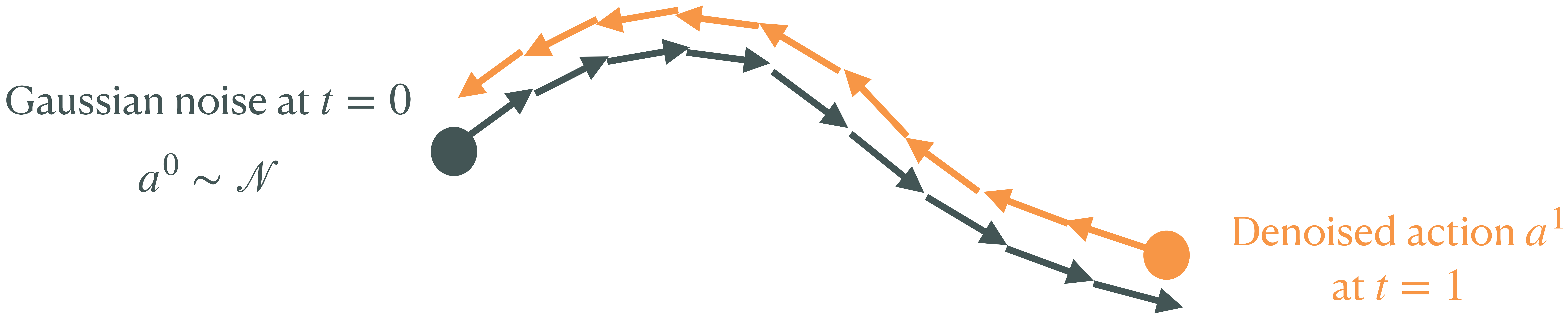


Advantages: model multi-modal data

PC: bair.berkeley.edu/blog/2020/12/07/offline/

RL with diffusion/flow policies

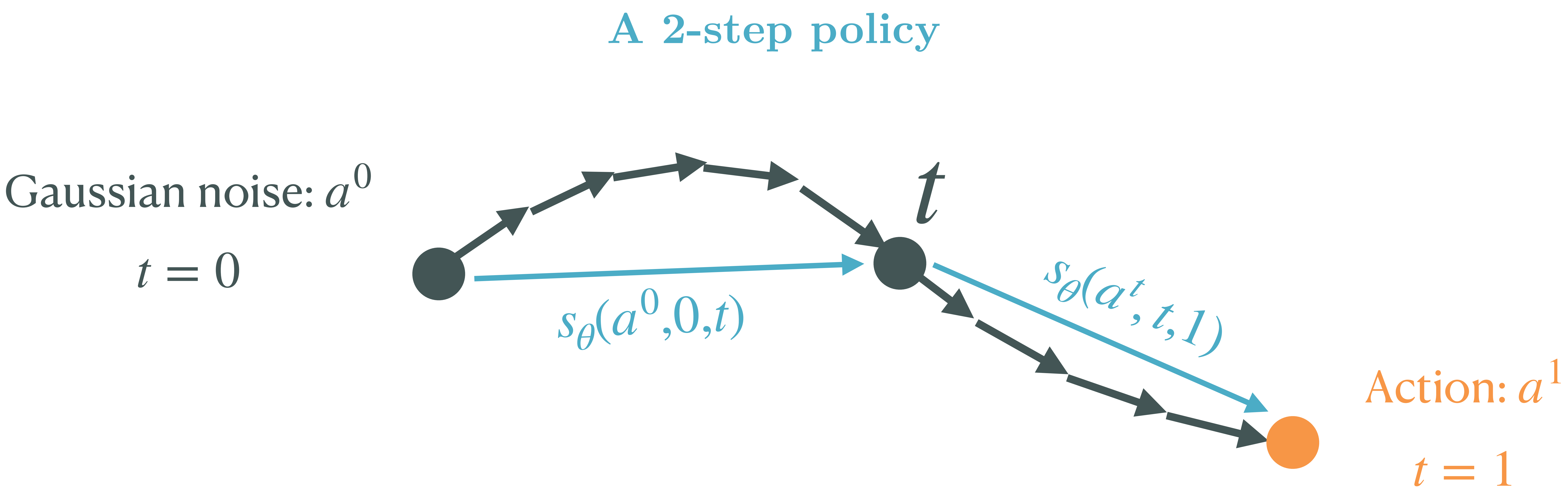
Flow policy generates action $a^1 \sim \pi(\cdot \mid s)$



Given $Q(s, a^1)$, optimize θ via backpropagation through time (BPTT)

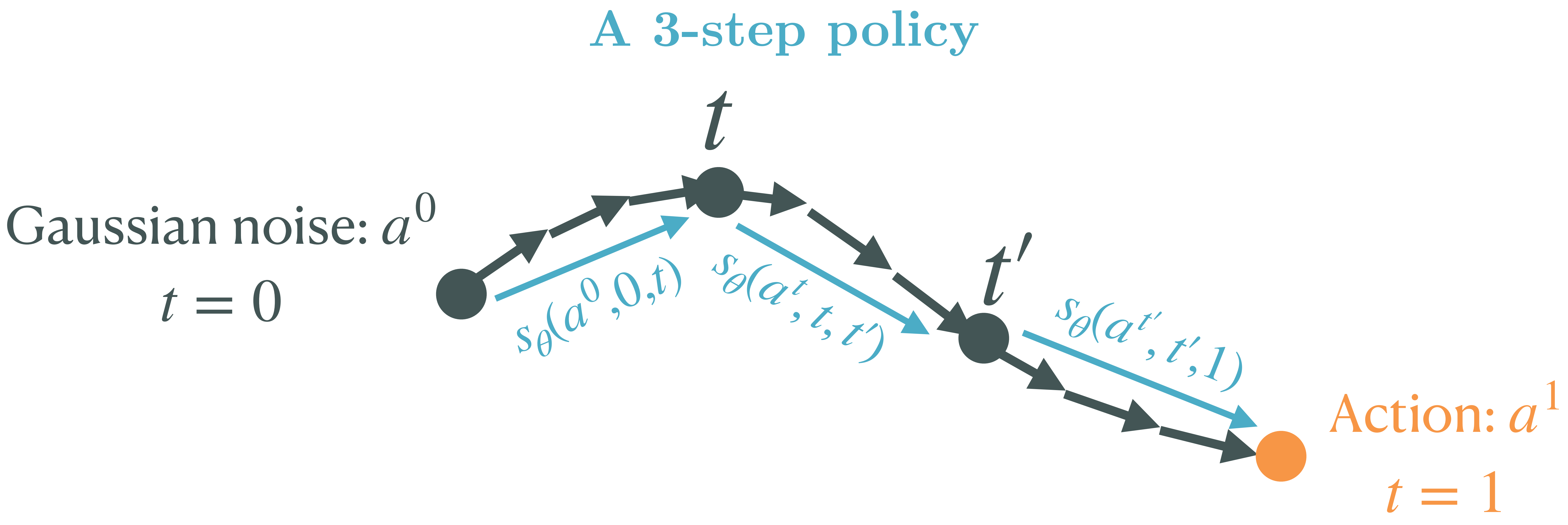
$$\frac{\partial Q}{\partial \theta} = \frac{\partial Q}{\partial a^1} \cdot \frac{\partial a^1}{\partial \theta}$$

Shortcut models: a flow model with "shortcuts"



s_θ: a differentiable function modeling a ***jump*** for some time interval t

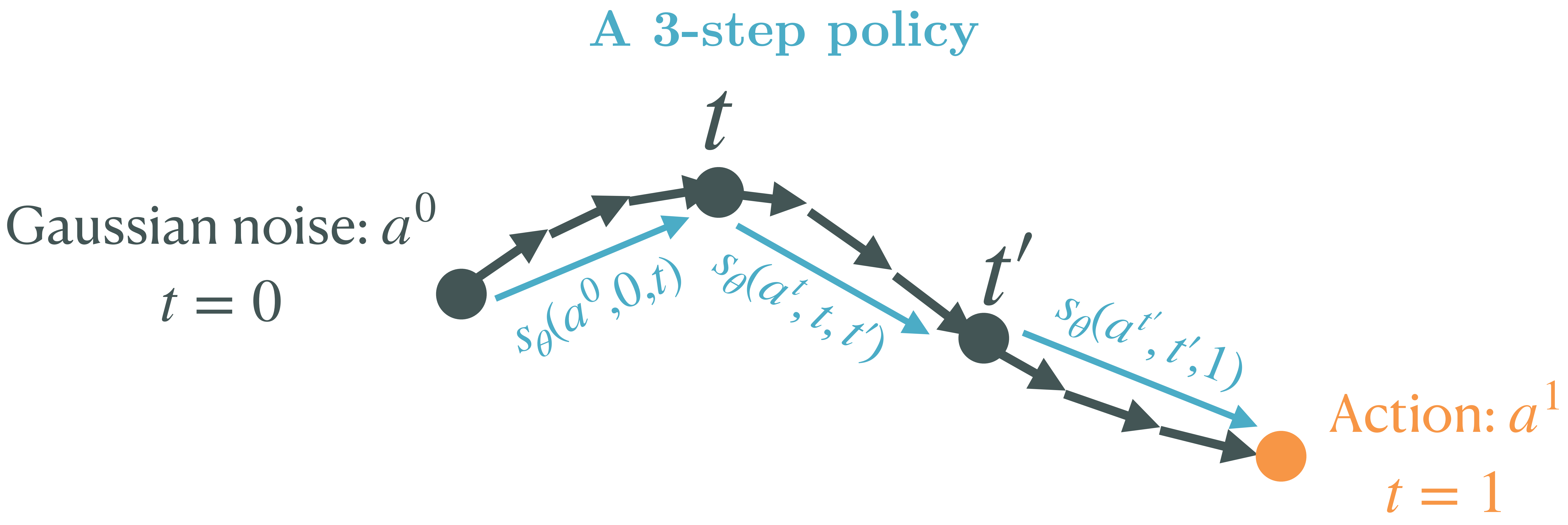
Shortcut models: a flow model with "shortcuts"



Key Advantage:

Unlike flow models, shortcut models are expressive with just a few steps

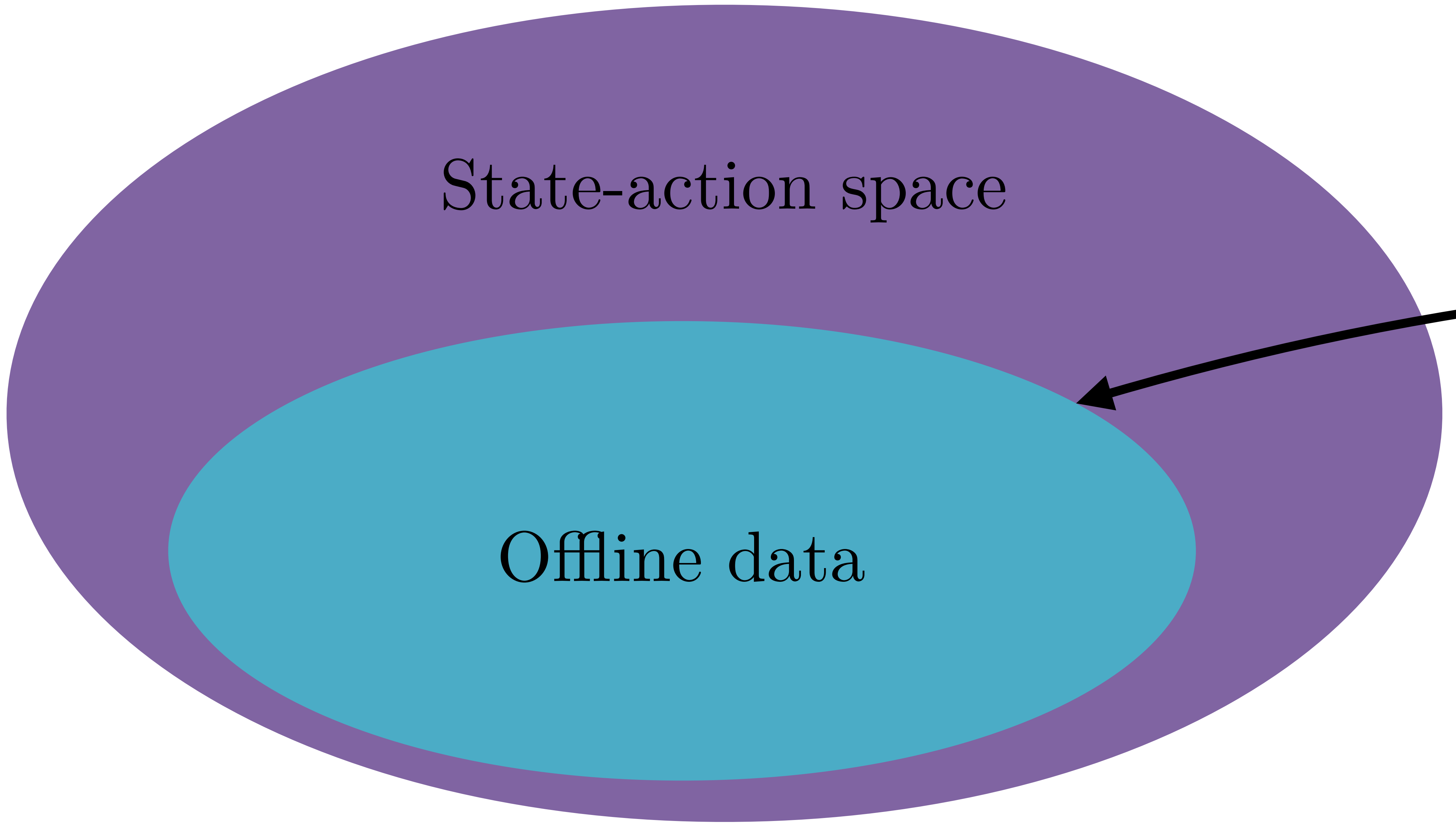
Optimizing shortcut policies



Given $Q(s, a^1)$, optimize θ via backpropagation k-steps

$$\frac{\partial Q}{\partial \theta} = \frac{\partial Q}{\partial a^1} \cdot \frac{\partial a^1}{\partial \theta}$$

Training a shortcut policy in offline RL setting



Key Challenge

Training with offline data alone easily runs into covariate shift problems

Goal

Design algorithms that restrict search space inside the offline data's coverage

Solution

Regularize to the offline data

How do we regularize to the offline data?

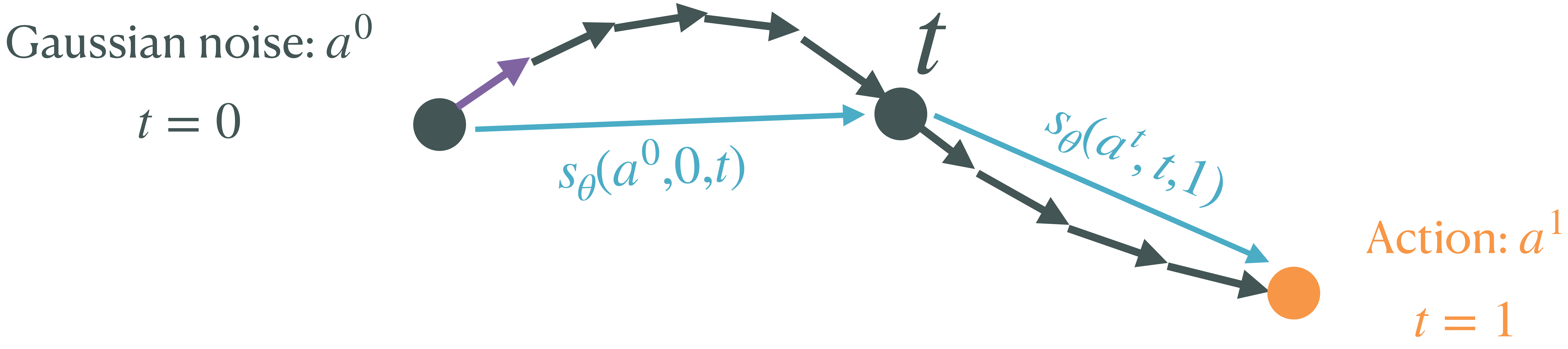
Learned policy

Behavior policy

Key idea

Match distributions of π_θ and π_B through *flow matching*

$\rightarrow$ match velocity of π_B's flow



When $t \rightarrow 0$, s_θ models

instantaneous velocity of generator's flow

How do we regularize to the offline data?

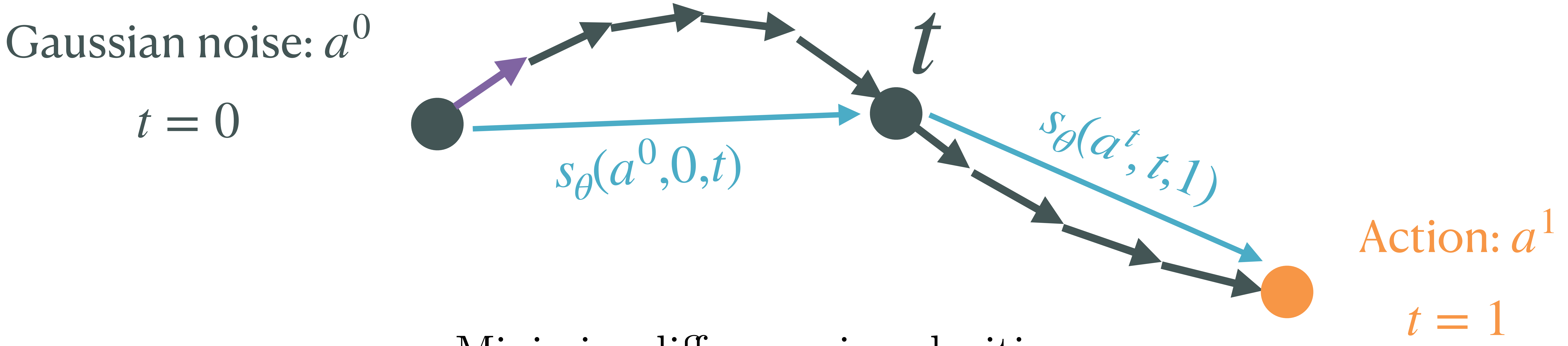
Learned policy

Behavior policy

Key idea

Match distributions of π_θ and π_B through *flow matching*

$\rightarrow$ match velocity of π_B's flow



Minimize difference in velocities

$$\min_\theta \| \underbrace{s_\theta(a_t, t, t+\Delta)}_{\text{flow at noised action } a_t} - \underbrace{\text{velocity}(\pi_B(\cdot \mid s), t)}_{\text{approximated with } \pi_B \text{ actions}} \|_2$$

Scalable **O**ffline **R**einforcement **L**earning (`SORL`)

1. *Value Learning*: Train Q^{π_n} with TD learning

Scalable **O**ffline **R**einforcement **L**earning (`SORL`)

1. *Value Learning*: Train Q^{π_n} with TD learning

2. *Policy Learning*: Train $\boldsymbol{\pi}_{n+1}$ via the following

Scalable **O**ffline **R**einforcement **L**earning (`SORL`)

1. *Value Learning*: Train Q^{π_n} with TD learning

2. *Policy Learning*: Train π_{n+1} via the following

$$\max_{\pi} \sum_{s \in \mathcal{D}} \mathbb{E}_{\epsilon} Q^{\pi_n}(s, \pi(s, \epsilon)) -$$

Offline data Noise Shortcut model

Scalable **O**ffline **R**einforcement **L**earning (SORL)

1. *Value Learning*: Train Q^{π_n} with TD learning

2. *Policy Learning*: Train π_{n+1} via the following

$$\max_{\pi} \underbrace{\sum_{s \in \mathcal{D}}}_{\text{Offline data}} \underbrace{\mathbb{E}_{\epsilon}}_{\text{Noise}} Q^{\pi_n}(s, \underbrace{\pi(s, \epsilon)}_{\text{Shortcut model}}) - \alpha_1 \underbrace{\mathrm{FlowMatching}(\pi, \pi_B)}_{\text{Regularization to offline data}}$$

Scalable **O**ffline **R**einforcement **L**earning (`SORL`)

1. *Value Learning*: Train Q^{π_n} with TD learning

2. *Policy Learning*: Train π_{n+1} via the following

$$\max_{\pi} \sum_{s \in \mathcal{D}} \mathbb{E}_{\epsilon} Q^{\pi_n}(s, \pi(s, \epsilon)) - \alpha_1 \, \mathrm{FlowMatching}(\pi, \pi_B) - \alpha_2 \, \mathrm{SelfConsistency}(\pi)$$

Offline data

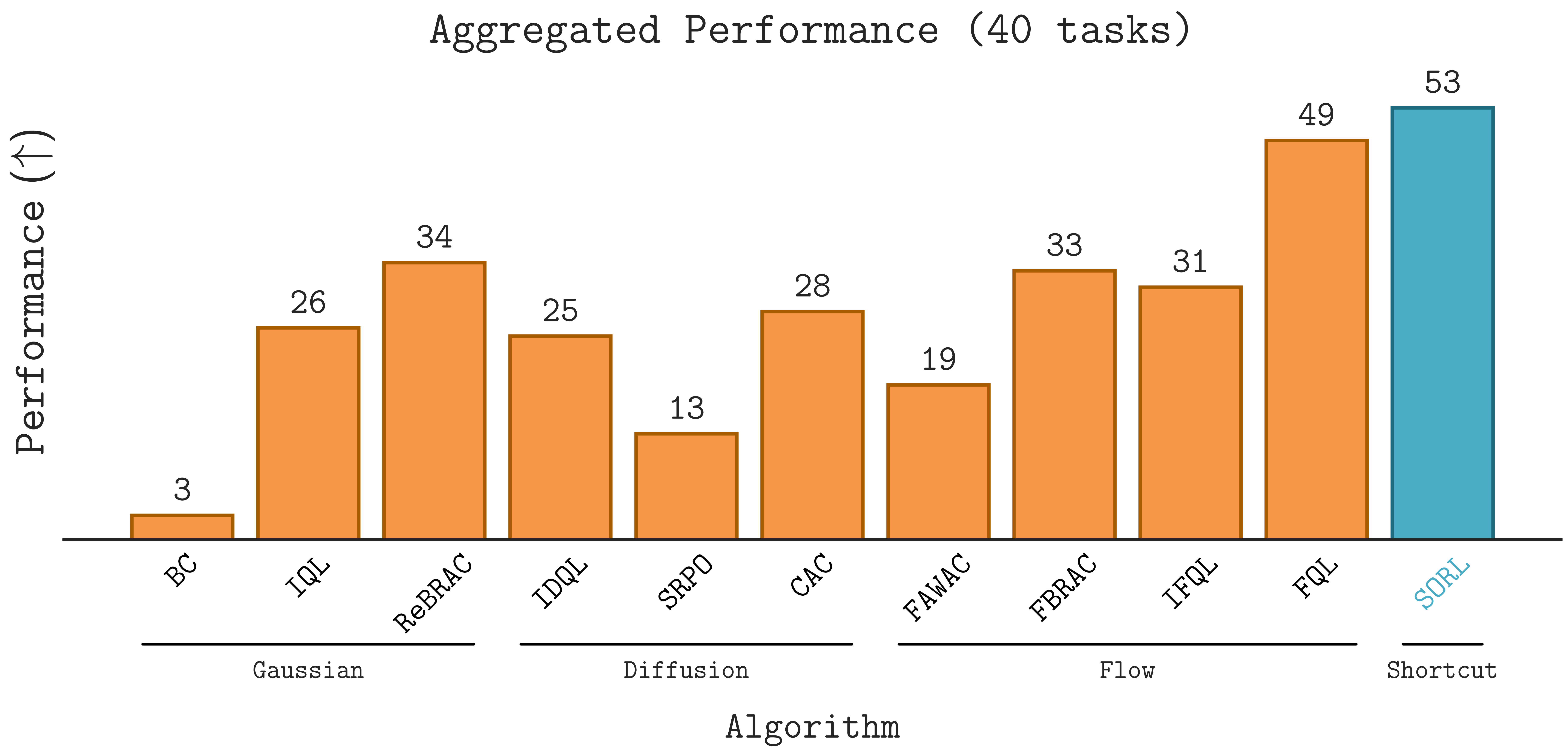
Noise

Shortcut model

Regularization to offline data

Consistency between shortcuts and ODE

SORL outperforms 10 baselines across 40 OGBench tasks



Axes of Scale in Offline RL

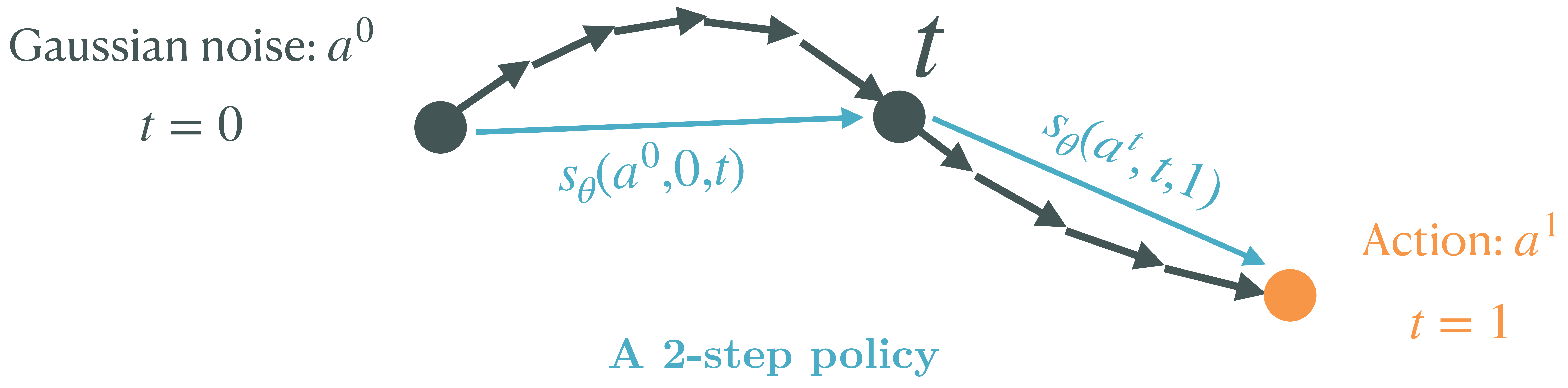
✅ Data

✅ Models

? Compute

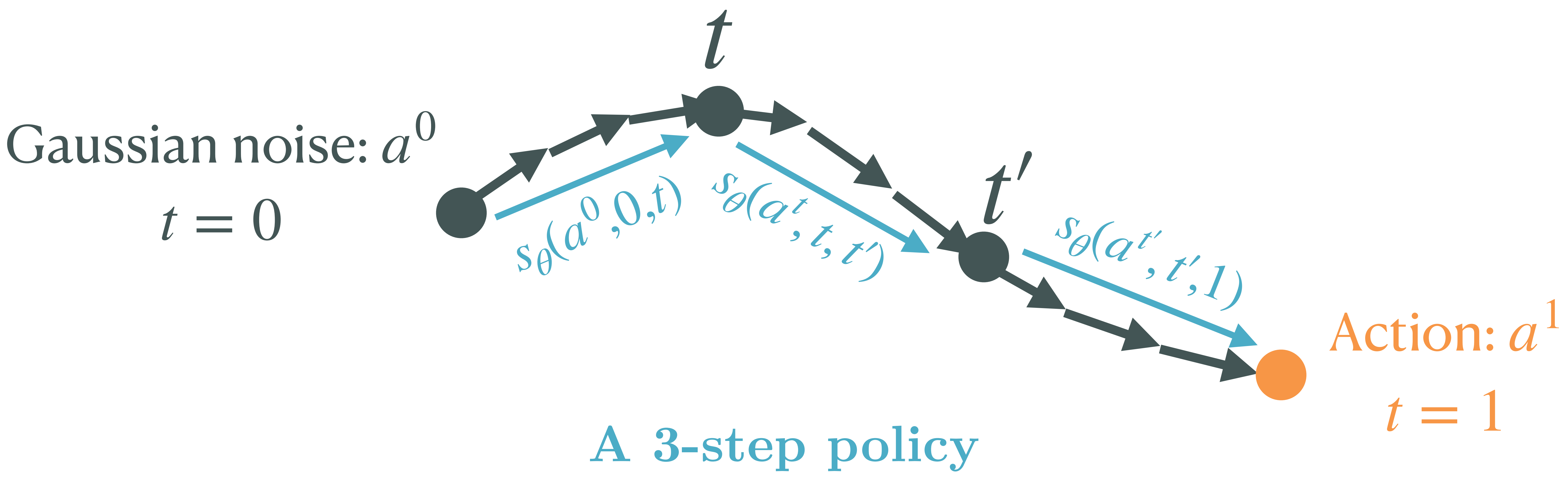
SORL enables test-time *sequential scaling*

At test-time, leverage more computation by using more "jumps"



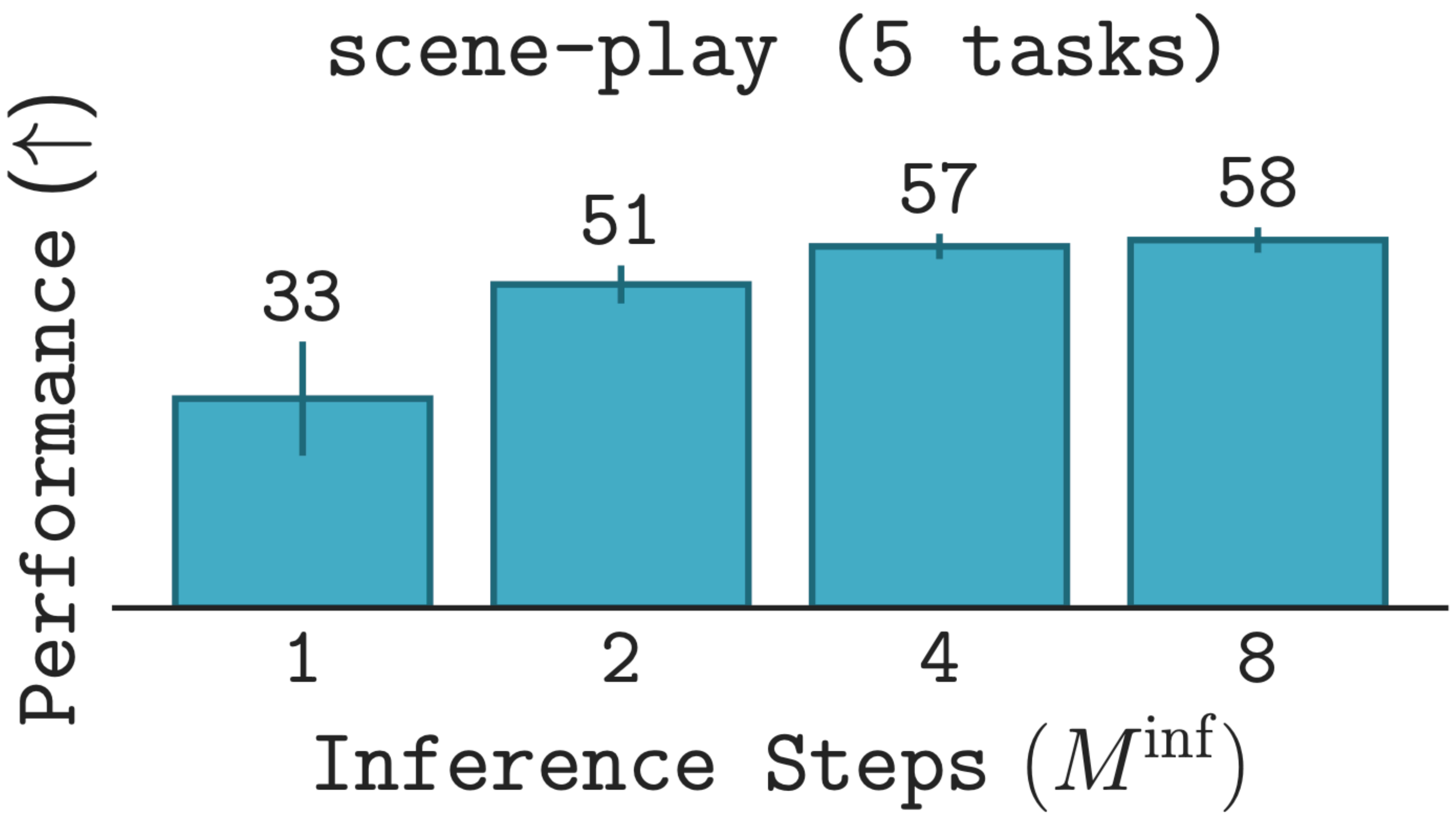
SORL enables test-time *sequential scaling*

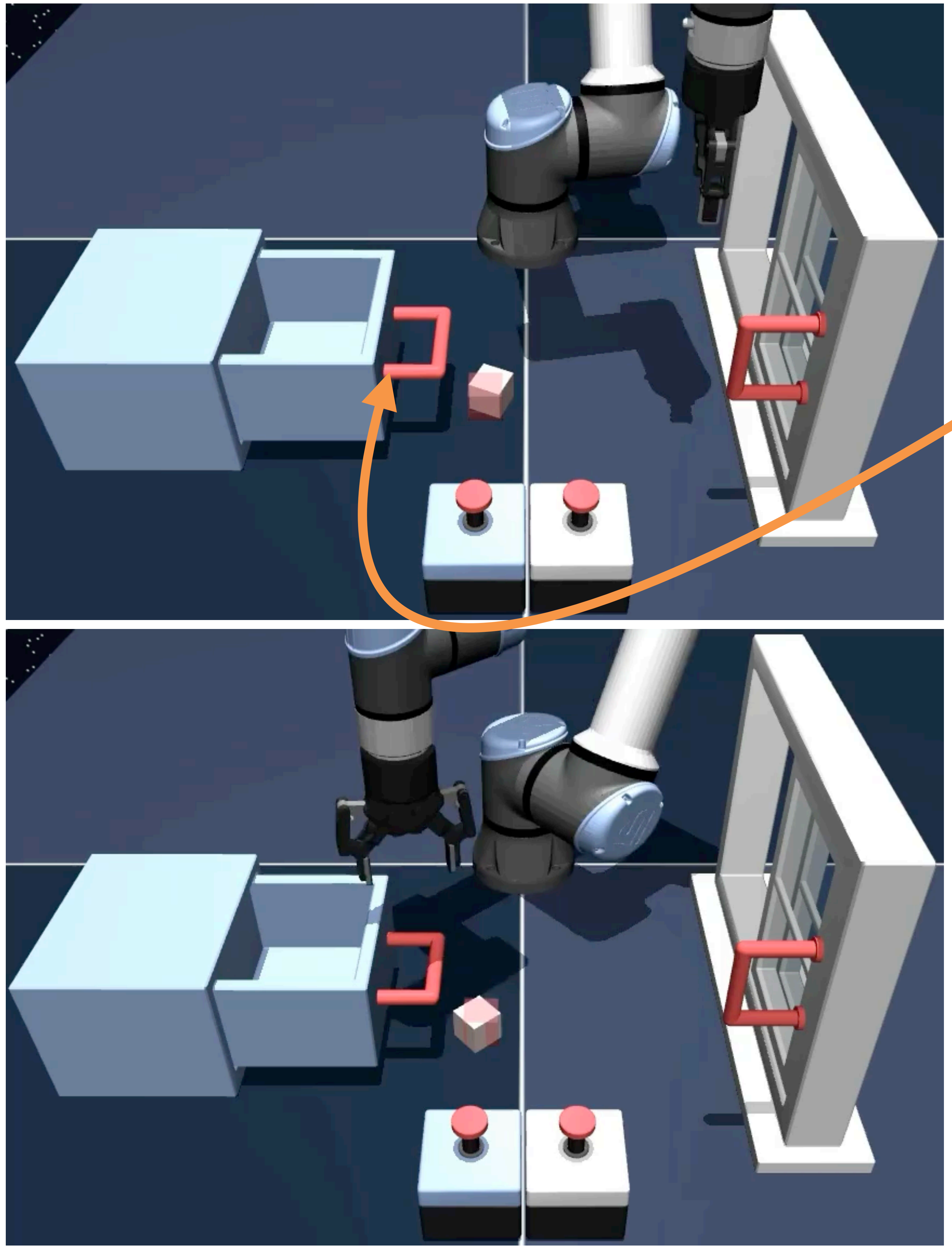
At test-time, leverage more computation by using more "jumps"



Policies with more steps can model richer distributions, enabling better optimization of reward

Q: Is sequential scaling helpful?



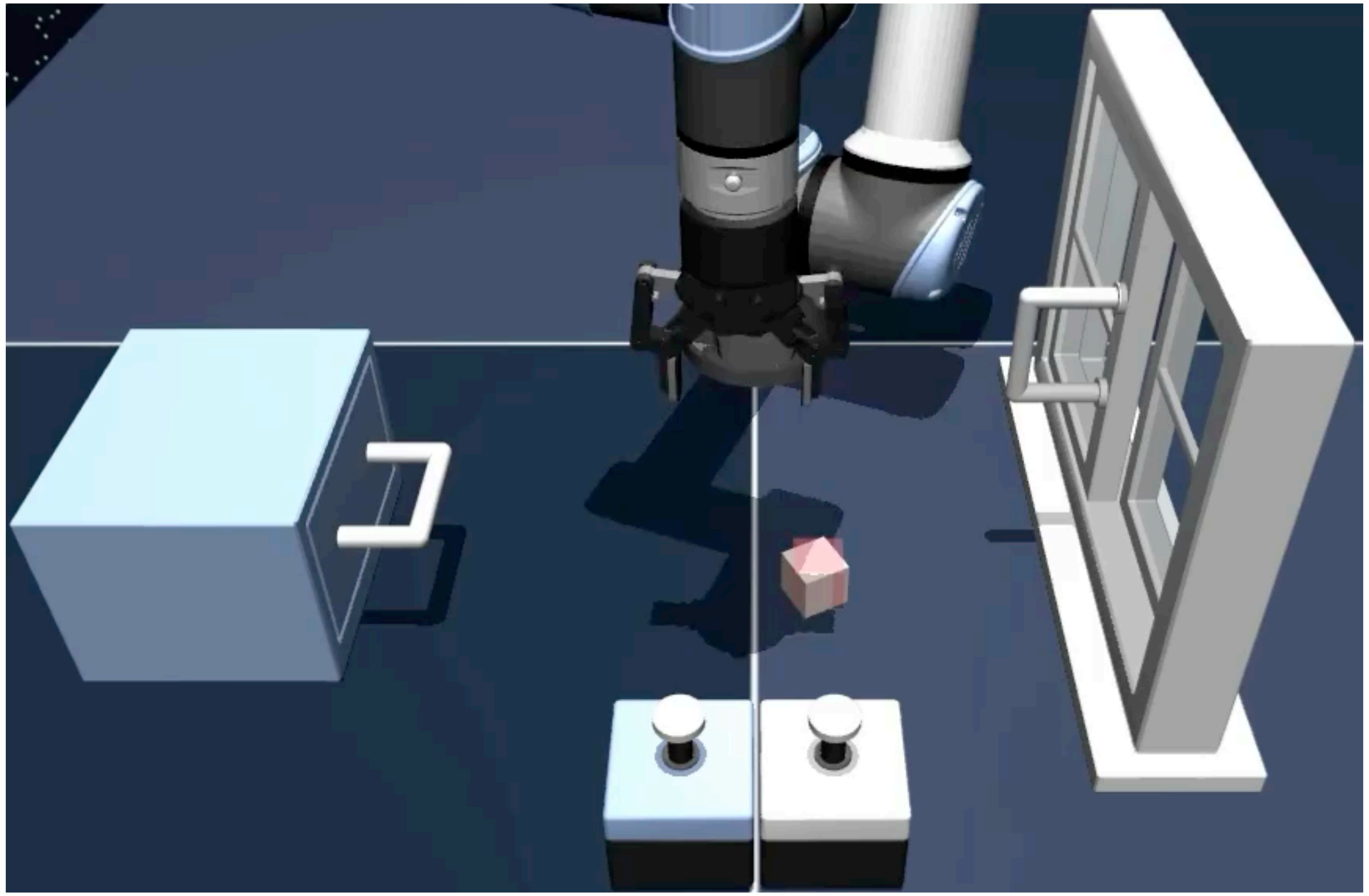


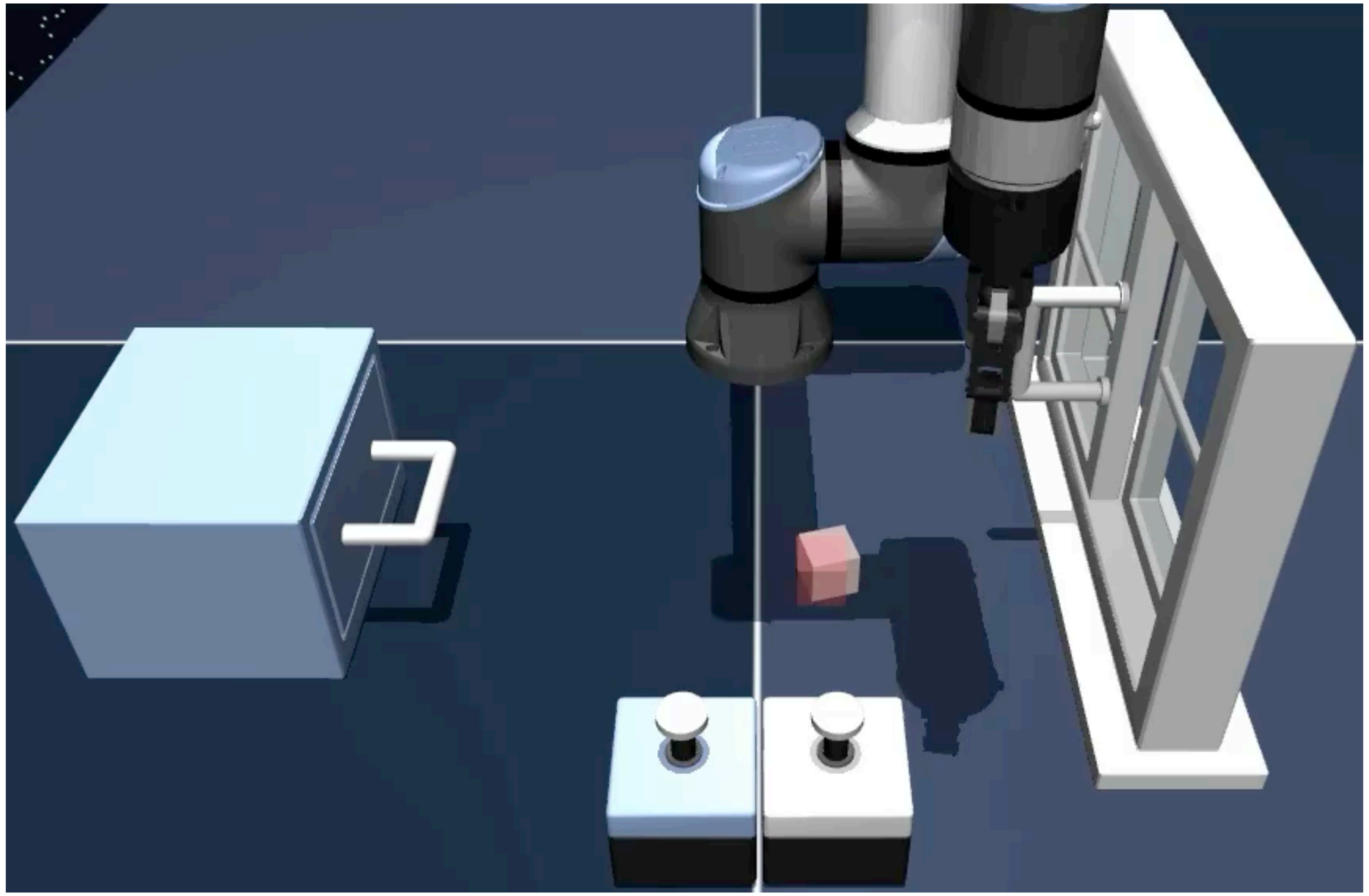
1-step policy struggles to close door

8-step policy solves task smoothly

Q: Is sequential scaling helpful?

Qualitatively, more inference steps leads to more precise actions





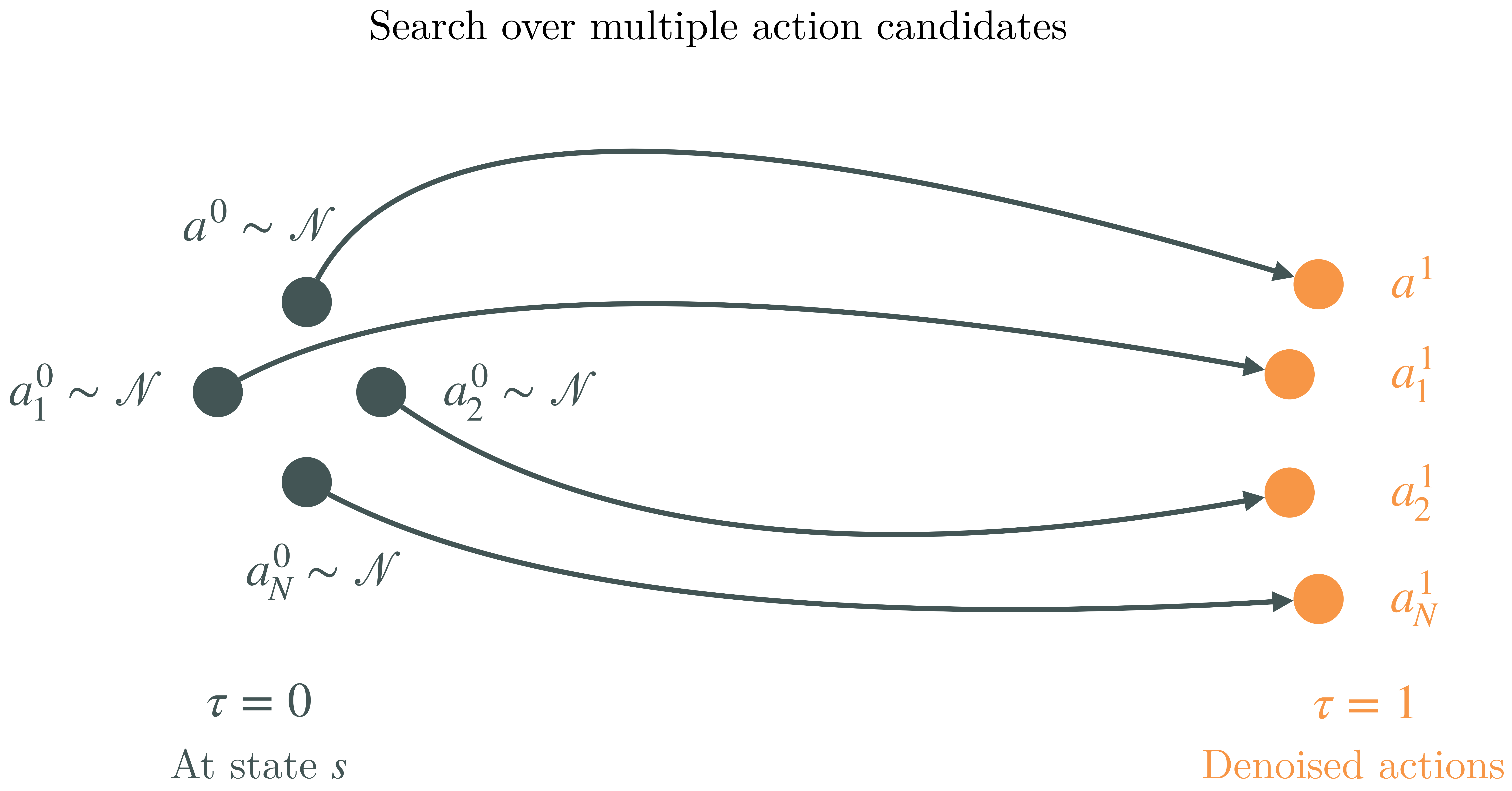
1-step policy *pushes* handle

8-step policy *grips* handle

`SORL` enables test-time *parallel scaling*

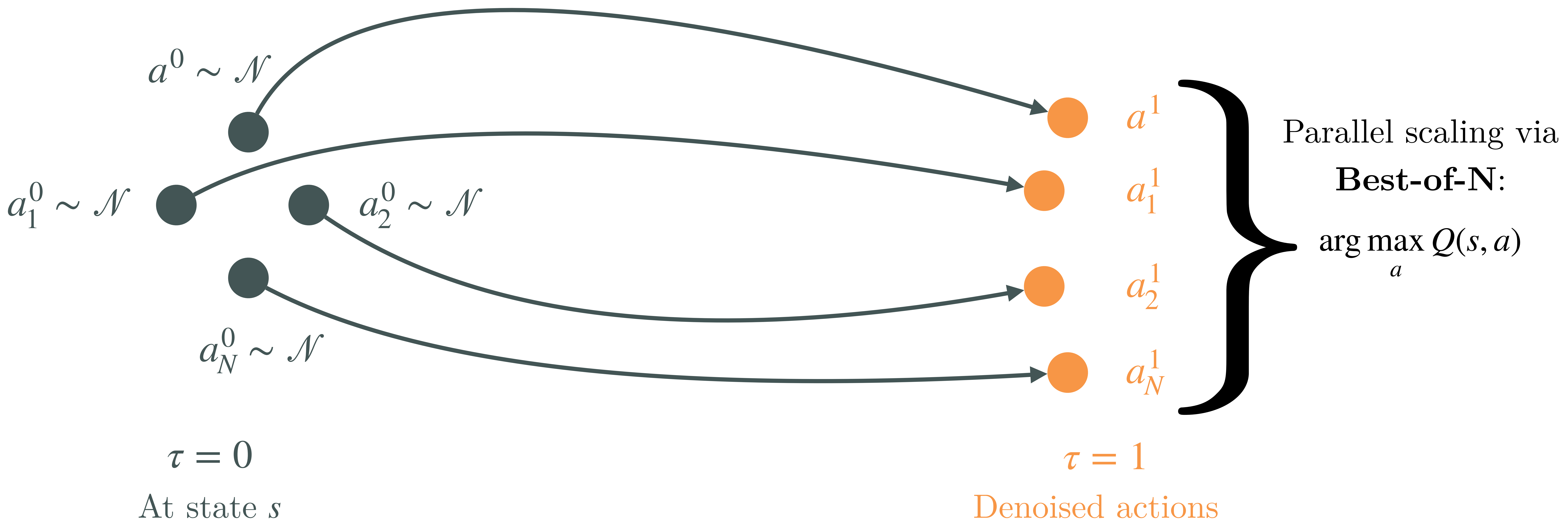
$\texttt{SORL}$ enables test-time *parallel scaling*

SORL enables test-time *parallel scaling*



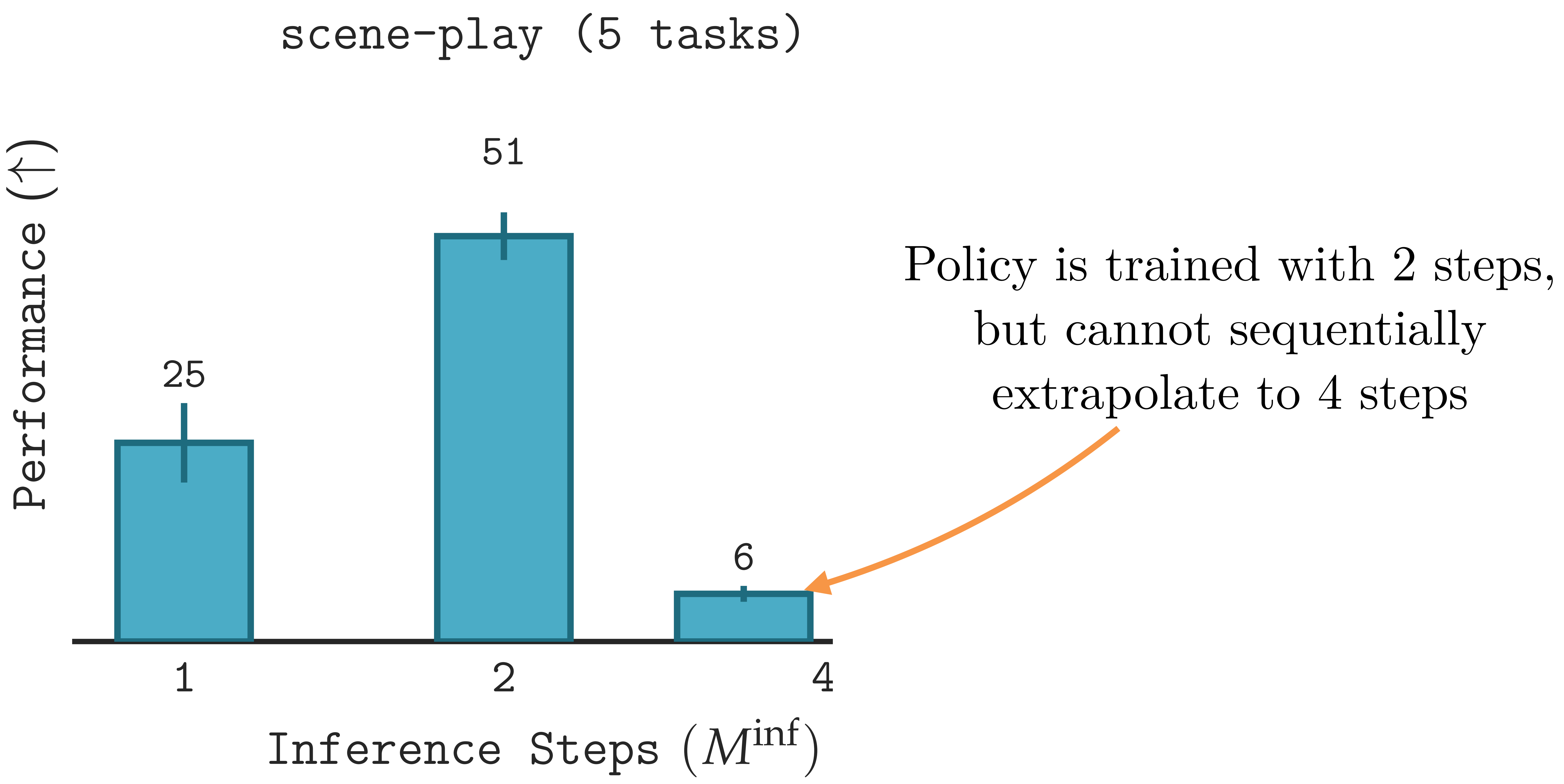
SORL enables test-time *parallel scaling*

Search over multiple action candidates, using the Q function as a *verifier*



Q: Is parallel scaling helpful?

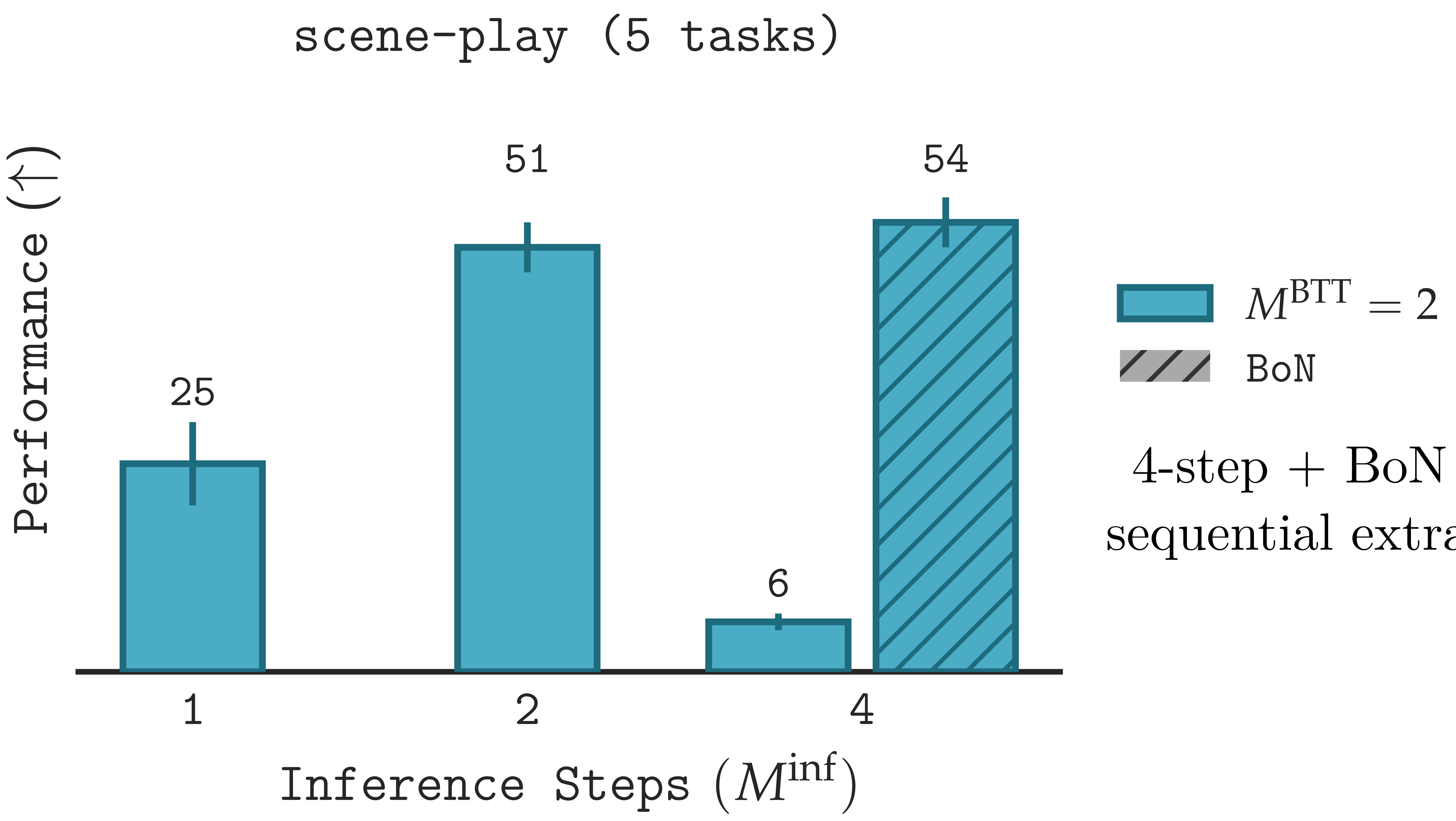
Unlike LLMs, improvement from BoN is not obvious



Q: Is parallel scaling helpful?

Unlike LLMs, improvement from BoN is not obvious

But BoN enables sequential extrapolation



4-step + BoN enables sequential extrapolation

Takeaways

Shortcut model is expressive while being suitable for RL training

Sequential and parallel scaling in general improves performance

In additional to scaling dataset, **leveraging rich generative models and scaling test-time computation** is a promising direction for RL

nico-espinosadice.github.io/projects/sorl

Scaling Offline RL via Efficient and Expressive Shortcut Models

Nicolas Espinosa Dice

Joint work with

Yiyi Zhang, Yiding Chen, Bradley Guo, Owen Oertell, Gokul Swamy, Kianté Brantley, Wen Sun