Why AI is harder in the Physical World 🤖

… and what to maybe do about it

Max Simchowitz CMU

Why are we working on AI?

Why are we working on AI?

"Vision of the Future" - Family Guy™

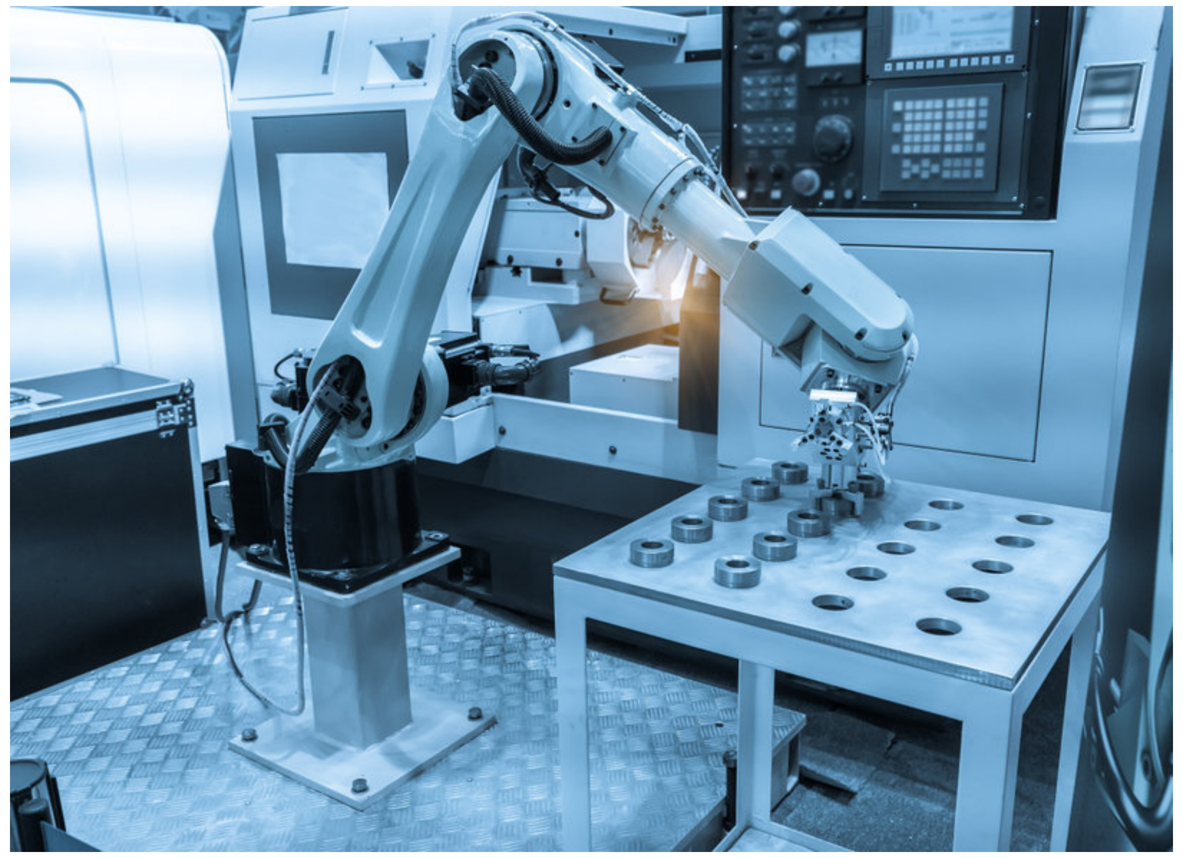
Why are we working on AI?

"Vision of the Future" - Family Guy™

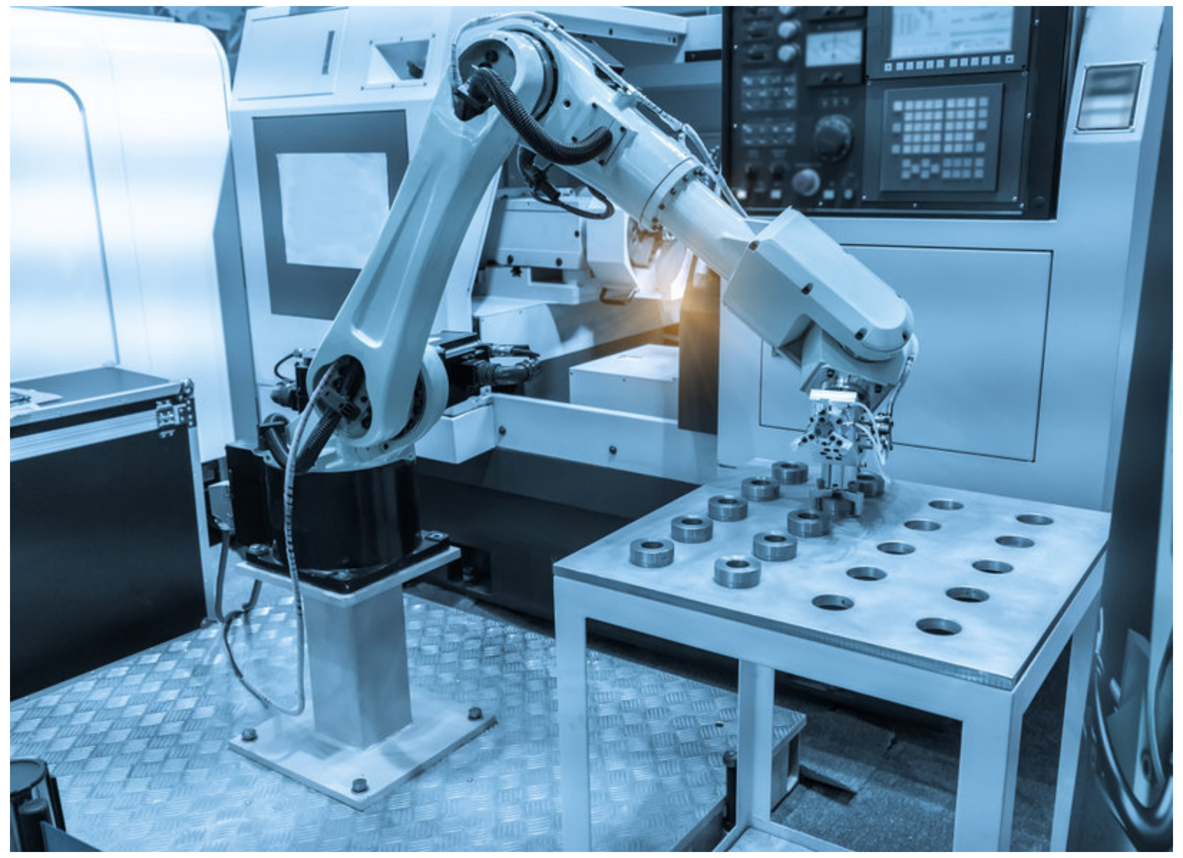
SOTA March 2020

AI in the Physical World 🤖

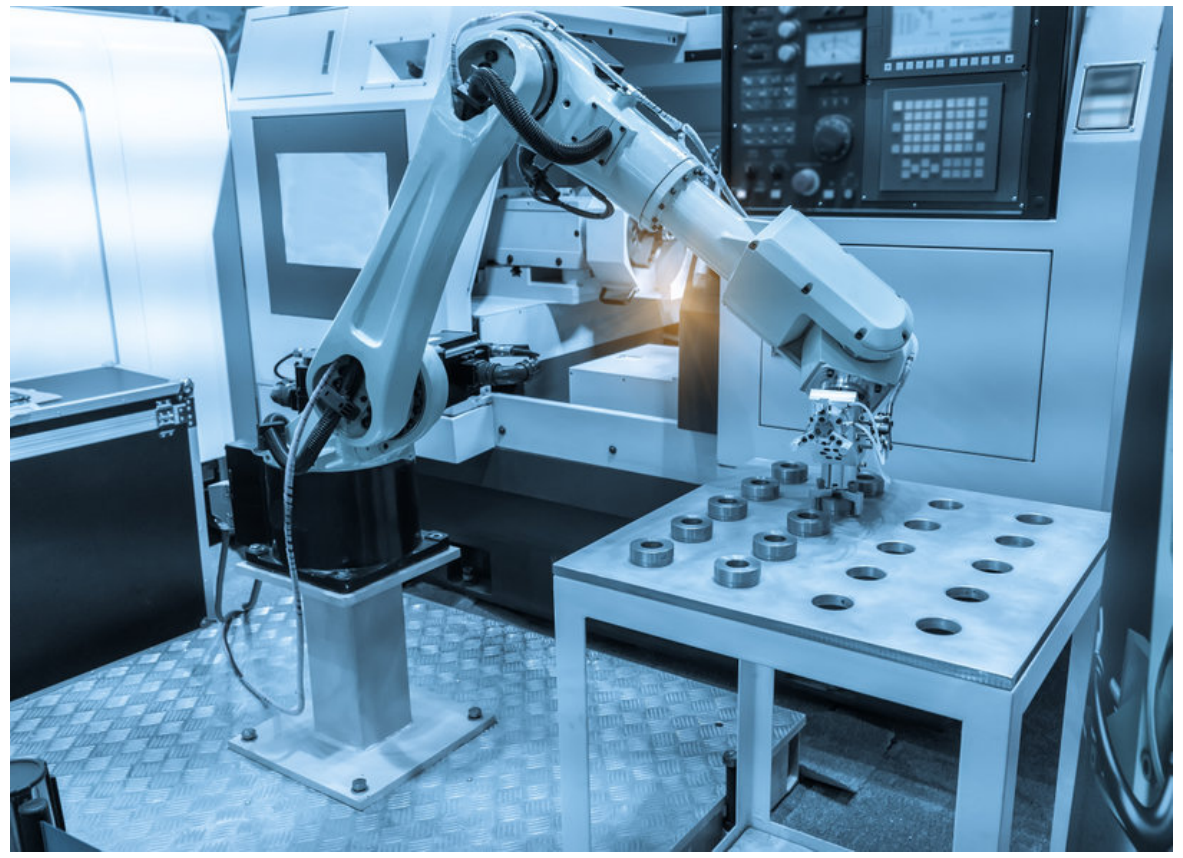
AI in the Physical World 🤖



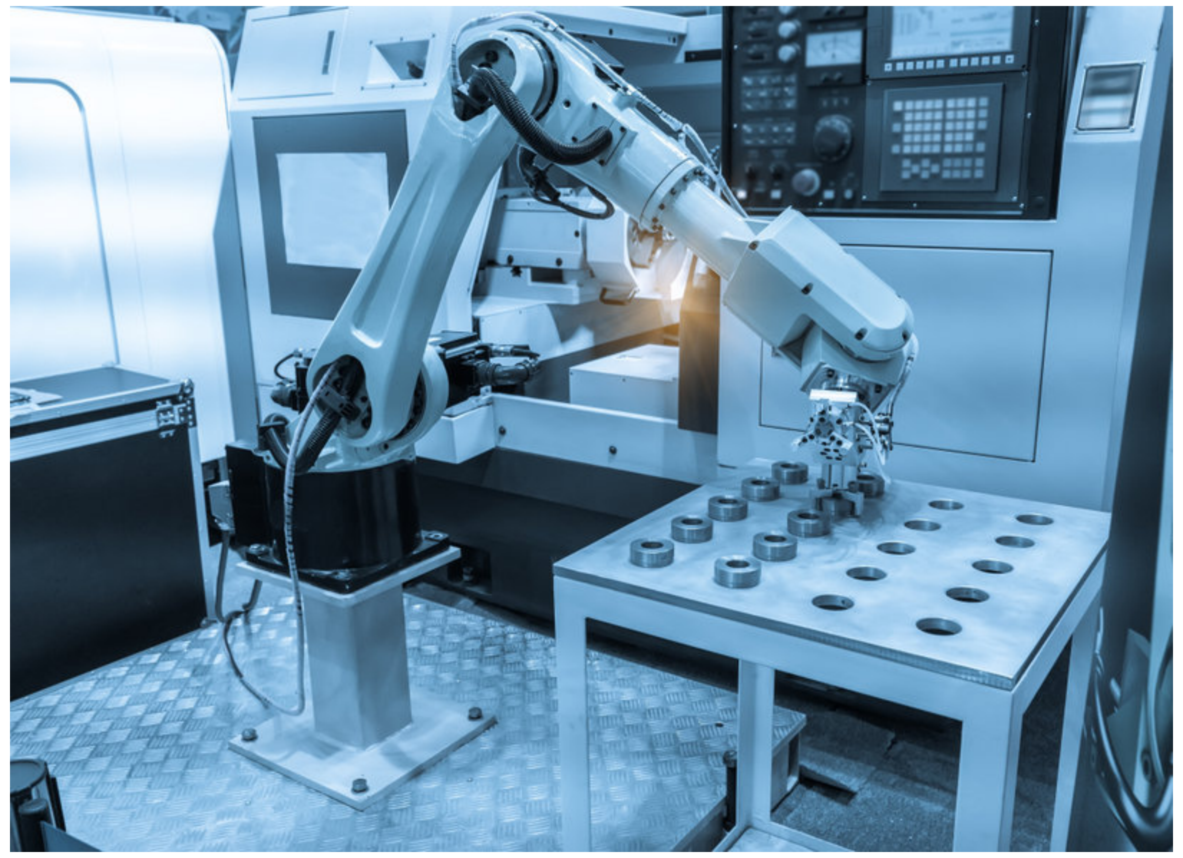
AI in the Physical World 🤖



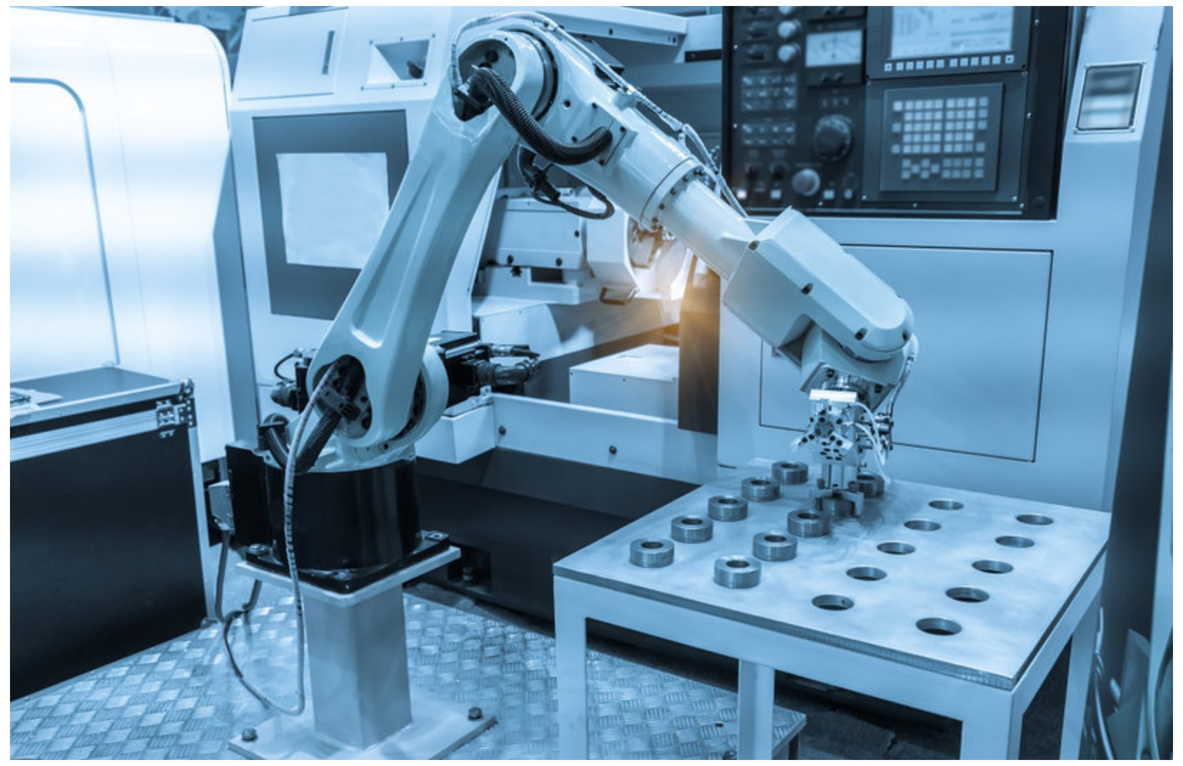
AI in the Physical World 🤖



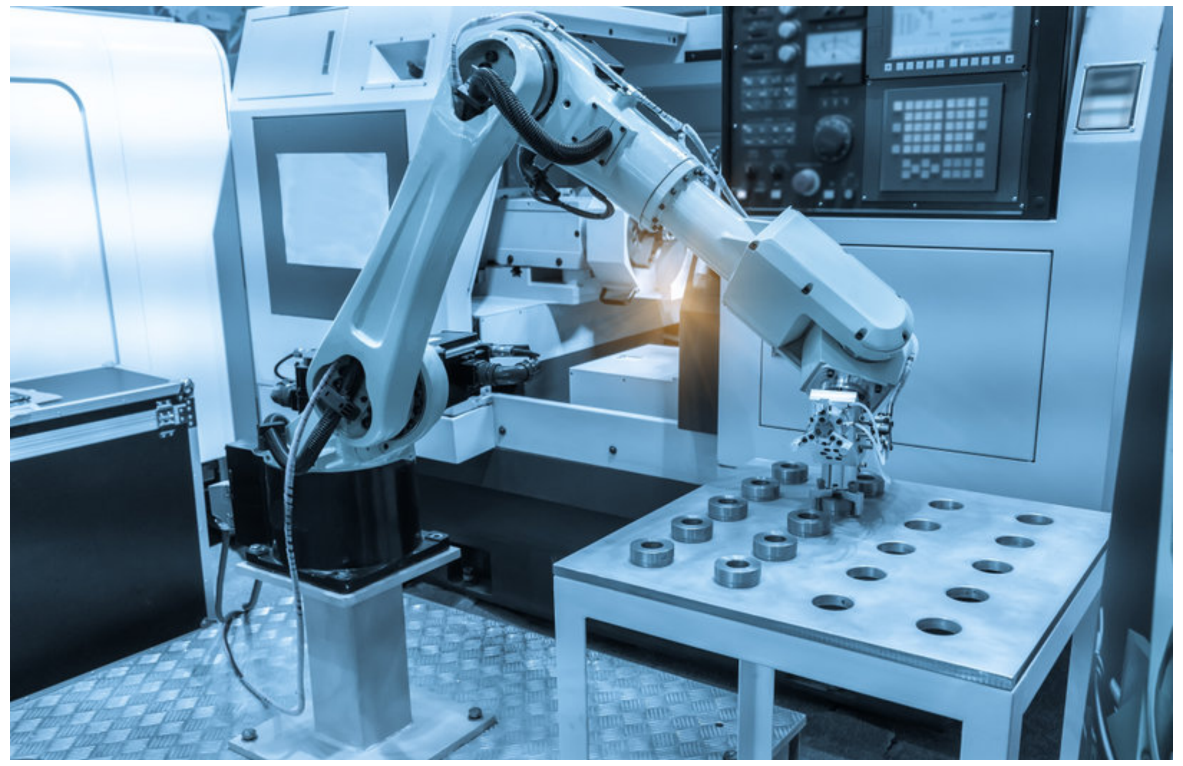
AI in the Physical World 🤖



The Physical World 🤖 v.s. The Discrete World 📚

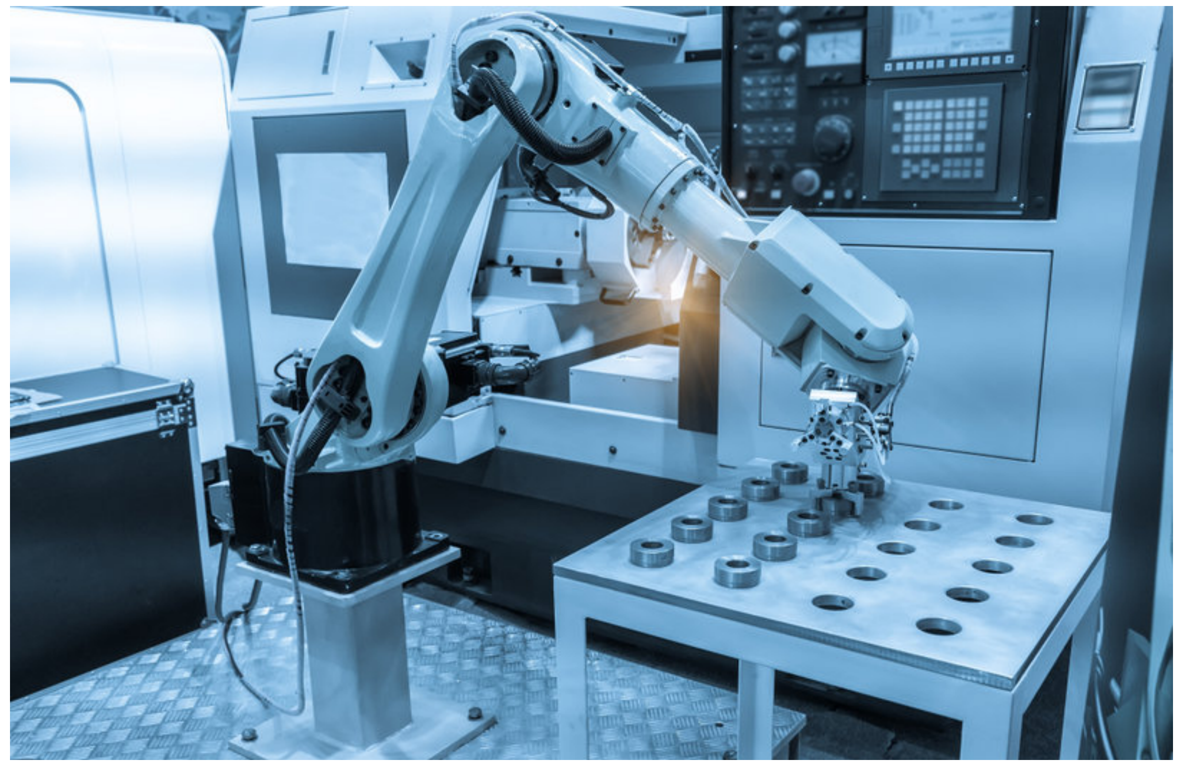


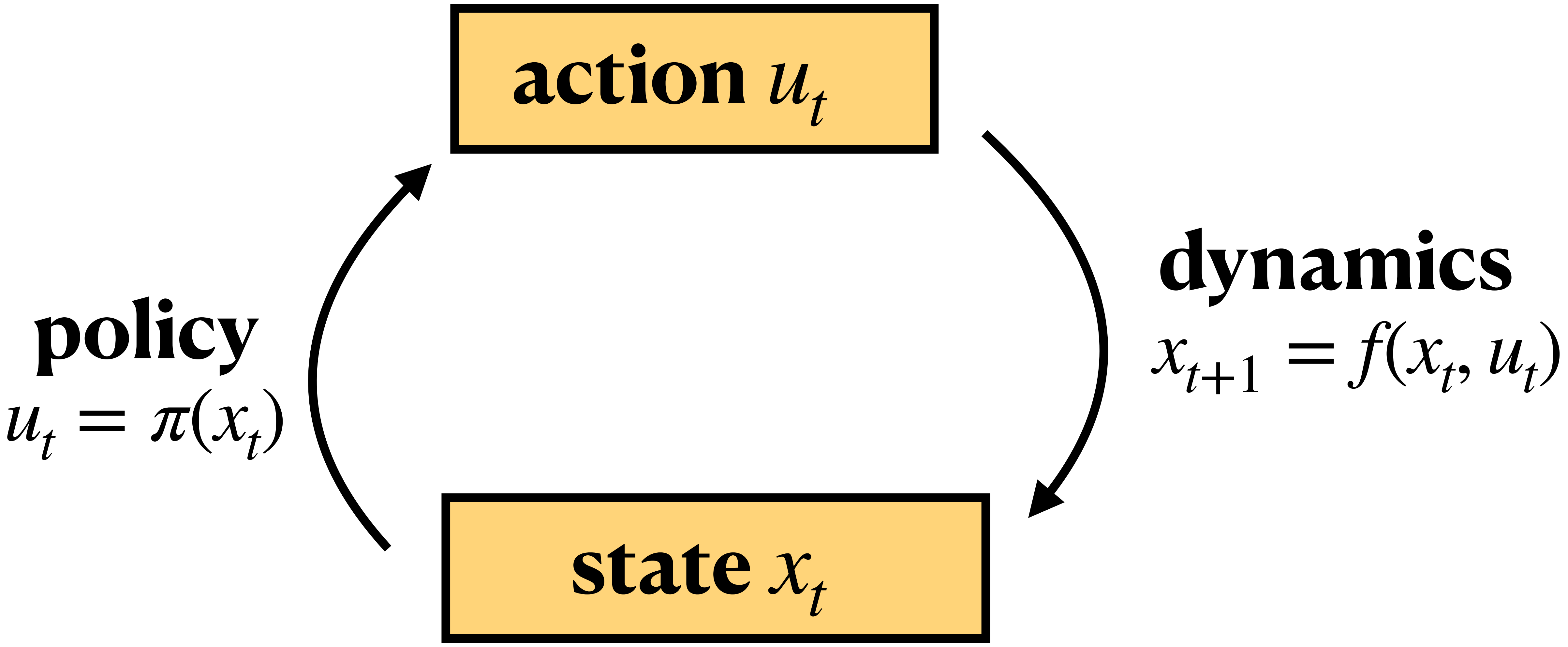
The Physical World 🤖 v.s. The Discrete World 📚



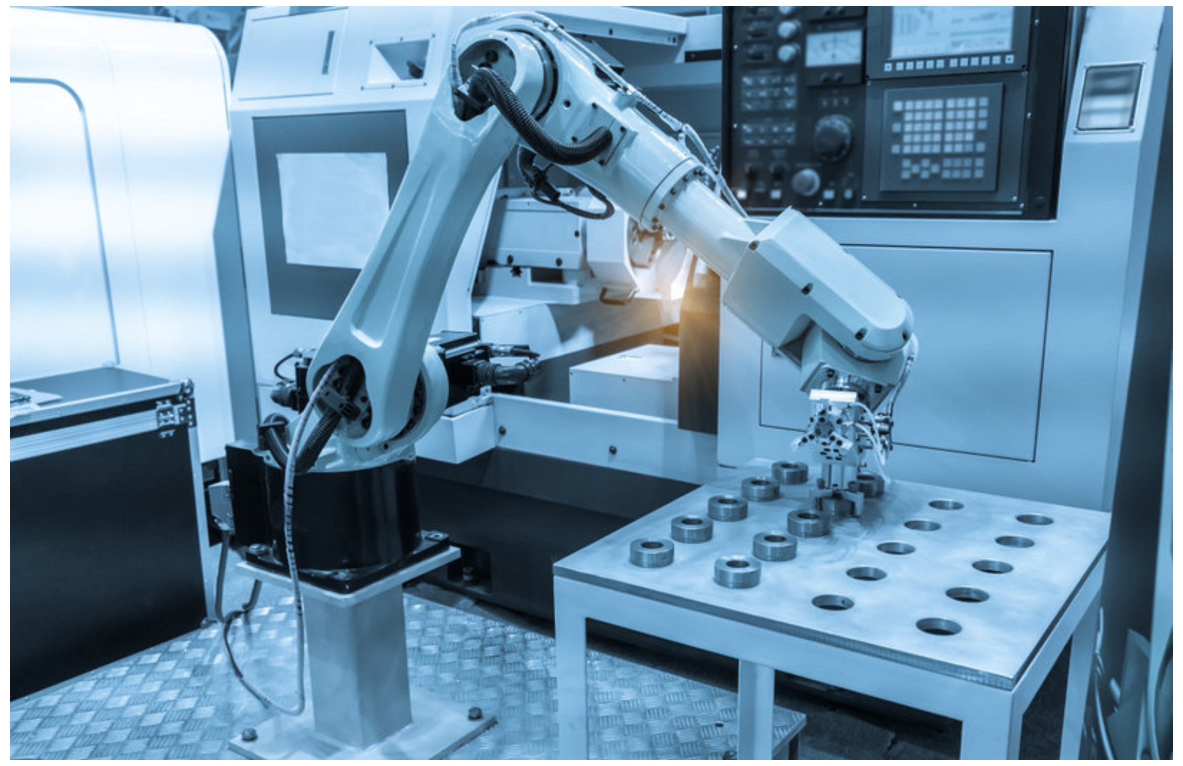
GPT-

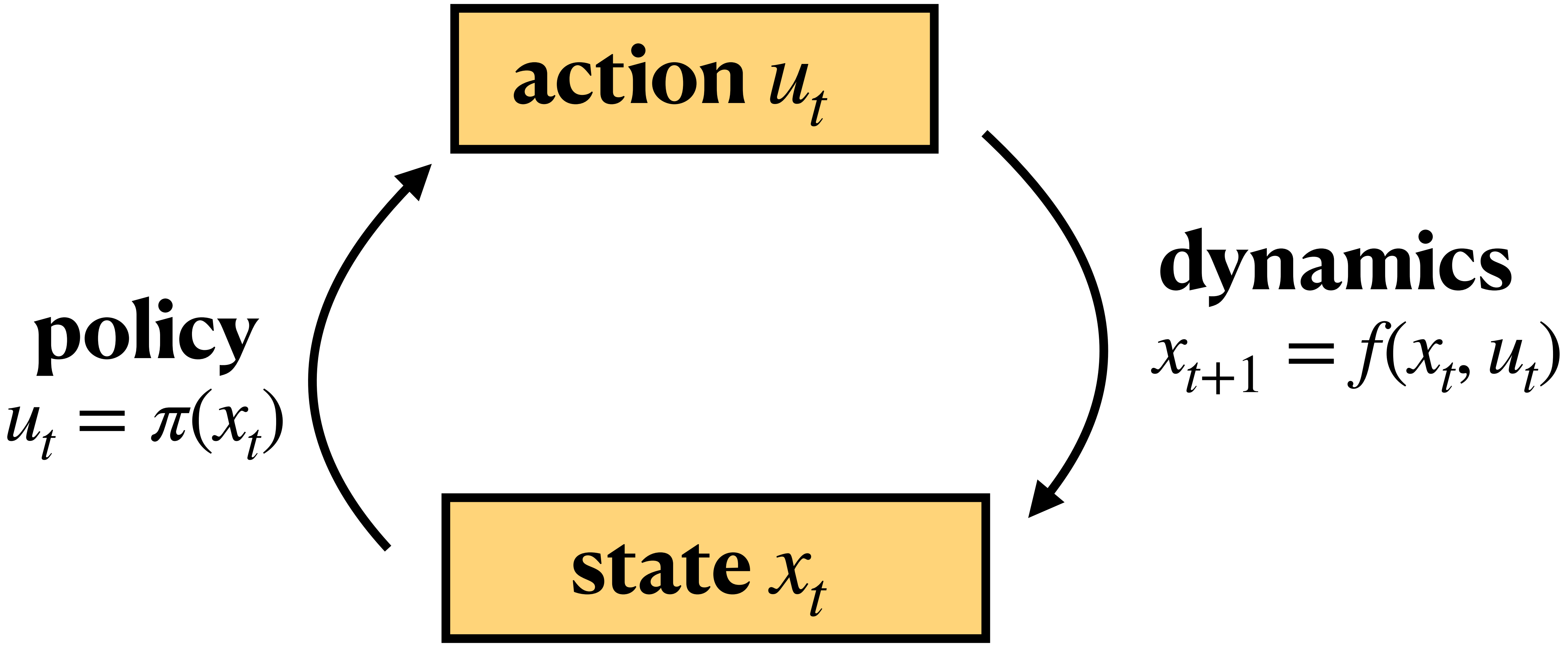
The Physical World 🤖 v.s. The Discrete World 📚

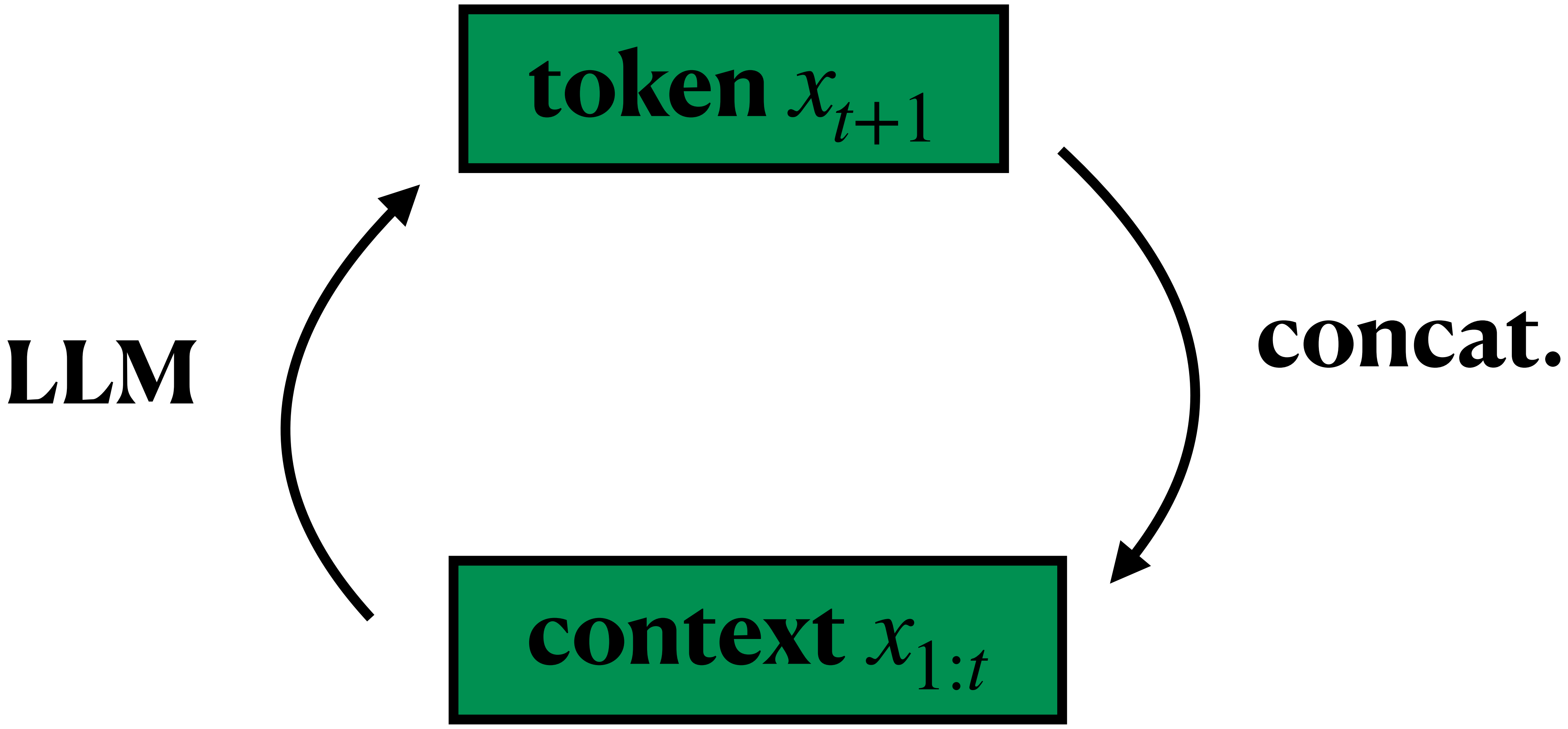




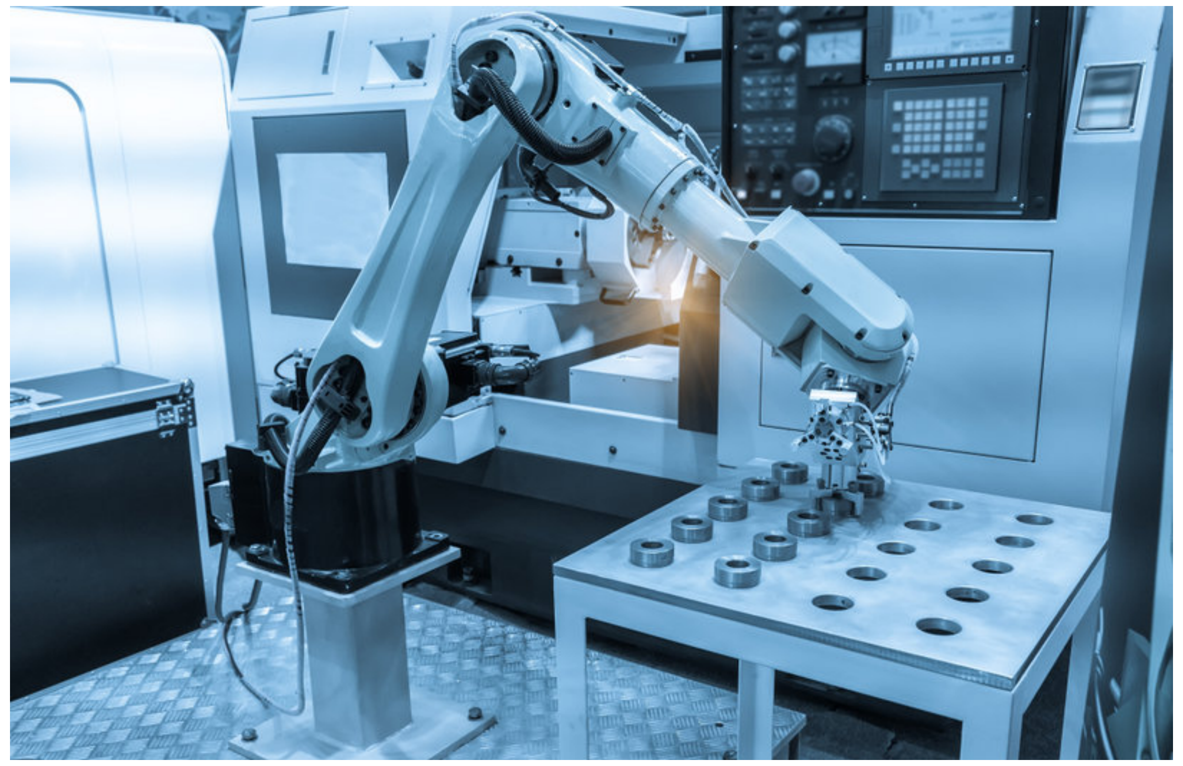
The Physical World 🤖 v.s. The Discrete World 📚







The Physical World 🤖 v.s. The Discrete World 📚



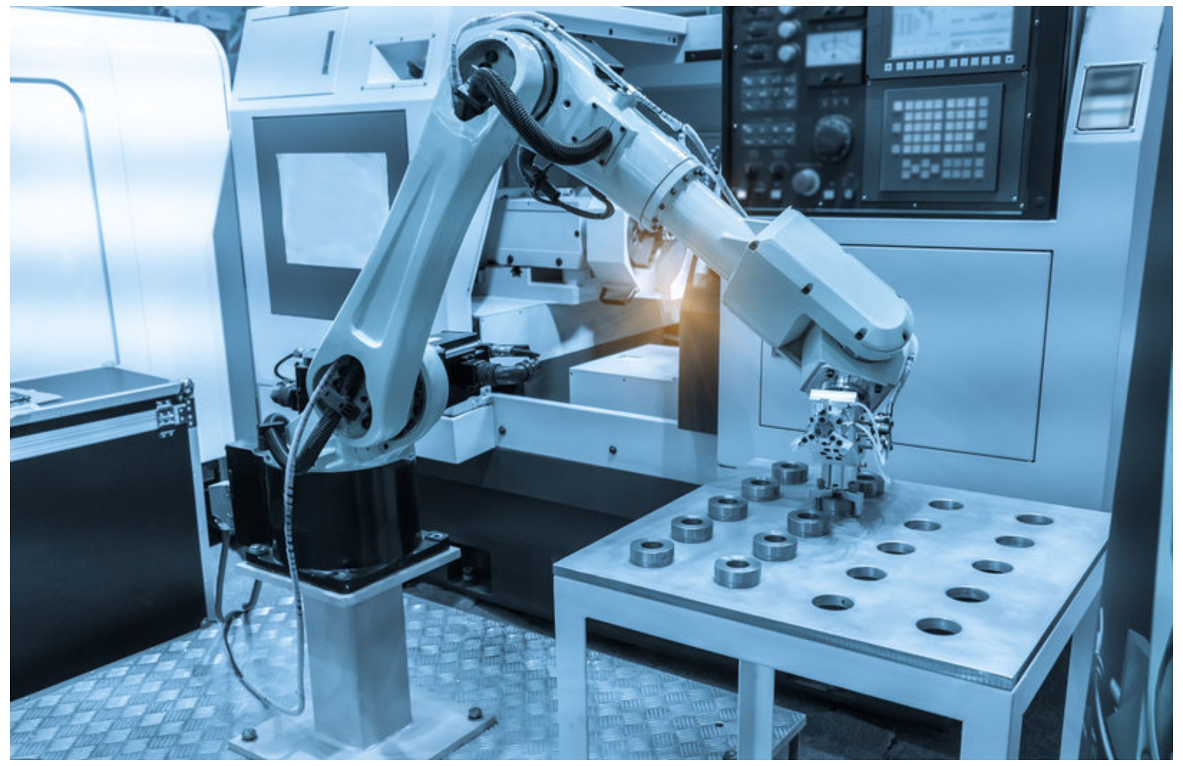
action u_t

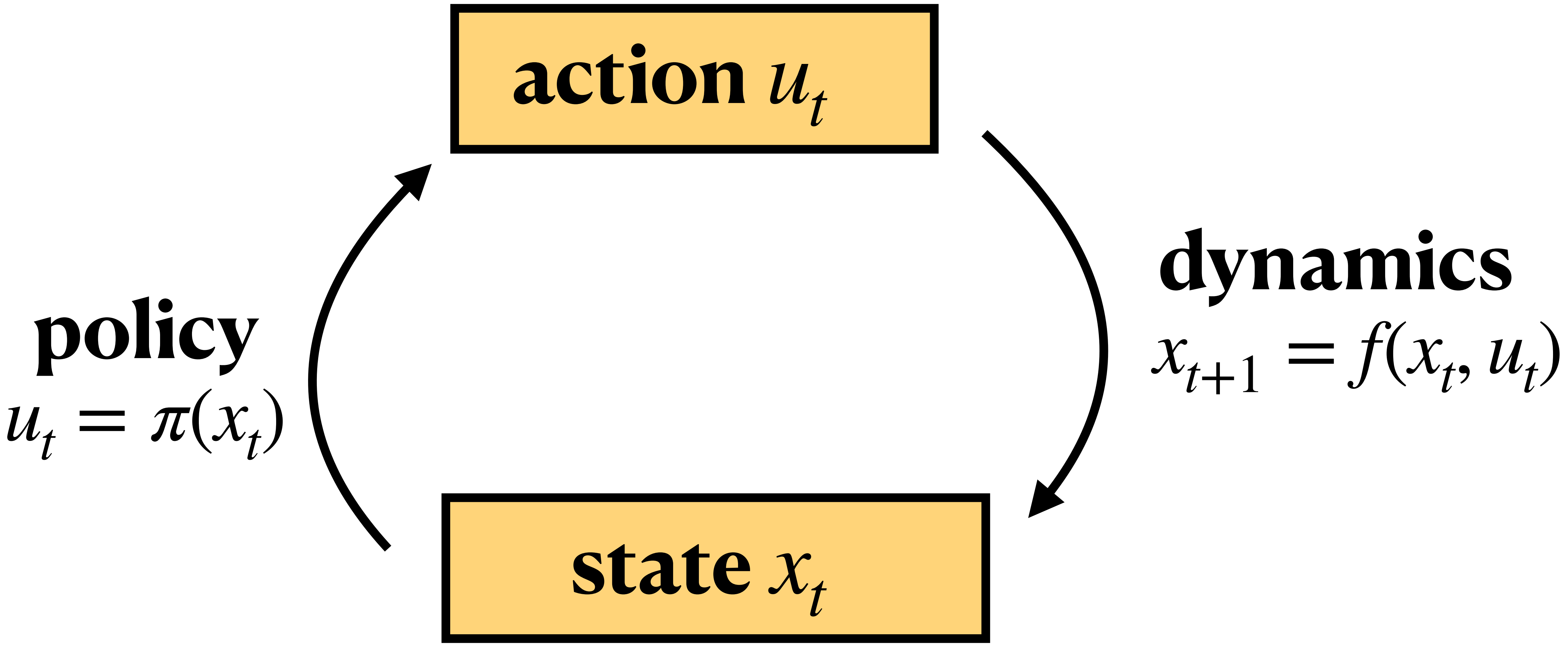
dynamics
$x_{t+1} = f(x_t, u_t)$

state x_t

policy
$u_t = \pi(x_t)$

The Physical World 🤖 v.s. The Discrete World 📚





1. Beholden to **external dynamics**
2. States and actions take **continuous values**

Pre-training in LLMs 📚 is Imitation

Pre-training in LLMs 📚 is Imitation

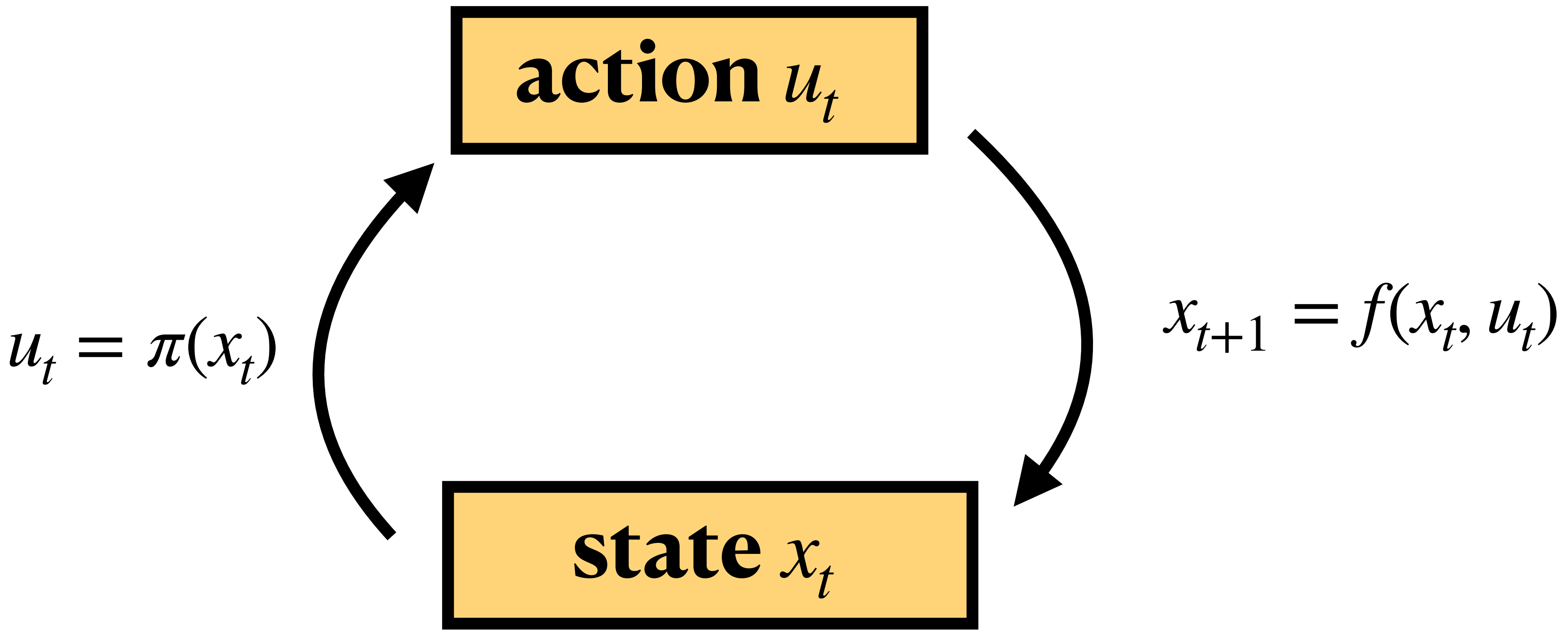
A **large language model (LLM)** is a type of machine learning model (source: Wikipedia)

Pre-training in LLMs 📚 is Imitation

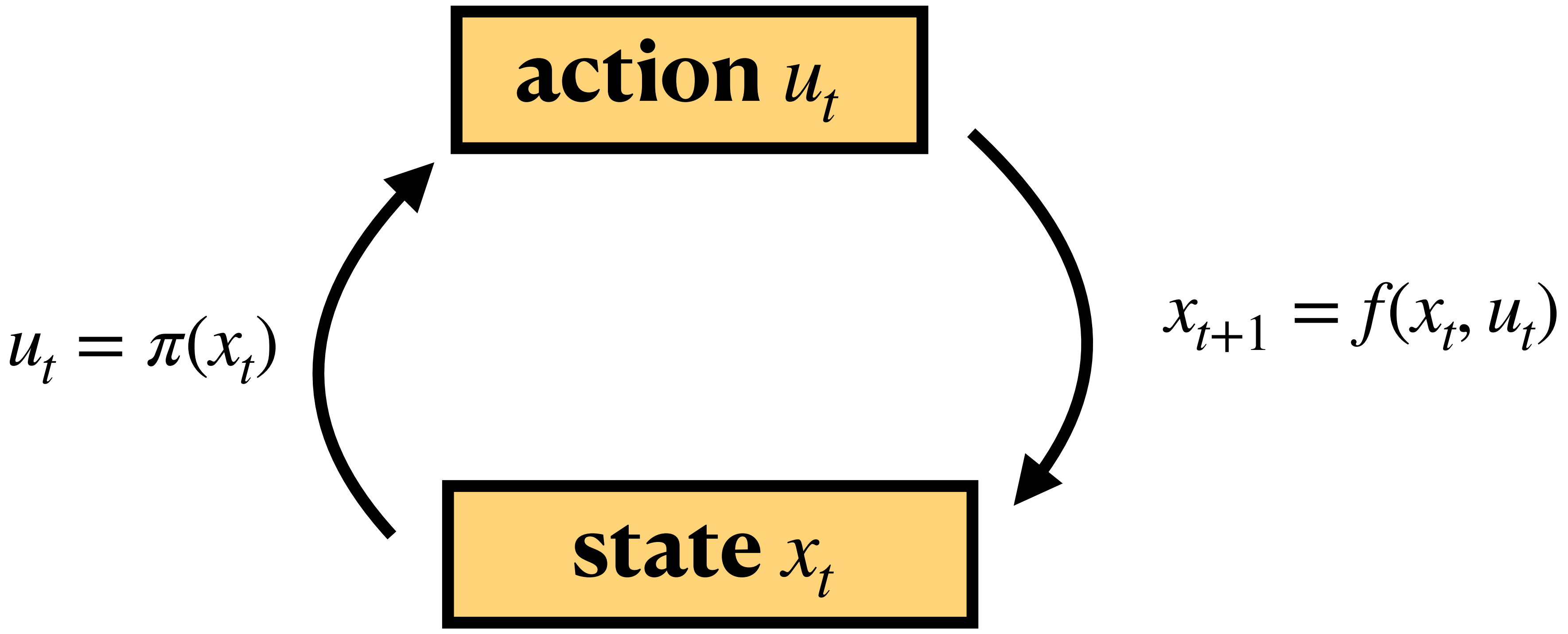
A **large language model** (**LLM**) is a type of machine learning model (source: Wikipedia)

We treat natural human language as an **expert demonstrator** which we aim to imitate.

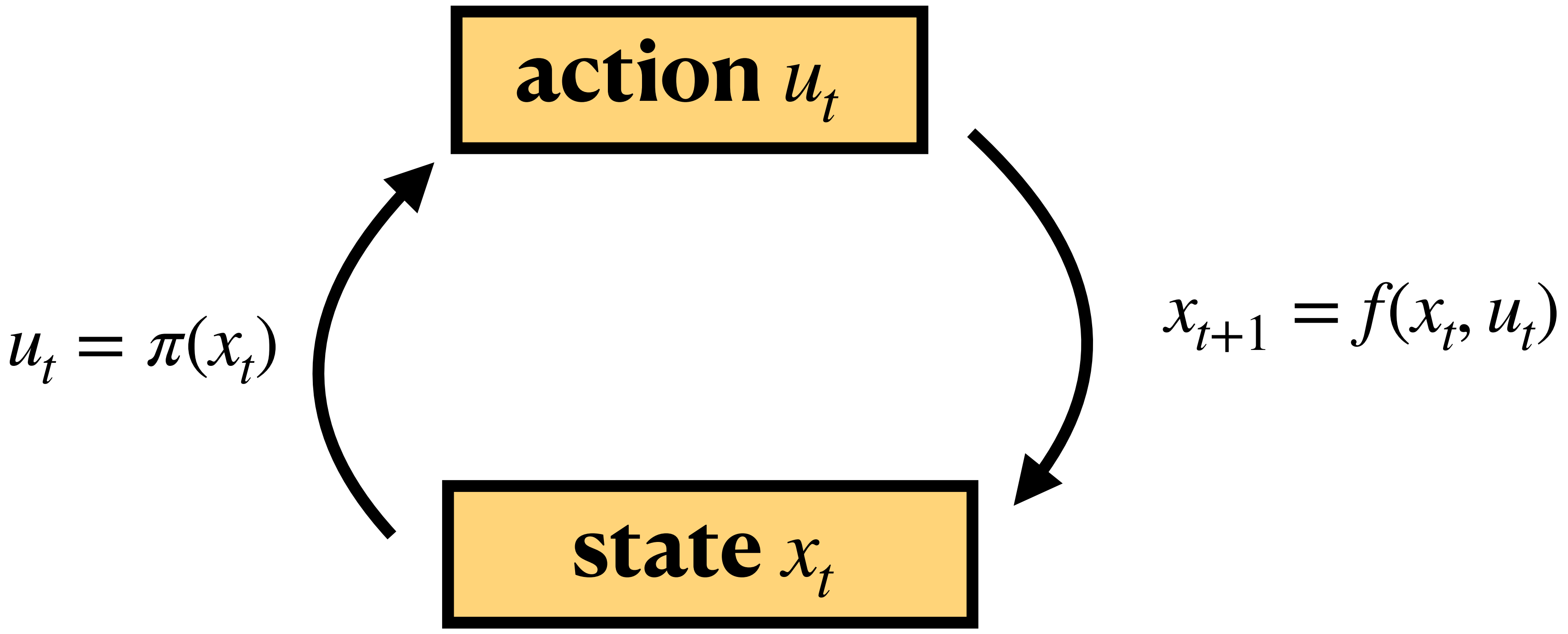
Imitation in the Physical World 🤖



Imitation in the Physical World 🤖

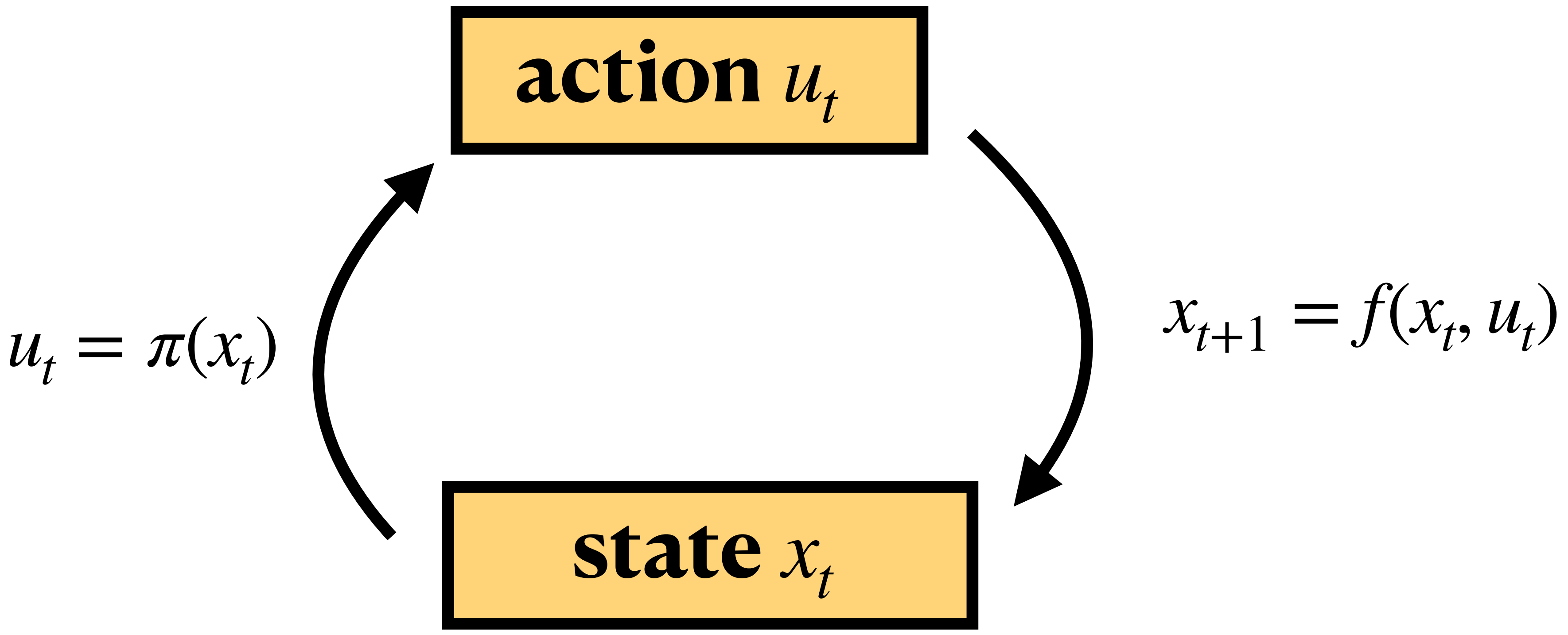


Imitation in the Physical World 🤖



We treat use a human **expert demonstrator** which we aim to imitate.

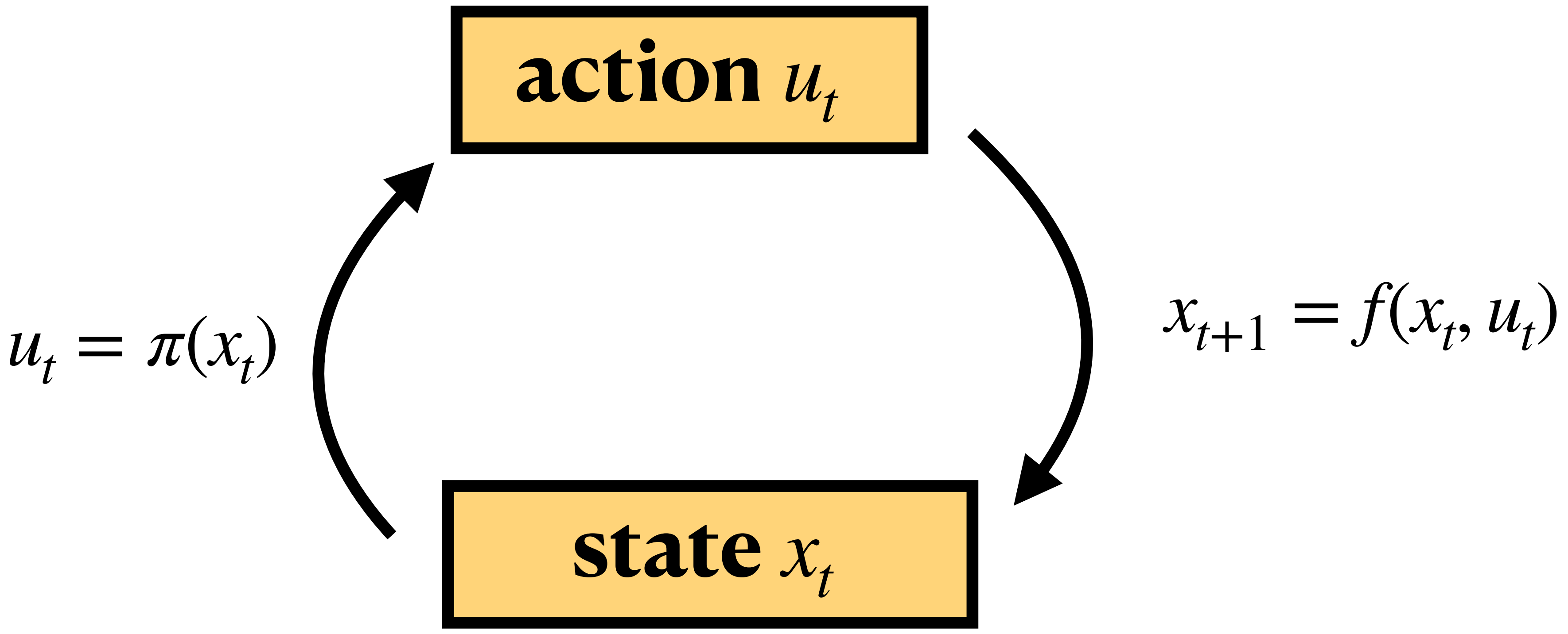
Imitation in the Physical World 🤖



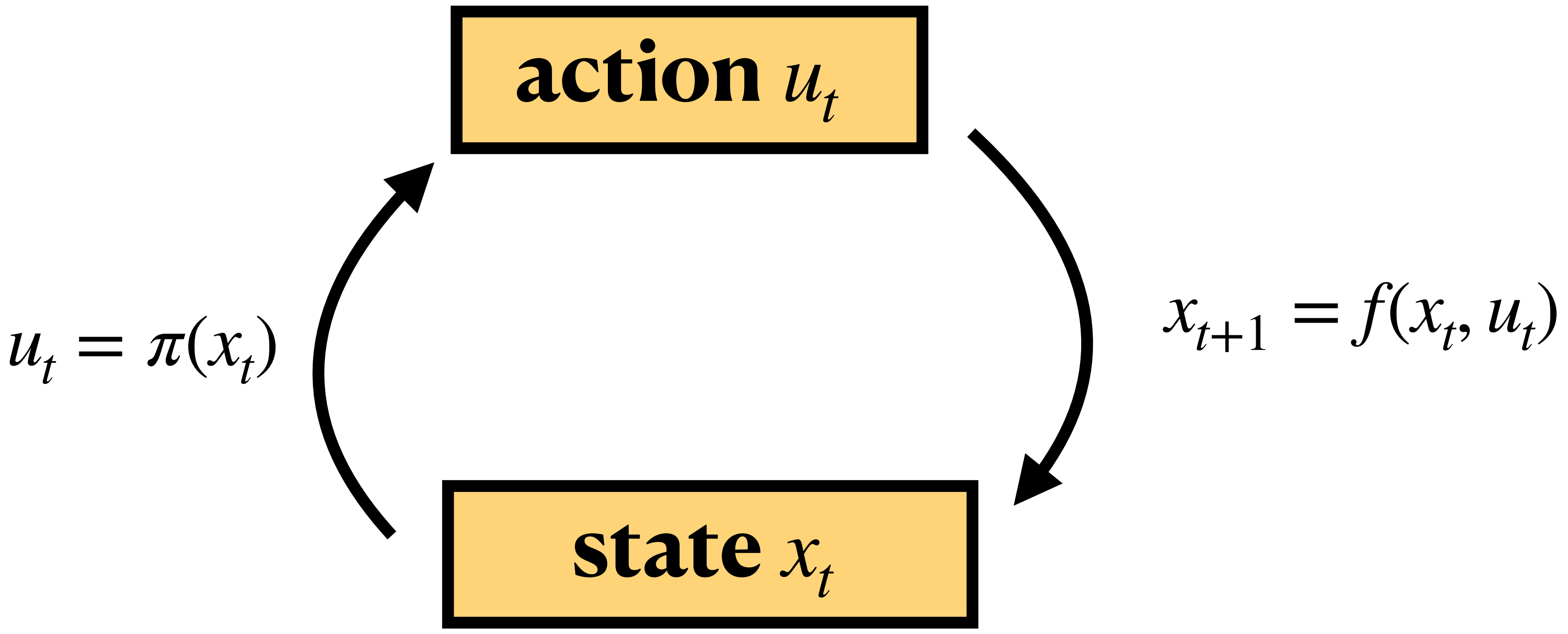
We treat use a human **expert demonstrator** which we aim to imitate.

Our aim is to predict a "**next action**" (robot action) from **observation** (pixels, tactile sensing.)

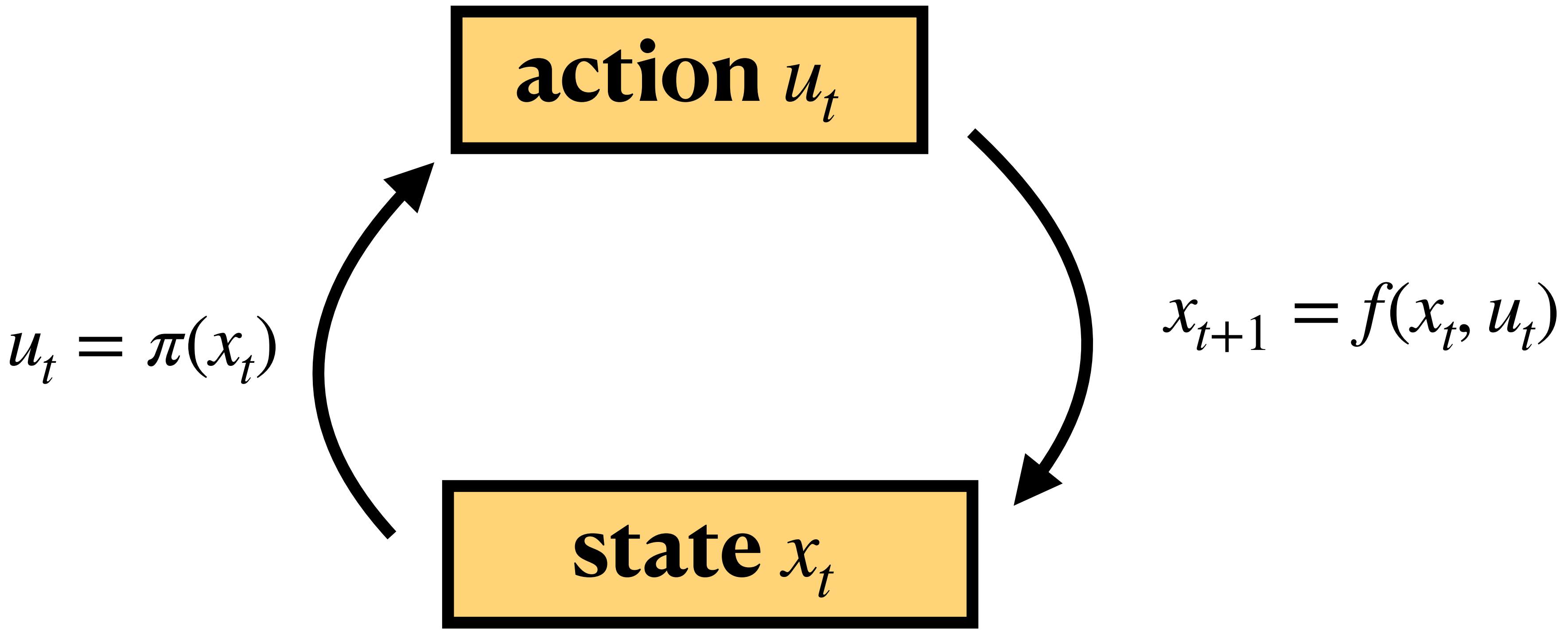
Imitation in the Physical World



Imitation in the Physical World 🤖



Imitation in the Physical World 🤖



How is **imitation** (e.g. pretraining) different in the **physical** v.s. **discrete** settings?

This Talk.

This Talk.

1. Introduce a formal setting of imitation learning (motivated by robotic pretraining).

This Talk.

1. Introduce a formal setting of imitation learning (motivated by robotic pretraining).

This Talk.

1. Introduce a formal setting of imitation learning (motivated by robotic pretraining).

2. Demonstrate how imitation is **considerably more challenging** in the **physical world** 🤖 than in the **discrete world** 📚.

This Talk.

1. Introduce a formal setting of imitation learning (motivated by robotic pretraining).

2. Demonstrate how imitation is **considerably more challenging** in the **physical world** 🤖 than in the **discrete world** 📚.

This Talk.

1. Introduce a formal setting of imitation learning (motivated by robotic pretraining).

2. Demonstrate how imitation is **considerably more challenging** in the **physical world** 🤖 than in the **discrete world** 📚**.**

3. Explain that that popular design decisions from today's world of robotics are not just **helpful**, but **indispensable.**

This Talk.

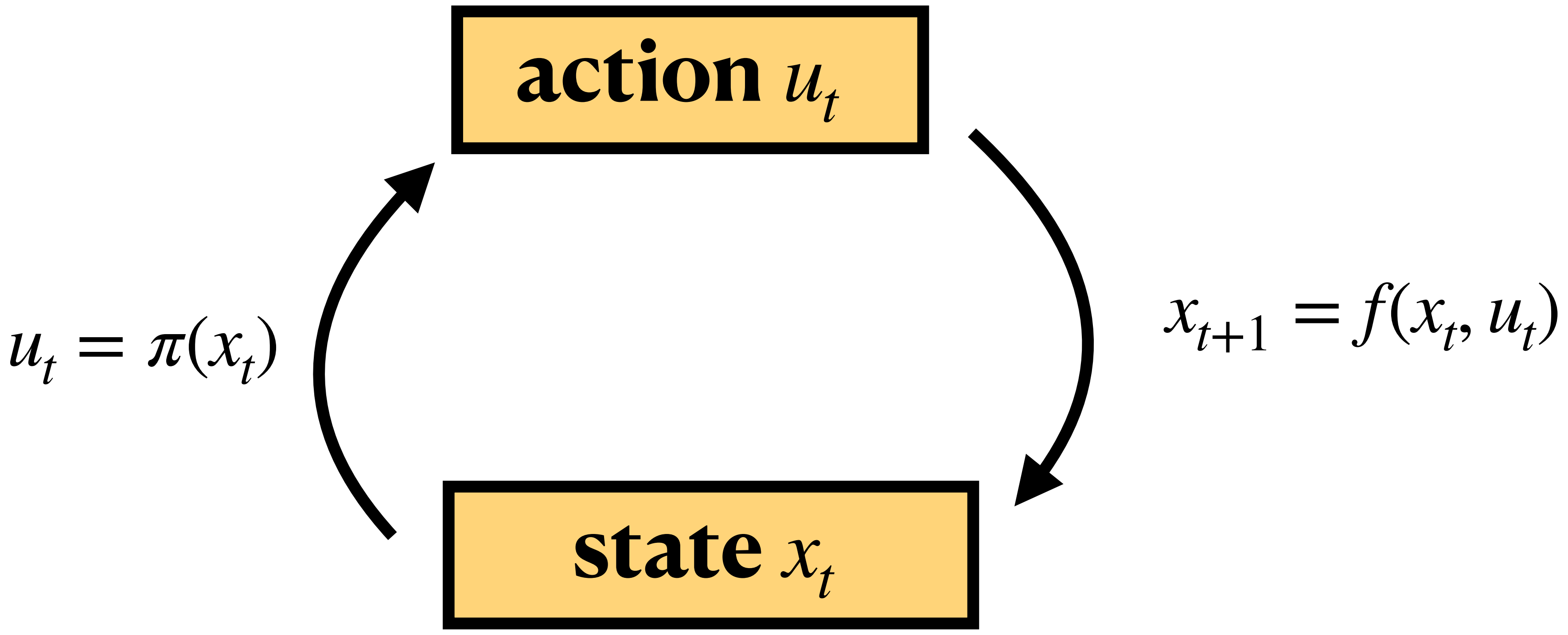
1. Introduce a formal setting of imitation learning (motivated by robotic pretraining).

2. Demonstrate how imitation is **considerably more challenging** in the **physical world** 🤖 than in the **discrete world** 📚**.**

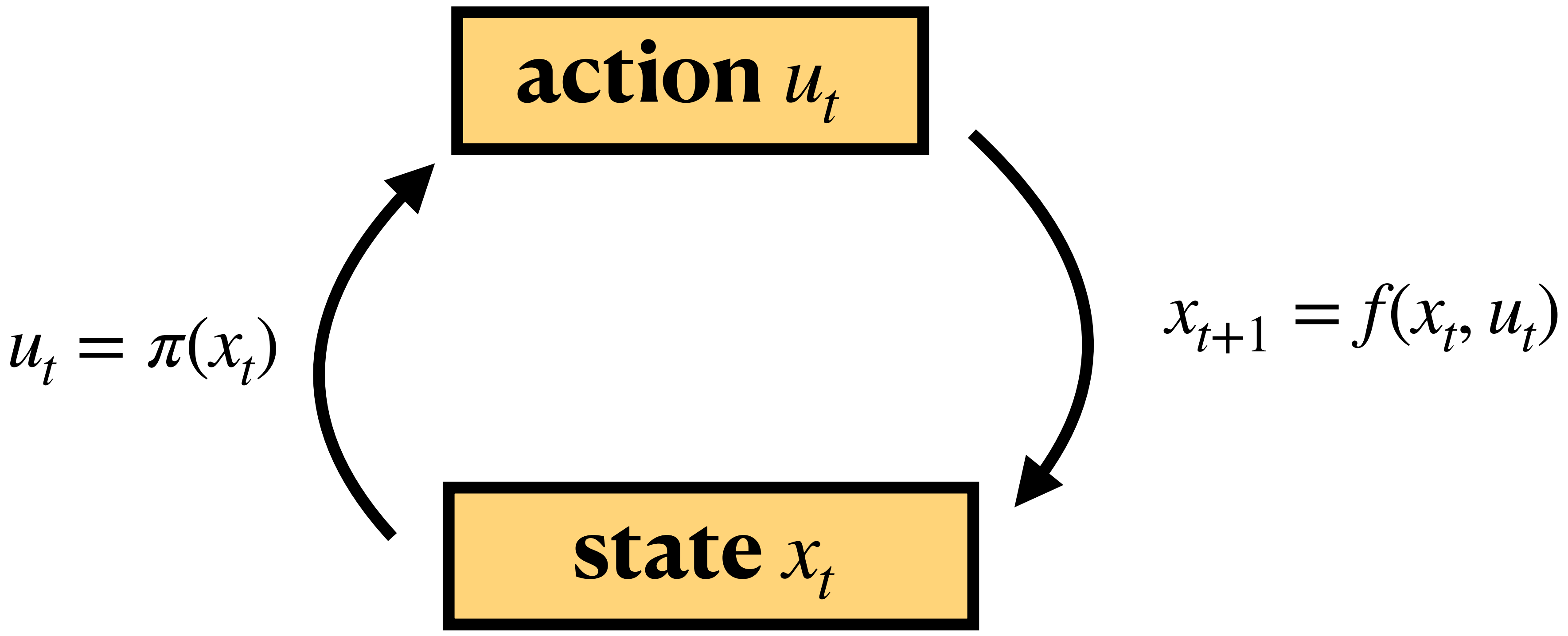
3. Explain that that popular design decisions from today's world of robotics are not just **helpful**, but **indispensable.**

(this is a theory talk)

Imitation in the Physical World 🤖

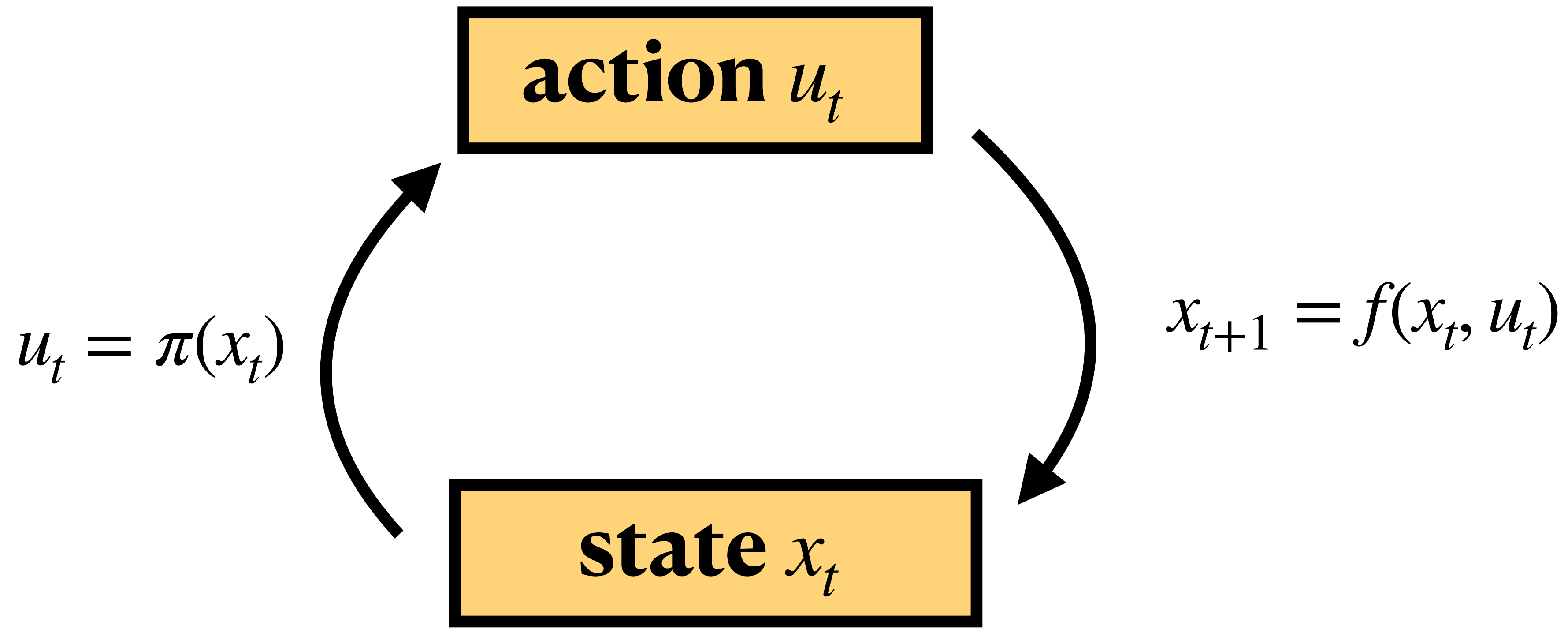


Imitation in the Physical World 🤖



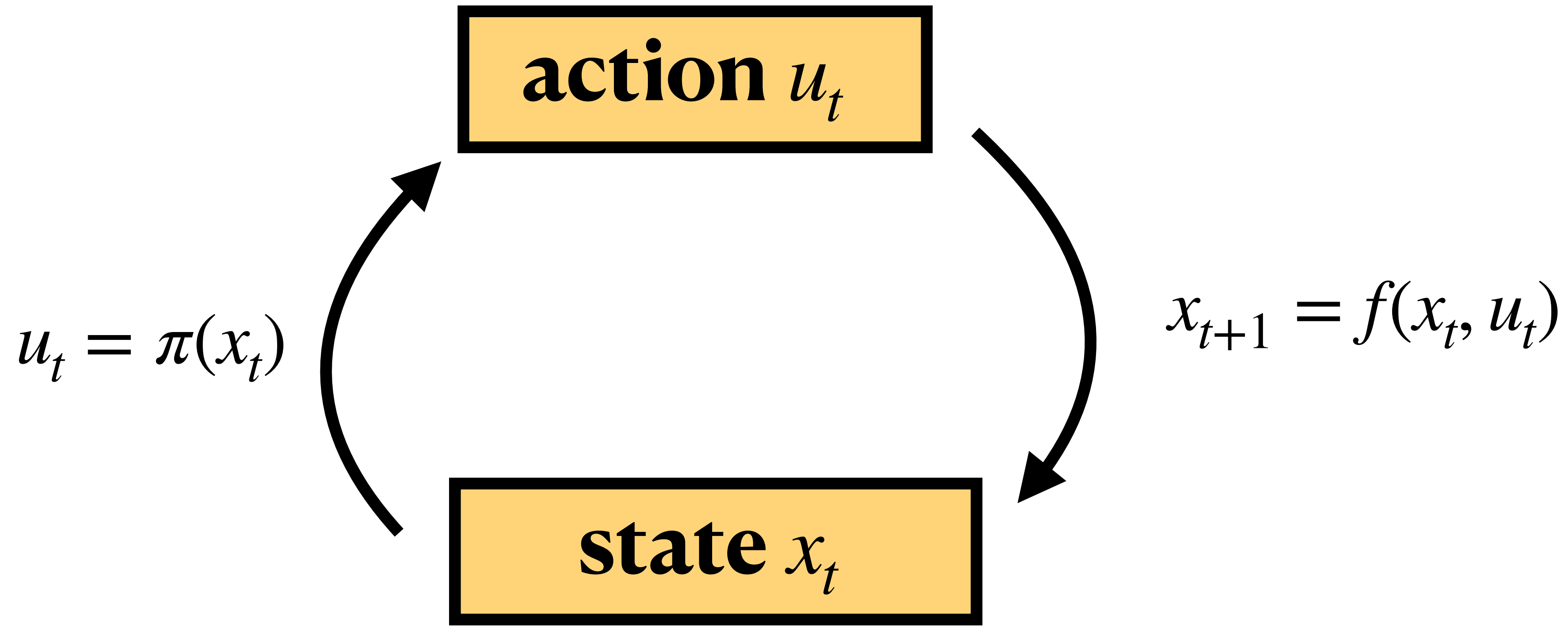
Imitation in the Physical World 🤖

Collect **n expert trajectories** $(x_{1:H}, u_{1:H}) \sim \mathbb{P}_{\pi^\star}$.



Imitation in the Physical World 🤖

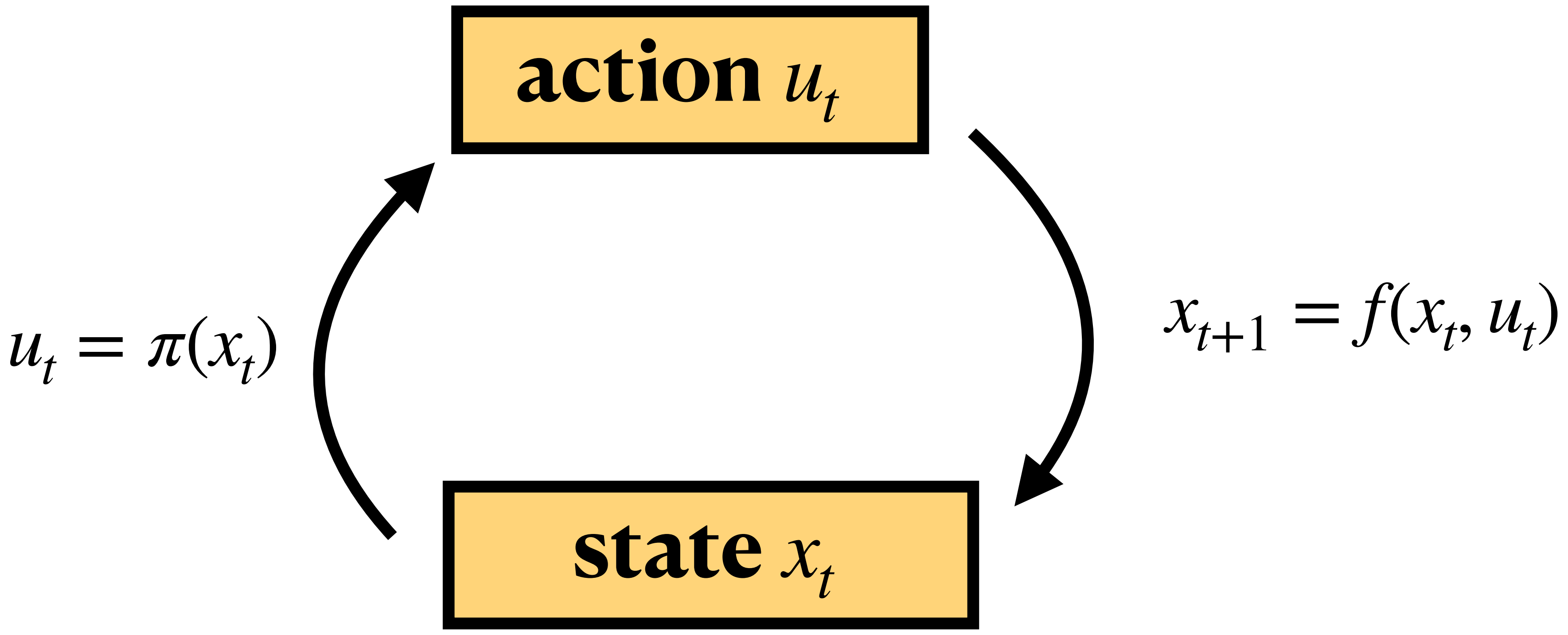
Collect **n expert trajectories** $(x_{1:H}, u_{1:H}) \sim \mathbb{P}_{\pi^\star}$.



Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

Imitation in the Physical World

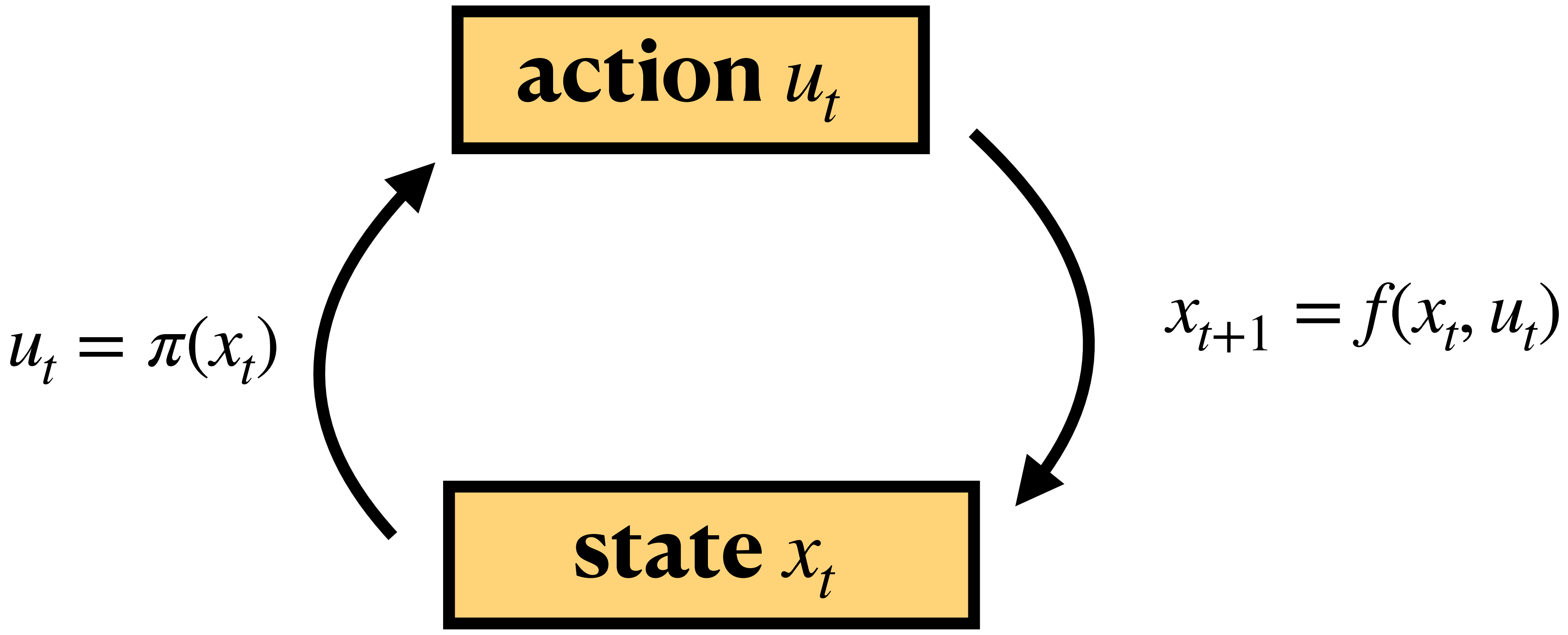
Collect **n expert trajectories** $(x_{1:H}, u_{1:H}) \sim \mathbb{P}_{\pi^\star}$.



Mininimize $\underbrace{\mathcal{R}_c(\hat{\pi}; \pi^\star)}_{\textbf{excess cost}} = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

Imitation in the Physical World 🤖

Collect **n expert trajectories** $(x_{1:H}, u_{1:H}) \sim \mathbb{P}_{\pi^\star}$.

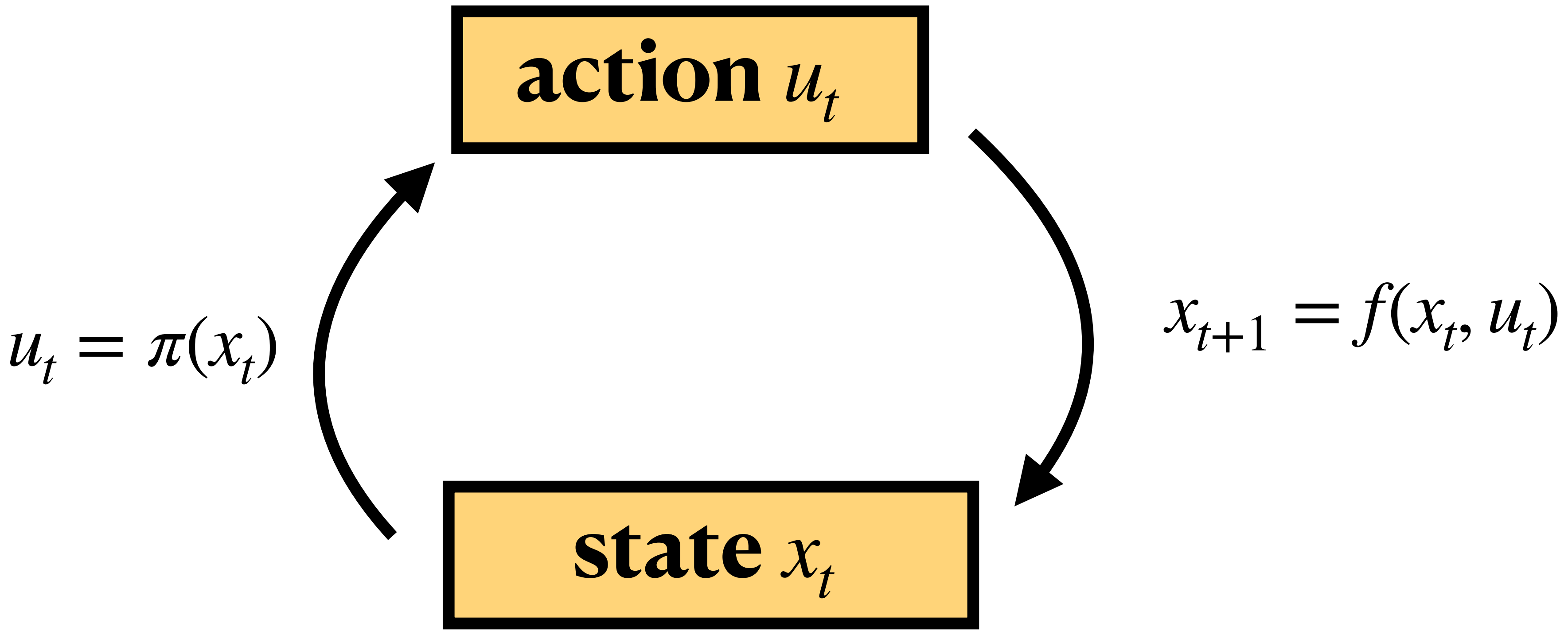


Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost **cost under imitator**

Imitation in the Physical World

Collect **n expert trajectories** $(x_{1:H}, u_{1:H}) \sim \mathbb{P}_{\pi^\star}$.

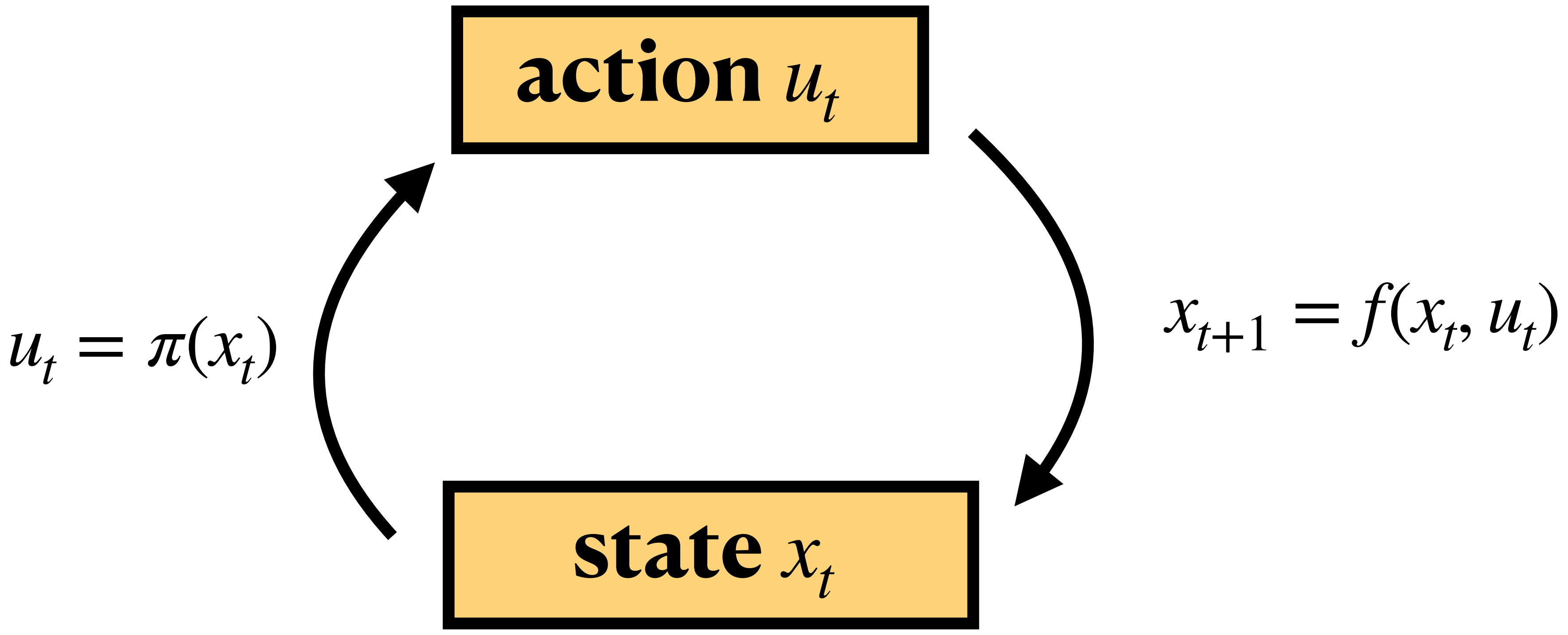


Mininimize $\mathcal{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost | **cost under imitator** | **cost under expert**

Imitation in the Physical World

Collect **n expert trajectories** $(x_{1:H}, u_{1:H}) \sim \mathbb{P}_{\pi^\star}$.



Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost — **cost under imitator** — **cost under expert**

“Horizon” H

Example Algorithm: Behavior Cloning.

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost — **cost under imitator** — **cost under expert**

Example Algorithm: Behavior Cloning.

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost — **cost under imitator** — **cost under expert**

The **Behavior Cloning Algorithm**: $\hat{\pi} \approx \arg\min_{\pi} \sum_{(x,u)\in \text{expert data}} \text{loss}(\pi, x, u)$

Example Algorithm: Behavior Cloning.

Mininimize $\mathcal{R}_c(\hat{\pi};\pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t,u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t,u_t)]$

excess cost — **cost under imitator** — **cost under expert**

The **Behavior Cloning Algorithm**: $\hat{\pi} \approx \arg\min_{\pi} \sum_{(x,u)\in \text{expert data}} \text{loss}(\pi, x, u)$

Example 1: $\text{loss}(\pi, x, u) = \|u - \pi(x)\|^2$

Example Algorithm: Behavior Cloning.

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost **cost under imitator** **cost under expert**

The **Behavior Cloning Algorithm**: $\hat{\pi} \approx \arg\min_{\pi} \sum_{(x,u) \in \text{expert data}} \text{loss}(\pi, x, u)$

Example 1: $\text{loss}(\pi, x, u) = \|u - \pi(x)\|^2$ ($\pi^\star$ is deterministic)

Example Algorithm: Behavior Cloning.

Mininimize $\mathcal{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost **cost under imitator** **cost under expert**

The **Behavior Cloning Algorithm**: $\hat{\pi} \approx \arg\min_{\pi} \sum_{(x,u)\in \text{expert data}} \text{loss}(\pi, x, u)$

Example 1: $\text{loss}(\pi, x, u) = \|u - \pi(x)\|^2$ ($\pi^\star$ is deterministic)

Example 2: $\text{loss}(\pi, x, u) = \mathbf{1}_{\pi(x)=u}$

Example Algorithm: Behavior Cloning.

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost | **cost under imitator** | **cost under expert**

The **Behavior Cloning Algorithm**: $\hat{\pi} \approx \arg\min_{\pi} \sum_{(x,u)\in \text{expert data}} \text{loss}(\pi, x, u)$

Example 1: $\text{loss}(\pi, x, u) = \|u - \pi(x)\|^2$ ($\pi^\star$ is deterministic)

Example 2: $\text{loss}(\pi, x, u) = \mathbf{1}_{\pi(x)=u}$ ($\pi^\star$ is discrete)

Example Algorithm: Behavior Cloning.

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost **cost under imitator** **cost under expert**

The **Behavior Cloning Algorithm**: $\hat{\pi} \approx \arg\min_{\pi} \sum_{(x,u)\in \text{expert data}} \text{loss}(\pi, x, u)$

Example 1: $\text{loss}(\pi, x, u) = \|u - \pi(x)\|^2$ ($\pi^\star$ is deterministic)

Example 2: $\text{loss}(\pi, x, u) = \mathbf{1}_{\pi(x)=u}$ ($\pi^\star$ is discrete)

Example 3: $\text{loss}(\pi, x, u) = \log \pi(u \mid x)$

Example Algorithm: Behavior Cloning.

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost **cost under imitator** **cost under expert**

The **Behavior Cloning Algorithm**: $\hat{\pi} \approx \arg\min_{\pi} \sum_{(x,u)\in \text{expert data}} \text{loss}(\pi, x, u)$

Example 1: $\text{loss}(\pi, x, u) = \|u - \pi(x)\|^2$ ($\pi^\star$ is deterministic)

Example 2: $\text{loss}(\pi, x, u) = \mathbf{1}_{\pi(x)=u}$ ($\pi^\star$ is discrete)

Example 3: $\text{loss}(\pi, x, u) = \log \pi(u \mid x)$ ($\pi^\star$ is discrete, or $\pi^\star(x)$ has density)

Example Algorithm: Behavior Cloning.

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost **cost under imitator** **cost under expert**

The **Behavior Cloning Algorithm**: $\hat{\pi} \approx \arg\min_{\pi} \sum_{(x,u)\in \text{expert data}} \text{loss}(\pi, x, u)$

Example 1: $\text{loss}(\pi, x, u) = \|u - \pi(x)\|^2$ ($\pi^\star$ is deterministic)

Example 2: $\text{loss}(\pi, x, u) = \mathbf{1}_{\pi(x)=u}$ ($\pi^\star$ is discrete)

Example 3: $\text{loss}(\pi, x, u) = \log \pi(u \mid x)$ ($\pi^\star$ is discrete, or $\pi^\star(x)$ has density)

Example 4: $\text{loss}(\pi, x, u) =$ (**Score Matching**)

Example Algorithm: Behavior Cloning.

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost **cost under imitator** **cost under expert**

The **Behavior Cloning Algorithm**: $\hat{\pi} \approx \arg\min_{\pi} \sum_{(x,u)\in \text{expert data}} \text{loss}(\pi, x, u)$

Example 1: $\text{loss}(\pi, x, u) = \|u - \pi(x)\|^2$ ($\pi^\star$ is deterministic)

Example 2: $\text{loss}(\pi, x, u) = \mathbf{1}_{\pi(x)=u}$ ($\pi^\star$ is discrete)

Example 3: $\text{loss}(\pi, x, u) = \log \pi(u \mid x)$ ($\pi^\star$ is discrete, or $\pi^\star(x)$ has density)

Example 4: $\text{loss}(\pi, x, u) =$ (**Score Matching**) (popular in robotics)

Example Algorithm: Behavior Cloning.

Mininimize $\mathcal{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost — **cost under imitator** — **cost under expert**

The **Behavior Cloning Algorithm**: $\hat{\pi} \approx \arg\min_{\pi} \sum_{(x,u)\in \text{expert data}} \text{loss}(\pi, x, u)$

Example Algorithm: Behavior Cloning.

Mininimize $\mathcal{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost | **cost under imitator** | **cost under expert**

The **Behavior Cloning Algorithm**: $\hat{\pi} \approx \arg\min_{\pi} \sum_{(x,u) \in \text{expert data}} \text{loss}(\pi, x, u)$

Compare to $\mathcal{R}_{\text{expert}}(\hat{\pi}; \pi^\star) = \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} \text{loss}(\hat{\pi}, \pi^\star, u_t)]$

loss of imitator under expert distribution

Example Algorithm: Behavior Cloning.

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost — **cost under imitator** — **cost under expert**

The **Behavior Cloning Algorithm**: $\hat{\pi} \approx \arg\min_{\pi} \sum_{(x,u) \in \text{expert data}} \text{loss}(\pi, x, u)$

Compare to $\mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star) = \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} \text{loss}(\hat{\pi}, \pi^\star, u_t)]$

loss of imitator under expert distribution

This can be minimized with **pure supervised learning**

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost **cost under imitator** **cost under expert**

Compare to $\mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star) = \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} \text{loss}(\hat{\pi}, \pi^\star, u_t)]$

loss of imitator under expert distribution

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost **cost under imitator** **cost under expert**

Compare to $\mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star) = \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} \text{loss}(\hat{\pi}, \pi^\star, u_t)]$

loss of imitator under expert distribution

The gap between these two is called the **compounding error problem.**

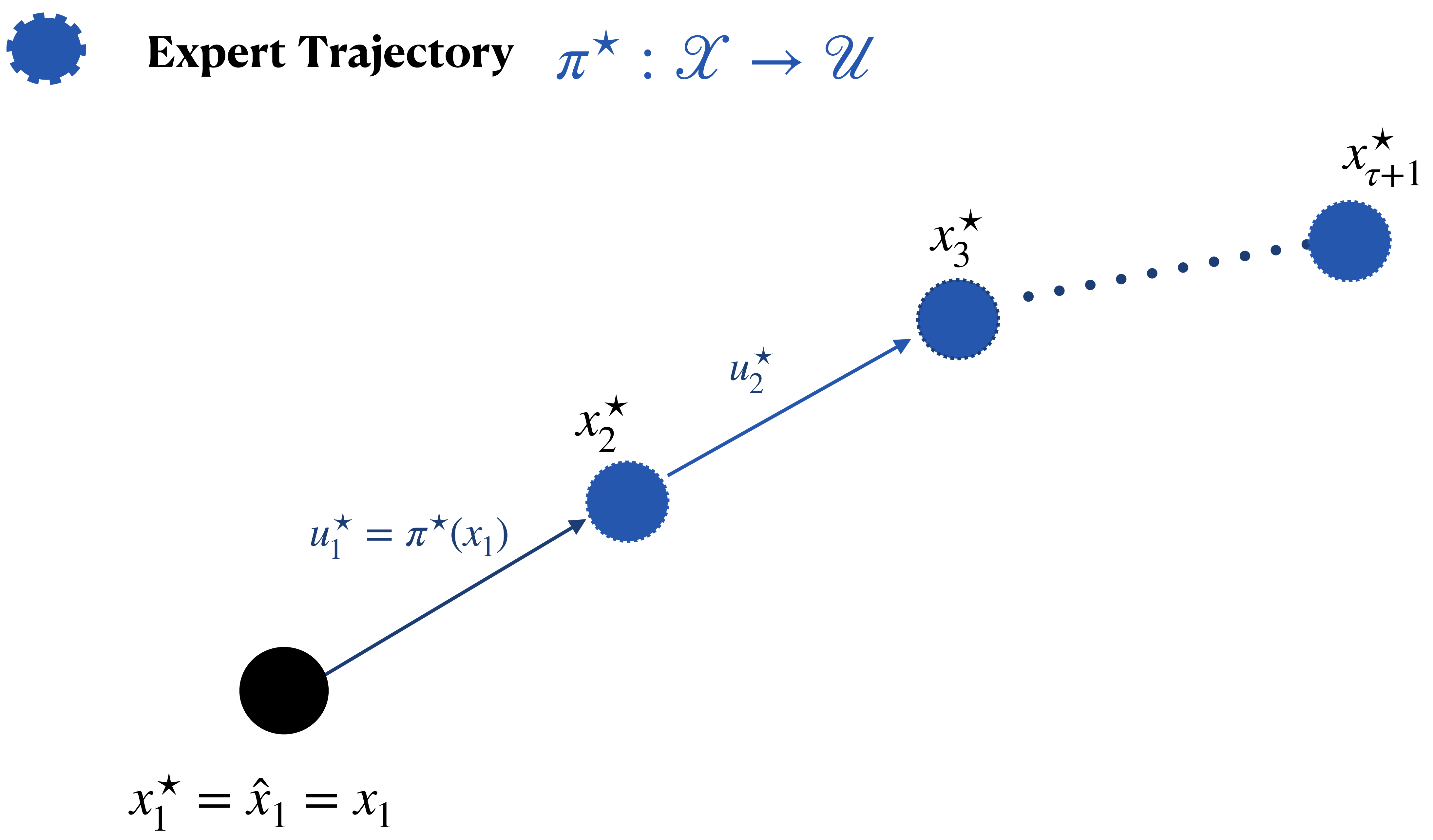
The Compounding Error Problem.

The Compounding Error Problem.

$$\mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star) = \mathbb{E}_{\pi^\star}\left[\sum_{h=1}^{H} \text{loss}(\hat{\pi}, \pi^\star, u_t)\right]$$

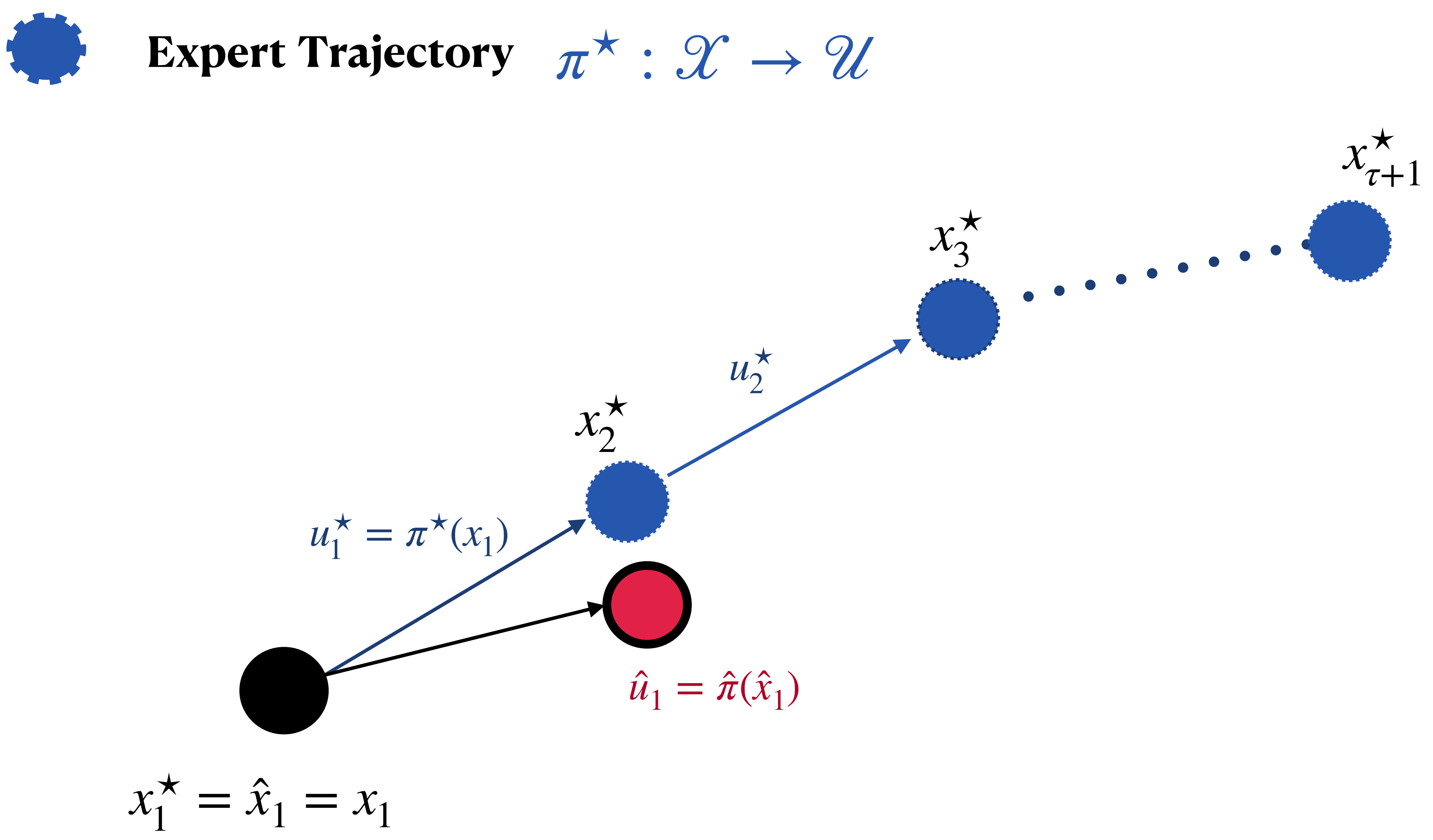
The Compounding Error Problem.

$$\mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star) = \mathbb{E}_{\pi^\star}[\,\textstyle\sum_{h=1}^{H} \text{loss}(\hat{\pi}, \pi^\star, u_t)]$$



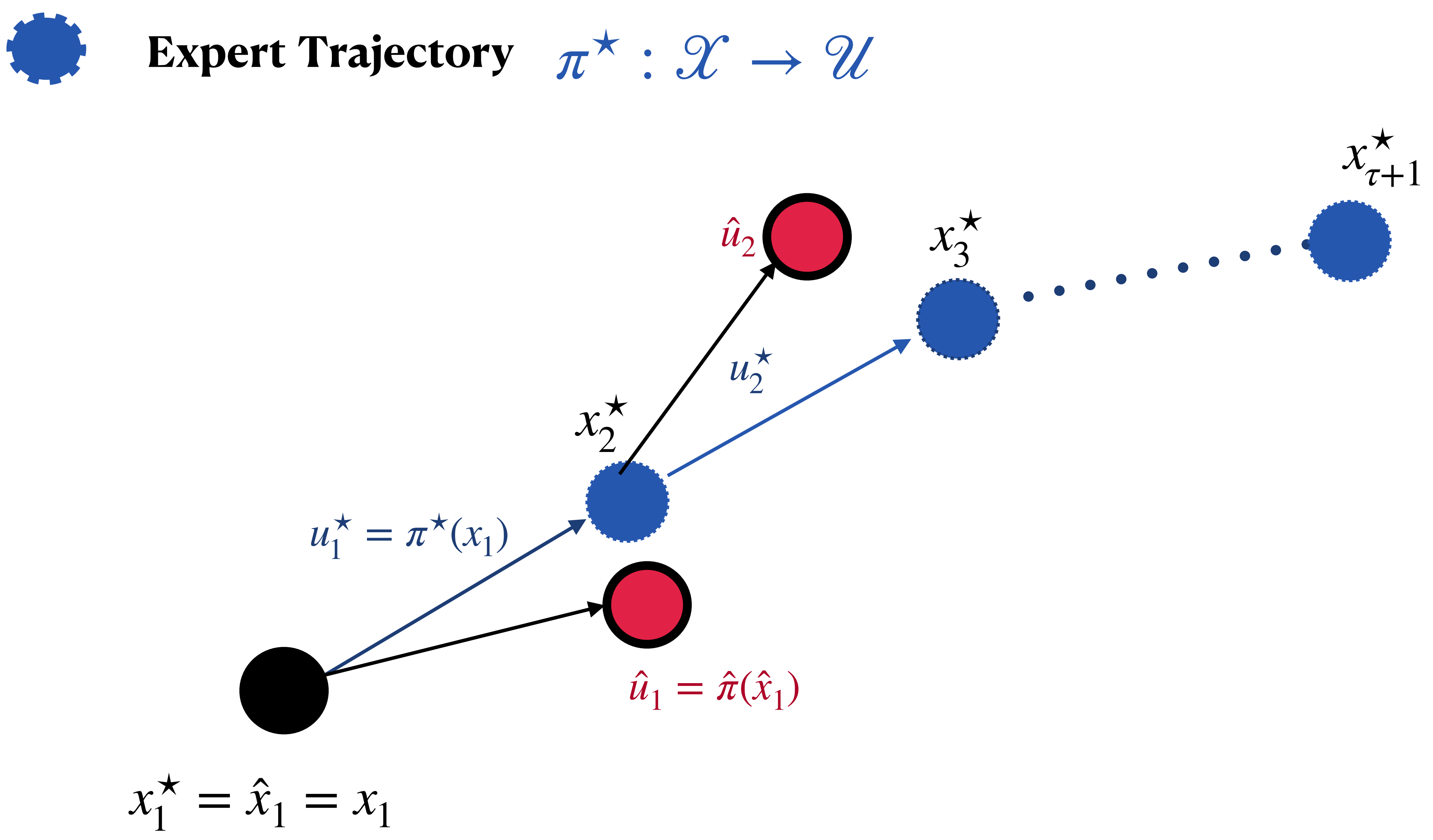
The Compounding Error Problem.

$$\mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star) = \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} \text{loss}(\hat{\pi}, \pi^\star, u_t)]$$



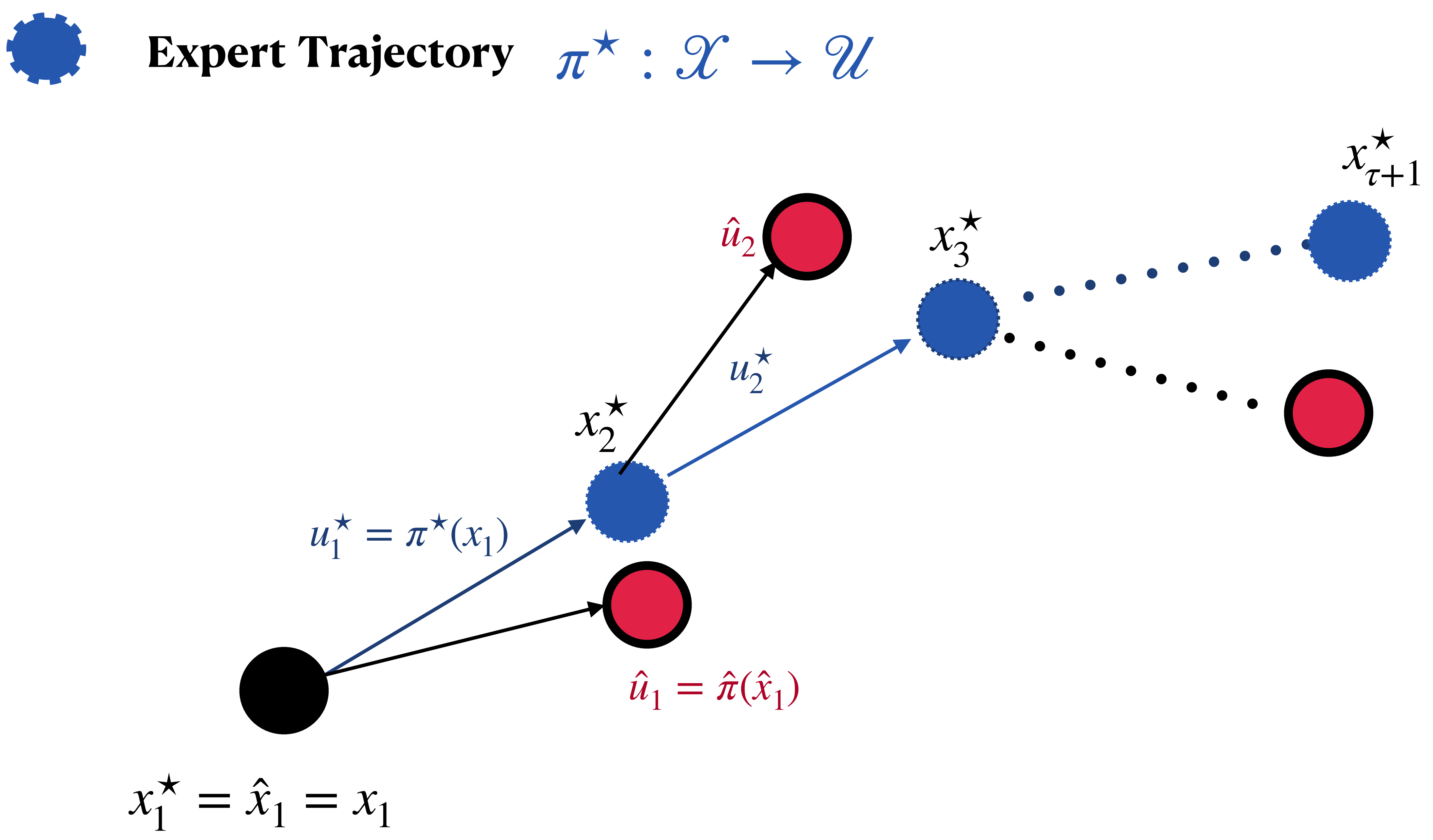
The Compounding Error Problem.

$$\mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star) = \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} \text{loss}(\hat{\pi}, \pi^\star, u_t)]$$



The Compounding Error Problem.

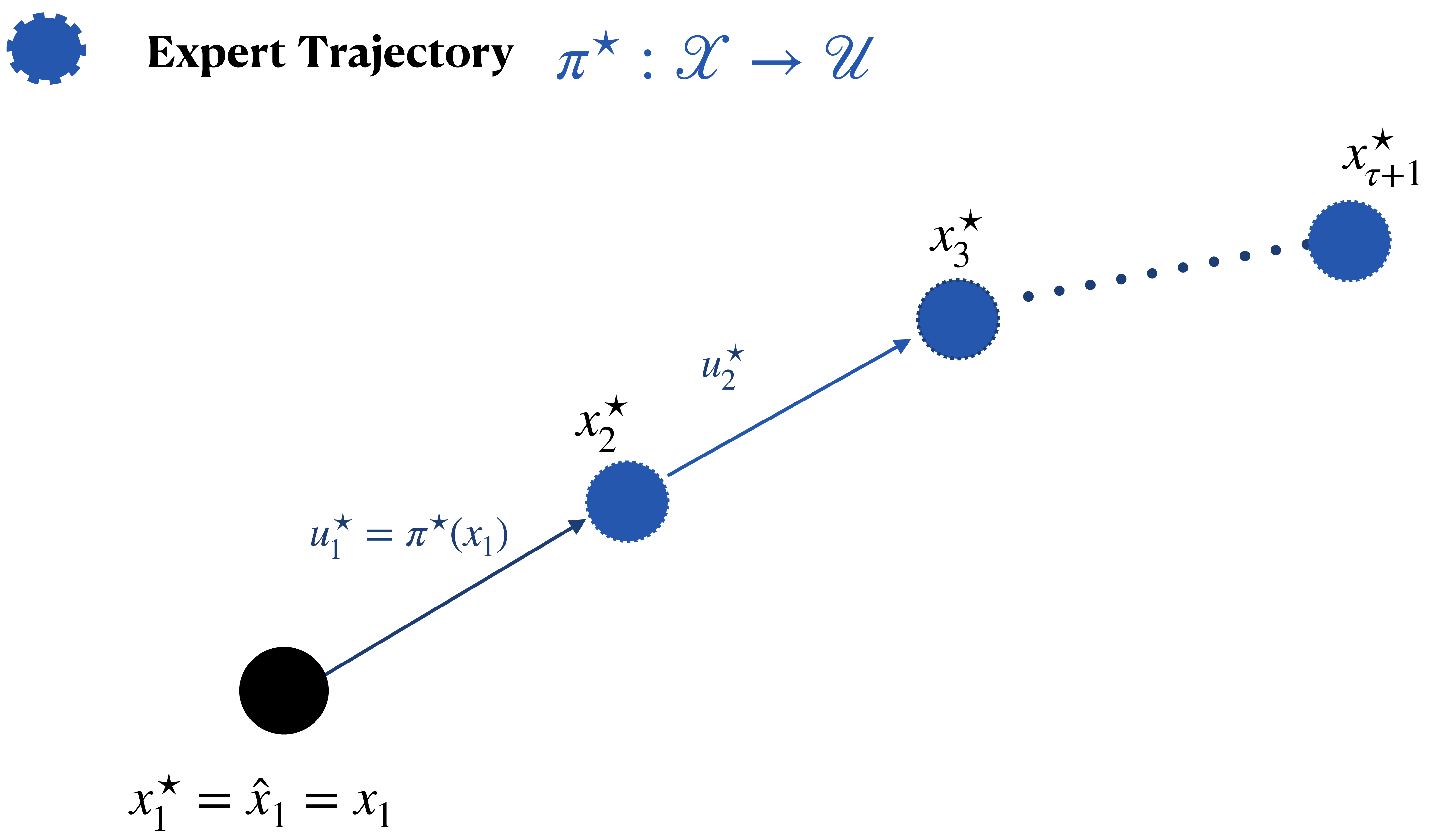
$$\mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star) = \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} \text{loss}(\hat{\pi}, \pi^\star, u_t)]$$



The Compounding Error Problem.

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

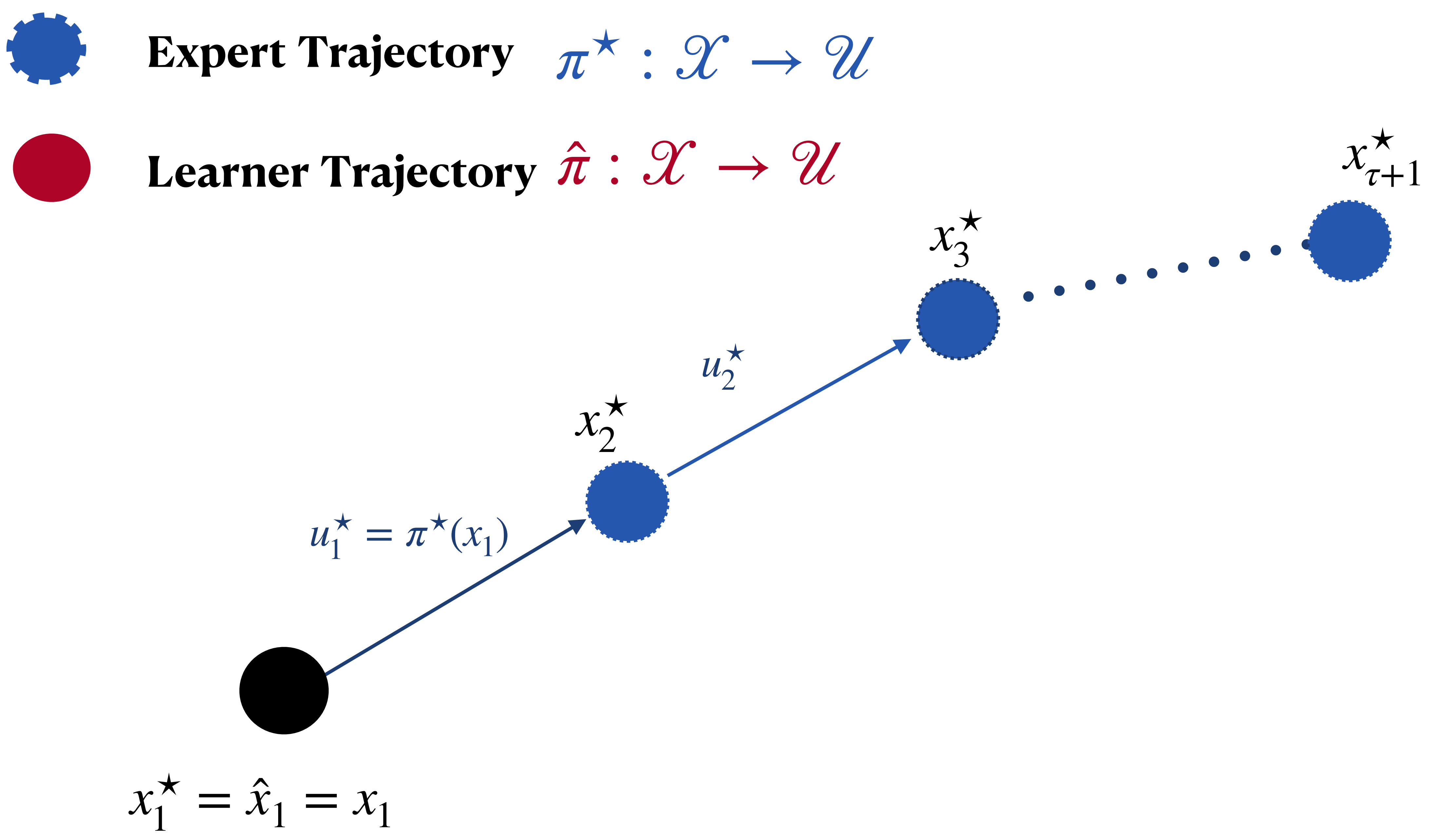
excess cost | **cost under imitator** | **cost under expert**



The Compounding Error Problem.

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost | **cost under imitator** | **cost under expert**



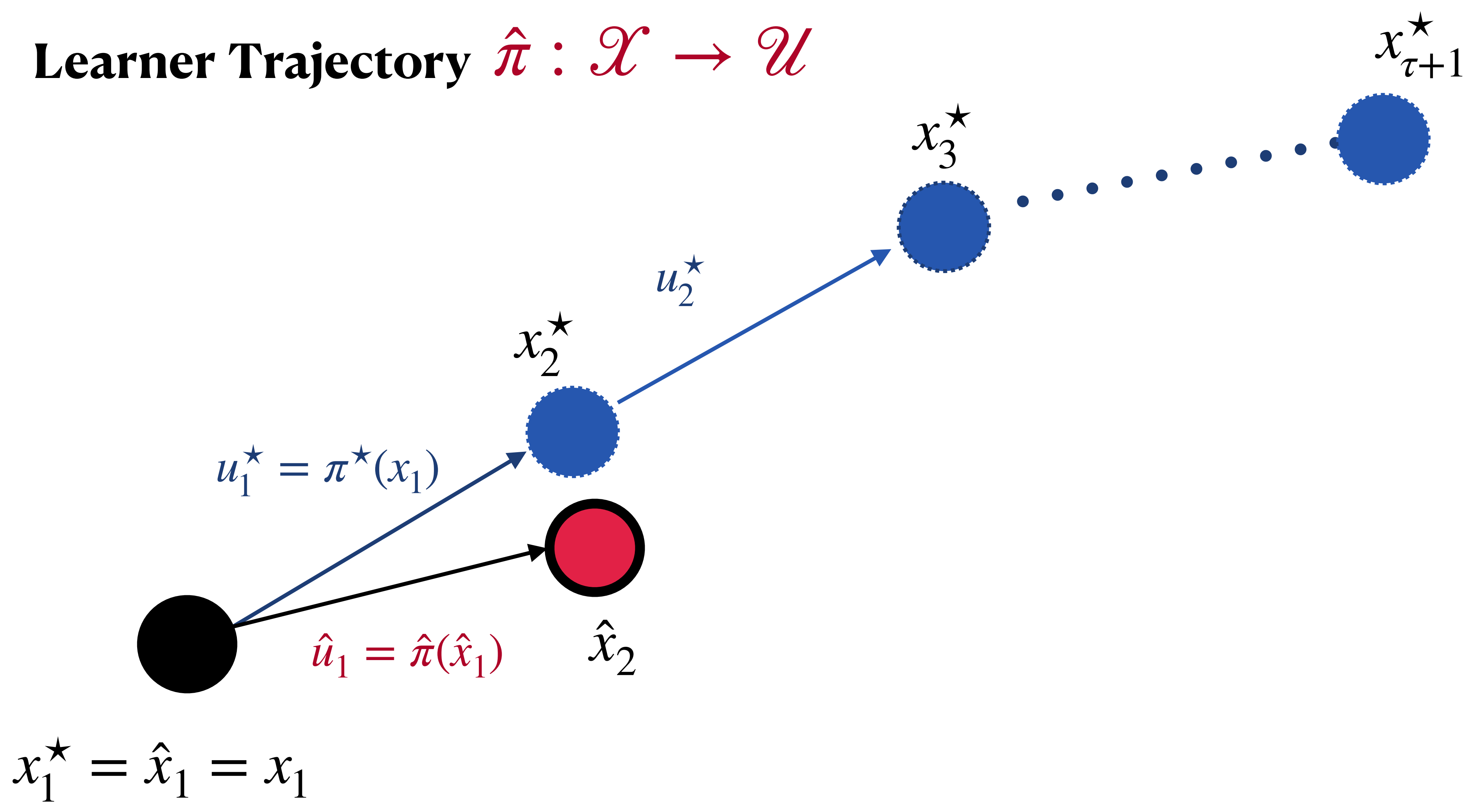
The Compounding Error Problem.

Mininimize $\mathcal{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost — **cost under imitator** — **cost under expert**

Expert Trajectory $\pi^\star : \mathcal{X} \to \mathcal{U}$

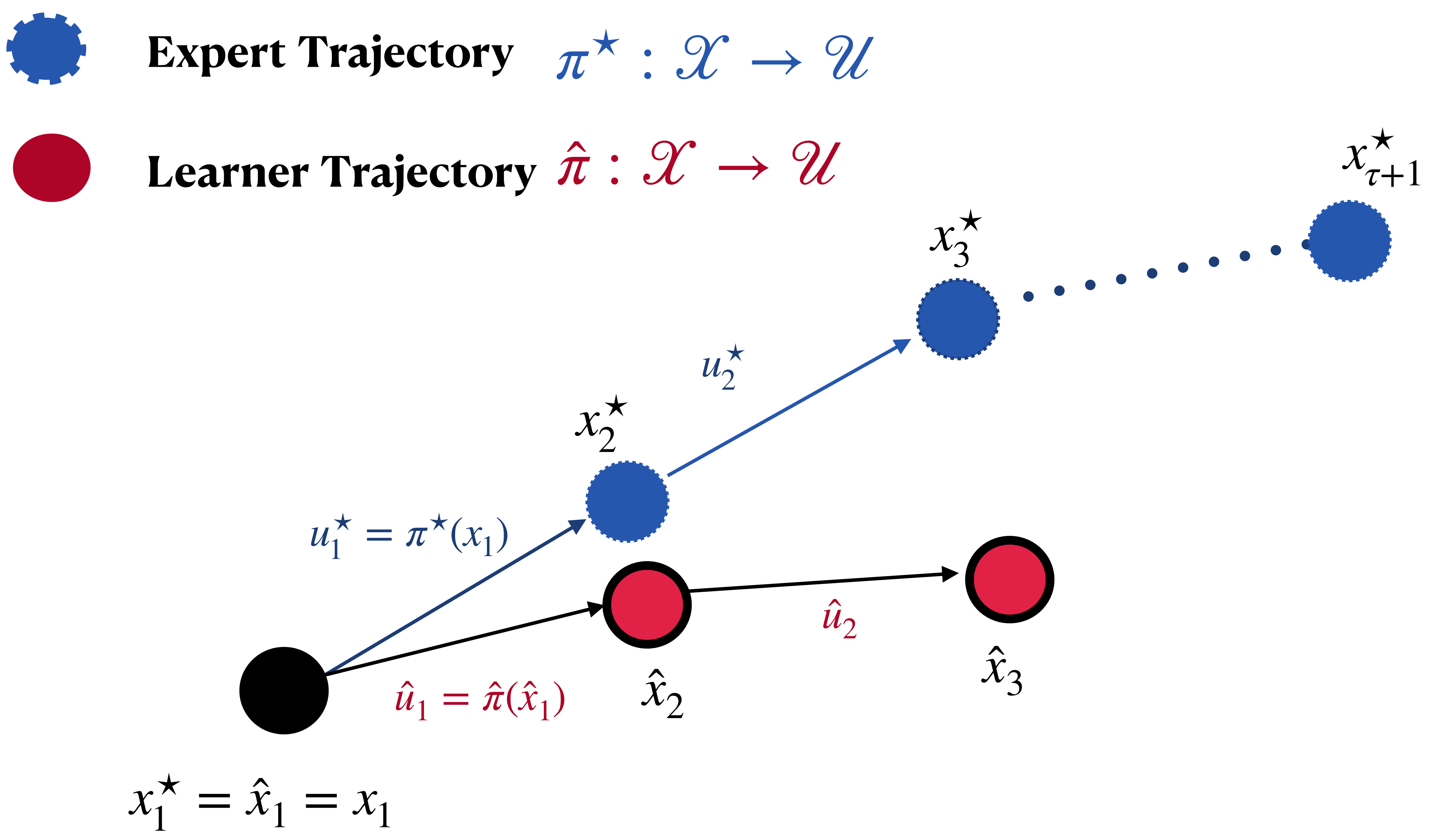
Learner Trajectory $\hat{\pi} : \mathcal{X} \to \mathcal{U}$



The Compounding Error Problem.

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

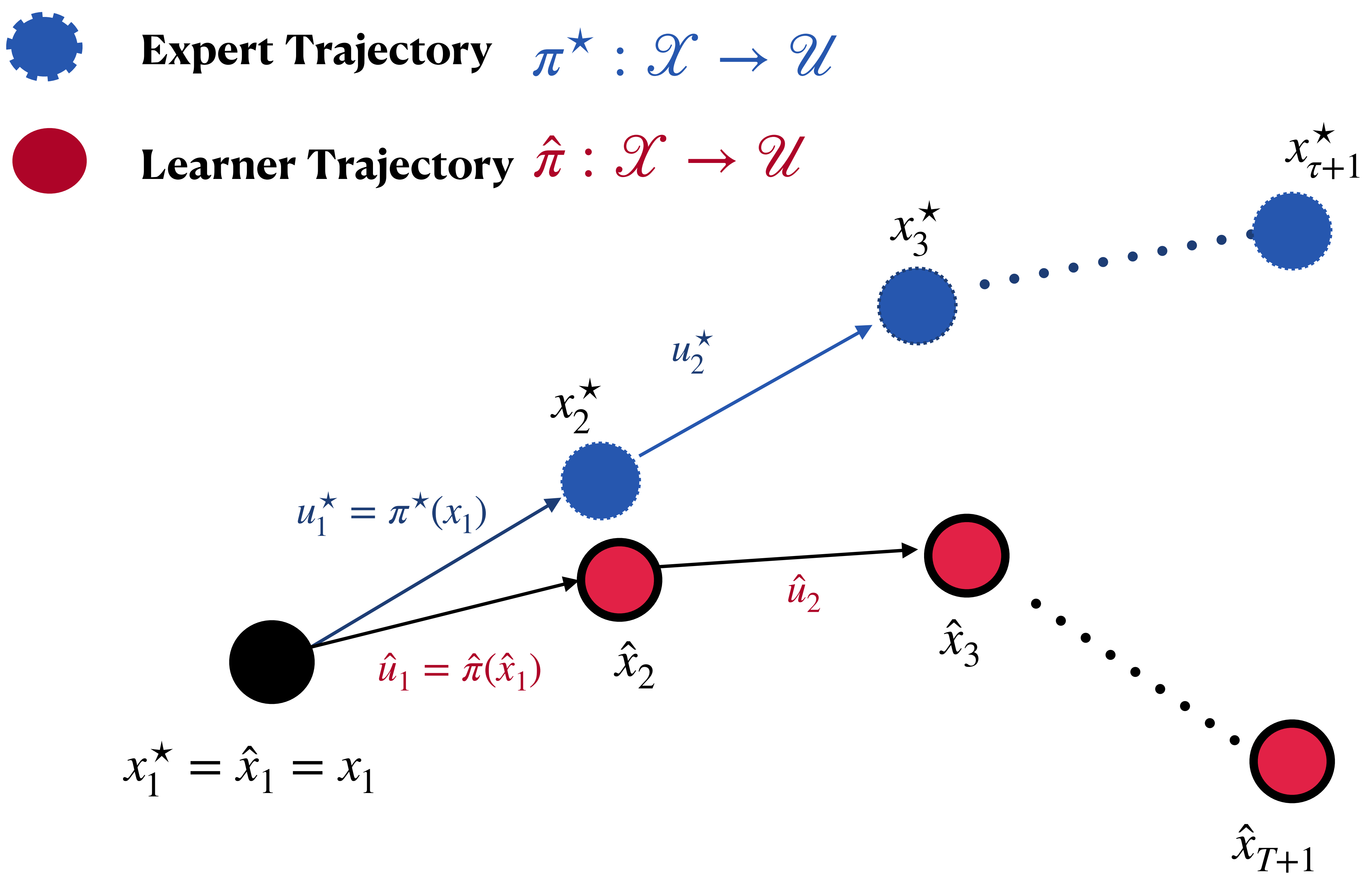
excess cost **cost under imitator** **cost under expert**



The Compounding Error Problem.

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

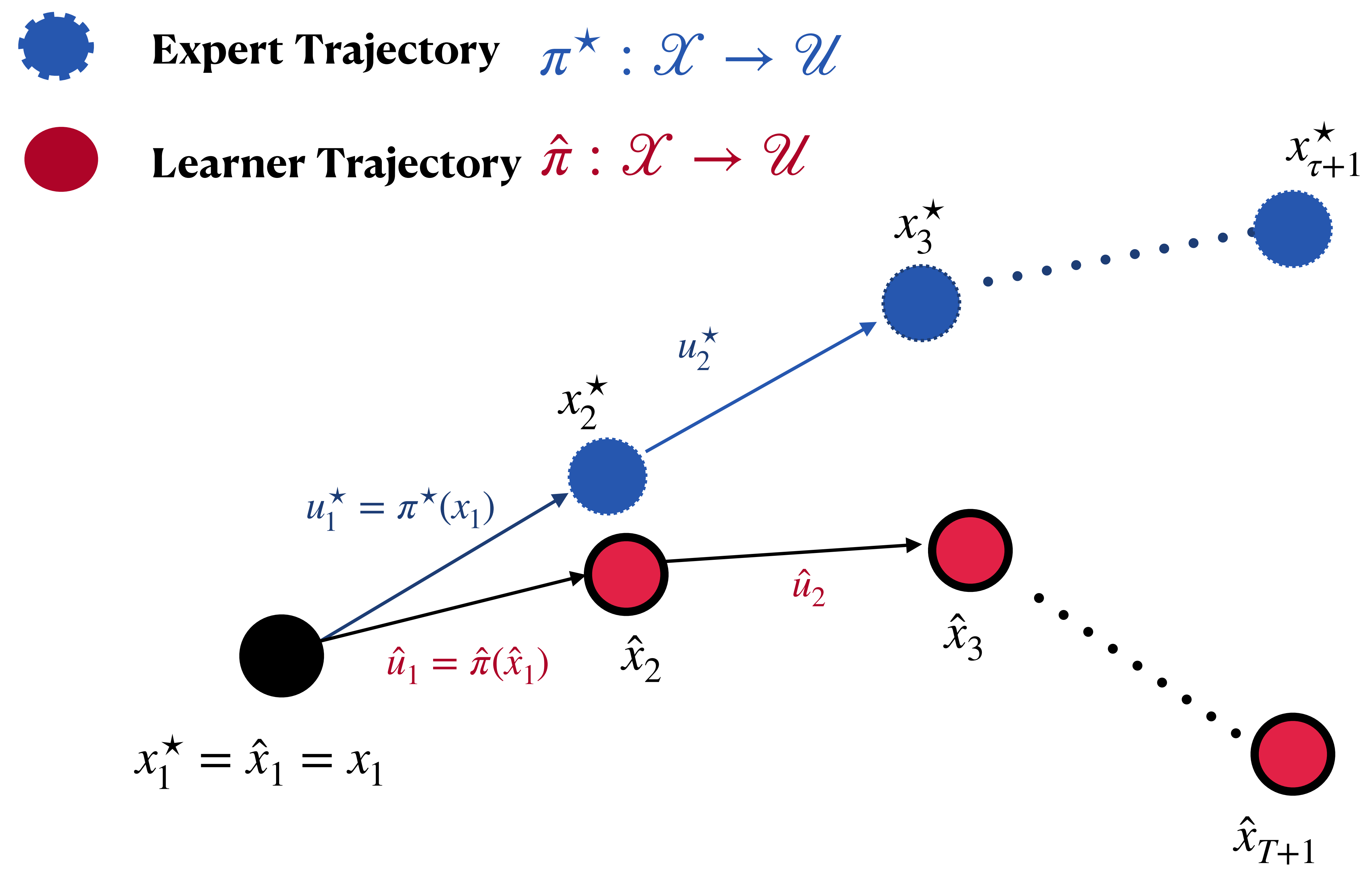
excess cost — **cost under imitator** — **cost under expert**



The Compounding Error Problem.

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost — **cost under imitator** — **cost under expert**

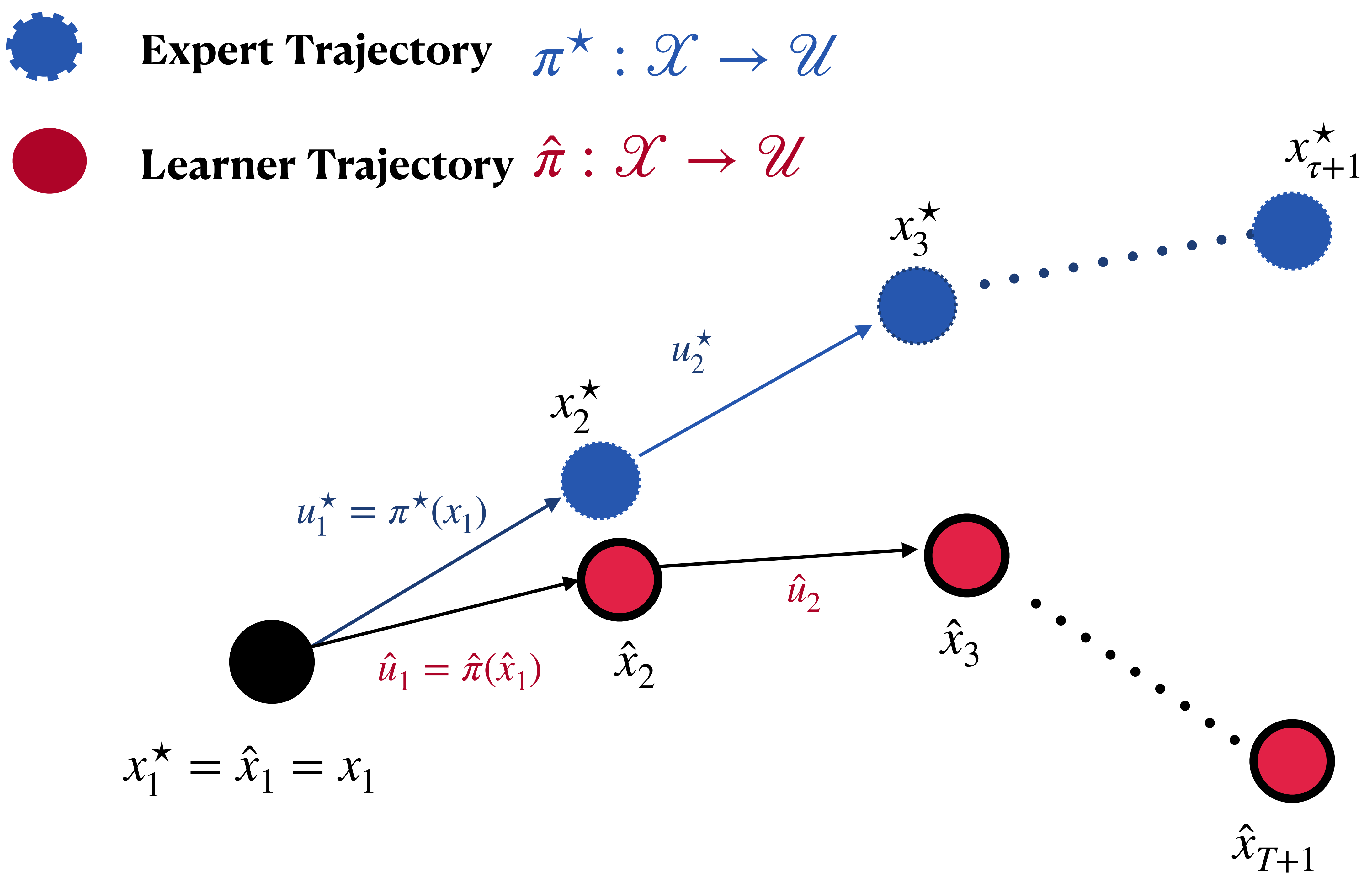


Challenge A: Error accumulates over time steps, larger with larger **H**.

The Compounding Error Problem.

Mininimize $\mathscr{R}_c(\hat{\pi};\pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t,u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t,u_t)]$

excess cost **cost under imitator** **cost under expert**



Challenge A: Error accumulates over time steps, larger with larger **H**.

Challenge B: After error has accumulated, we are now **out of distribution.**

Compounding In the Discrete World 📚

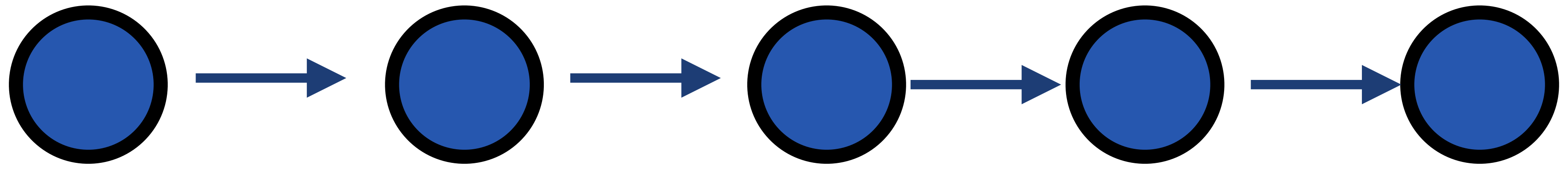
Mininimize $\mathcal{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost **cost under imitator** **cost under expert**

Compounding In the Discrete World 📚

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

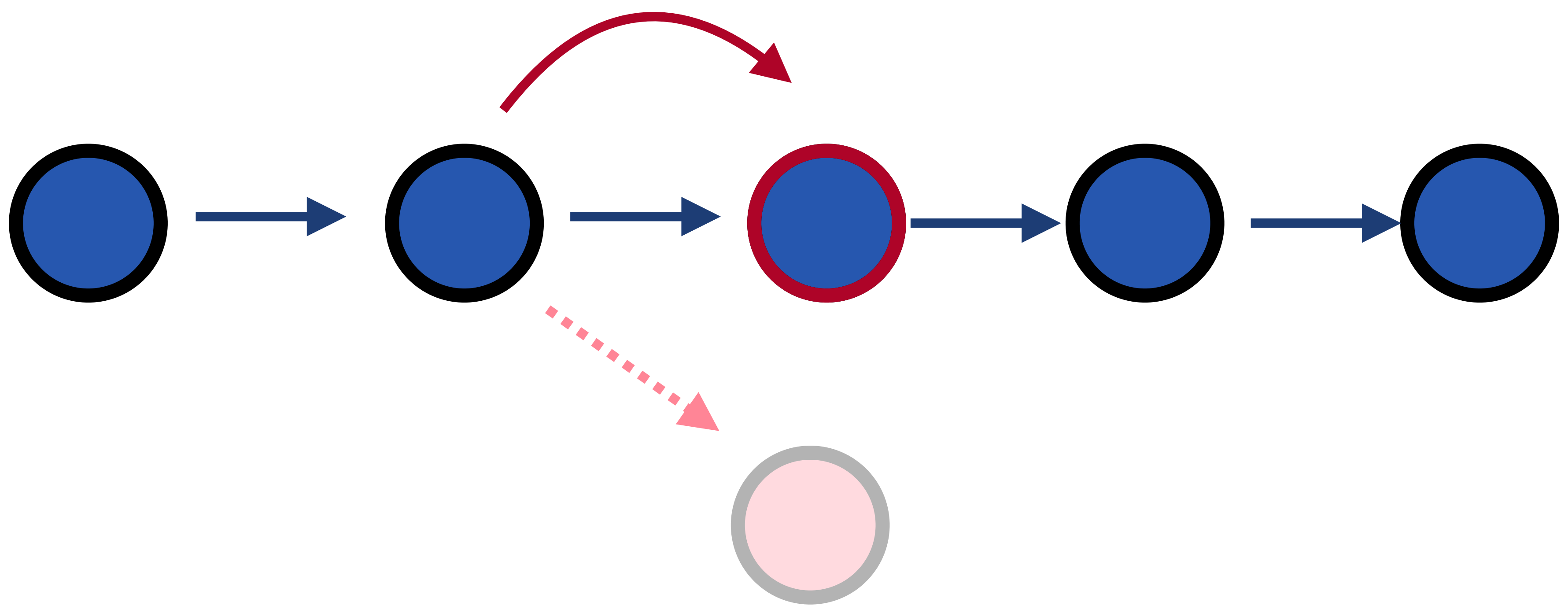
excess cost **cost under imitator** **cost under expert**



Compounding In the Discrete World 📚

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

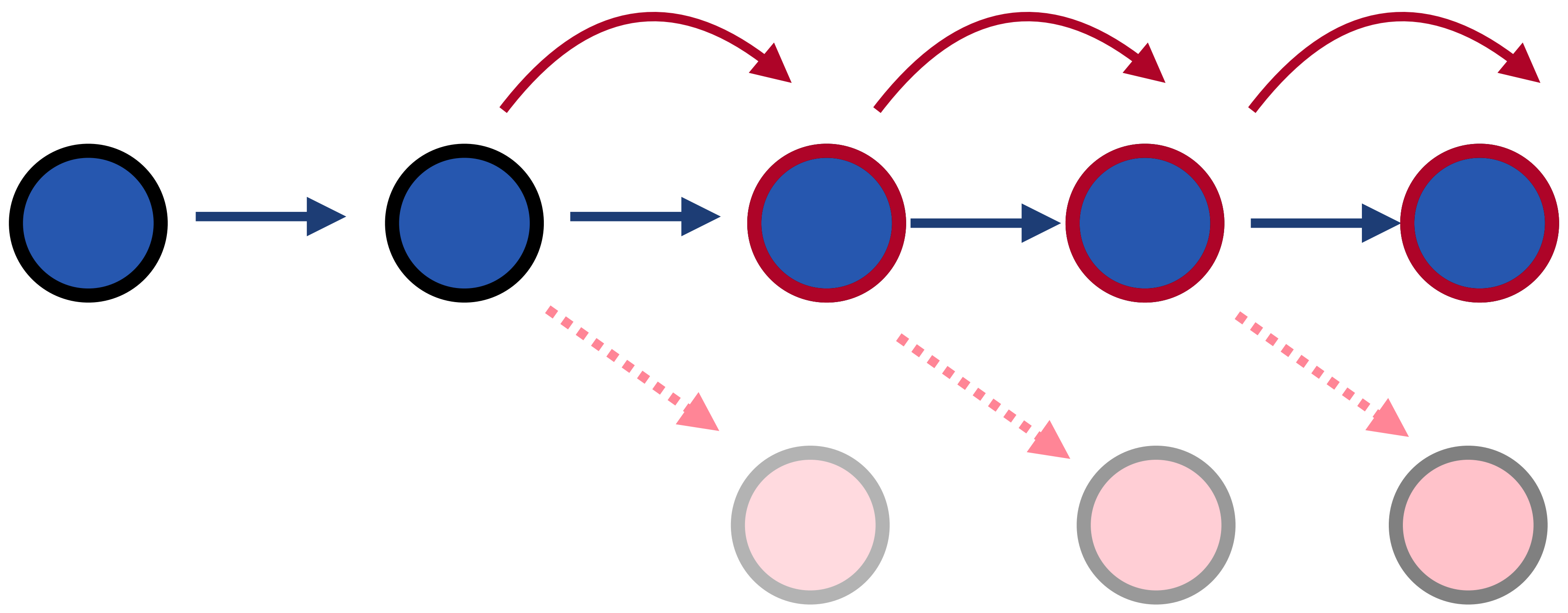
excess cost | **cost under imitator** | **cost under expert**



Compounding In the Discrete World 📚

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

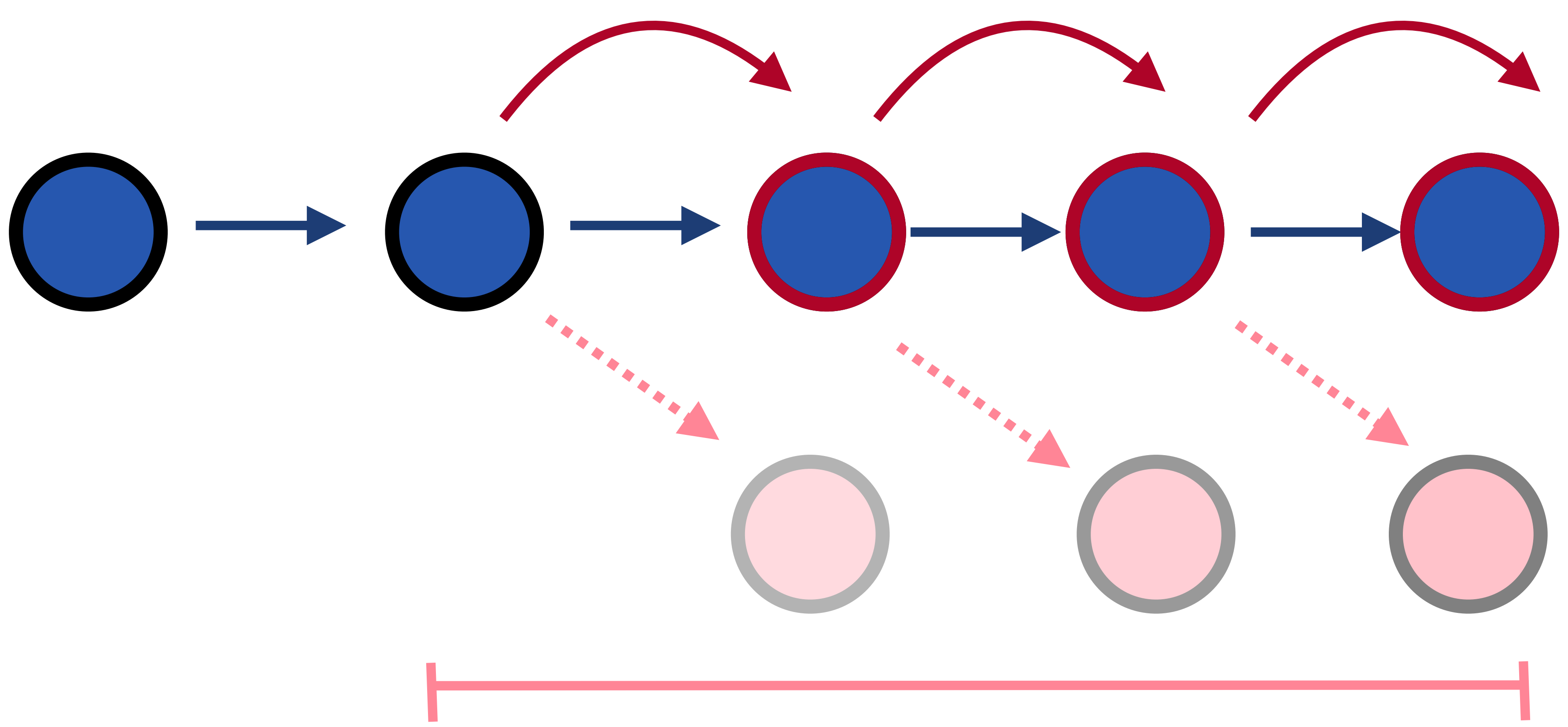
excess cost — **cost under imitator** — **cost under expert**



Compounding In the Discrete World 📚

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost **cost under imitator** **cost under expert**

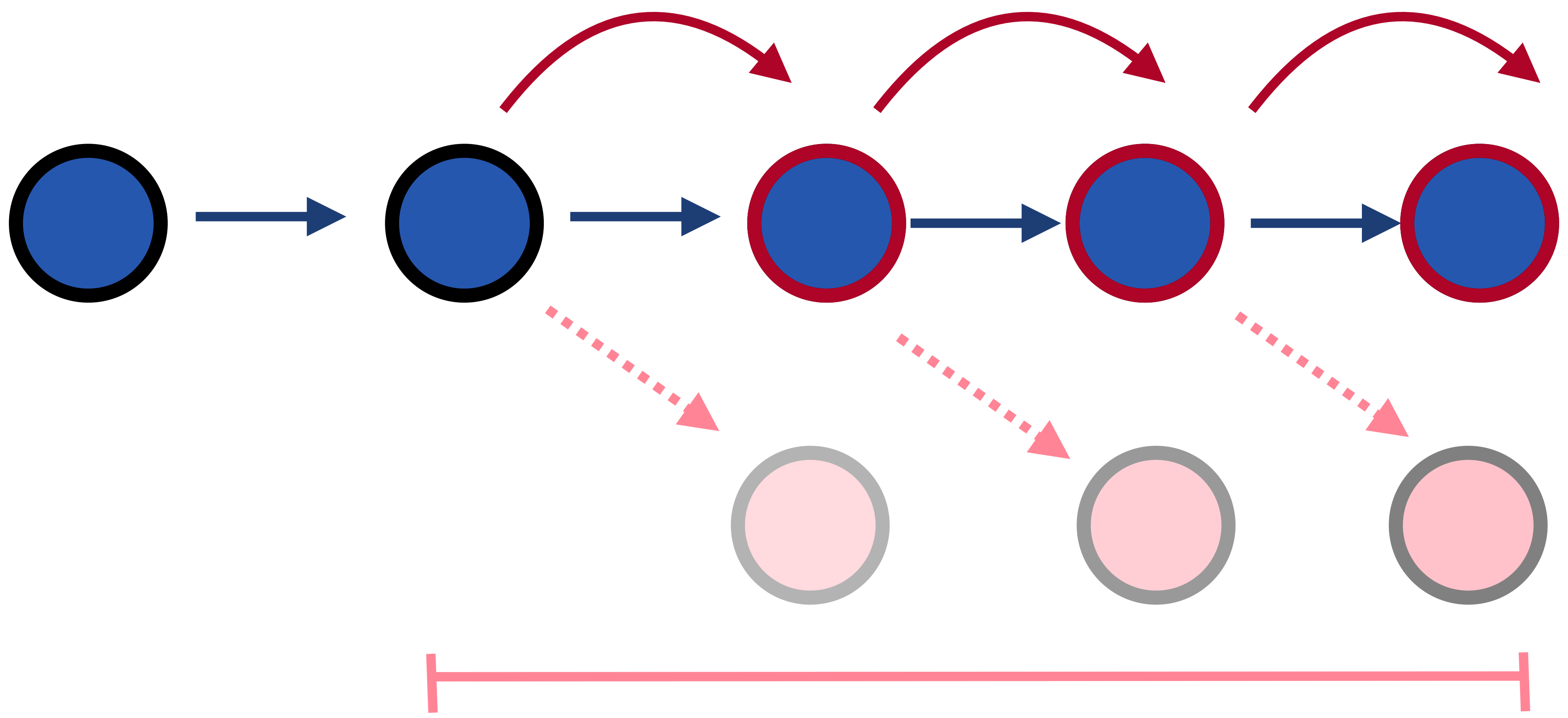


probabilistic errors** accumulate **at most linearly.

Compounding In the Discrete World 📚

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost — **cost under imitator** — **cost under expert**



probabilistic errors *accumulate* ***at most linearly.***

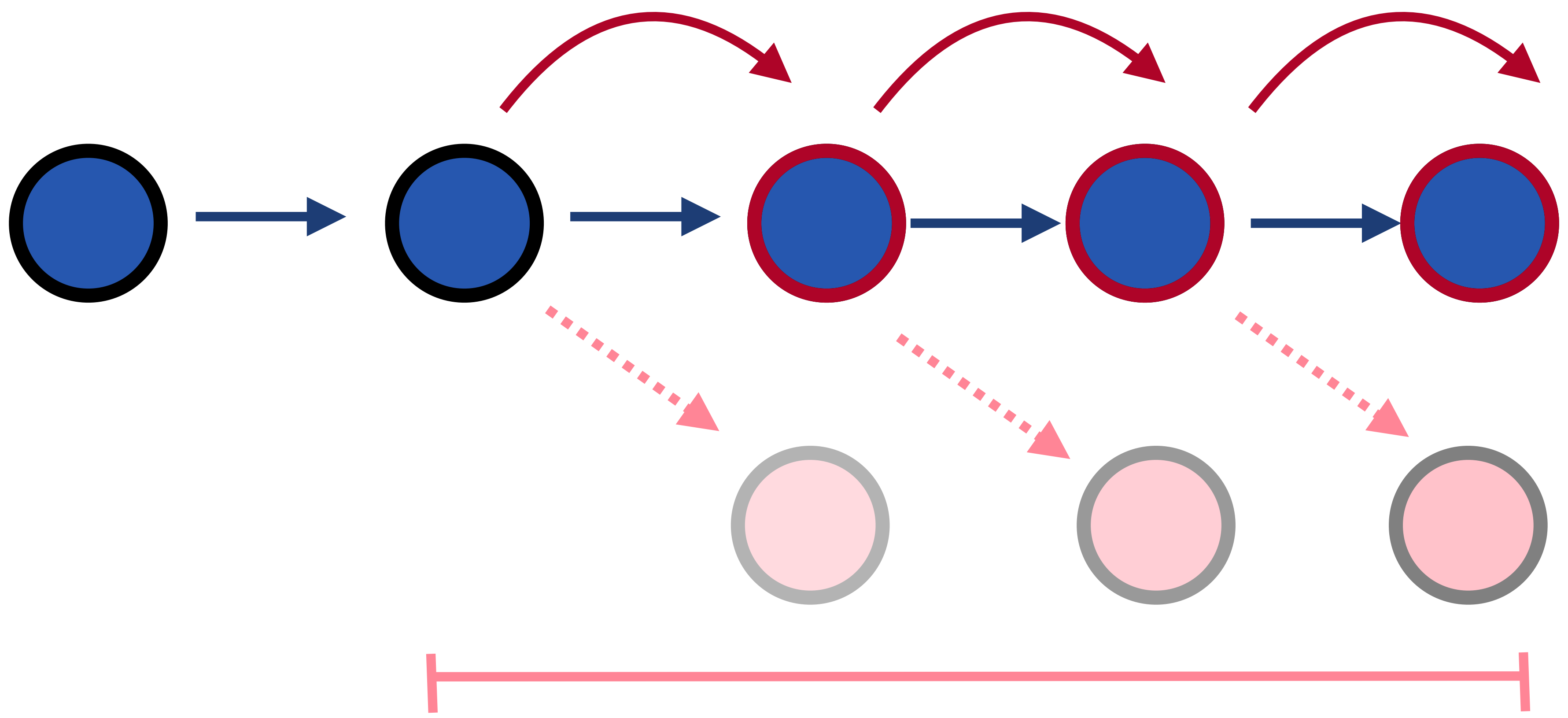
Theorem: If $\text{loss}(\pi, x, u) = \mathbf{1}_{\pi(x)=u}$ is the **zero-one loss**, and that $c(x, u)$ is bounded in [0,1]. Then, for all $(\hat{\pi}; \pi^\star)$

$$\mathscr{R}_c(\hat{\pi}; \pi^\star) \leq H \cdot \mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star)$$

Compounding In the Discrete World 📚

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost — **cost under imitator** — **cost under expert**



probabilistic errors *accumulate* ***at most linearly.***

Theorem: If $\text{loss}(\pi, x, u) = \mathbf{1}_{\pi(x)=u}$ is the **zero-one loss**, and that $c(x, u)$ is bounded in $[0,1]$. Then, for all $(\hat{\pi}; \pi^\star)$

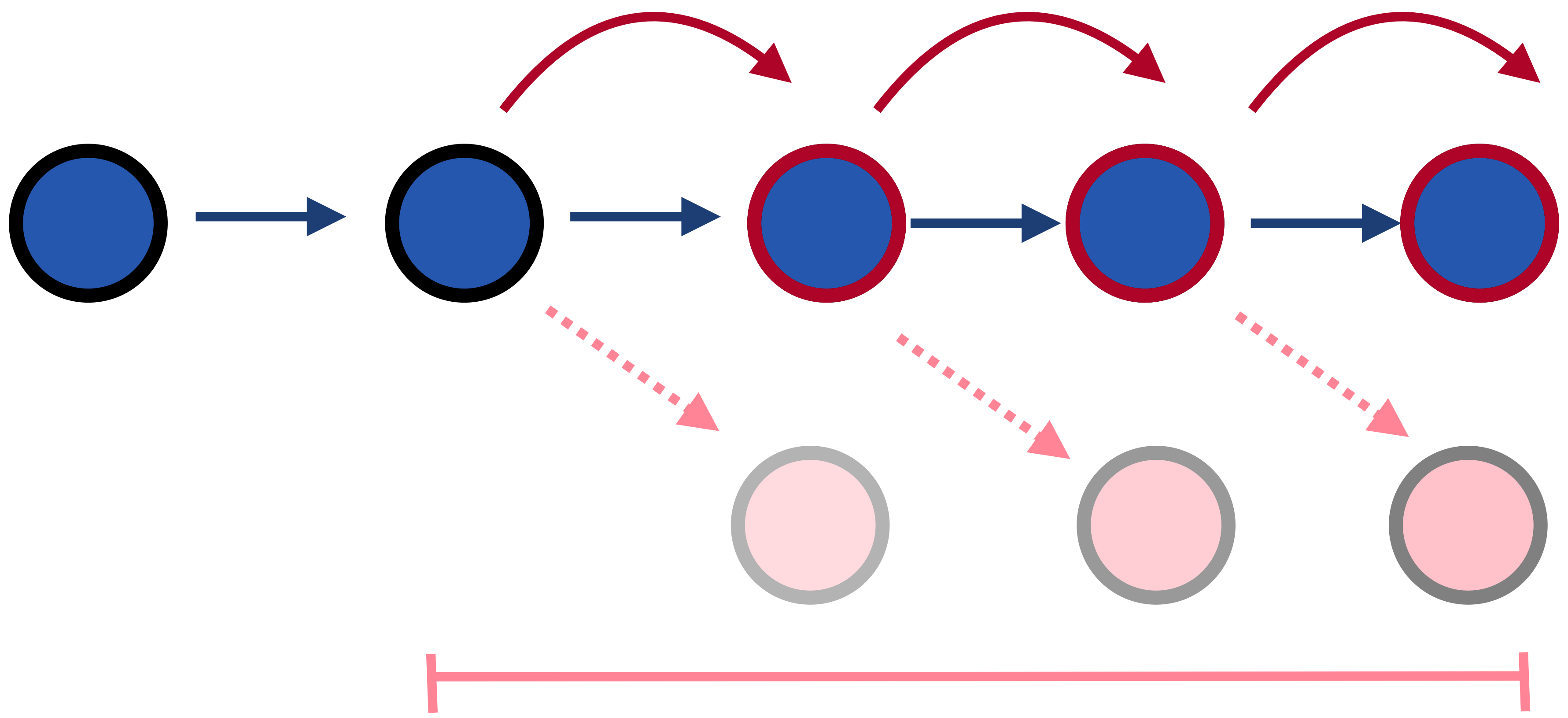
$$\mathscr{R}_c(\hat{\pi}; \pi^\star) \leq H \cdot \mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star)$$

Improvements due to Foster et al. '24 for the **Log Loss.**

Compounding In the Discrete World 📚

Mininimize $\mathscr{R}_c(\hat{\pi}; \pi^\star) = \mathbb{E}_{\hat{\pi}}[\sum_{h=1}^{H} c(x_t, u_t)] - \mathbb{E}_{\pi^\star}[\sum_{h=1}^{H} c(x_t, u_t)]$

excess cost — **cost under imitator** — **cost under expert**



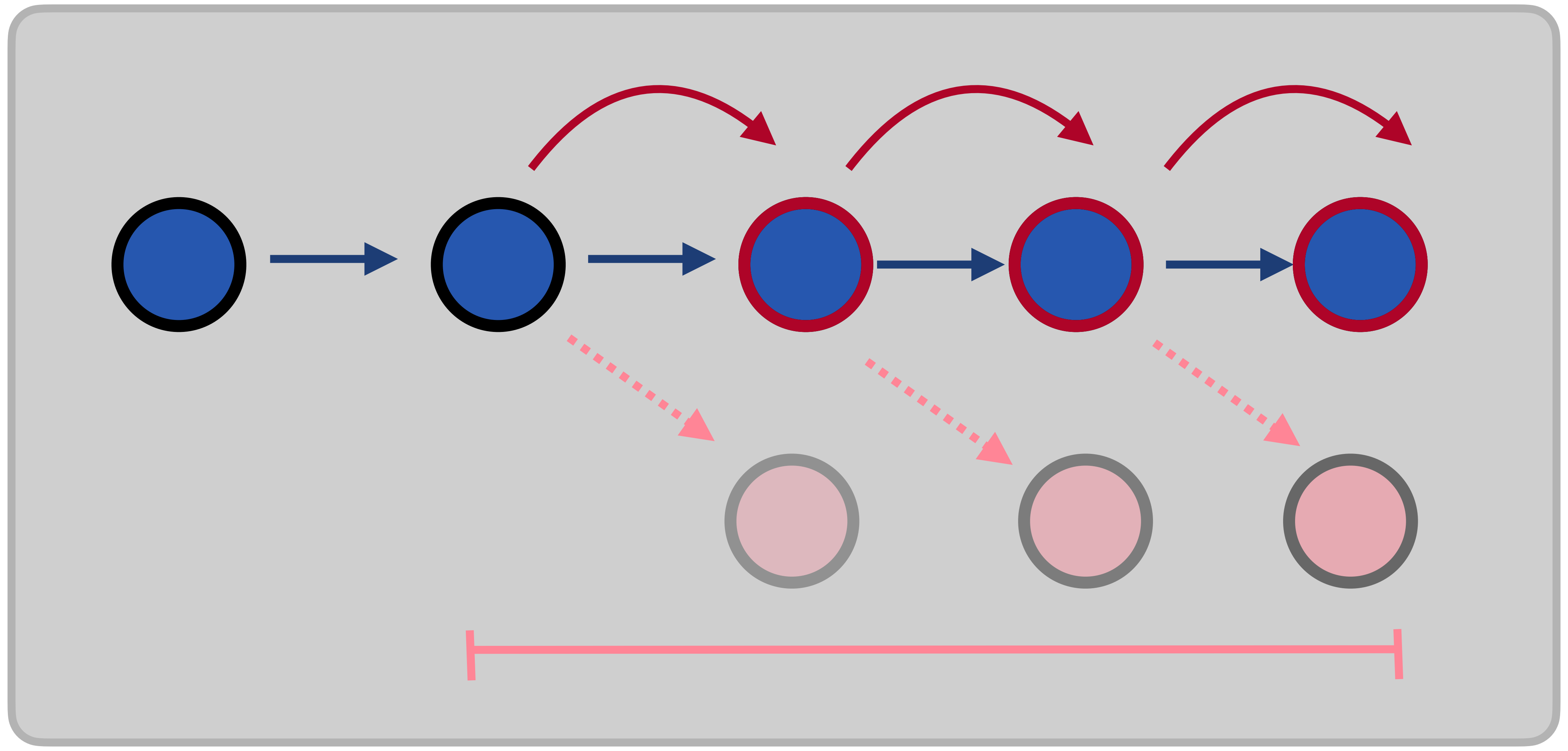
Theorem: If $\text{loss}(\pi, x, u) = \mathbf{1}_{\pi(x)=u}$ is the **zero-one loss**, and that $c(x,u)$ is bounded in $[0,1]$. Then, for all $(\hat{\pi}; \pi^\star)$

$$\mathscr{R}_c(\hat{\pi}; \pi^\star) \leq H \cdot \mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star)$$

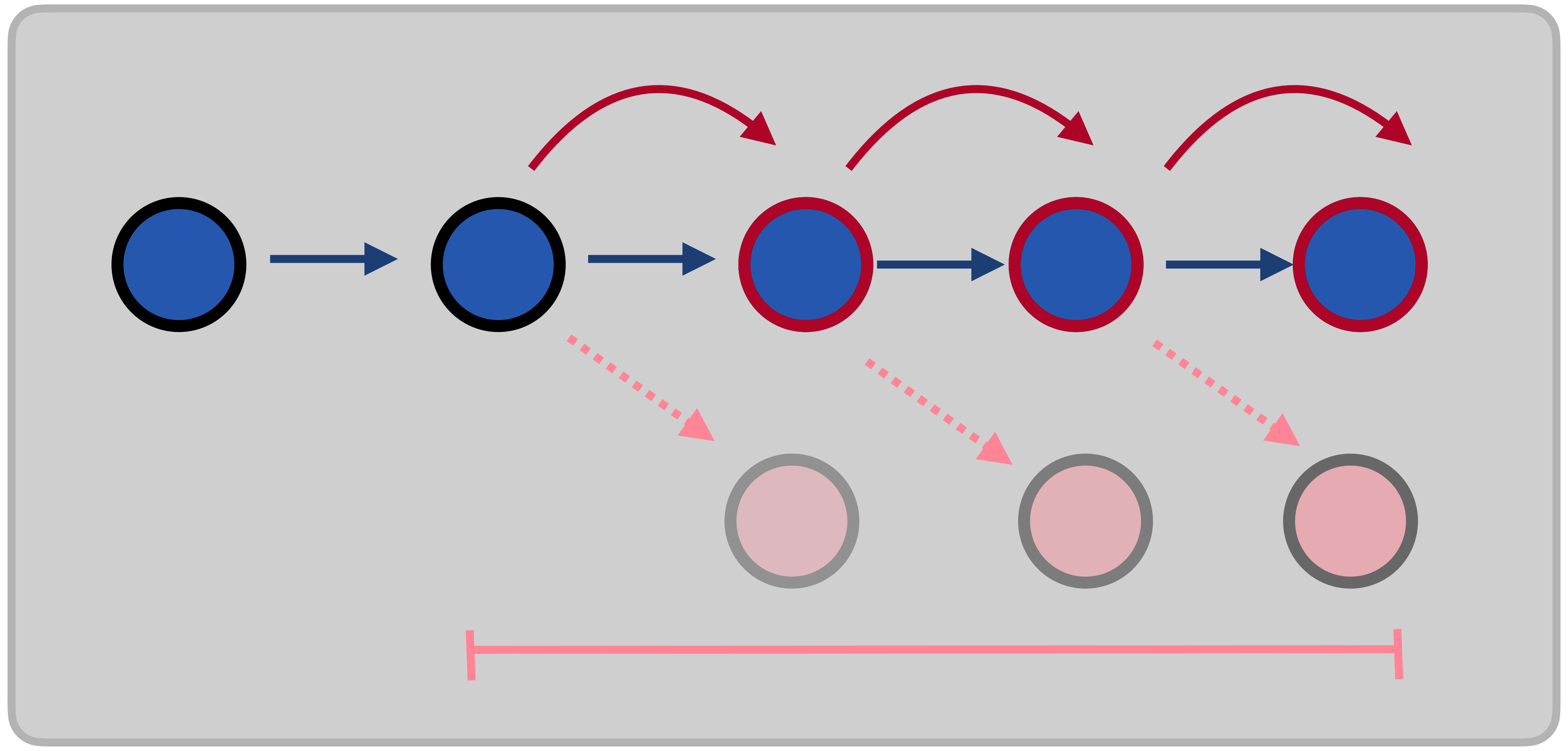
Crucially relies probabilistic errors + ***discreteness of actions!***

Compounding in Physical World 🤖?

Compounding in Physical World 🤖?

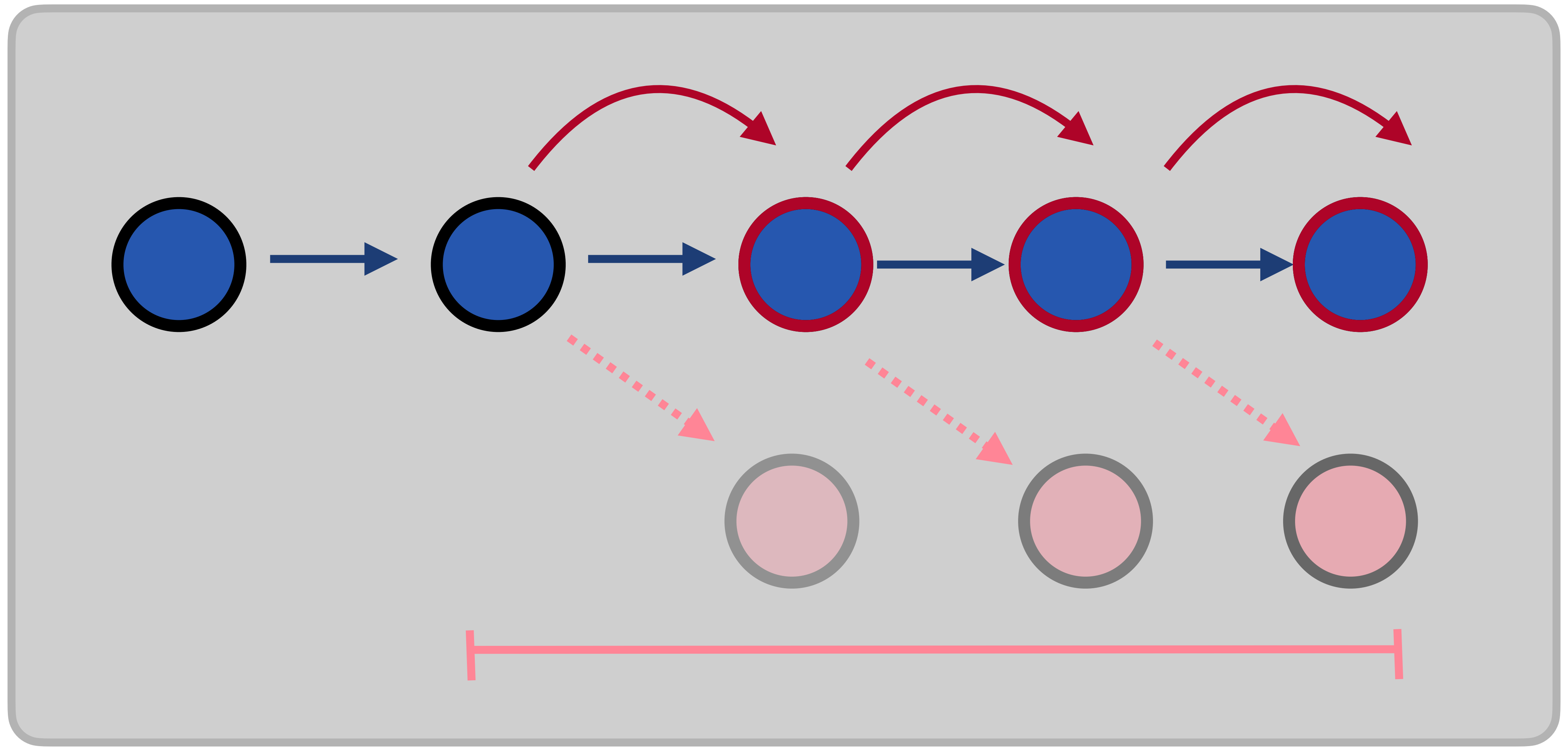


Compounding in Physical World 🤖?

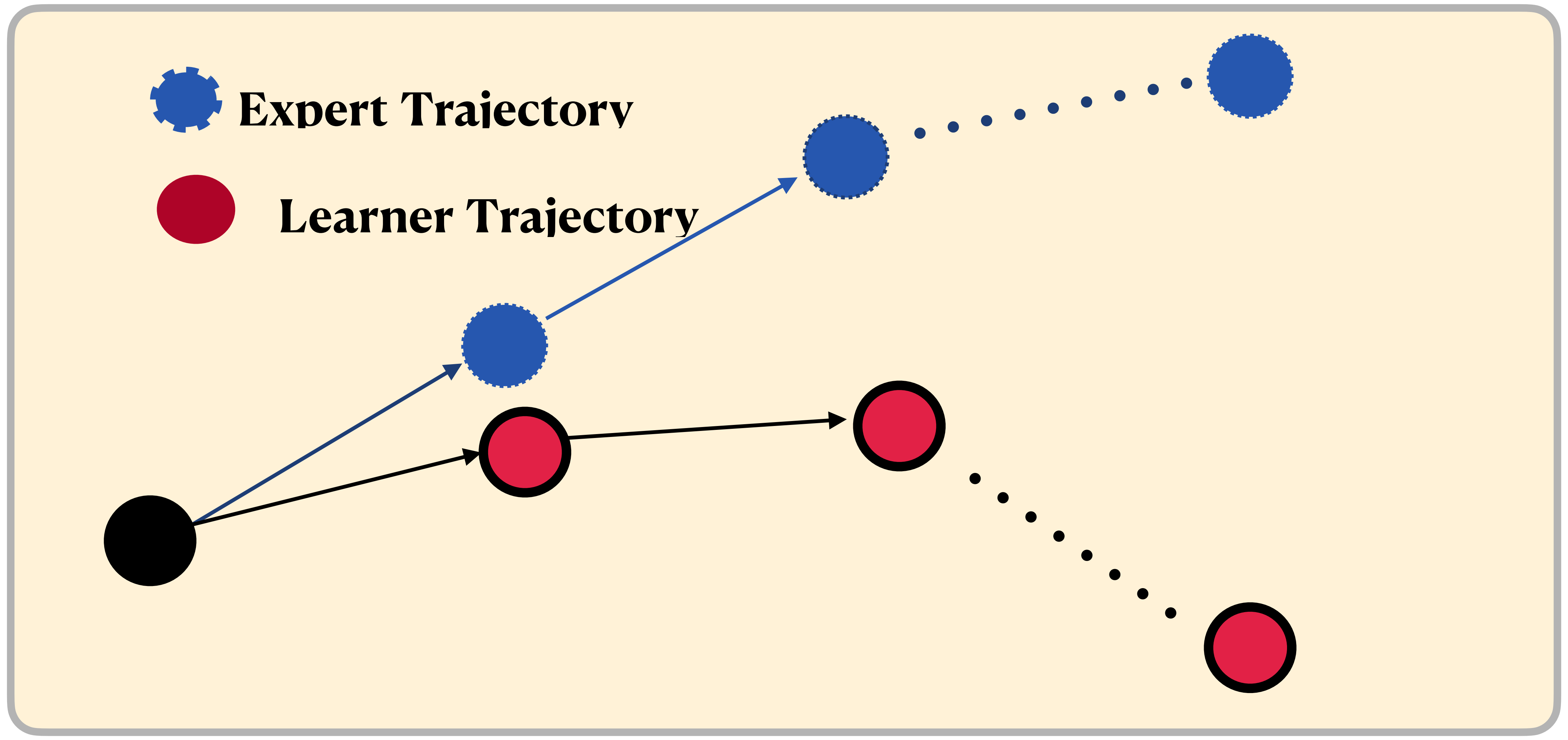


📚 ***Limited Compounding w/ Probabilistic Error?***

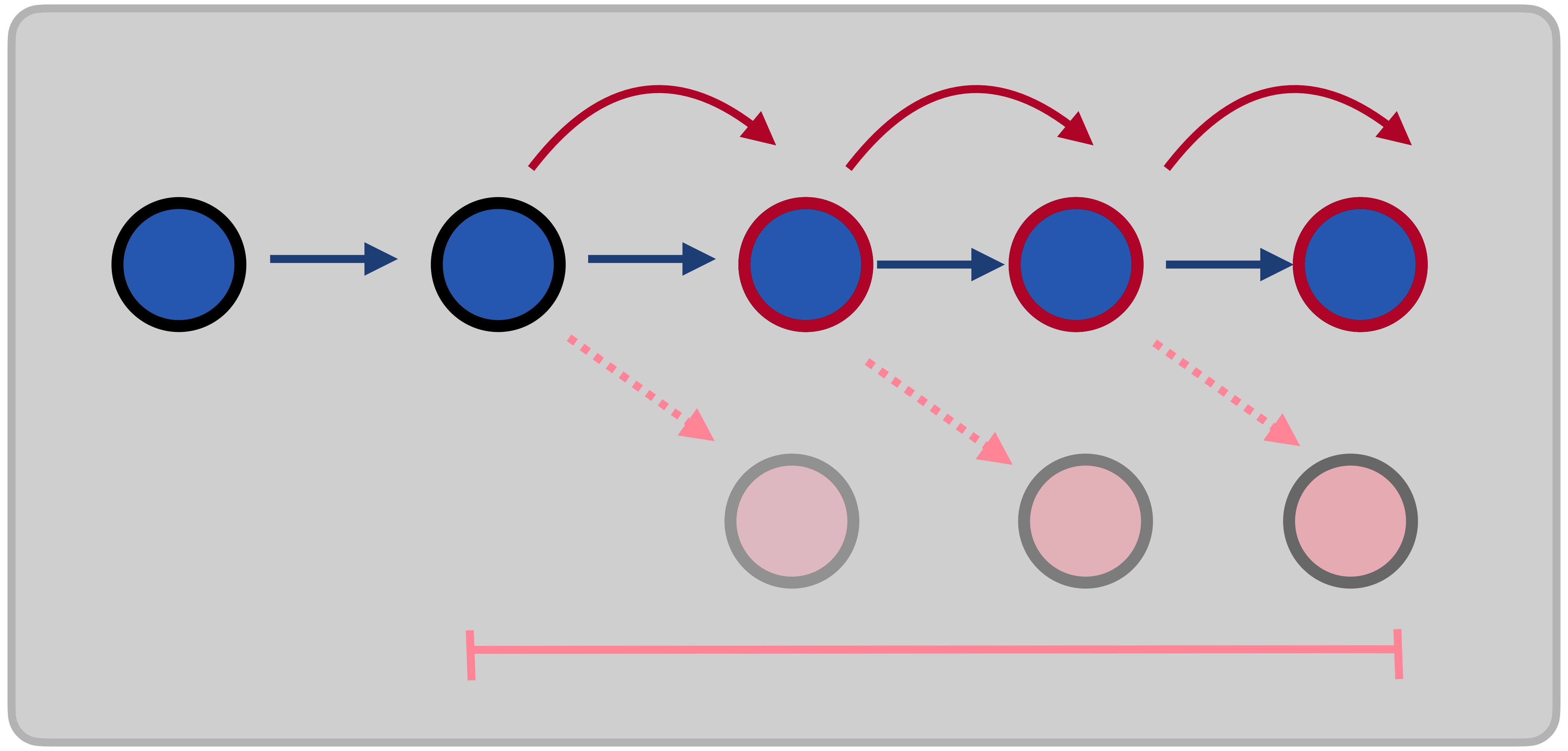
Compounding in Physical World 🤖?



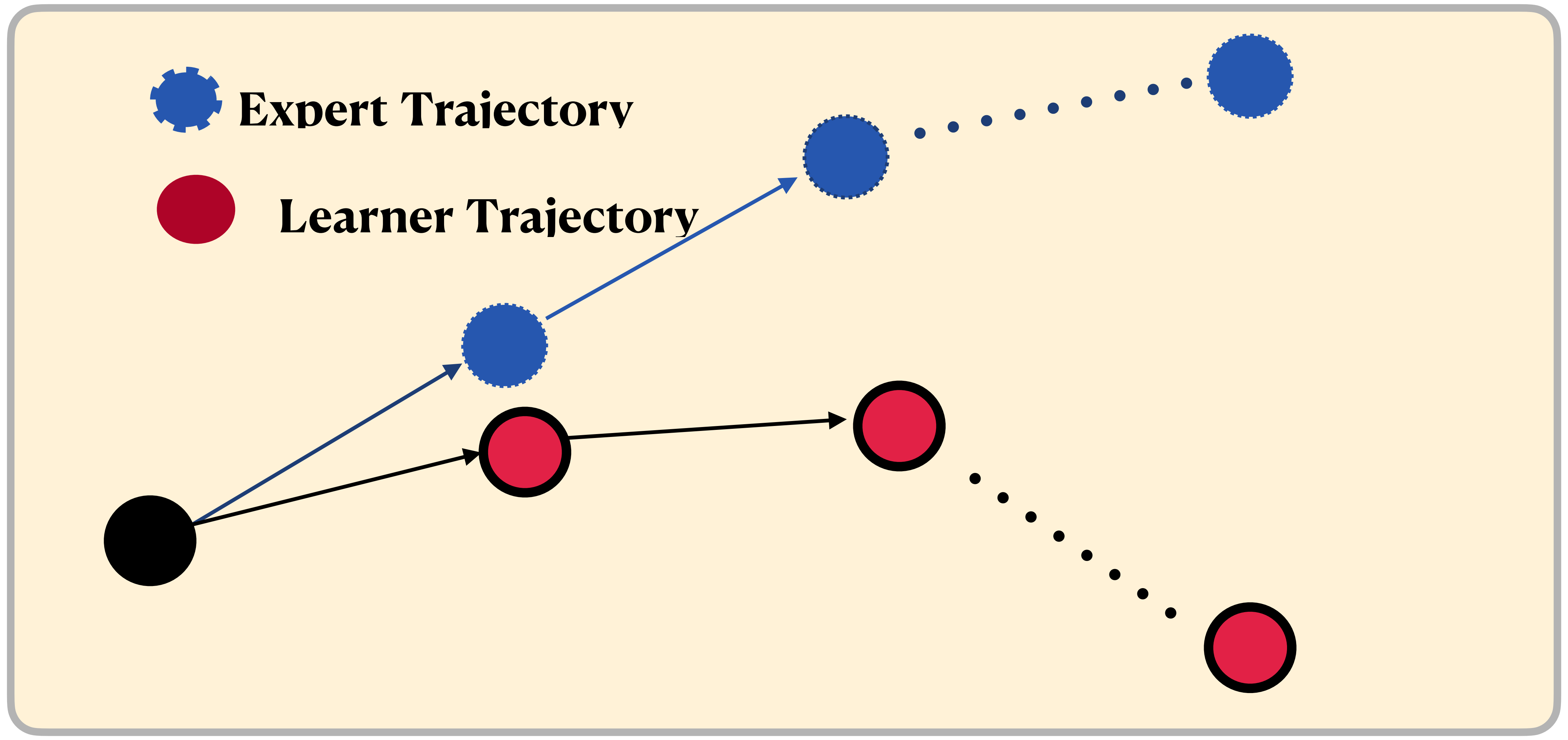
📚 ***Limited Compounding w/ Probabilistic Error?***



Compounding in Physical World 🤖?

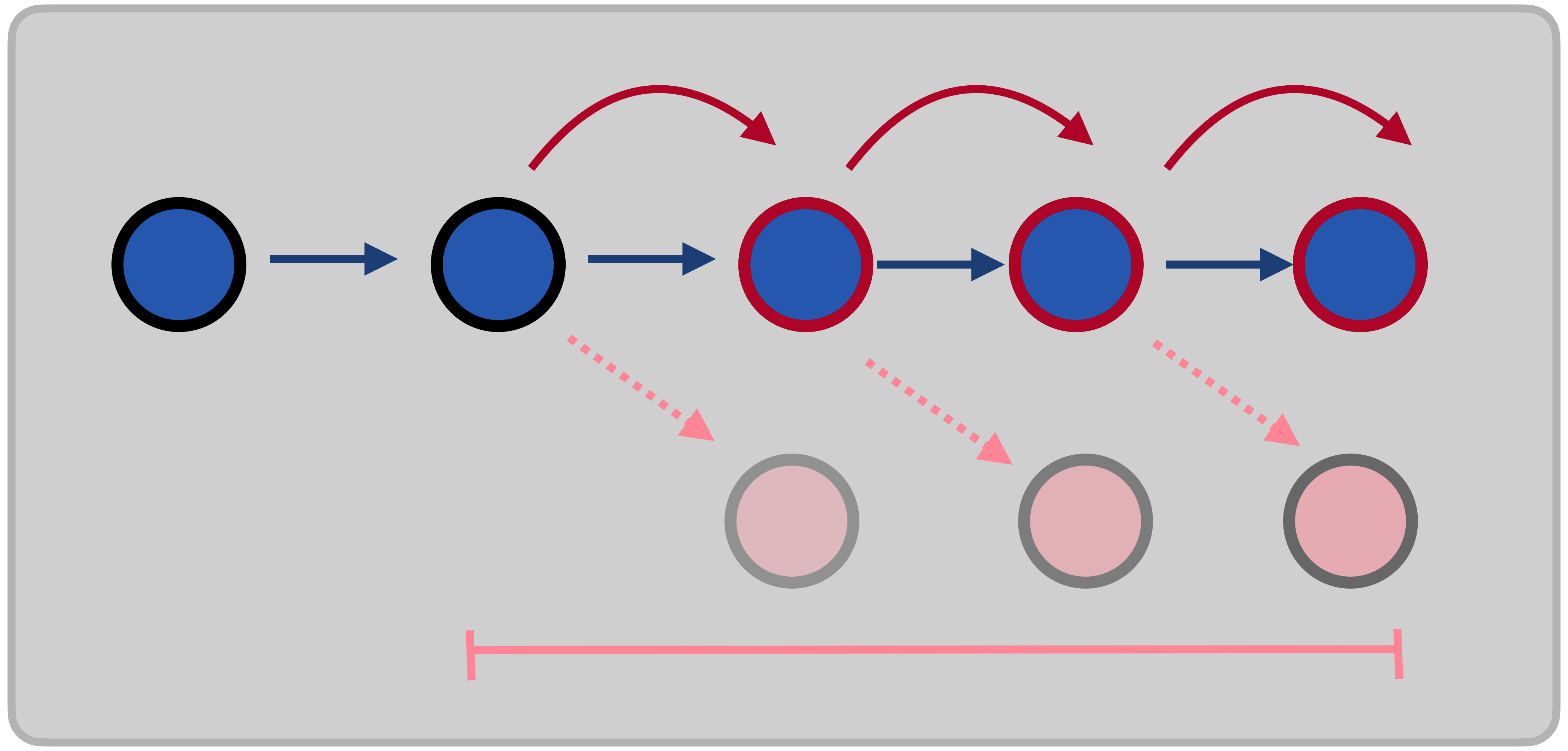


📚 ***Limited Compounding w/ Probabilistic Error?***

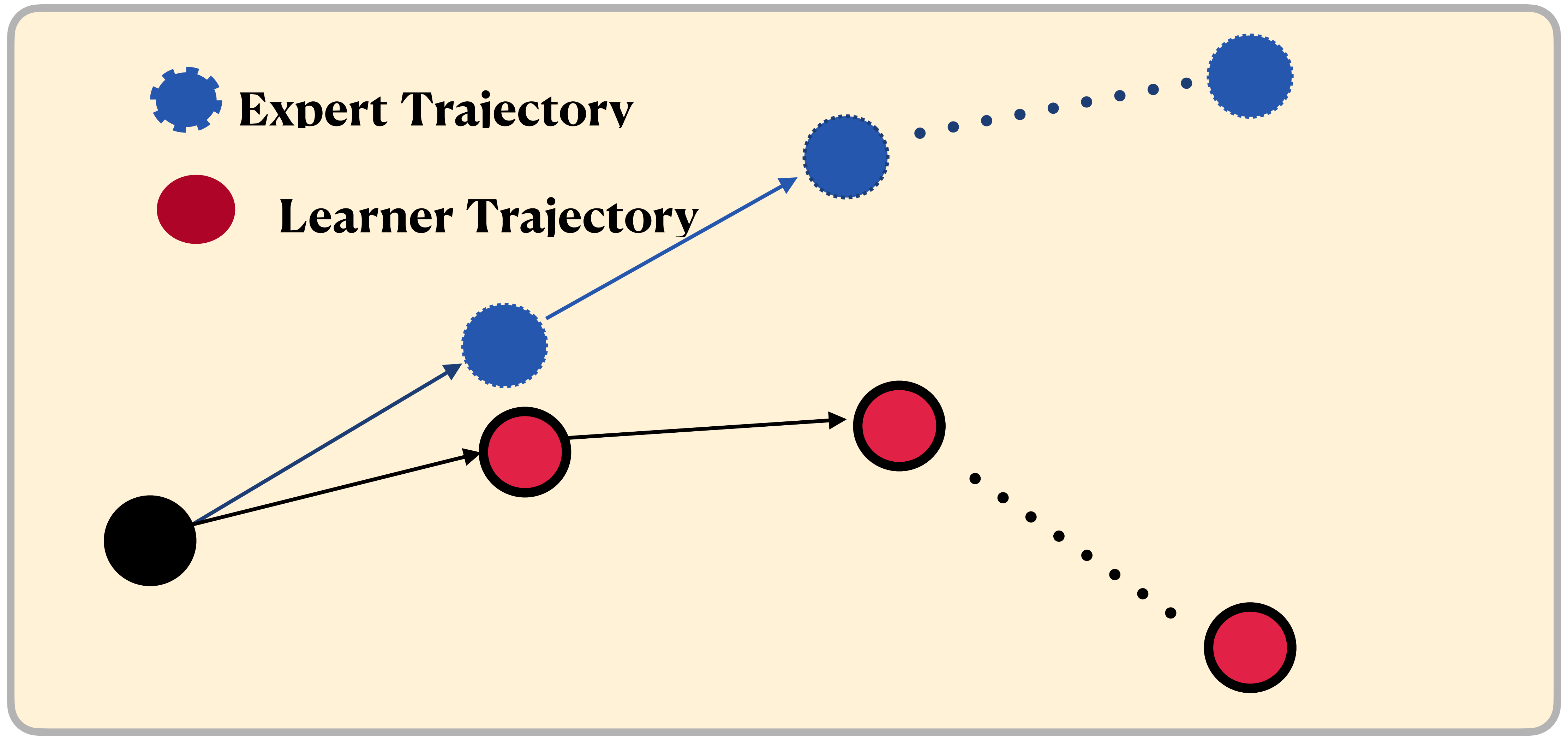


Perturbative Error! 🤖

Compounding in Physical World 🤖?



📚 ***Limited Compounding w/ Probabilistic Error?***



Perturbative Error! 🤖

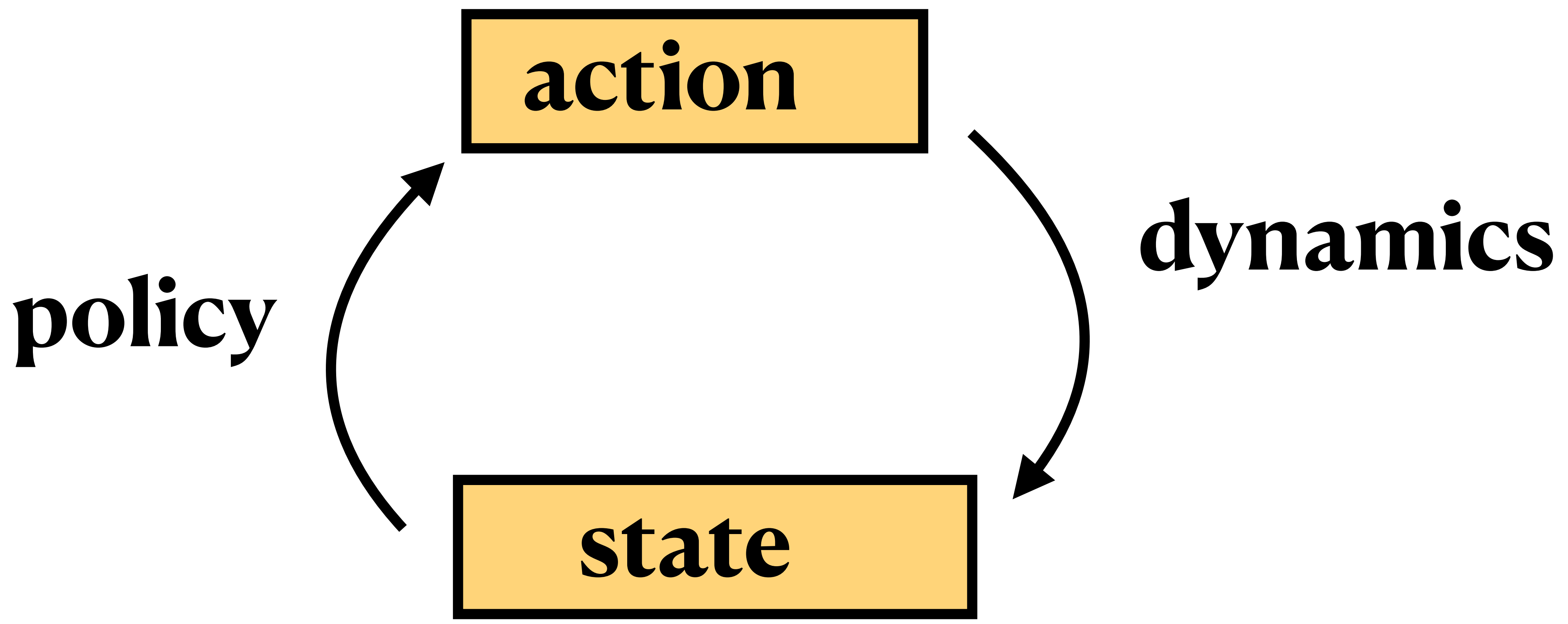
Sometime much worse?

Act 2: "Learning in the Physical World is Harder"

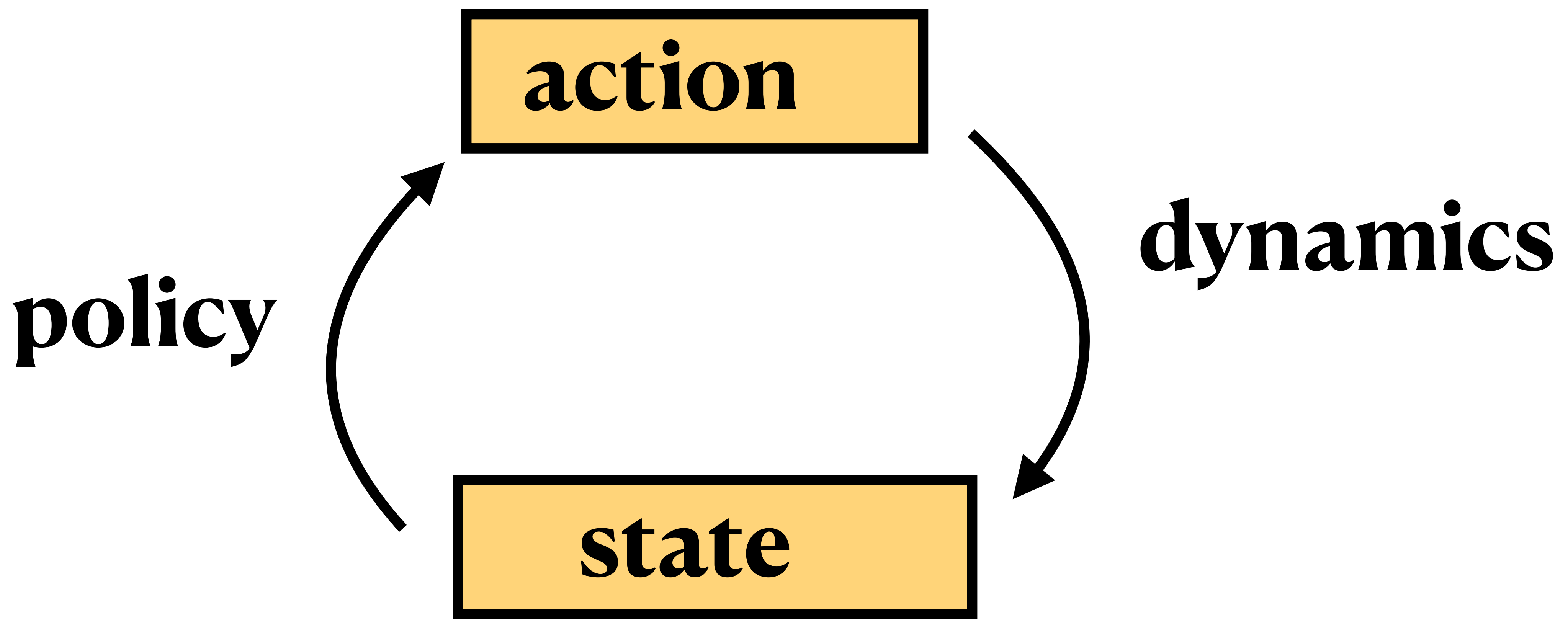
w/ Daniel Pfrommer, Ali Jadbabaie (MIT).

An Informal Theorem 🤖

An Informal Theorem 🤖

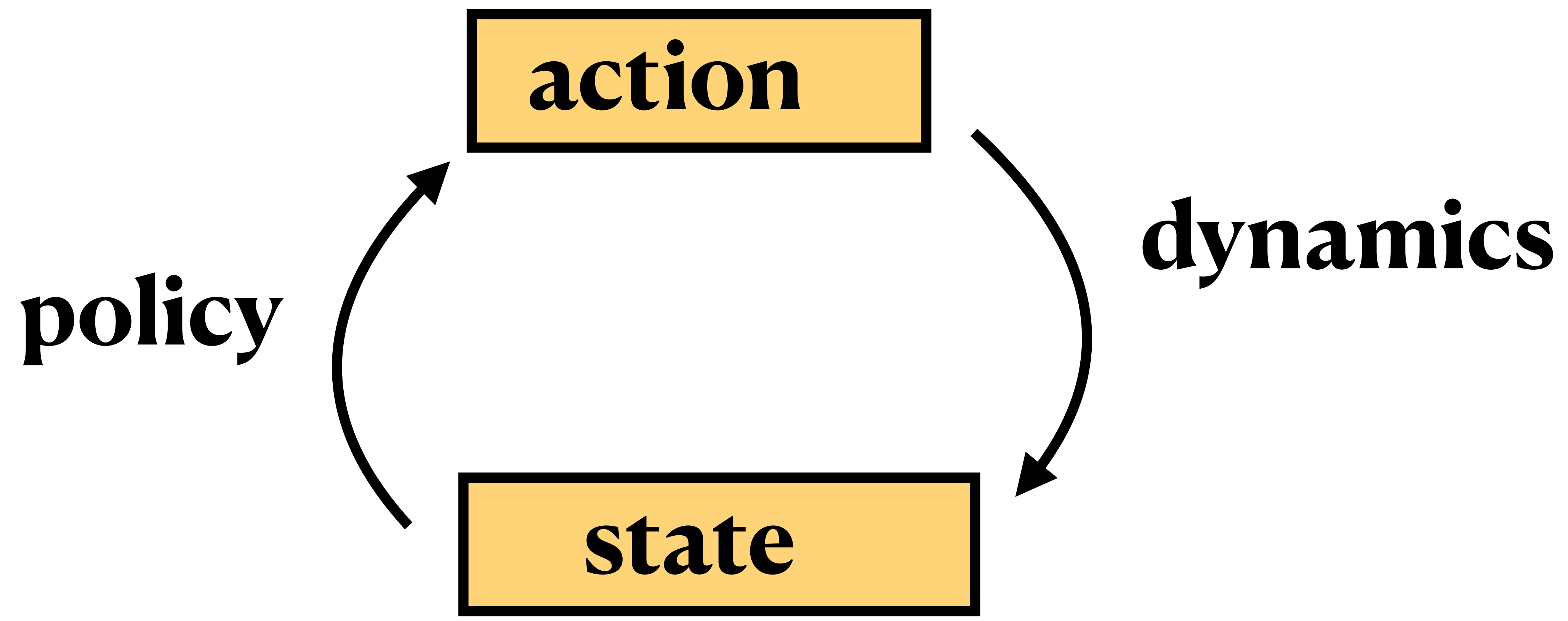


An Informal Theorem 🤖



Assumptions: “Things are nice”

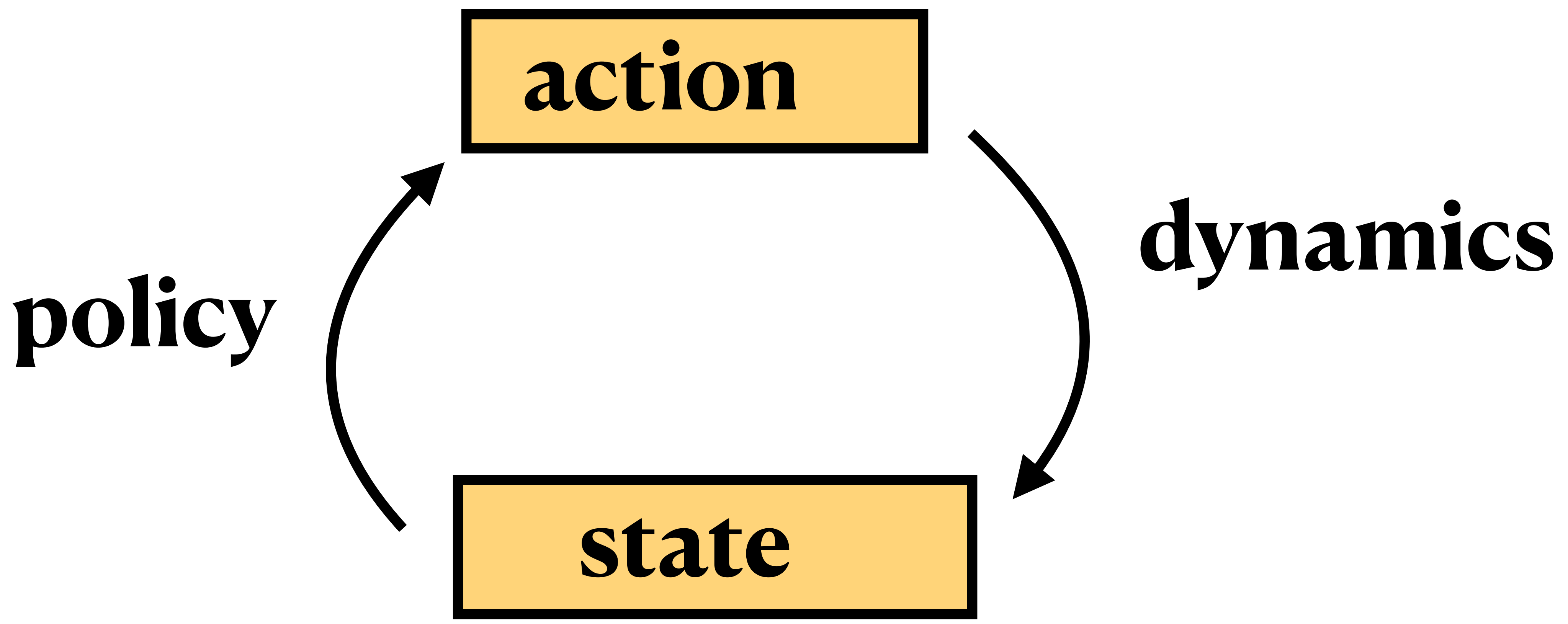
An Informal Theorem 🤖



Assumptions: "Things are nice"

1. dynamics + policy are smooth+deterministic

An Informal Theorem 🤖

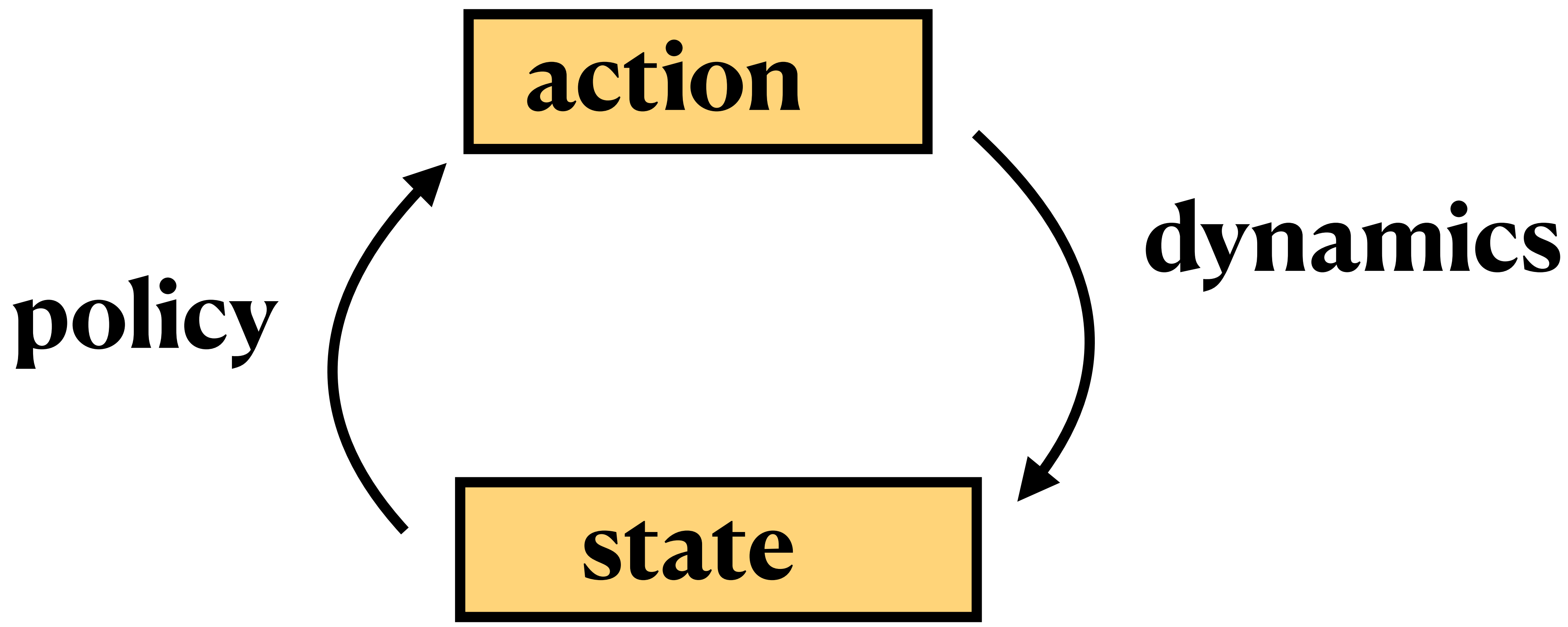


Assumptions: "Things are nice"

1. dynamics + policy are smooth+deterministic

2. cost function is smooth and bounded.

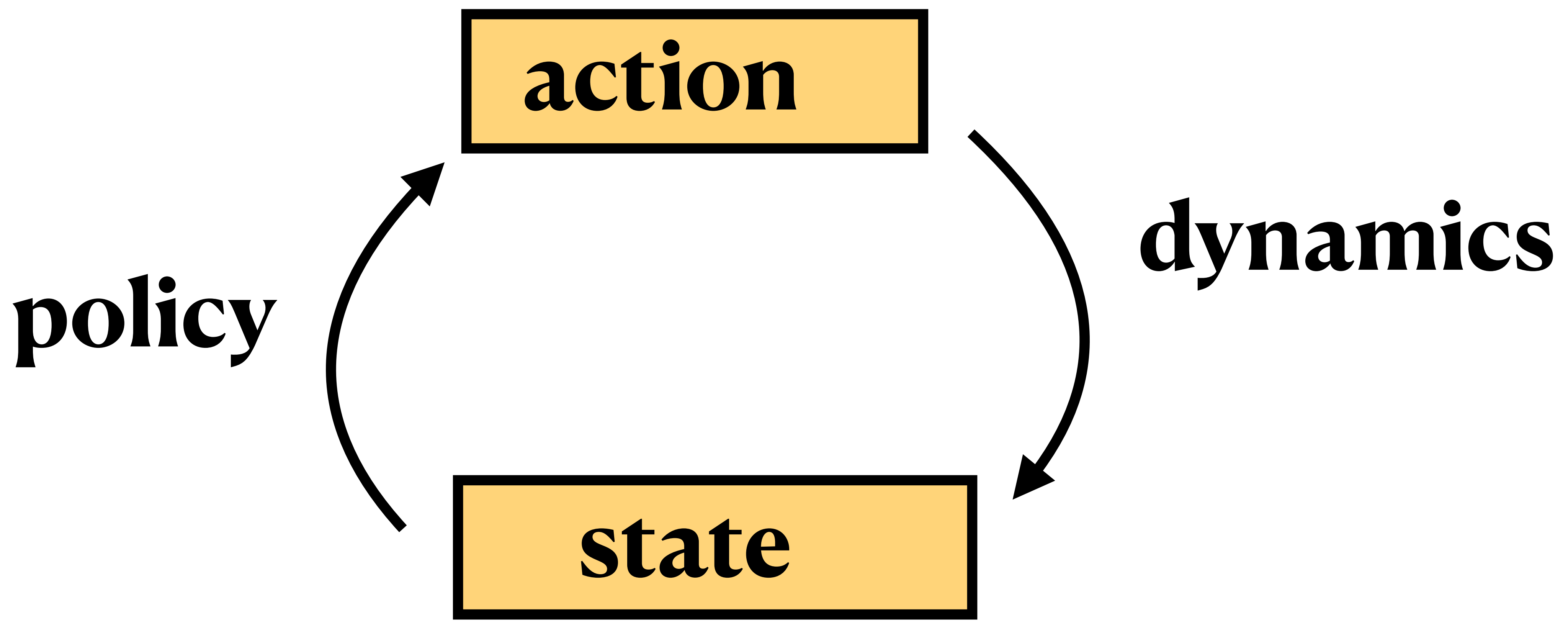
An Informal Theorem 🤖



Assumptions: "Things are nice"

1. dynamics + policy are smooth+deterministic
2. cost function is smooth and bounded.
3. dynamics are stable.

An Informal Theorem 🤖

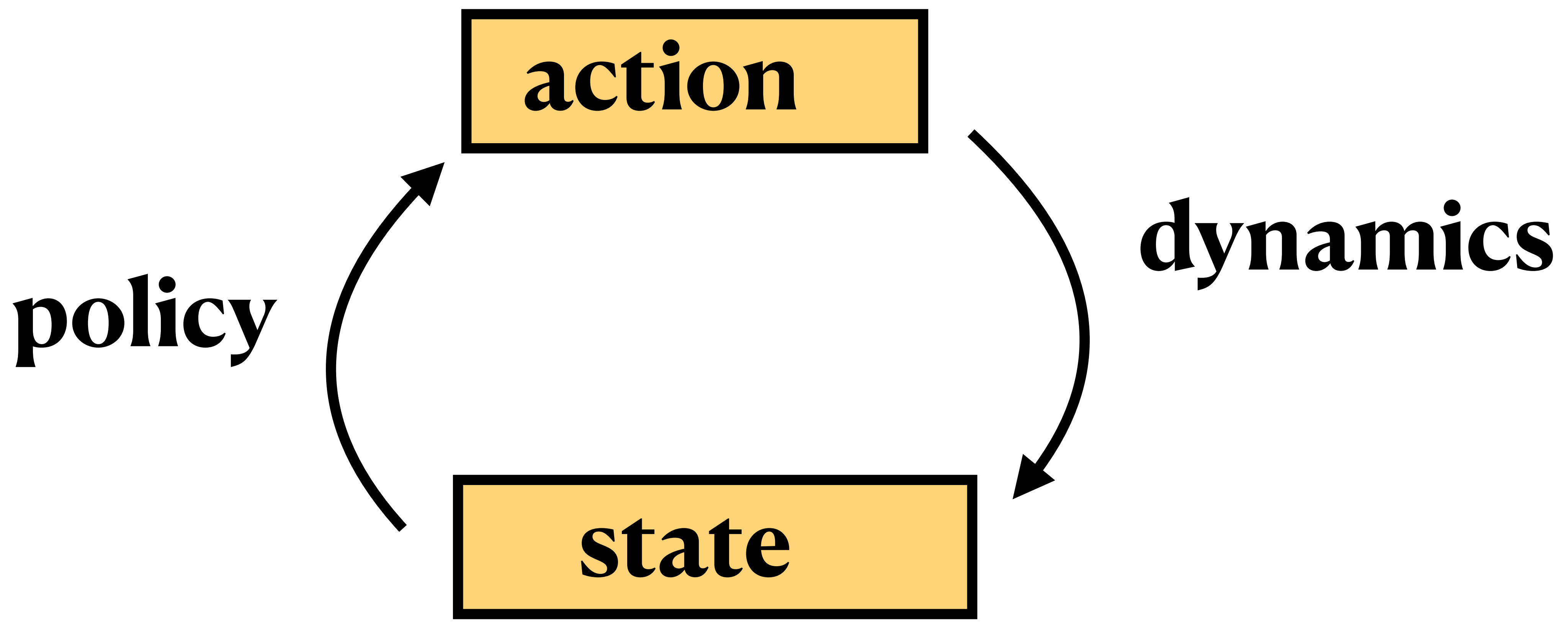


Assumptions: "Things are nice"

1. dynamics + policy are smooth+deterministic
2. cost function is smooth and bounded.
3. dynamics are stable.

Takeaway: learning in the physical world 🤖 can be hard even if the problems seems benign

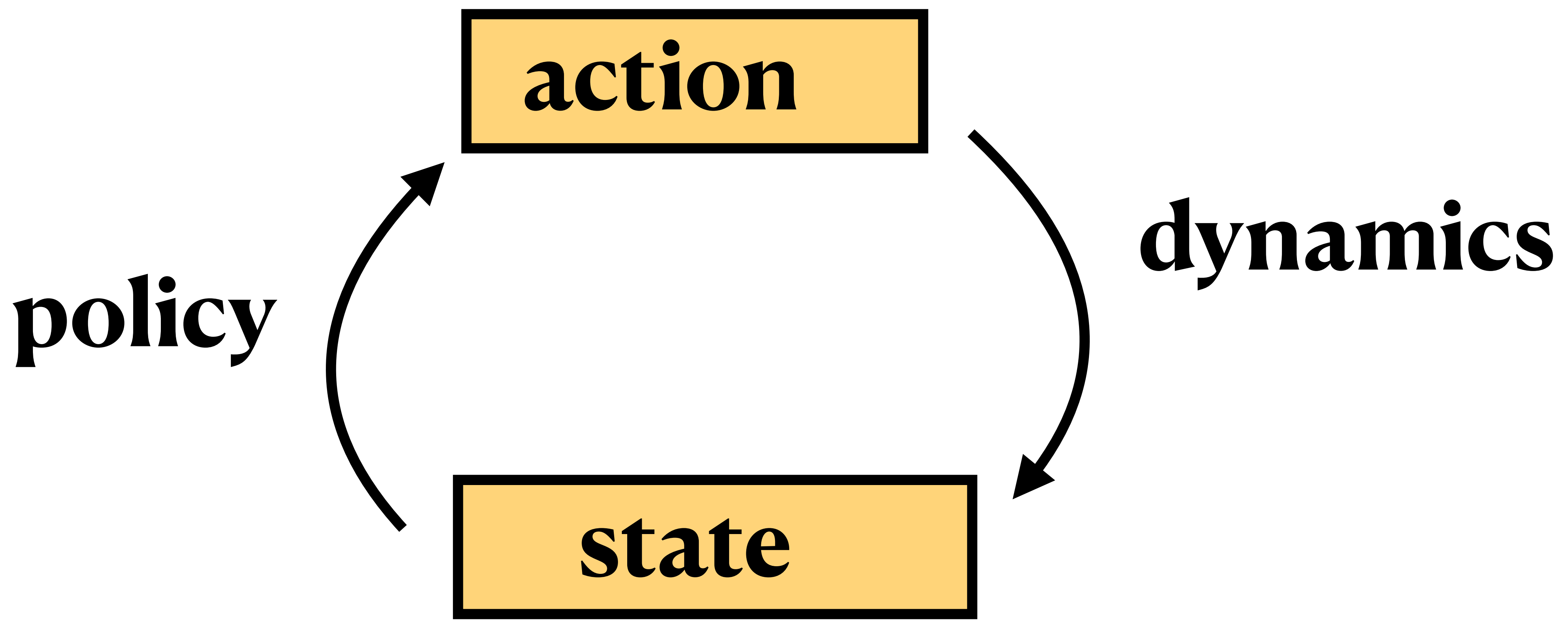
An Informal Theorem 🤖



Assumptions: "Things are nice"

1. dynamics + policy are smooth+deterministic
2. cost function is smooth and bounded.
3. **dynamics are stable.**

An Informal Theorem 🤖

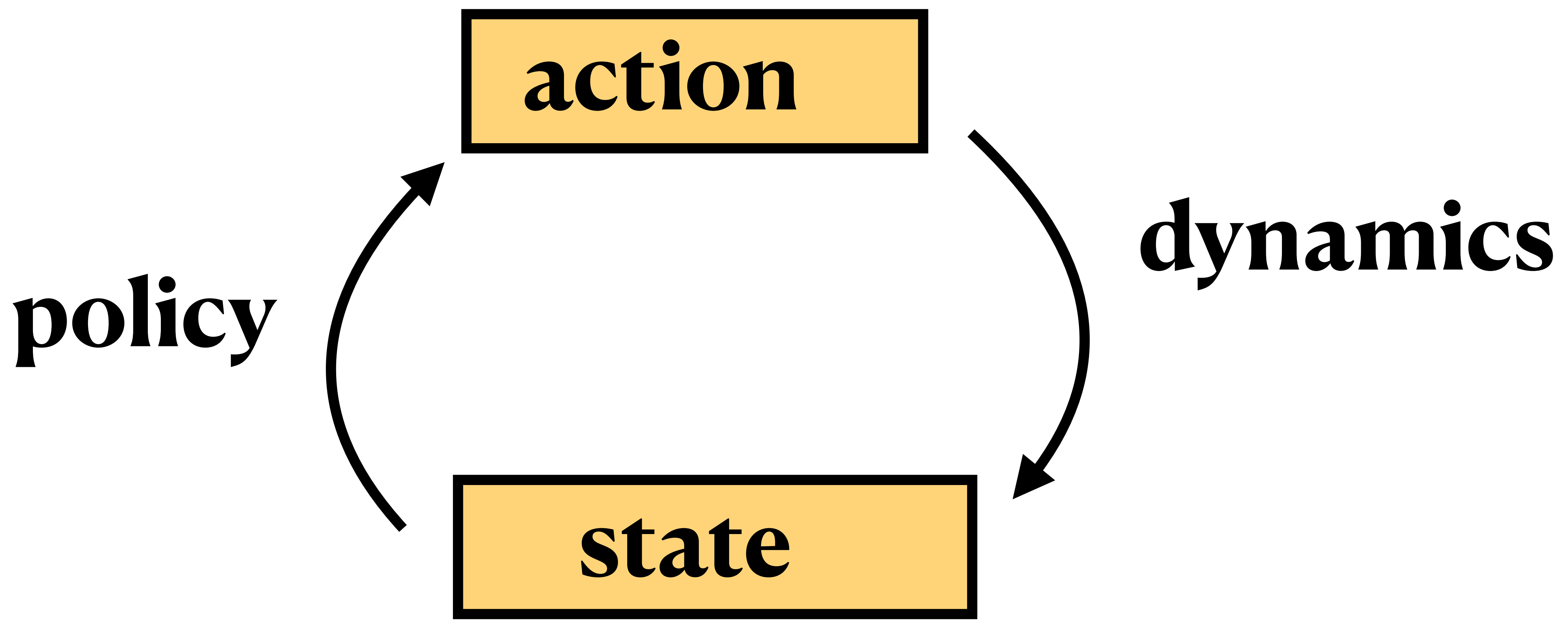


Assumptions: “Things are nice”

1. dynamics + policy are smooth+deterministic
2. cost function is smooth and bounded.
3. **dynamics are stable.**

Theorem (SPJ): Let **n** be the number of expert trajectories. For any $\epsilon(n) \propto n^{-k}$, there exists a family of **nice behavior cloning problems** where:

An Informal Theorem 🤖



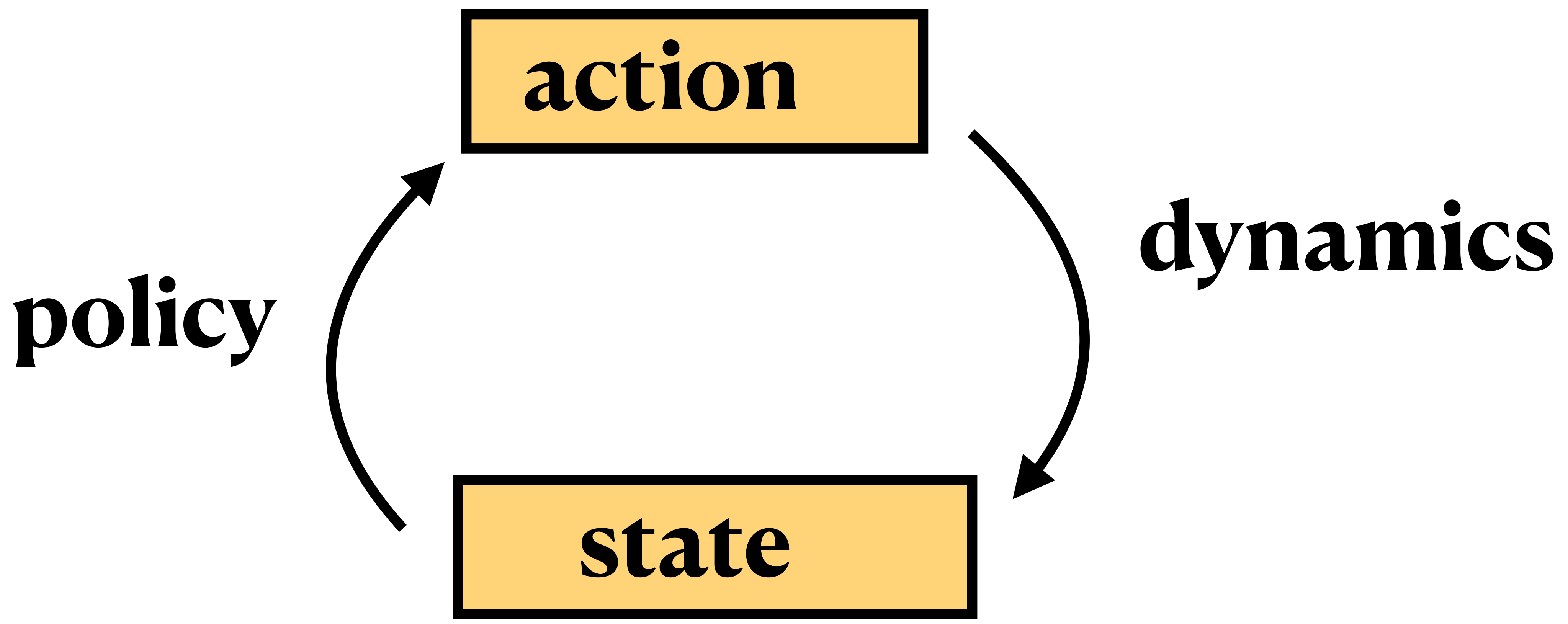
Assumptions: "Things are nice"

1. dynamics + policy are smooth+deterministic
2. cost function is smooth and bounded.
3. **dynamics are stable.**

Theorem (SPJ): Let **n** be the number of expert trajectories. For any $\epsilon(n) \propto n^{-k}$, there exists a family of **nice behavior cloning problems** where:

Expert Error $\mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star) \leq \epsilon(n)$

An Informal Theorem 🤖



Assumptions: "Things are nice"

1. dynamics + policy are smooth+deterministic
2. cost function is smooth and bounded.
3. **dynamics are stable.**

Theorem (SPJ): Let **n** be the number of expert trajectories. For any $\epsilon(n) \propto n^{-k}$, there exists a family of **nice behavior cloning problems** where:

Expert Error $\mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star) \leq \epsilon(n)$ **Cost** $\mathscr{R}_c(\hat{\pi}; \pi^\star) \gtrsim \min\left\{2^H \epsilon(n), 1\right\}$

An Informal Theorem 🤖

Theorem (SPJ): Let **n** be the number of expert trajectories. For any $\epsilon(n) \propto n^{-k}$, there exists a family of **nice behavior cloning problems** s.t.:

Expert Error $\mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star) \leq \epsilon(n)$

Cost $\mathscr{R}_c(\hat{\pi}; \pi^\star) \gtrsim \min\left\{2^H \epsilon(n), 1\right\}$

An Informal Theorem 🤖

Theorem (SPJ): Let **n** be the number of expert trajectories. For any $\epsilon(n) \propto n^{-k}$, there exists a family of **nice behavior cloning problems** s.t.:

Expert Error $\mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star) \leq \epsilon(n)$ **Cost** $\mathscr{R}_c(\hat{\pi}; \pi^\star) \gtrsim \min\left\{2^H \epsilon(n), 1\right\}$

Behavior Cloning achieve this!

An Informal Theorem 🤖

Theorem (SPJ): Let **n** be the number of expert trajectories. For any $\epsilon(n) \propto n^{-k}$, there exists a family of **nice behavior cloning problems** s.t.:

Expert Error $\mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star) \leq \epsilon(n)$

Cost $\mathscr{R}_c(\hat{\pi}; \pi^\star) \gtrsim \min\left\{2^H \epsilon(n), 1\right\}$

Behavior Cloning achieve this!

Any* algorithm using **expert data** suffers.

An Informal Theorem 🤖

Theorem (SPJ): Let **n** be the number of expert trajectories. For any $\epsilon(n) \propto n^{-k}$, there exists a family of **nice behavior cloning problems** s.t.:

Expert Error $\mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^{\star}) \leq \epsilon(n)$

Cost $\mathscr{R}_{c}(\hat{\pi}; \pi^{\star}) \gtrsim \min\left\{2^{H}\epsilon(n), 1\right\}$

Behavior Cloning achieve this!

Any* algorithm using **expert data** suffers.

Including: inverse RL and offline RL!

An Informal Theorem 🤖

Theorem (SPJ): Let **n** be the number of expert trajectories. For any $\epsilon(n) \propto n^{-k}$, there exists a family of **nice behavior cloning problems** s.t.:

Expert Error $\mathcal{R}_{\text{expert}}(\hat{\pi}; \pi^\star) \leq \epsilon(n)$

Cost $\mathcal{R}_c(\hat{\pi}; \pi^\star) \gtrsim \min\left\{2^H \epsilon(n), 1\right\}$

Behavior Cloning achieve this!

Any* algorithm using **expert data** suffers.

Including: inverse RL and offline RL!

Exponential worse that $\mathcal{R}_c(\hat{\pi}; \pi^\star) \leq H \cdot \epsilon(n)$ **if** 📚

An Informal Theorem 🤖

Theorem (SPJ): Let **n** be the number of expert trajectories. For any $\epsilon(n) \propto n^{-k}$, there exists a family of **nice behavior cloning problems** s.t.:

Expert Error $\mathcal{R}_{\text{expert}}(\hat{\pi}; \pi^{\star}) \leq \epsilon(n)$

Cost $\mathcal{R}_c(\hat{\pi}; \pi^{\star}) \gtrsim \min\left\{2^H \epsilon(n), 1\right\}$

Behavior Cloning achieve this!

Any* algorithm using **expert data** suffers.

Including: inverse RL and offline RL!

Exponential worse that $\mathcal{R}_c(\hat{\pi}; \pi^{\star}) \leq H \cdot \epsilon(n)$ **if** 📚

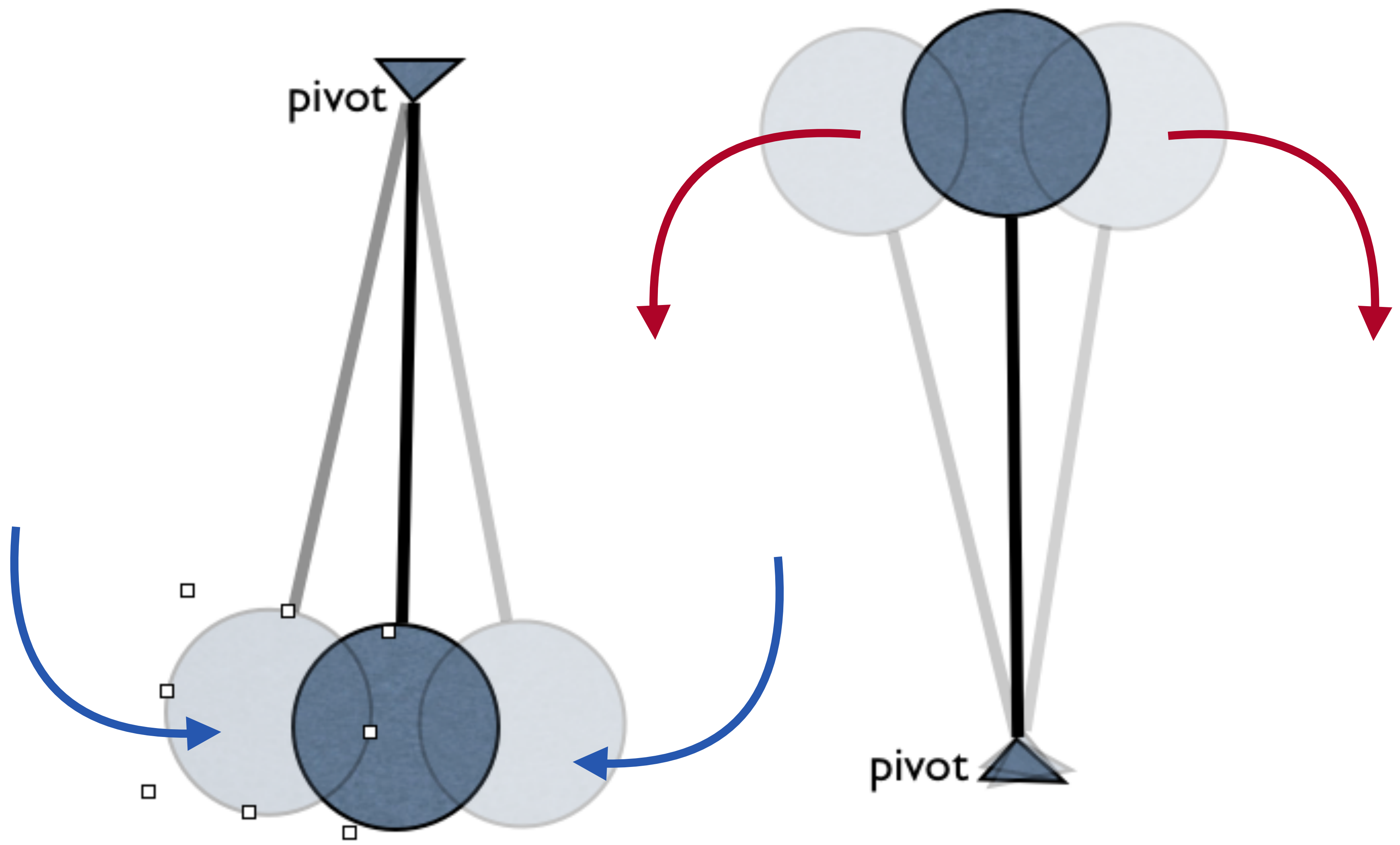
***Caveat for the end.....**

Control Theoretic Stability 🤸

Control Theoretic Stability 🤸

Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

Control Theoretic Stability

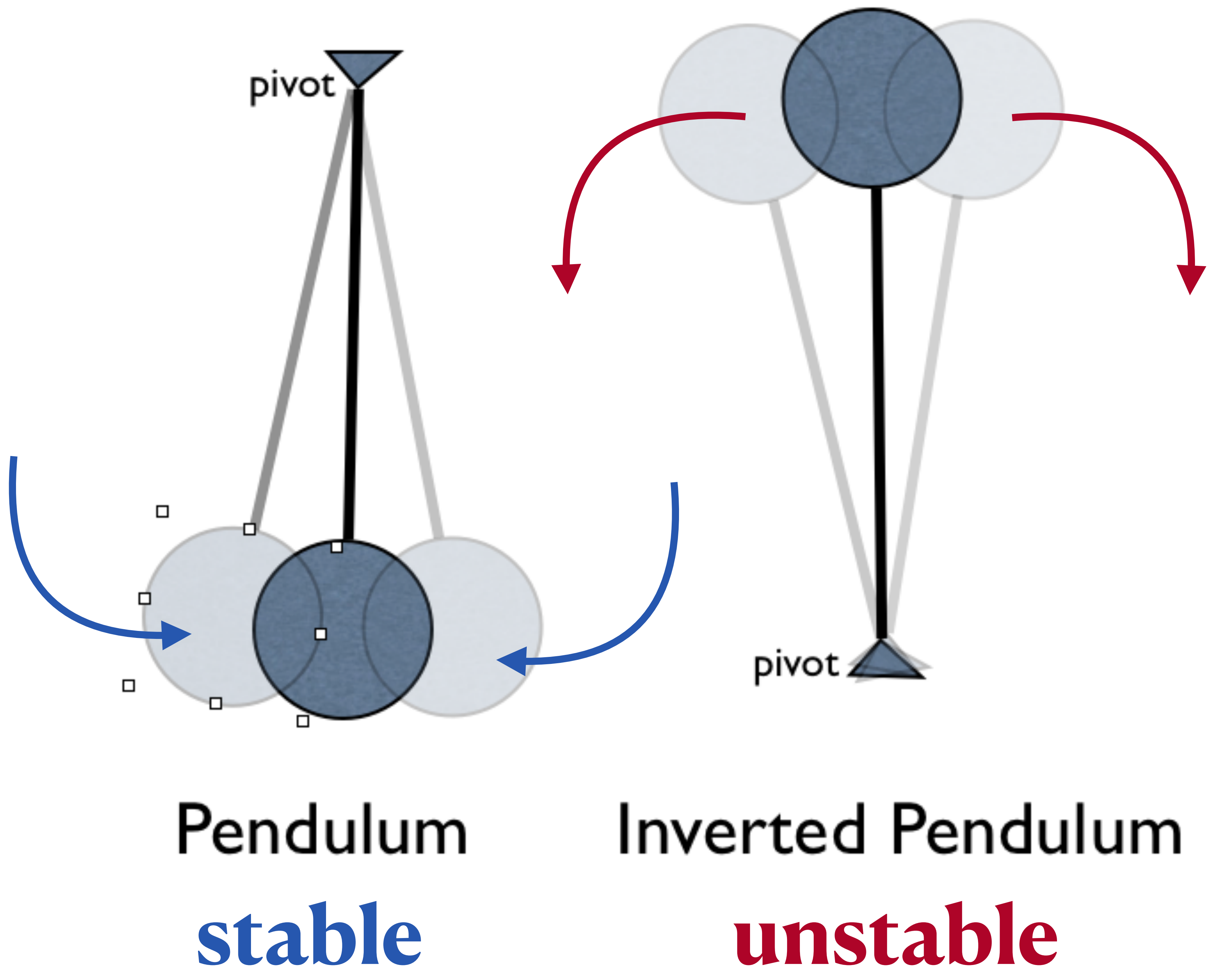


Pendulum
stable

Inverted Pendulum
unstable

Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

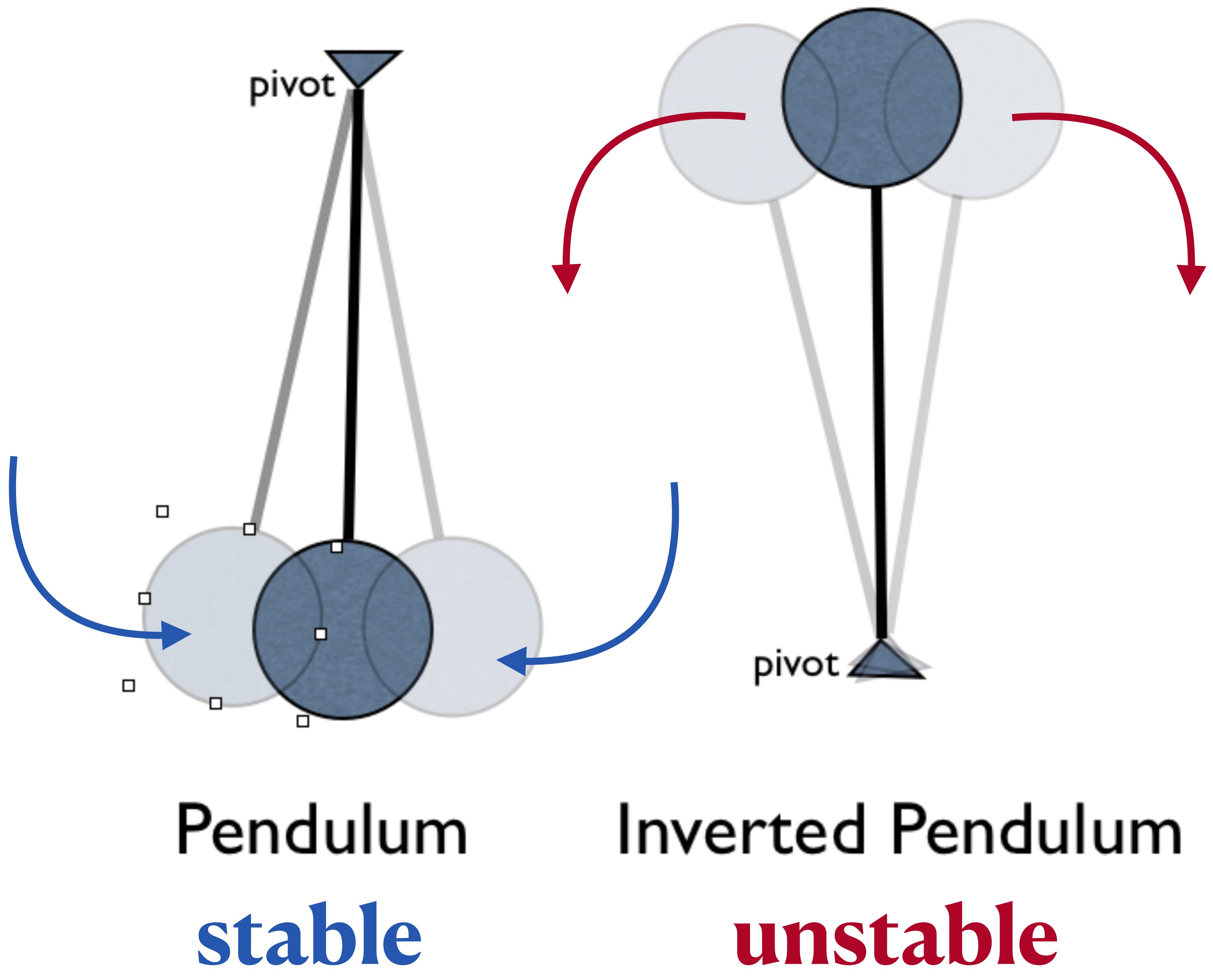
Control Theoretic Stability 🤸



Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

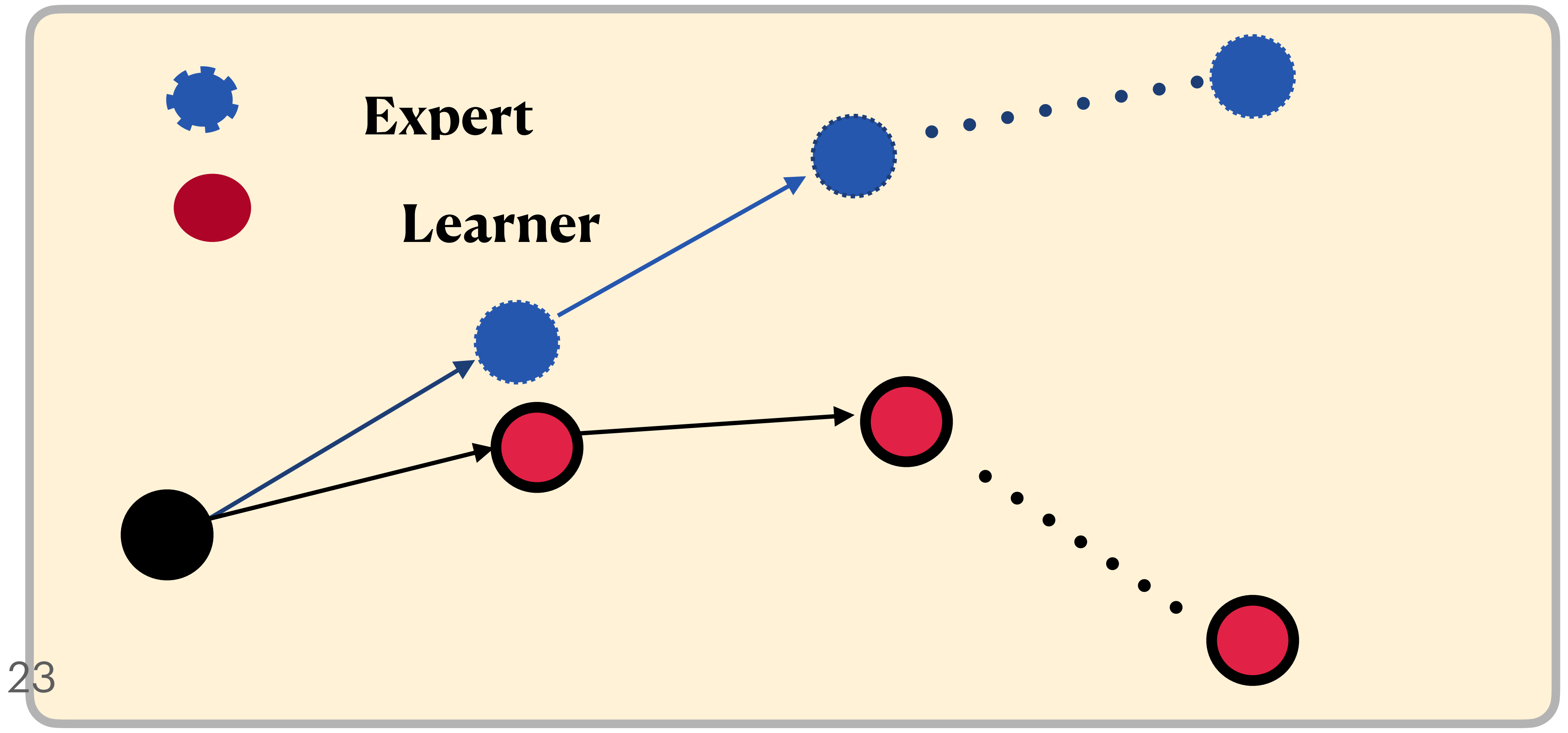
naturally related to compounding error.

Control Theoretic Stability 🤸

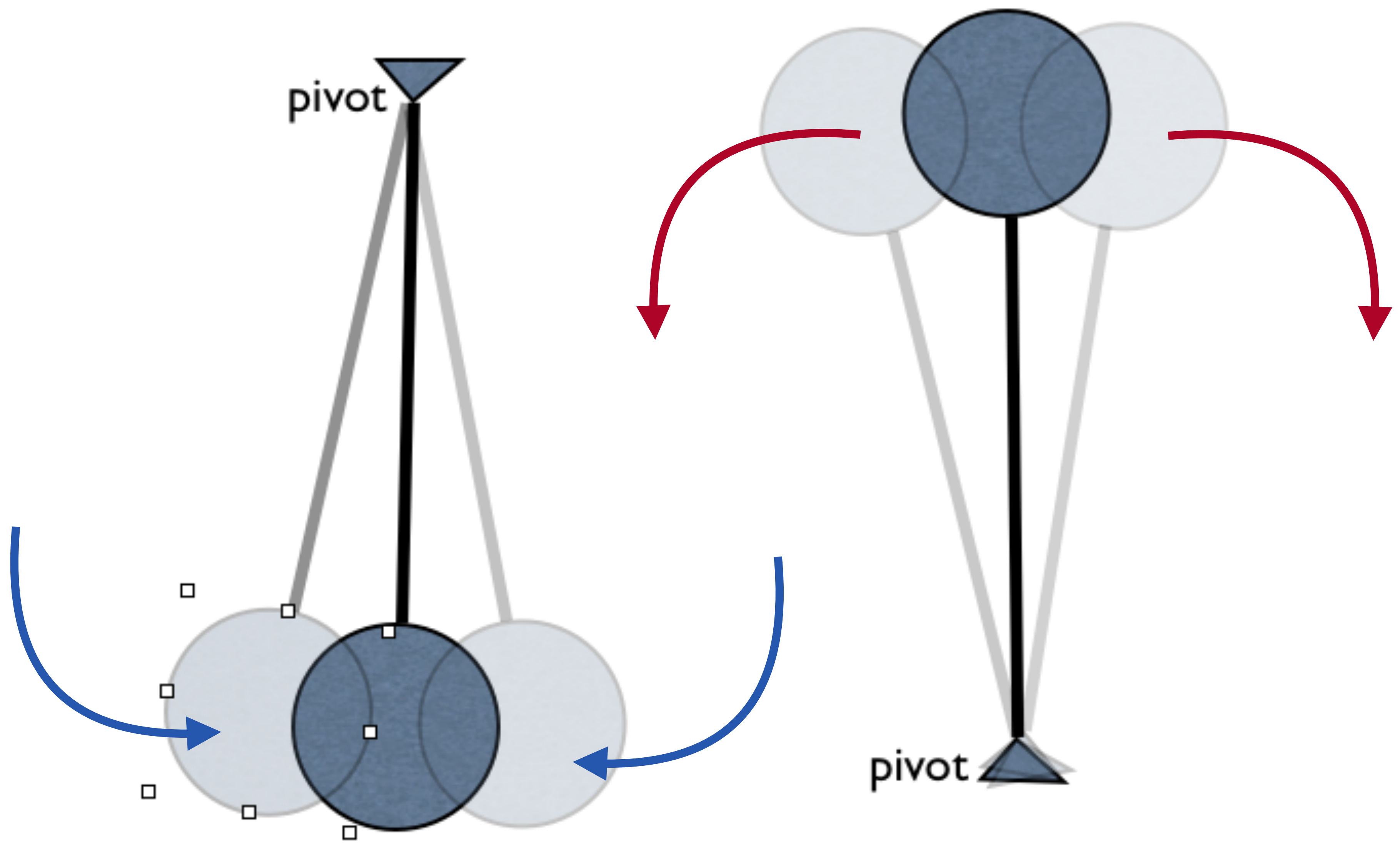


Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

naturally related to compounding error.



Control Theoretic Stability 🤸

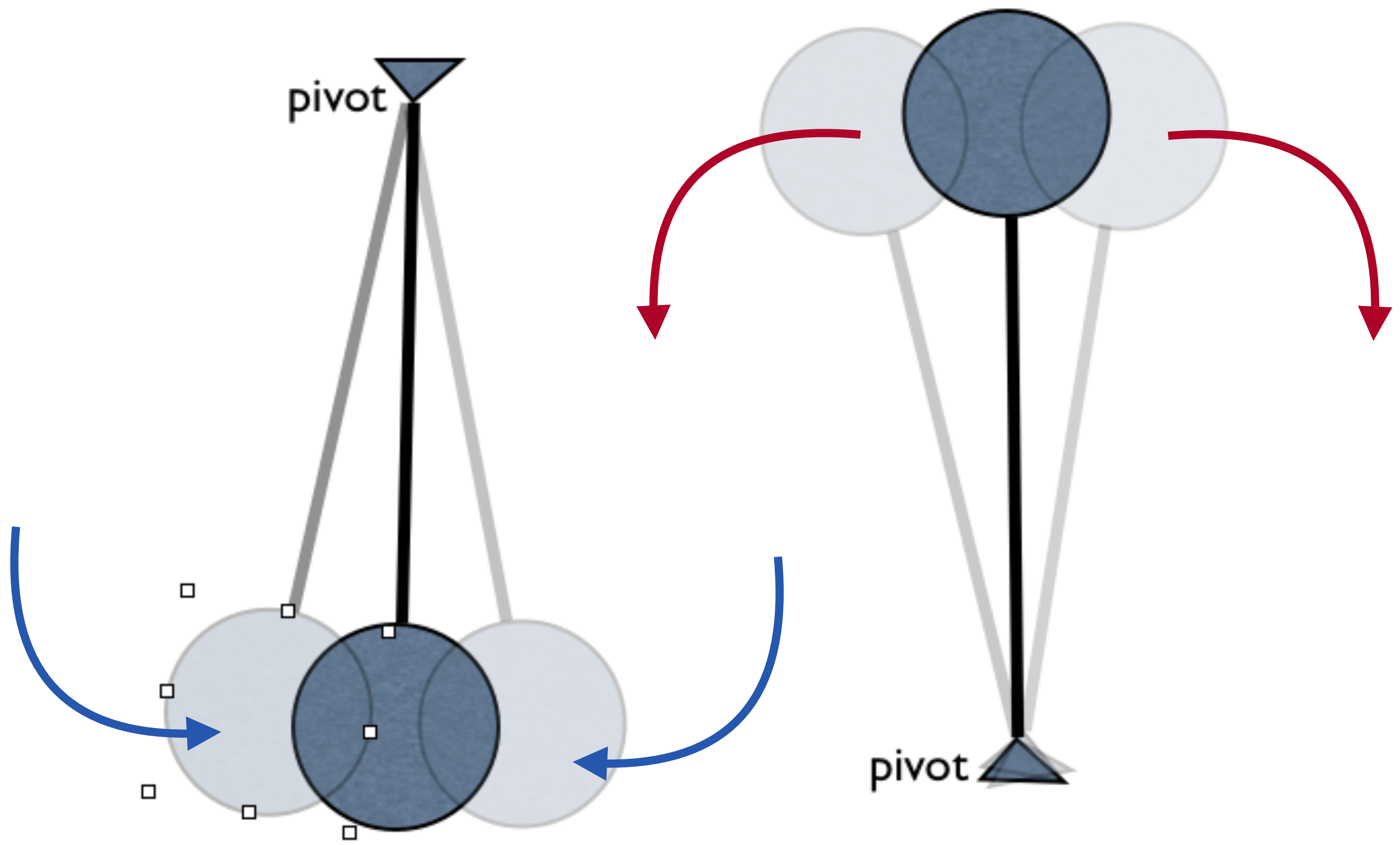


Pendulum
stable

Inverted Pendulum
unstable

Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

Control Theoretic Stability



Pendulum **stable**

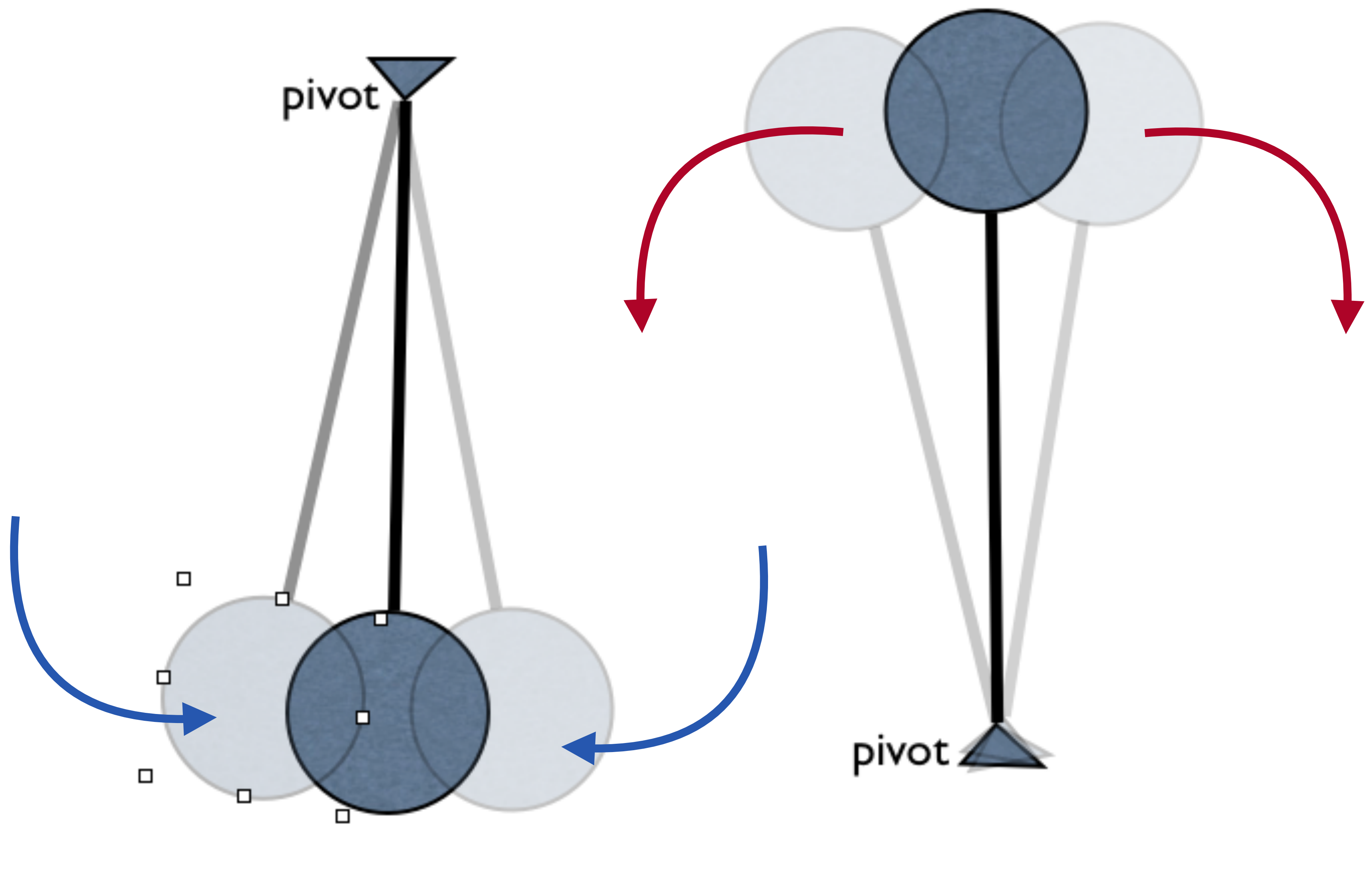
Inverted Pendulum **unstable**

Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

Definition: Dynamics $x_{t+1} = f(x_t, u_t)$ are (C, ρ)-**stable** if, for same initial condition $x_1 = x'_1$

$$\|x'_{t+1} - x'_{t+1}\| \leq C \sum_{s \leq t} \rho^{t-s} \|u_s - u'_s\|$$

Control Theoretic Stability 🤸



stable **unstable**

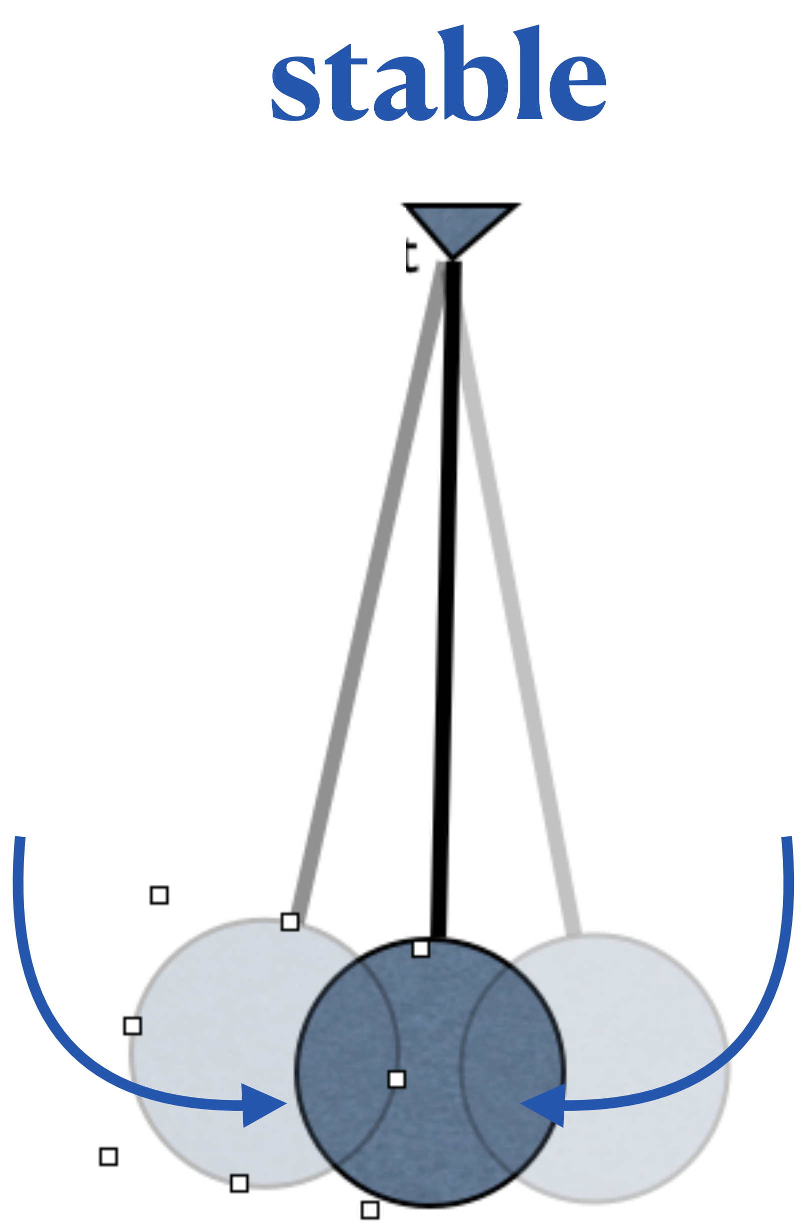
Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

Definition: Dynamics $x_{t+1} = f(x_t, u_t)$ are (C, ρ) -**stable** if, for same initial condition $x_1 = x'_1$

$$\|x'_{t+1} - x'_{t+1}\| \leq C \sum_{s \leq t} \rho^{t-s} \|u_s - u'_s\|$$

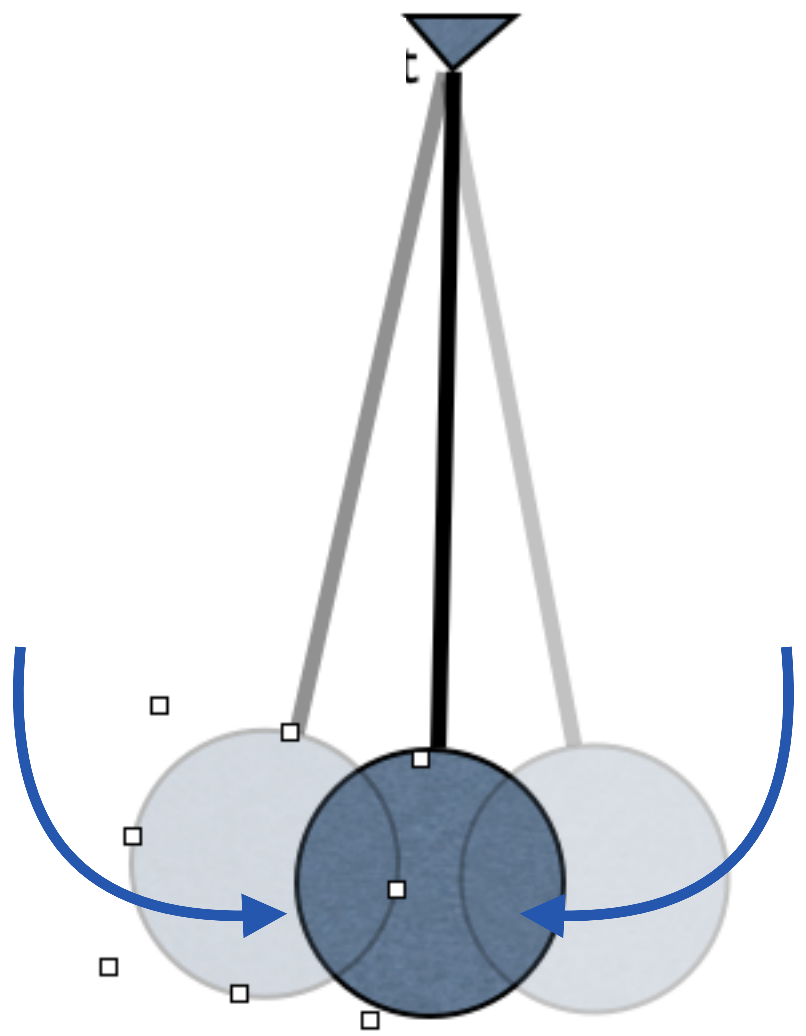
$\rho \in (0,1)$ = **exponentially quick forgetting of mistakes**

Control Theoretic Stability 🤸

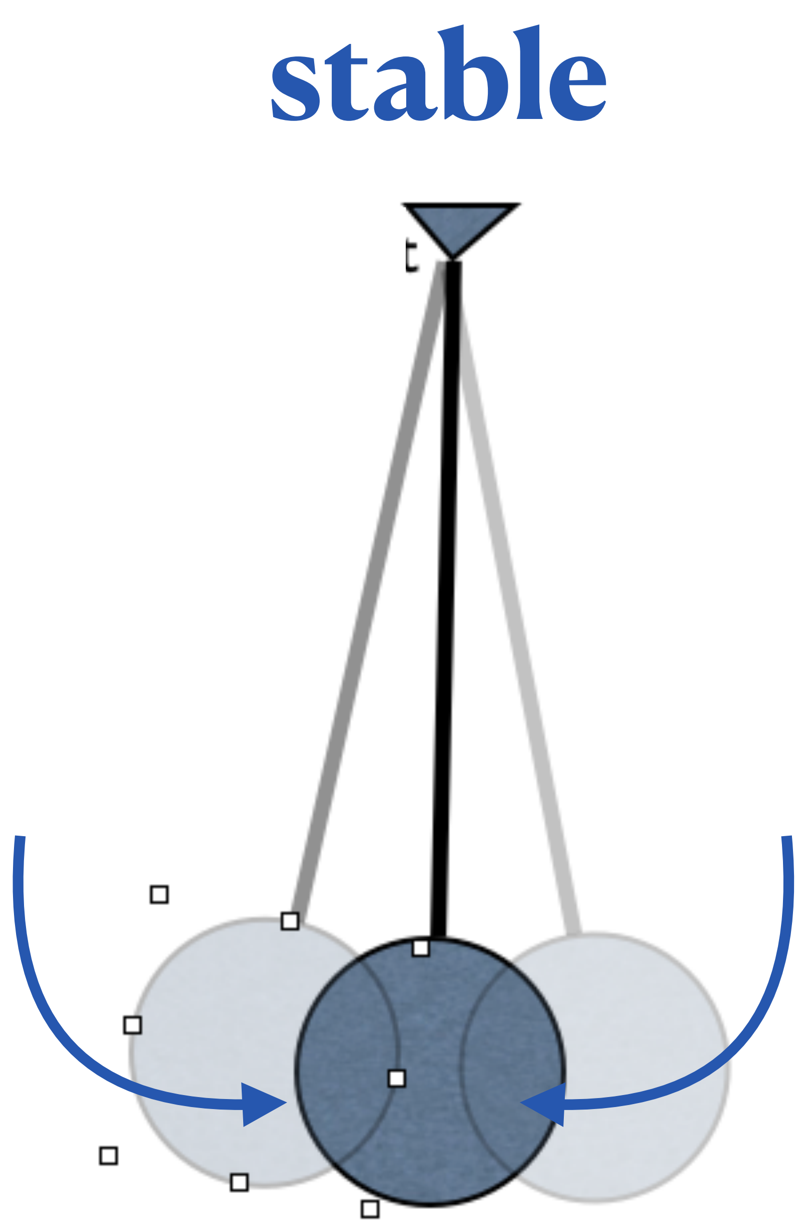


Control Theoretic Stability 🤸

We assume that the following are (C, ρ) stable



Control Theoretic Stability 🤸

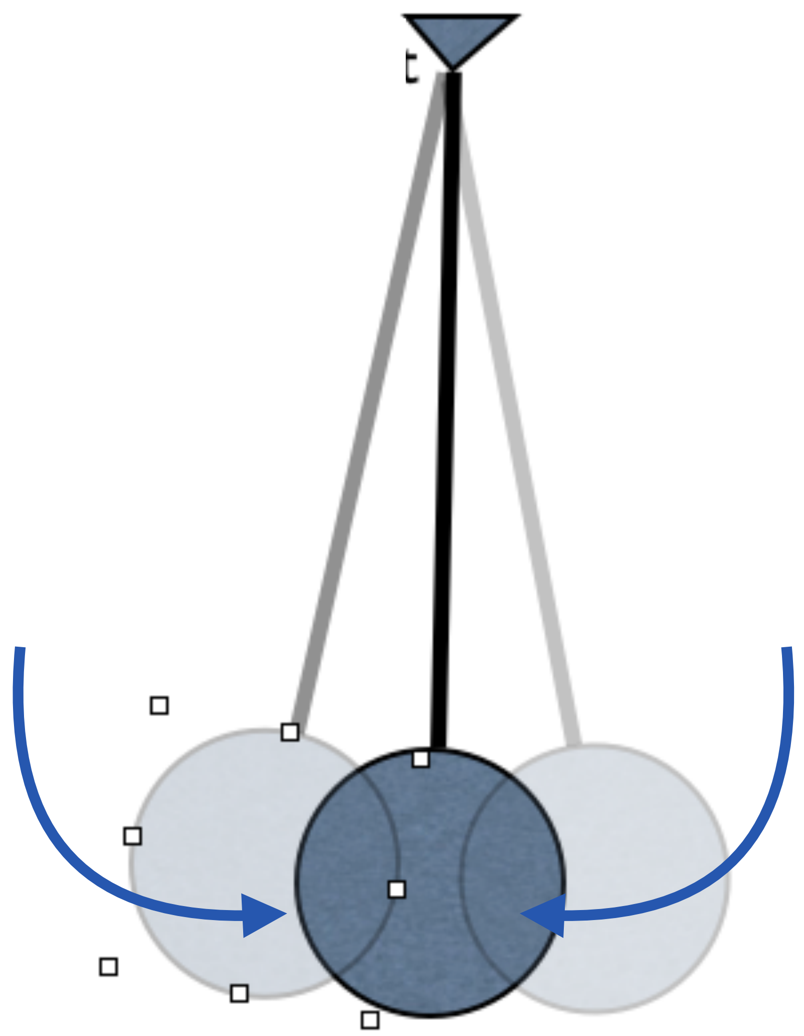


We assume that the following are (C, ρ) **stable**

1. **"open loop"** $(x, u) \to f(x, u)$

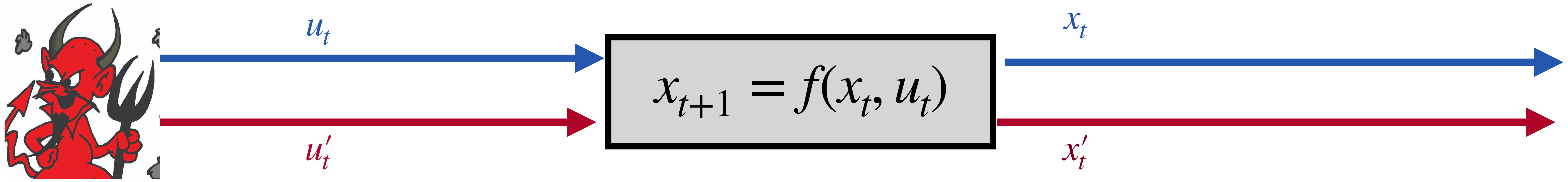
Control Theoretic Stability

stable



We assume that the following are (C, ρ) stable

1. **"open loop"** $(x, u) \to f(x, u)$



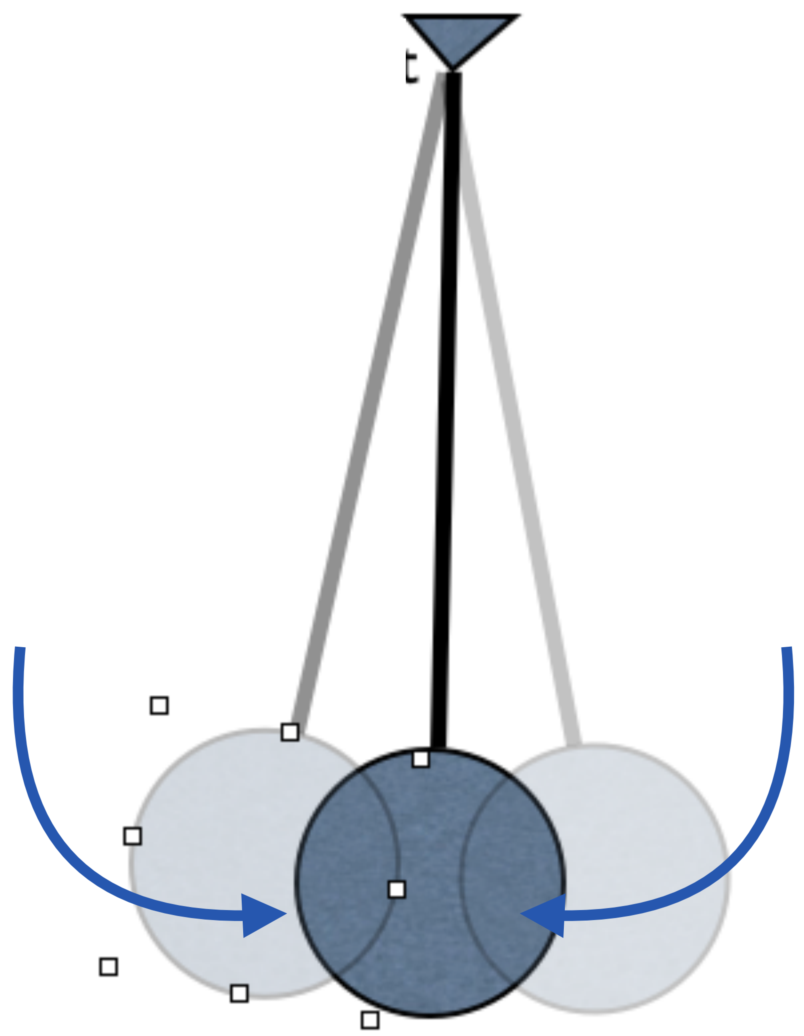
open-loop stable

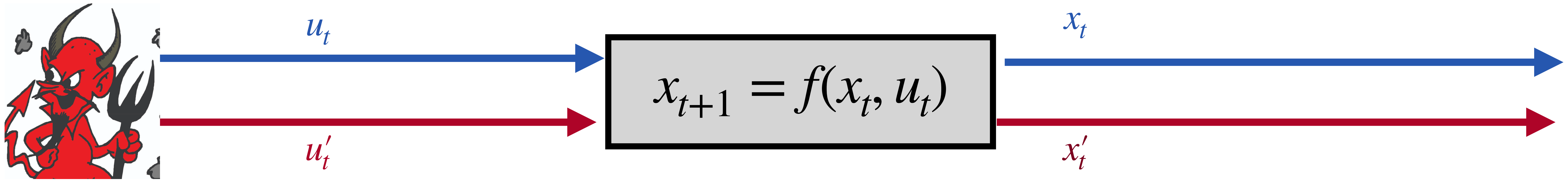
Control Theoretic Stability

We assume that the following are (C, ρ) stable

1. **"open loop"** $(x, u) \to f(x, u)$
2. **"closed loop"** $(x, u) \to f(x, \pi^\star(x) + u)$

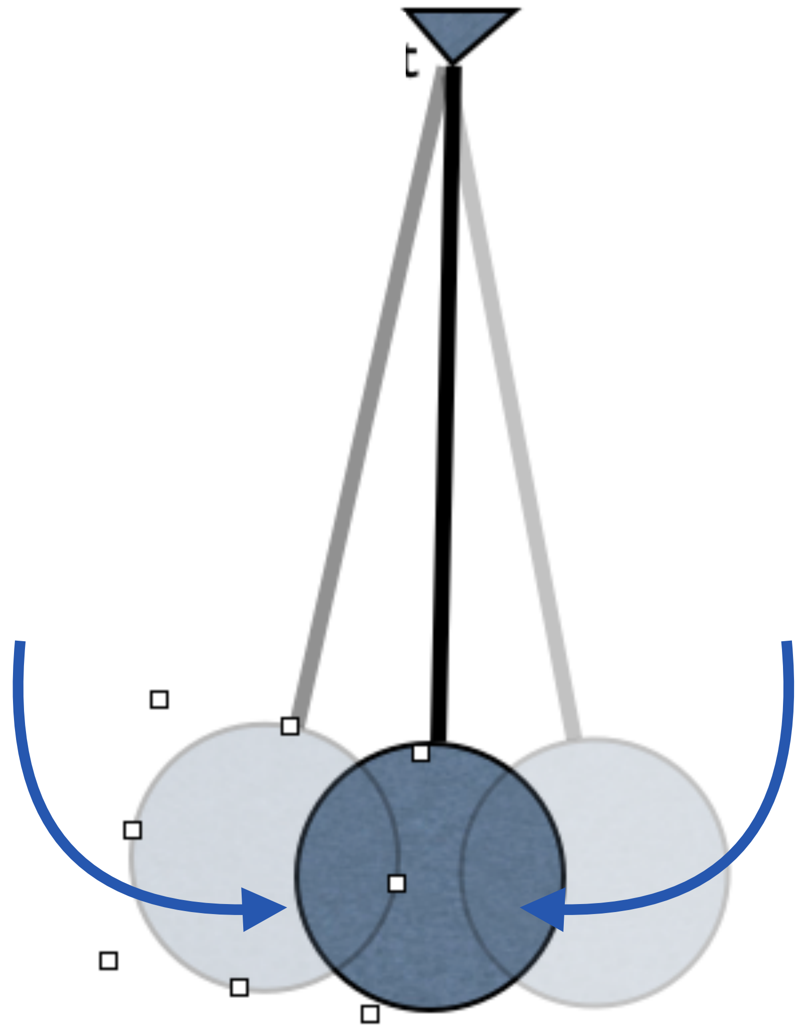
stable





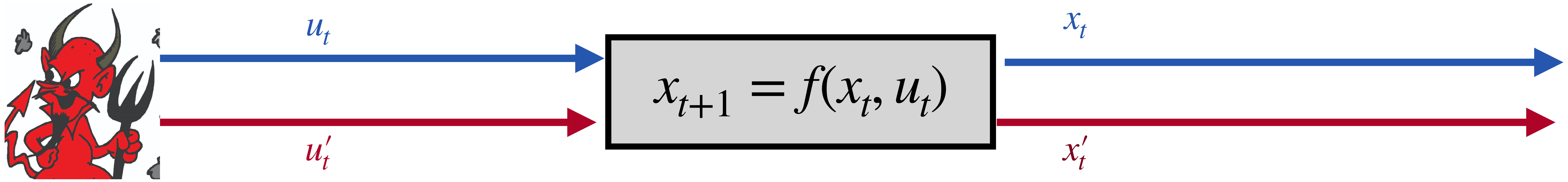
open-loop stable

Control Theoretic Stability

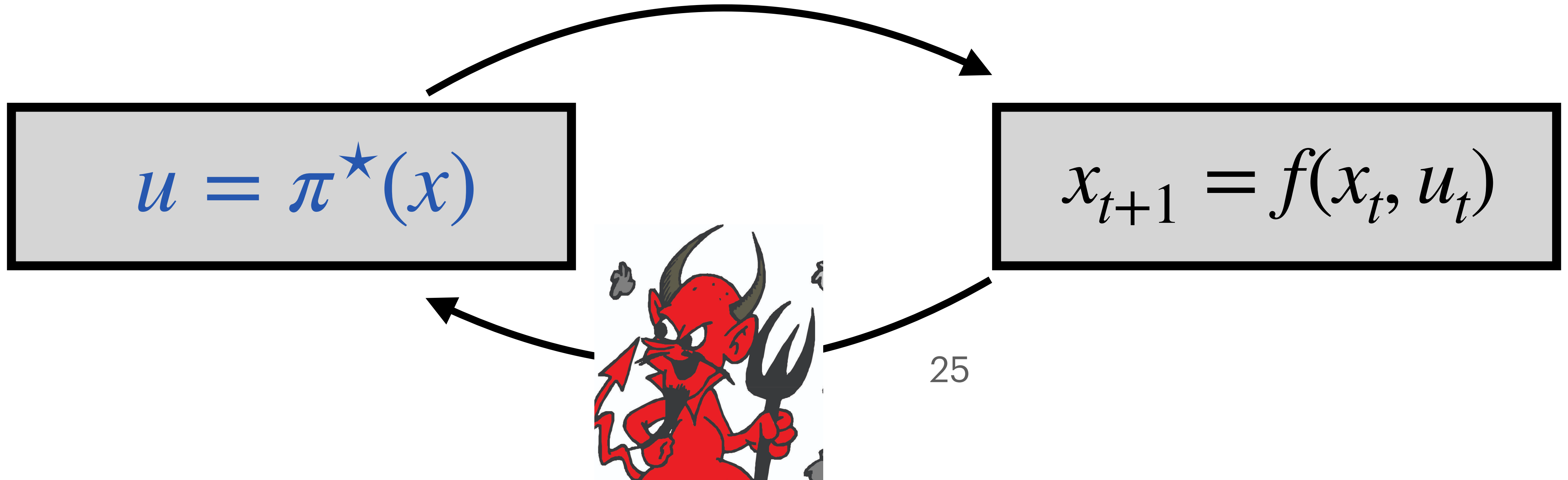


We assume that the following are (C, ρ) stable

1. **"open loop"** $(x, u) \to f(x, u)$
2. **"closed loop"** $(x, u) \to f(x, \pi^\star(x) + u)$



open-loop stable

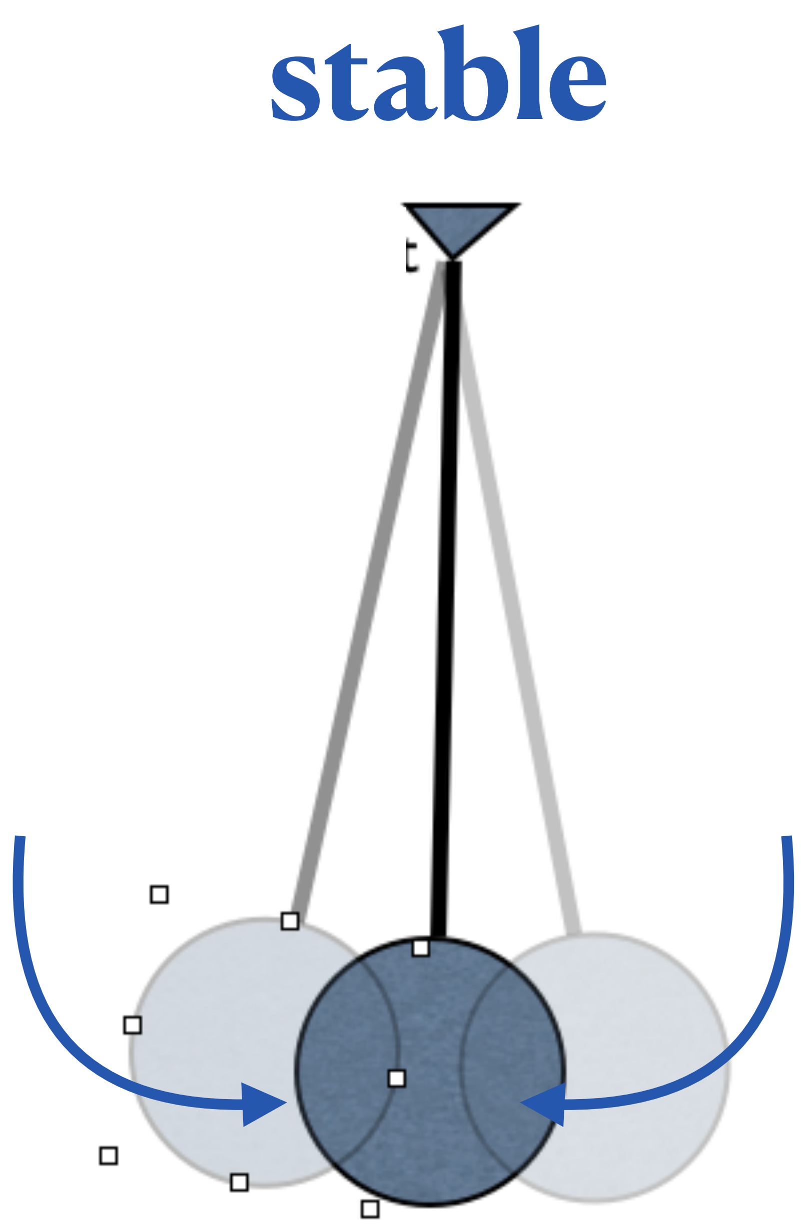


closed-loop stable

This is surprising.

We assume that the following are (C, ρ) **stable**

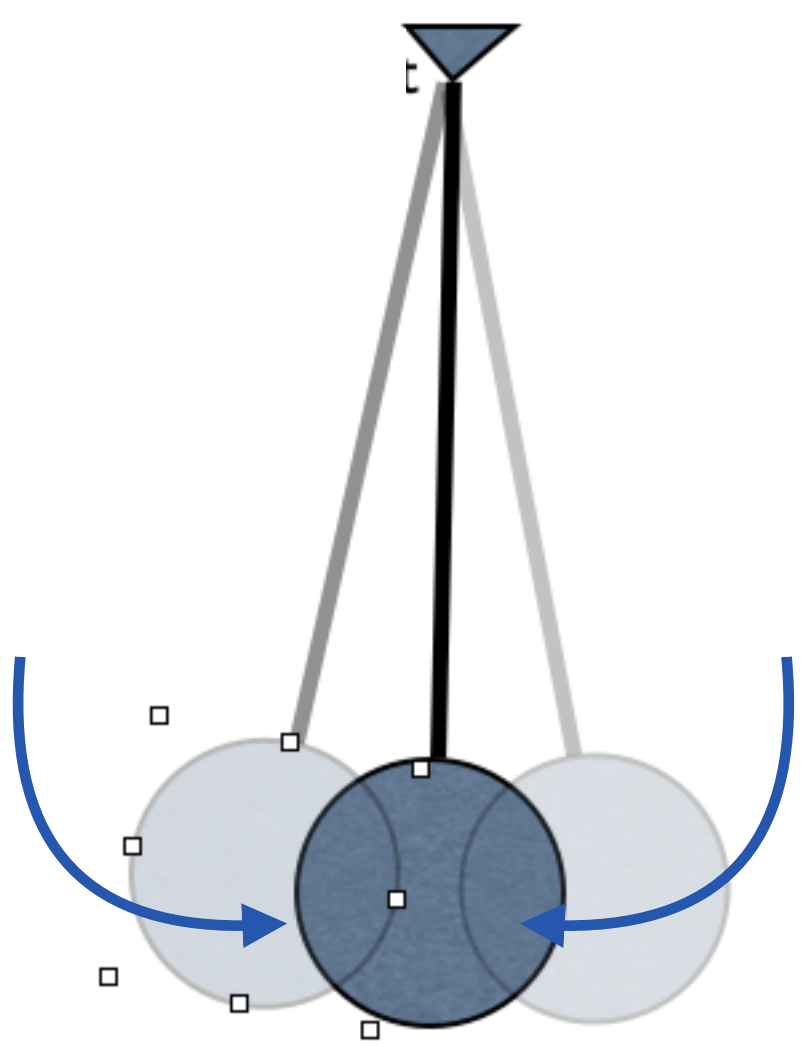
1. "open loop" $(x, u) \to f(x, u)$
2. "closed loop" $(x, u) \to f(x, \pi^\star(x) + u)$



This is surprising.

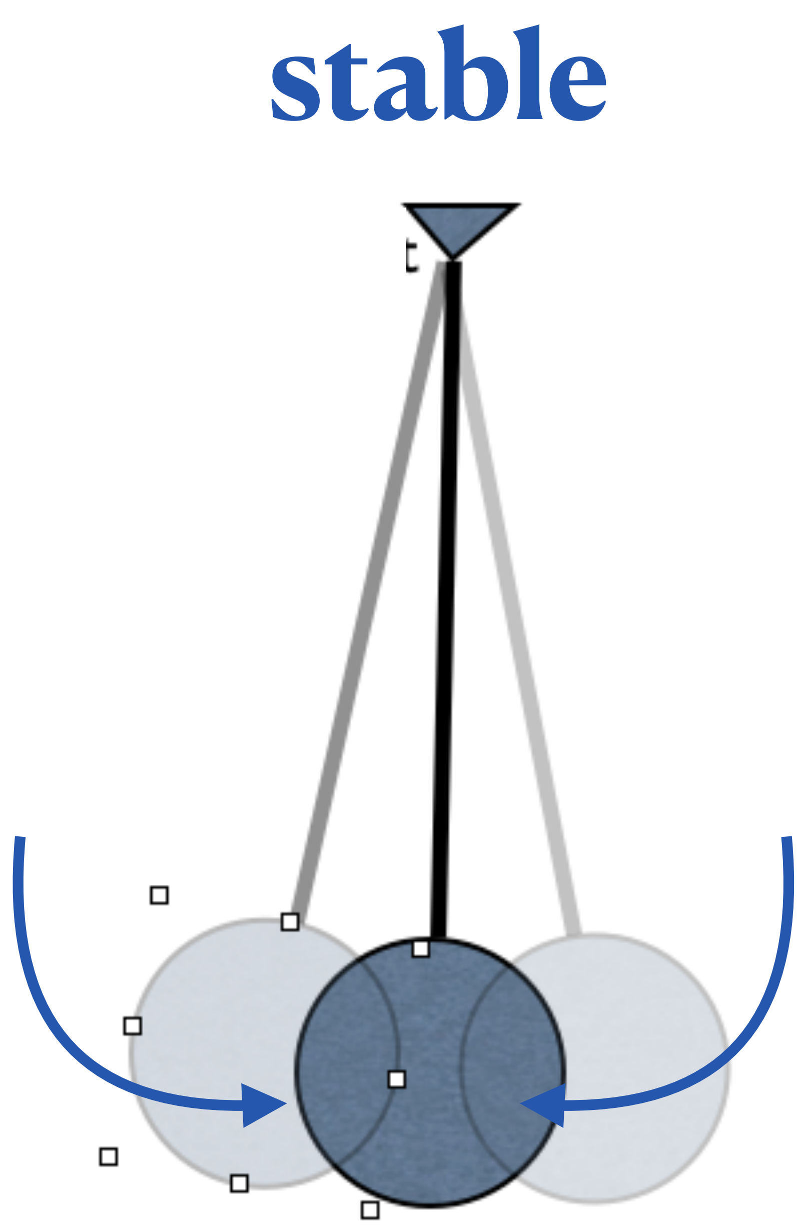
We assume that the following are (C, ρ) **stable**

1. **"open loop"** $(x, u) \to f(x, u)$
2. **"closed loop"** $(x, u) \to f(x, \pi^\star(x) + u)$



Recall error compound exponentially $\mathcal{R}_c(\hat{\pi}; \pi^\star) \gtrsim 2^H \cdot \mathcal{R}_{\text{expert}}(\hat{\pi}; \pi^\star)$

This is surprising.



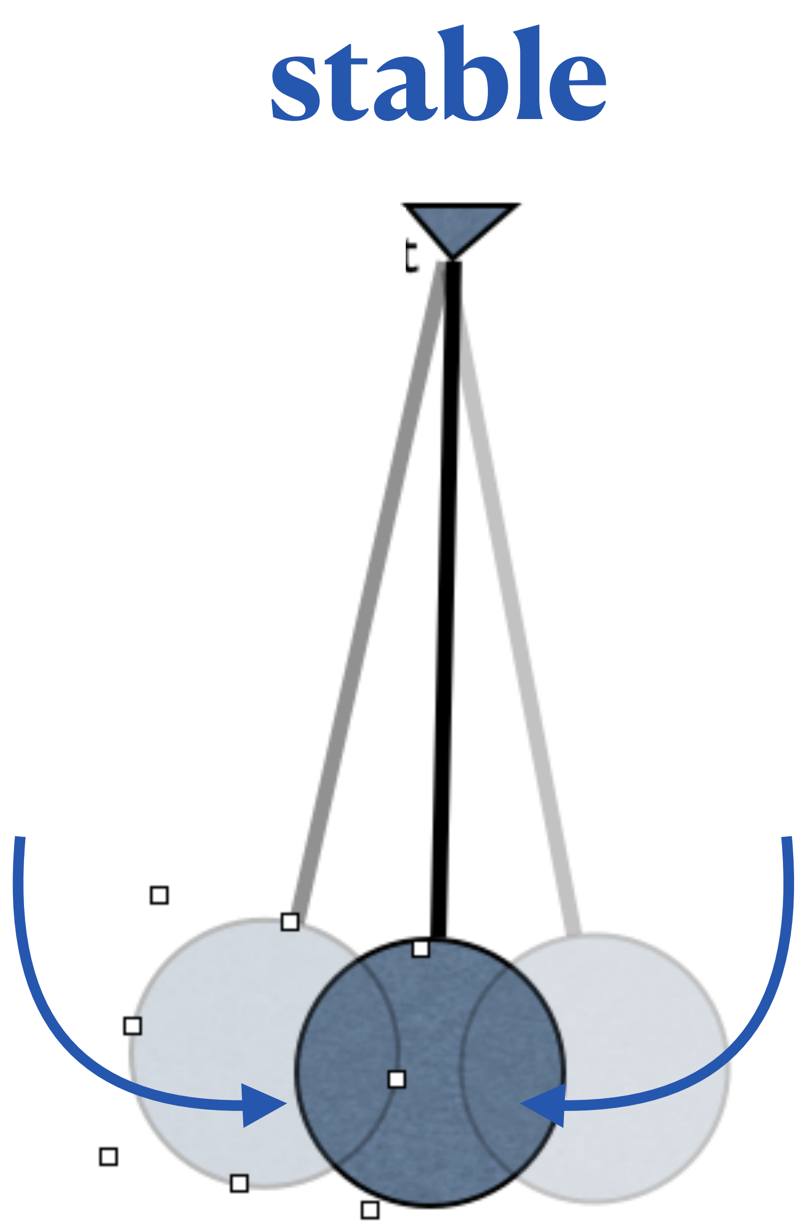
We assume that the following are (C, ρ) **stable**

1. **"open loop"** $(x, u) \to f(x, u)$
2. **"closed loop"** $(x, u) \to f(x, \pi^\star(x) + u)$

Recall error compound exponentially $\mathscr{R}_c(\hat{\pi}; \pi^\star) \gtrsim 2^H \cdot \mathscr{R}_{\mathrm{expert}}(\hat{\pi}; \pi^\star)$

But dynamics forget mistakes exponentially quickly

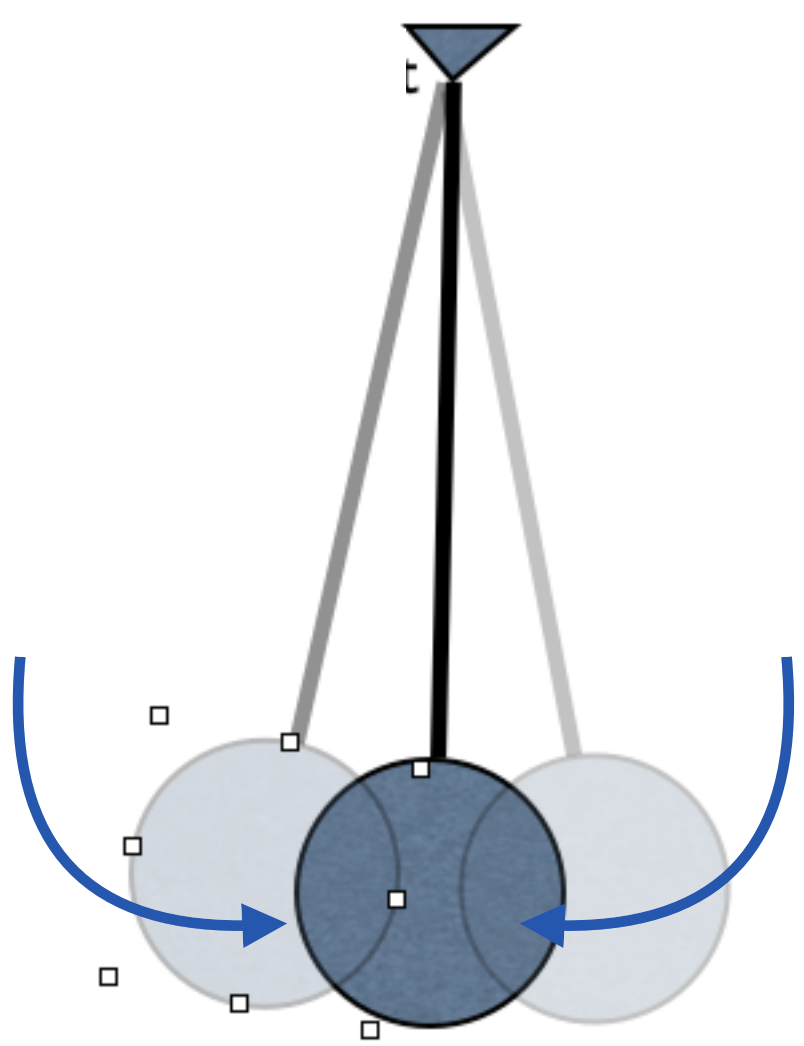
This is surprising.



Recall error compound **exponentially** $\mathscr{R}_c(\hat{\pi}; \pi^\star) \gtrsim 2^H \cdot \mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star)$

But dynamics forget mistakes **exponentially quickly**

This is surprising.

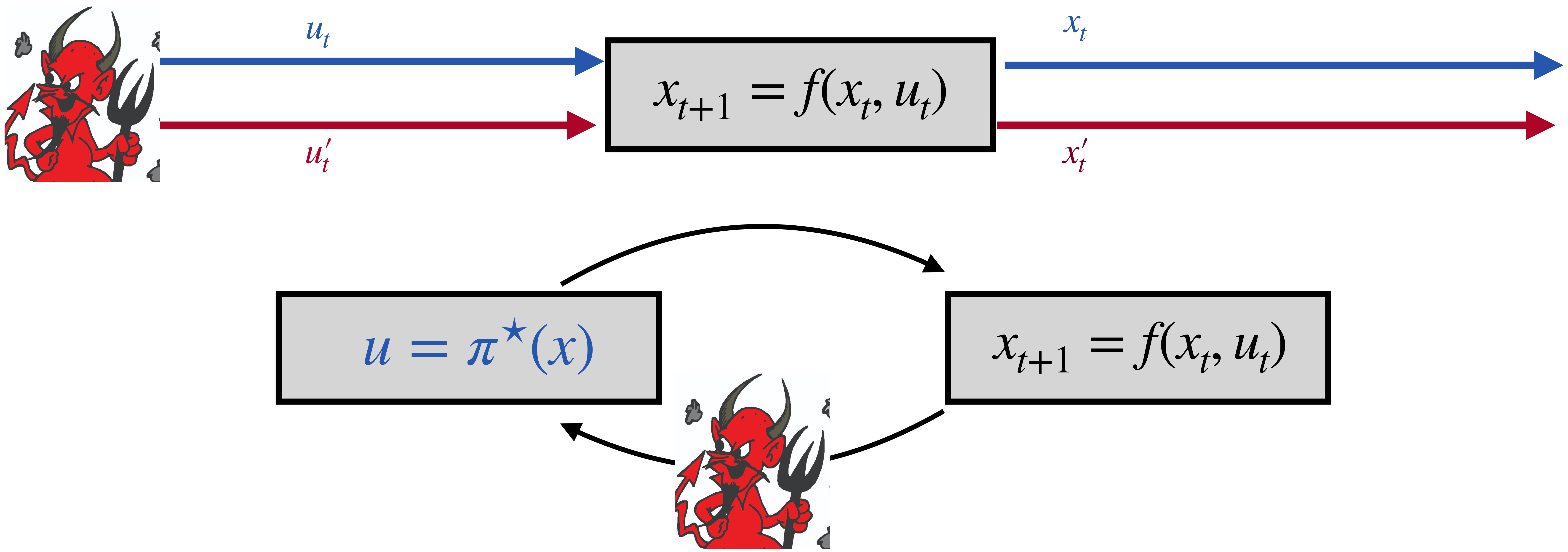


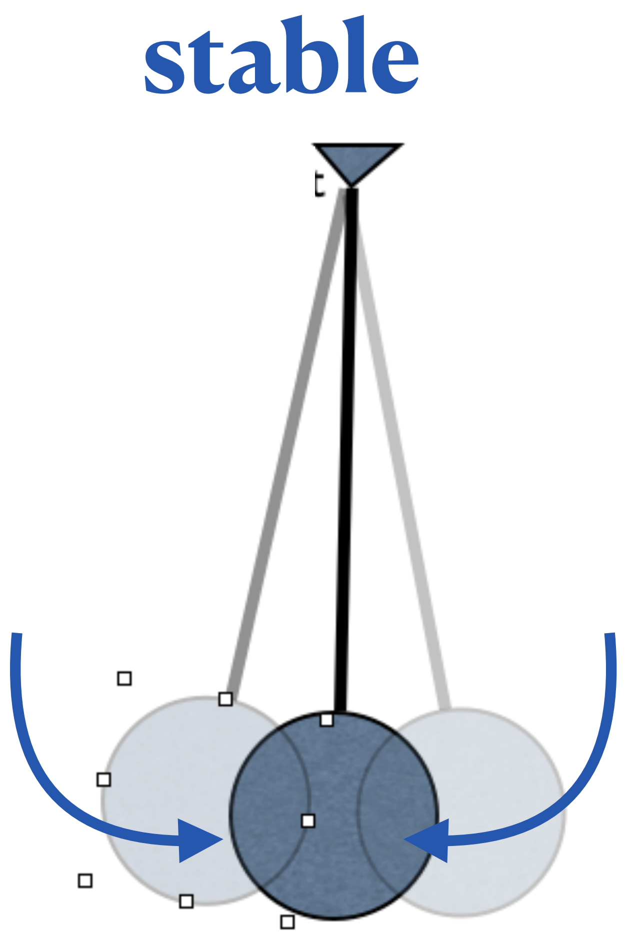
Takeaway: learning in the physical world 🤖 can be hard even if the problems seems benign

Recall error compound exponentially $\mathscr{R}_c(\hat{\pi}; \pi^\star) \gtrsim 2^H \cdot \mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^\star)$

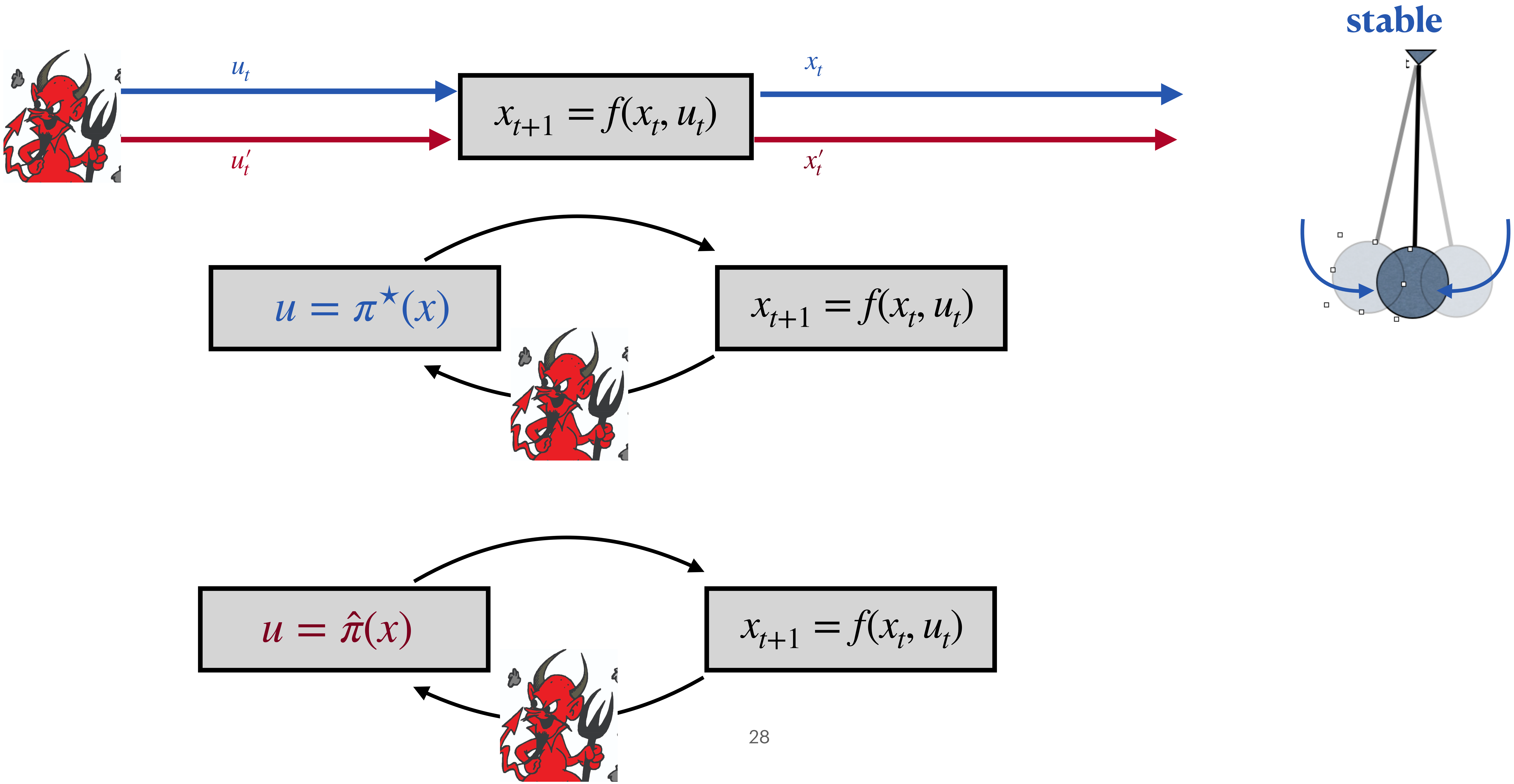
But dynamics forget mistakes exponentially quickly

The Catch.



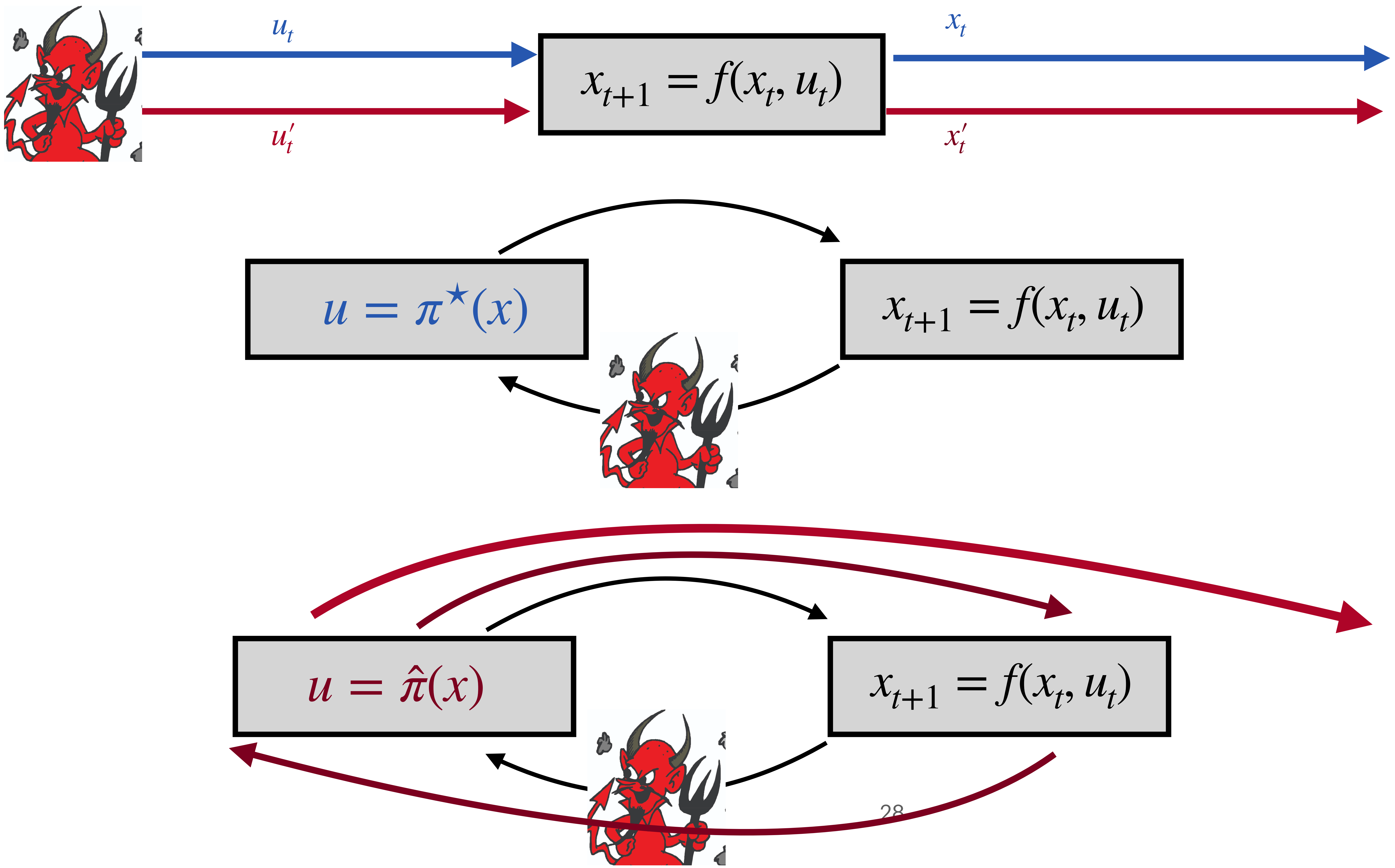


The Catch.

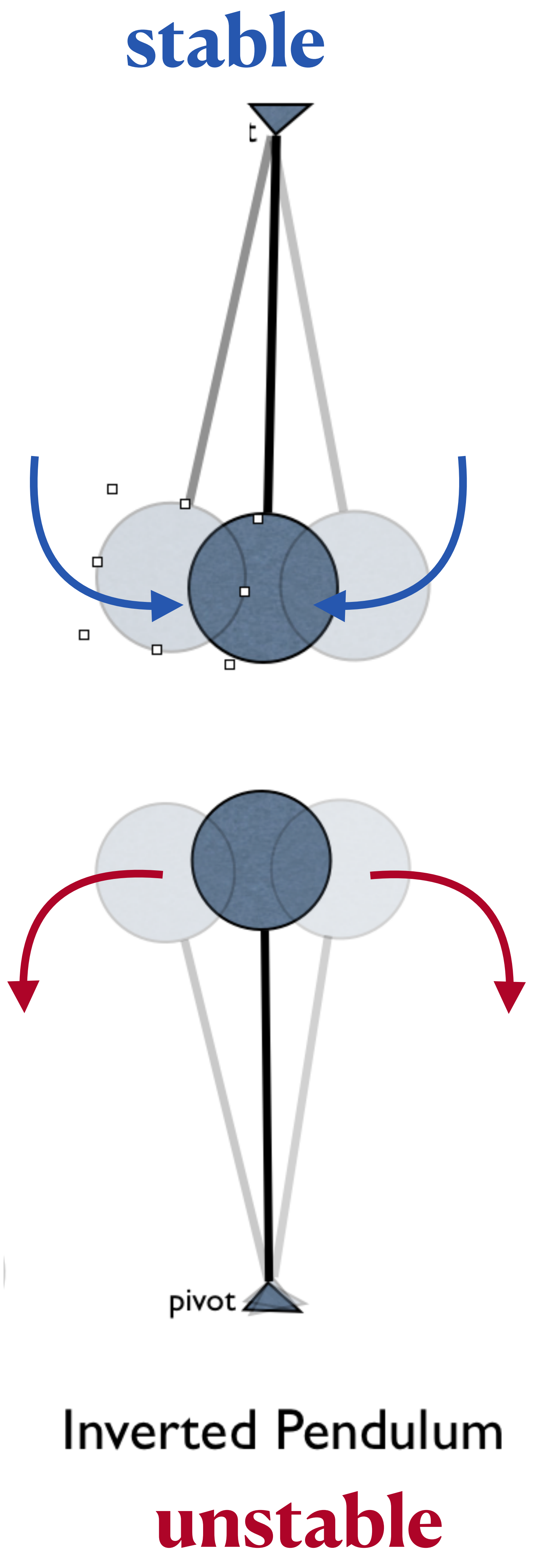


stable
u_t
$x_{t+1} = f(x_t, u_t)$
x_t
u'_t
x'_t
$u = \pi^\star(x)$
$x_{t+1} = f(x_t, u_t)$
$u = \hat{\pi}(x)$
$x_{t+1} = f(x_t, u_t)$

The Catch.



u_t
$x_{t+1} = f(x_t, u_t)$
x_t
u'_t
x'_t
$u = \pi^\star(x)$
$x_{t+1} = f(x_t, u_t)$
$u = \hat{\pi}(x)$
$x_{t+1} = f(x_t, u_t)$



stable
pivot
Inverted Pendulum
unstable

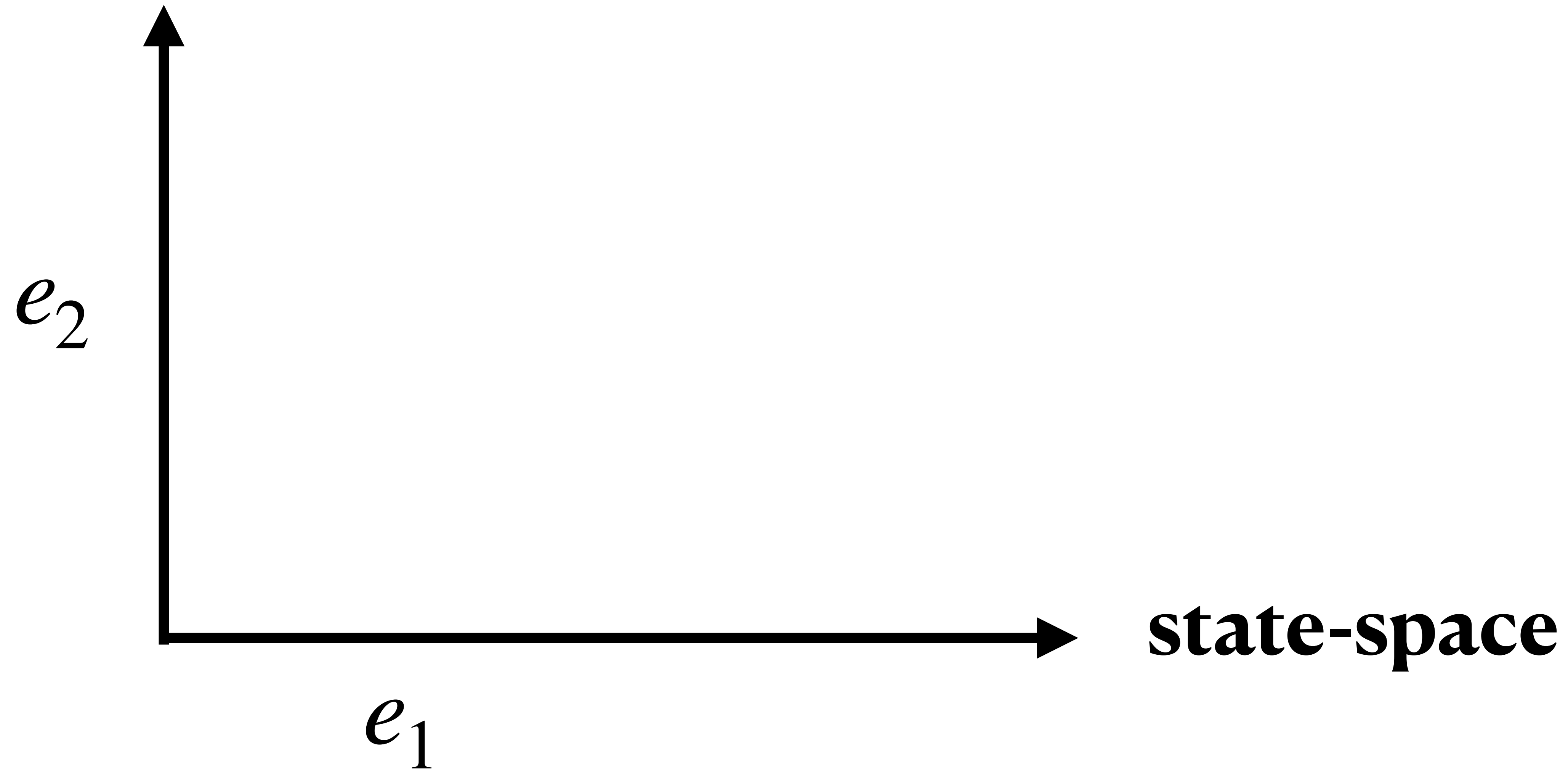
Proof Idea: “Catch 22”

Proof Idea: "Catch 22"

Step 1: Lower Bound for Linear Systems. There exists a pair of 2 dimensional linear dynamical system $x_{t+1} = A_i x_t + u_t$ and associated linear control policies $\pi_i(x) = K_i x$ s.t.

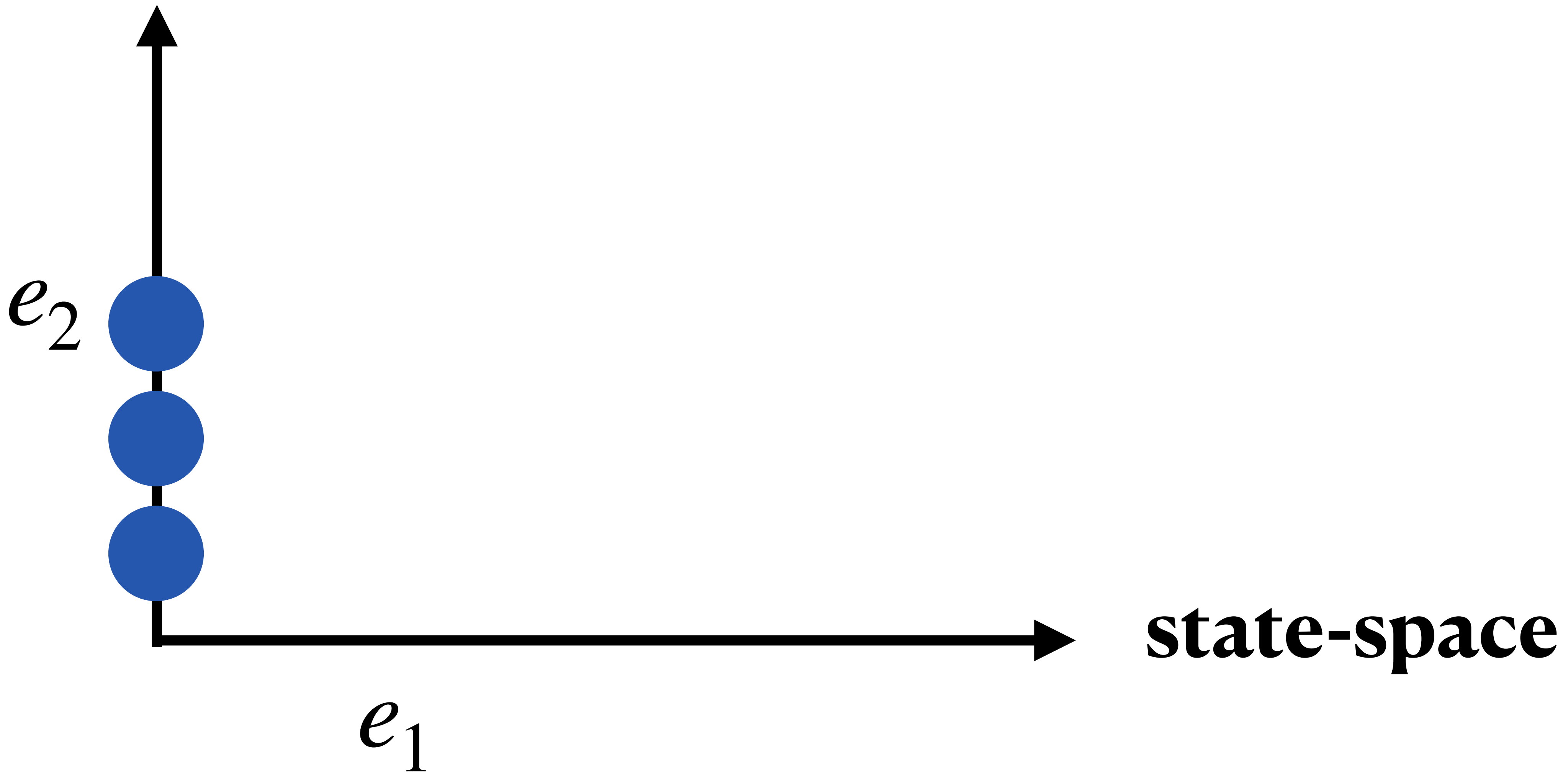
Proof Idea: “Catch 22”

Step 1: Lower Bound for Linear Systems. There exists a pair of 2 dimensional linear dynamical system $x_{t+1} = A_i x_t + u_t$ and associated linear control policies $\pi_i(x) = K_i x$ s.t.



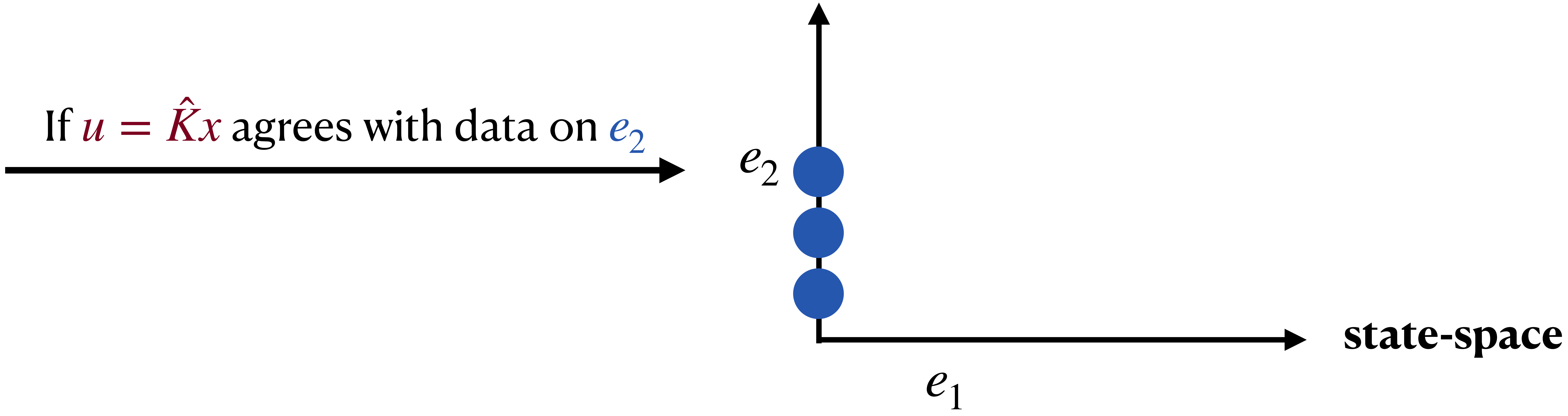
Proof Idea: "Catch 22"

Step 1: Lower Bound for Linear Systems. There exists a pair of 2 dimensional linear dynamical system $x_{t+1} = A_i x_t + u_t$ and associated linear control policies $\pi_i(x) = K_i x$ s.t.



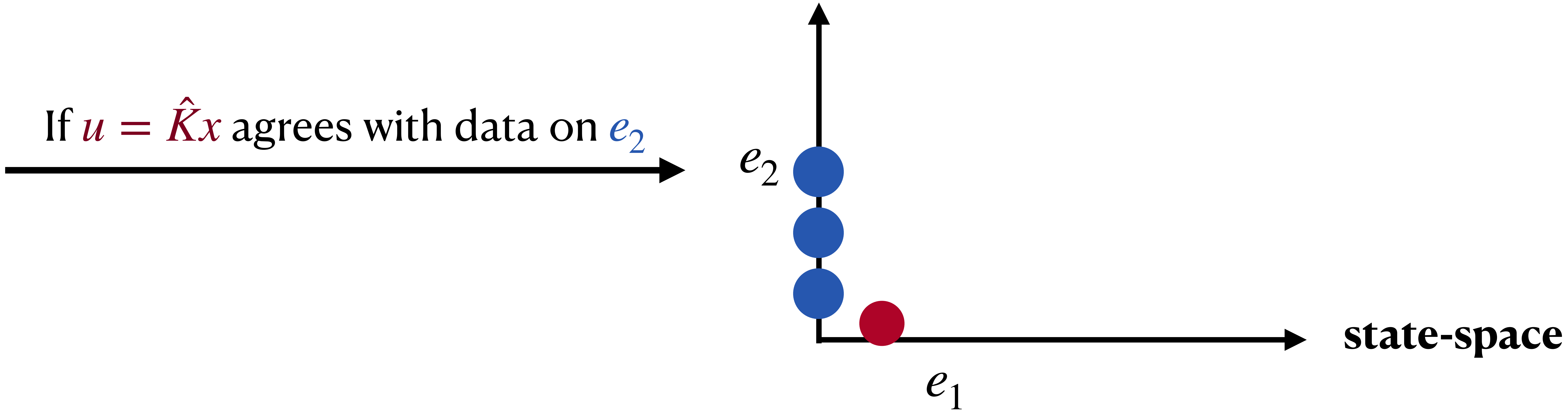
Proof Idea: “Catch 22”

Step 1: Lower Bound for Linear Systems. There exists a pair of 2 dimensional linear dynamical system $x_{t+1} = A_i x_t + u_t$ and associated linear control policies $\pi_i(x) = K_i x$ s.t.



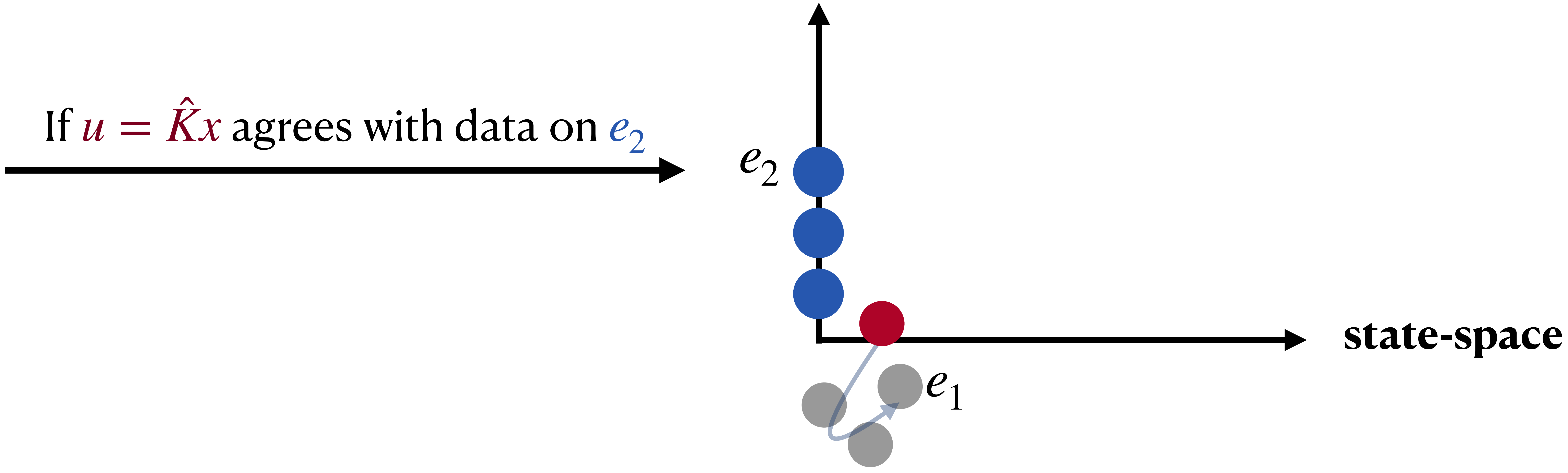
Proof Idea: “Catch 22”

Step 1: Lower Bound for Linear Systems. There exists a pair of 2 dimensional linear dynamical system $x_{t+1} = A_i x_t + u_t$ and associated linear control policies $\pi_i(x) = K_i x$ s.t.



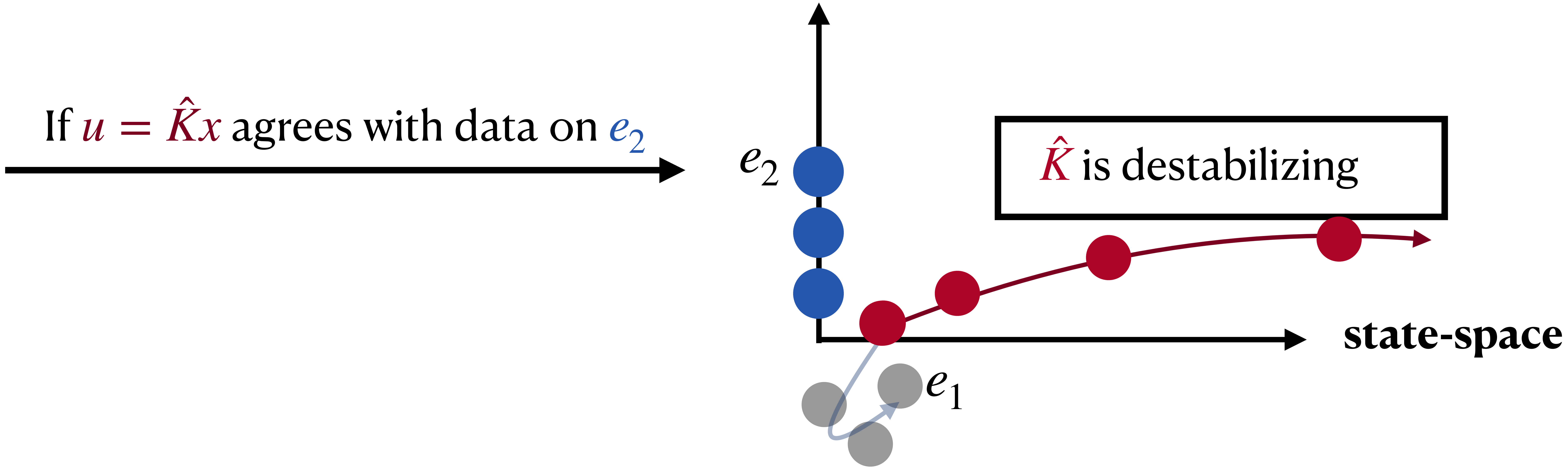
Proof Idea: "Catch 22"

Step 1: Lower Bound for Linear Systems. There exists a pair of 2 dimensional linear dynamical system $x_{t+1} = A_i x_t + u_t$ and associated linear control policies $\pi_i(x) = K_i x$ s.t.



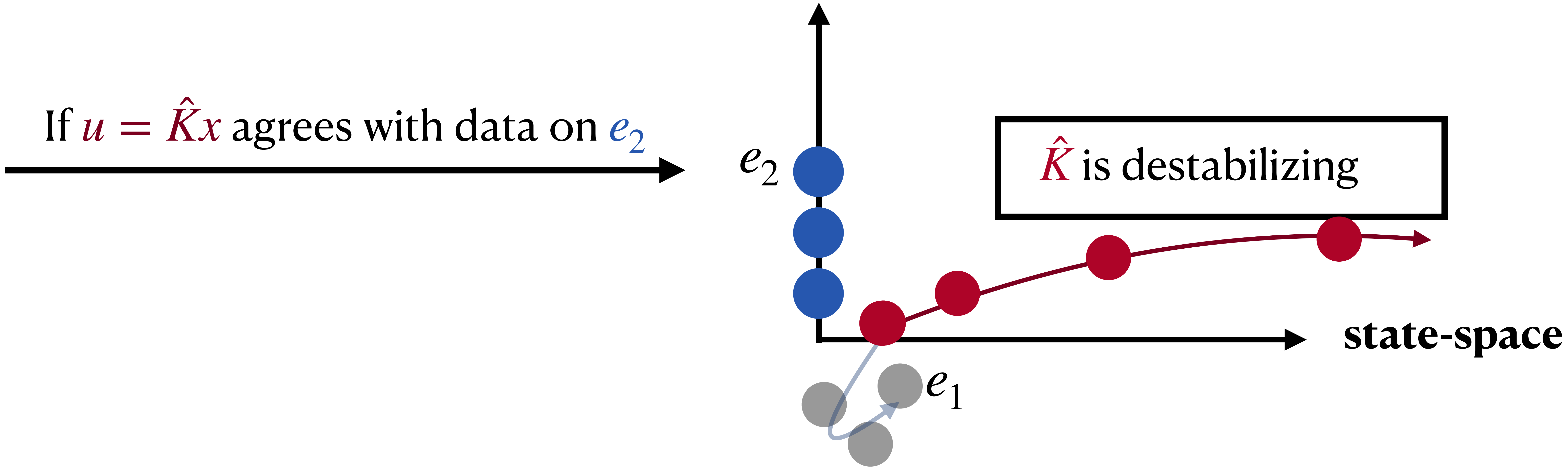
Proof Idea: "Catch 22"

Step 1: Lower Bound for Linear Systems. There exists a pair of 2 dimensional linear dynamical system $x_{t+1} = A_i x_t + u_t$ and associated linear control policies $\pi_i(x) = K_i x$ s.t.



Proof Idea: "Catch 22"

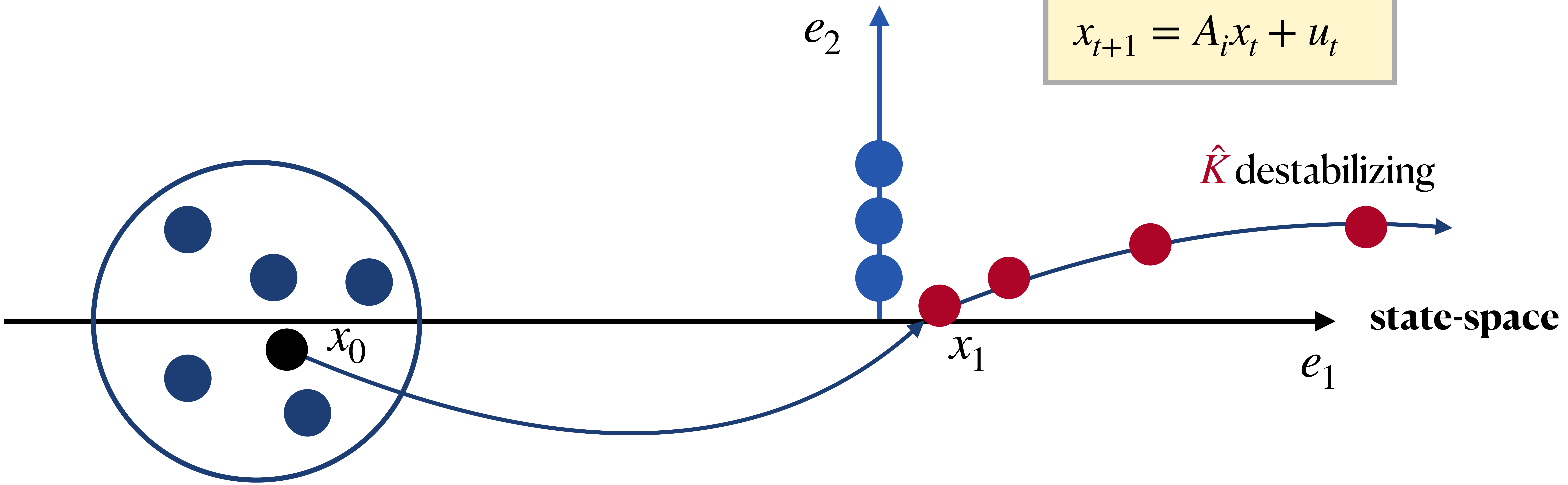
Step 1: Lower Bound for Linear Systems. There exists a pair of 2 dimensional linear dynamical system $x_{t+1} = A_i x_t + u_t$ and associated linear control policies $\pi_i(x) = K_i x$ s.t.



Learned policies cannot both follow the expert and stabilize unknown dynamics

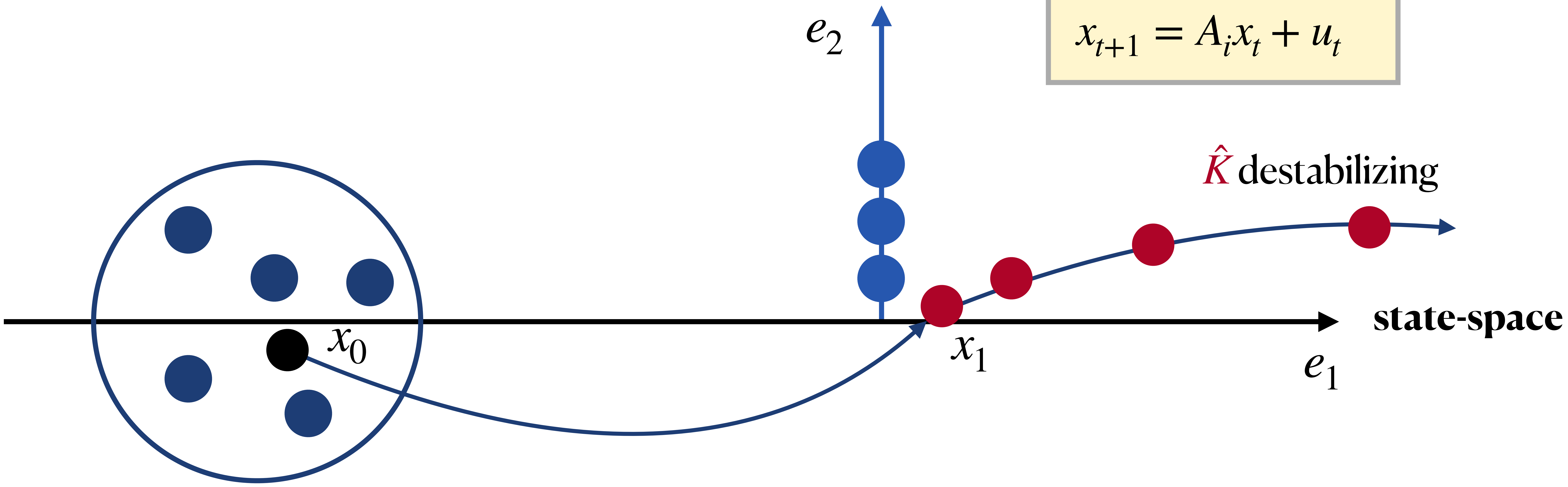
Proof Idea: 'Catch 22'

$$x_{t+1} = A_i x_t + u_t$$



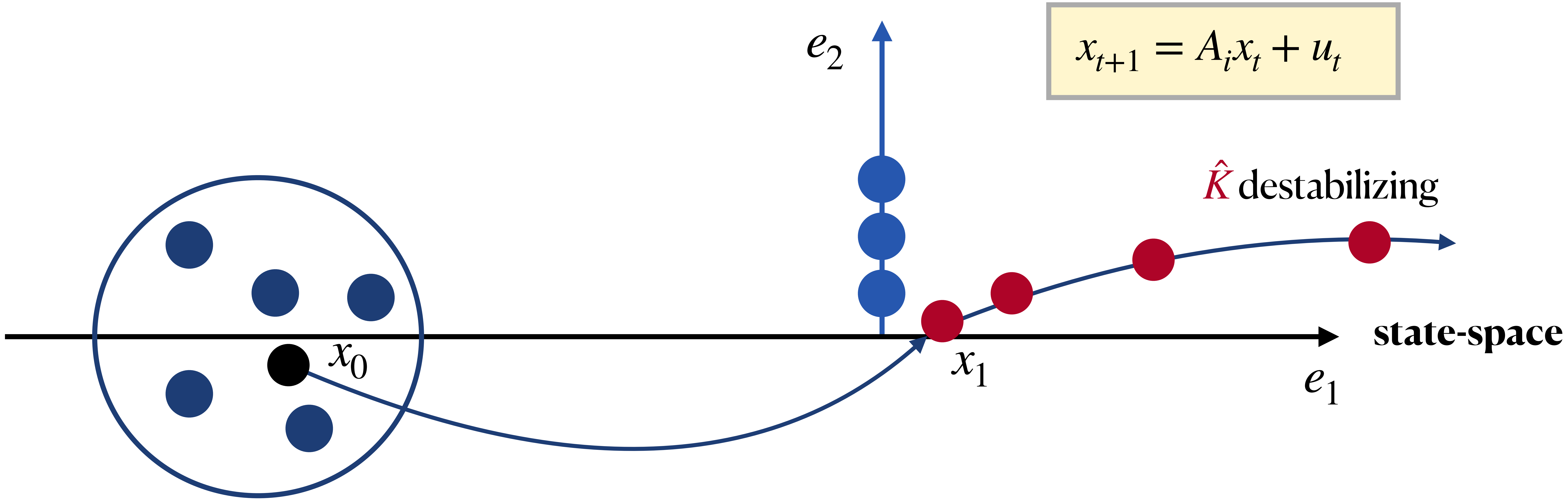
Proof Idea: 'Catch 22'

$$x_{t+1} = A_i x_t + u_t$$



Step 2: Carefully embed a nonparametric learning problem as a source of original error, which becomes amplified by dynamical instability.

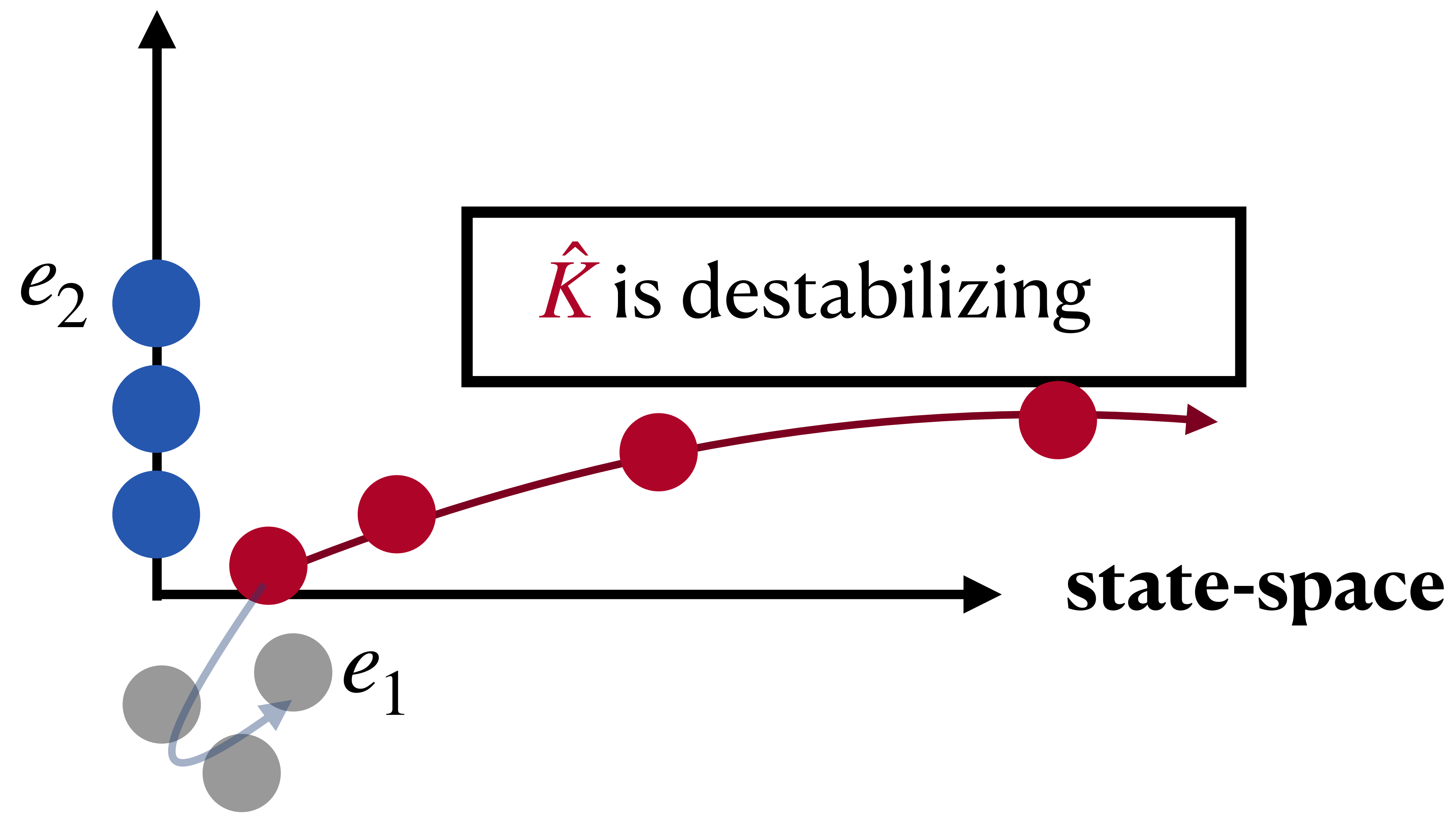
Proof Idea: 'Catch 22'



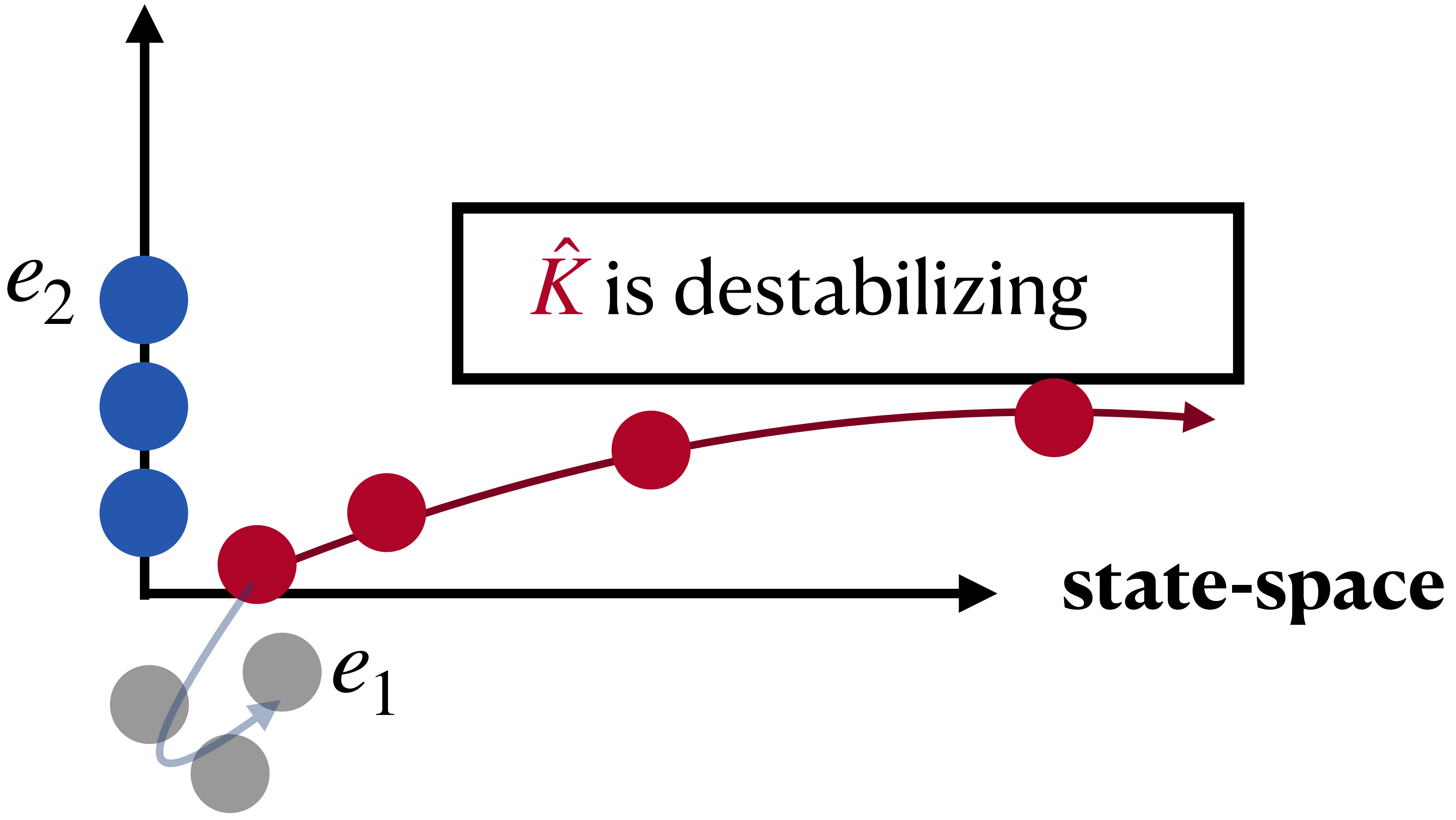
Step 2: Carefully embed a nonparametric learning problem as a source of original error, which becomes amplified by dynamical instability.

Note: This does not arise in the classical bound due to absence of "metric" error

Proof Idea: 'Catch 22'

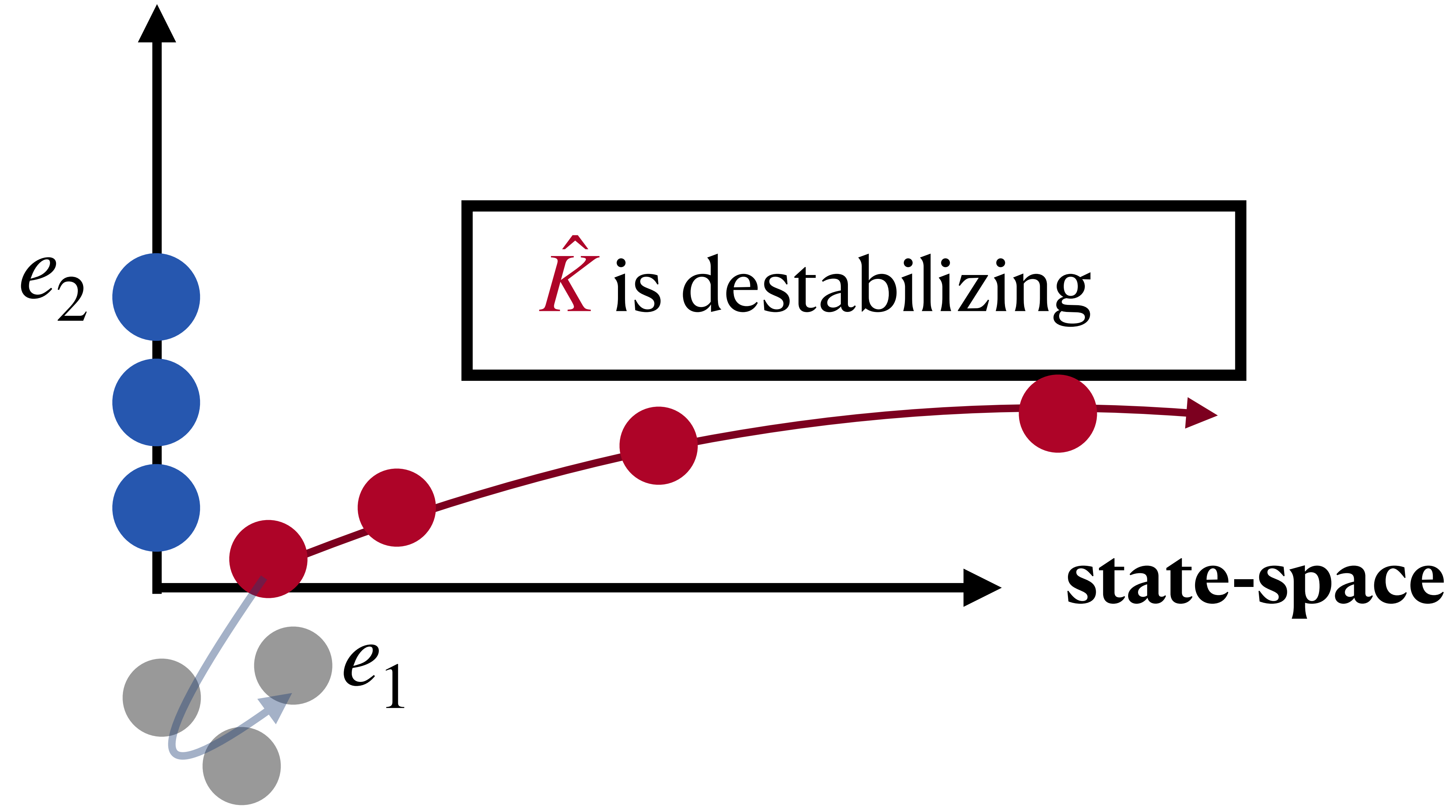


Proof Idea: 'Catch 22'



Learned policies cannot both follow the expert and stabilize unknown dynamics

Proof Idea: ‘Catch 22’



Learned policies cannot both follow the expert and stabilize unknown dynamics

Because the Physical World 🤖 involves “perturbative error,” pushing us out of distribution, learning can be much harder!

Act 3: “What to do about it?”

*w/ **Thomas Zhang,** Daniel Pfrommer, Nikolai Matni (UPenn+MIT)*

The Caveat

The Caveat

While the negative result holds if dynamics are unstable, it only applies to stable dynamics if $\hat{\pi}$ is a "simple policy"

The Caveat

While the negative result holds if dynamics are unstable, it only applies to stable dynamics if $\hat{\pi}$ is a "simple policy"

$$\hat{\pi}(x) = \pi^{\text{determ}}(x) + (\textbf{independent noise})$$

note: expert is also 'simple'

The Caveat

While the negative result holds if dynamics are unstable, it only applies to stable dynamics if $\hat{\pi}$ is a "simple policy"

$$\hat{\pi}(x) = \pi^{\text{determ}}(x) + (\textbf{independent noise})$$

note: expert is also 'simple'

Unlike language pertaining, naive imitation does not work. However, **better policy representation** + **better data** can overcome the challenges of **physical world learning** 🤖.

Action Chunking

Action Chunking

Definition: Action Chunking is the practice of predicting a sequence of $k \geq 1$ inputs $(u_1, u_2, \ldots, u_k)$ at a time, and committing to them.

Action Chunking

Definition: Action Chunking is the practice of predicting a sequence of $k \geq 1$ inputs $(u_1, u_2, \ldots, u_k)$ at a time, and committing to them.

One of the most essential practices in modern robotics, but hitherto mysterious.

What We Get from Action Chunking

Theorem (ZPMS): Given an open-loop stable system, there exists a fixed **k** such that (independent of data amount **n**), s.t. **k**-action chunking gives

$$\mathcal{R}_c(\hat{\pi}; \pi^\star) \leq C_{sys}\, \mathcal{R}_{expert}(\hat{\pi}; \pi^\star)$$

independent of horizon!

What We Get from Action Chunking

Theorem (ZPMS): Given an open-loop stable system, there exists a fixed **k** such that (independent of data amount **n**), s.t. **k**-action chunking gives

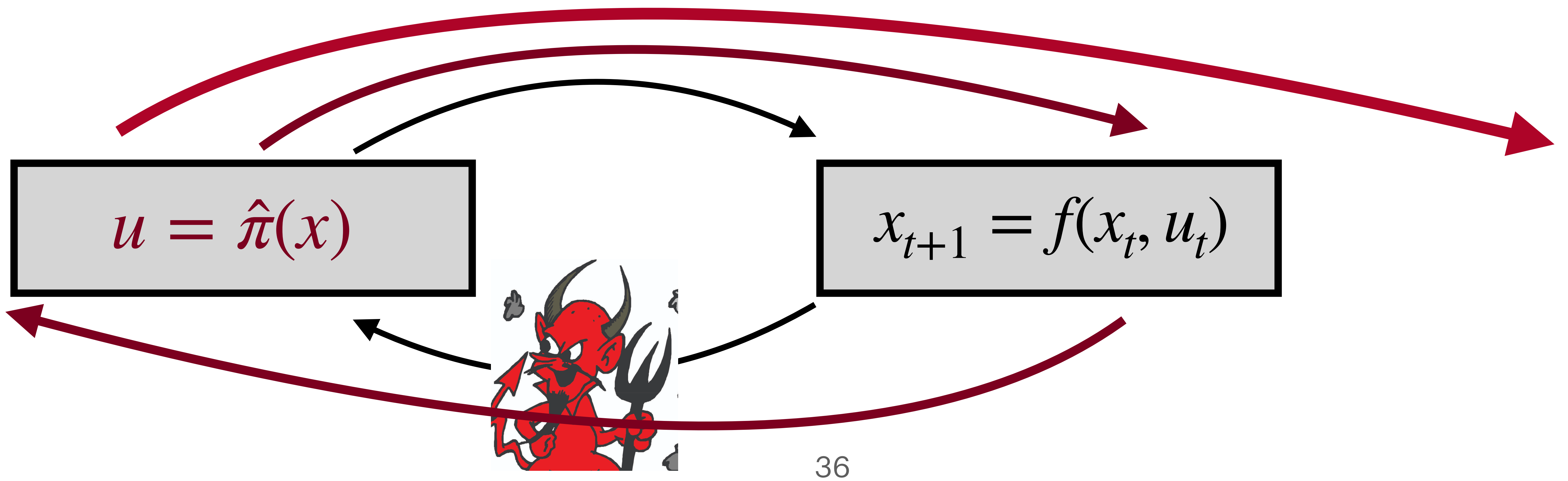
$$\mathcal{R}_{\mathrm{c}}(\hat{\pi}; \pi^{\star}) \leq C_{\mathrm{sys}}\, \mathcal{R}_{\mathrm{expert}}(\hat{\pi}; \pi^{\star})$$

What We Get from Action Chunking

Theorem (ZPMS): Given an open-loop stable system, there exists a fixed **k** such that (independent of data amount **n**), s.t. **k**-action chunking gives

$$\mathcal{R}_c(\hat{\pi}; \pi^\star) \leq C_{sys}\, \mathcal{R}_{expert}(\hat{\pi}; \pi^\star)$$

Proof Idea: recall that, without action chunking,



spiral out of control.

What We Get from Action Chunking

Theorem (ZPMS): Given an open-loop stable system, there exists a fixed **k** such that (independent of data amount **n**), s.t. **k**-action chunking gives

$$\mathscr{R}_{\mathrm{c}}(\hat{\pi}; \pi^\star) \leq C_{\mathrm{sys}}\, \mathscr{R}_{\mathrm{expert}}(\hat{\pi}; \pi^\star)$$

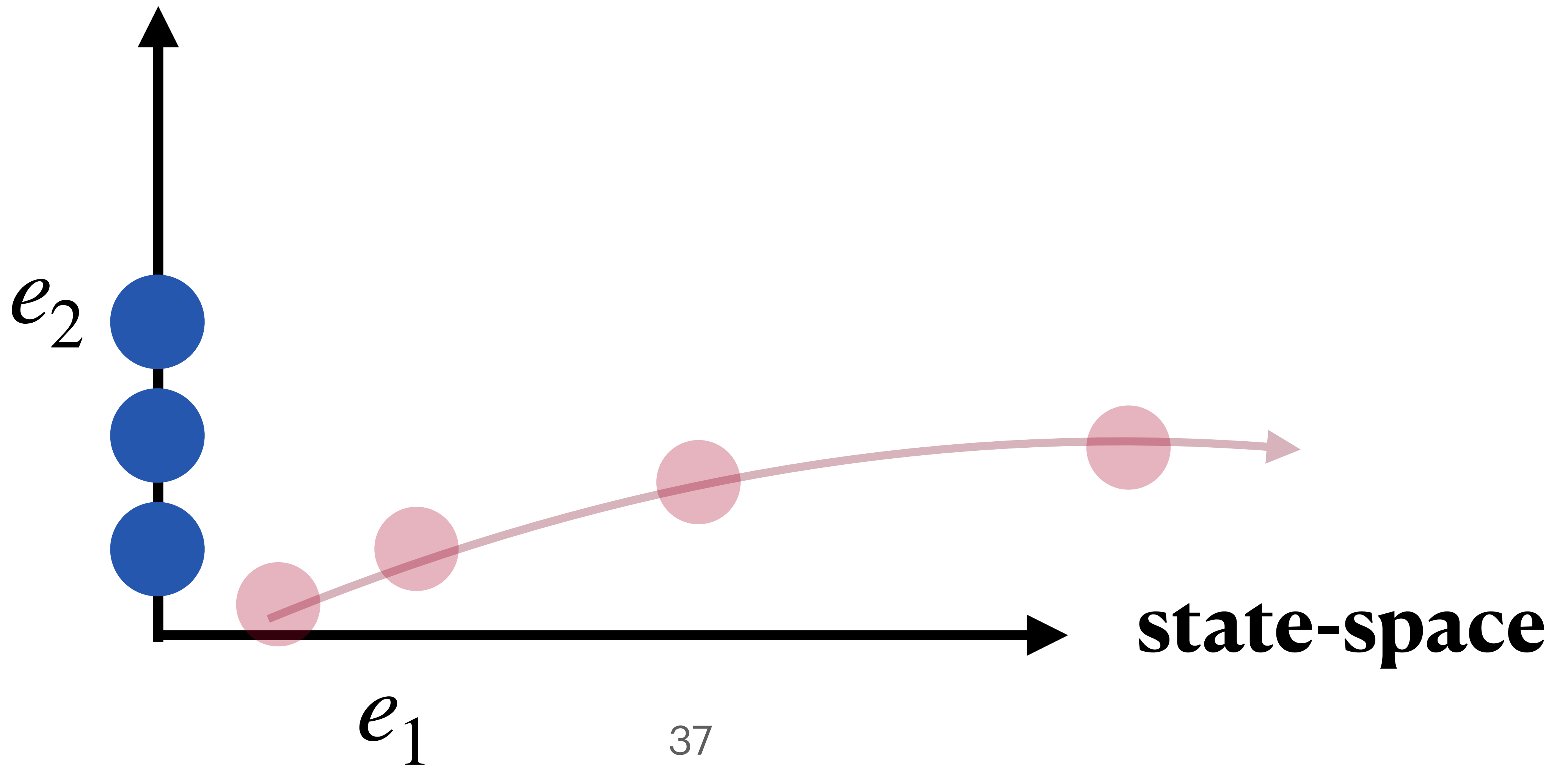
Proof: By updating the policy rarely, you leverage **passive stability of dynamics.**

What We Get from Action Chunking

Theorem (ZPMS): Given an open-loop stable system, there exists a fixed **k** such that (independent of data amount **n**), s.t. **k**-action chunking gives

$$\mathscr{R}_{c}(\hat{\pi}; \pi^{\star}) \leq C_{\text{sys}} \, \mathscr{R}_{\text{expert}}(\hat{\pi}; \pi^{\star})$$

Proof: By updating the policy rarely, you leverage **passive stability of dynamics.**

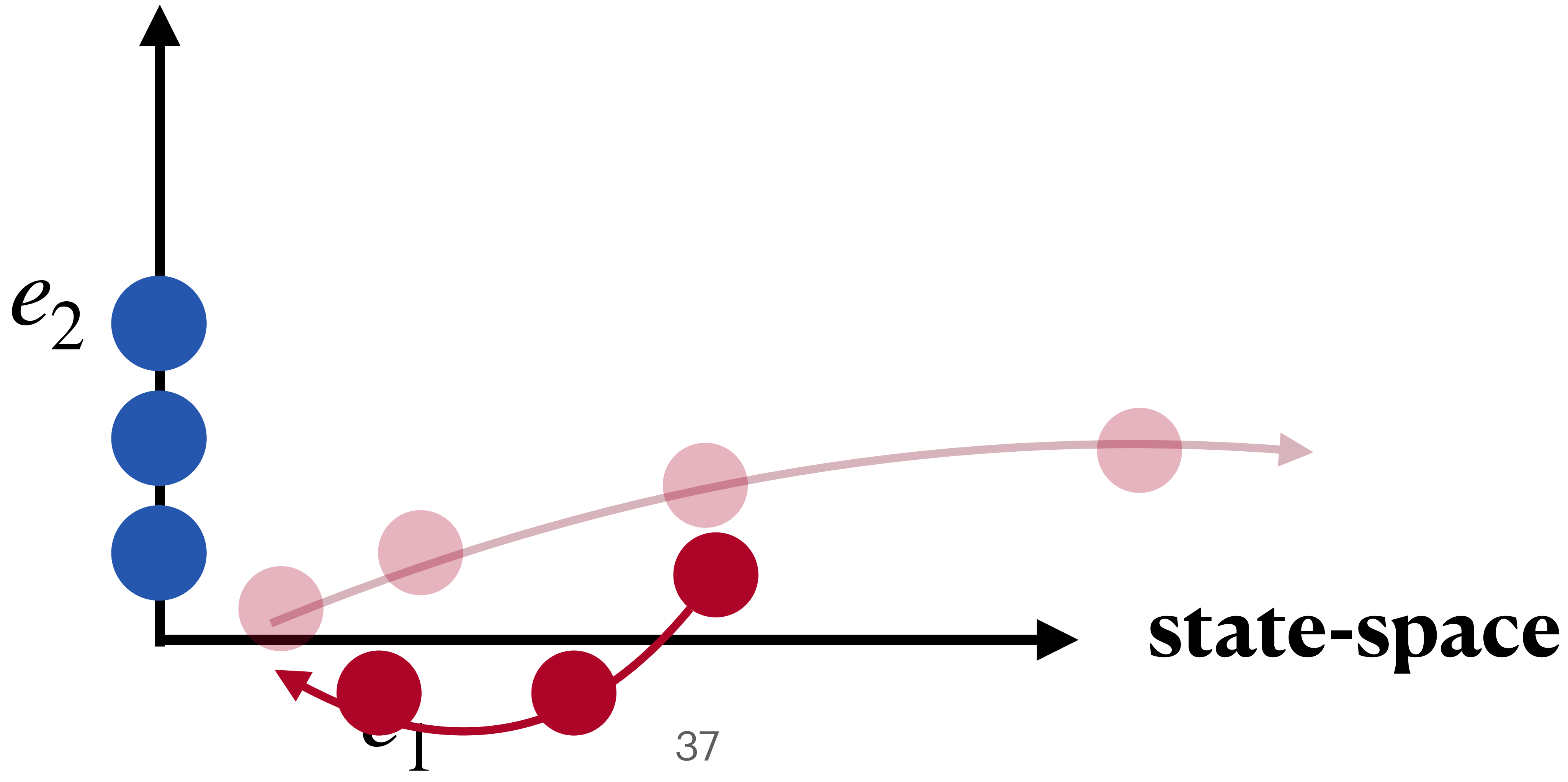


What We Get from Action Chunking

Theorem (ZPMS): Given an open-loop stable system, there exists a fixed **k** such that (independent of data amount **n**), s.t. **k**-action chunking gives

$$\mathcal{R}_{c}(\hat{\pi}; \pi^{\star}) \leq C_{sys}\, \mathcal{R}_{expert}(\hat{\pi}; \pi^{\star})$$

Proof: By updating the policy rarely, you leverage **passive stability of dynamics.**

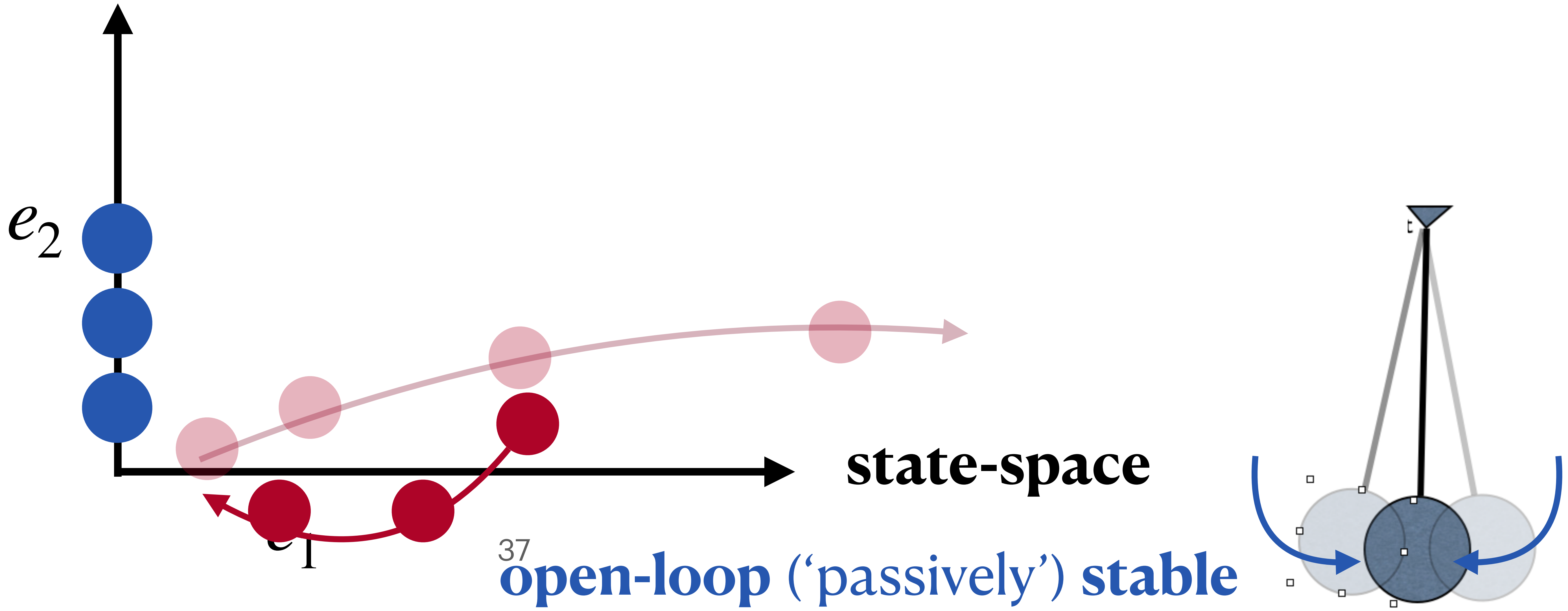


What We Get from Action Chunking

Theorem (ZPMS): Given an open-loop stable system, there exists a fixed **k** such that (independent of data amount **n**), s.t. **k**-action chunking gives

$$\mathcal{R}_c(\hat{\pi}; \pi^\star) \leq C_{\text{sys}} \, \mathcal{R}_{\text{expert}}(\hat{\pi}; \pi^\star)$$

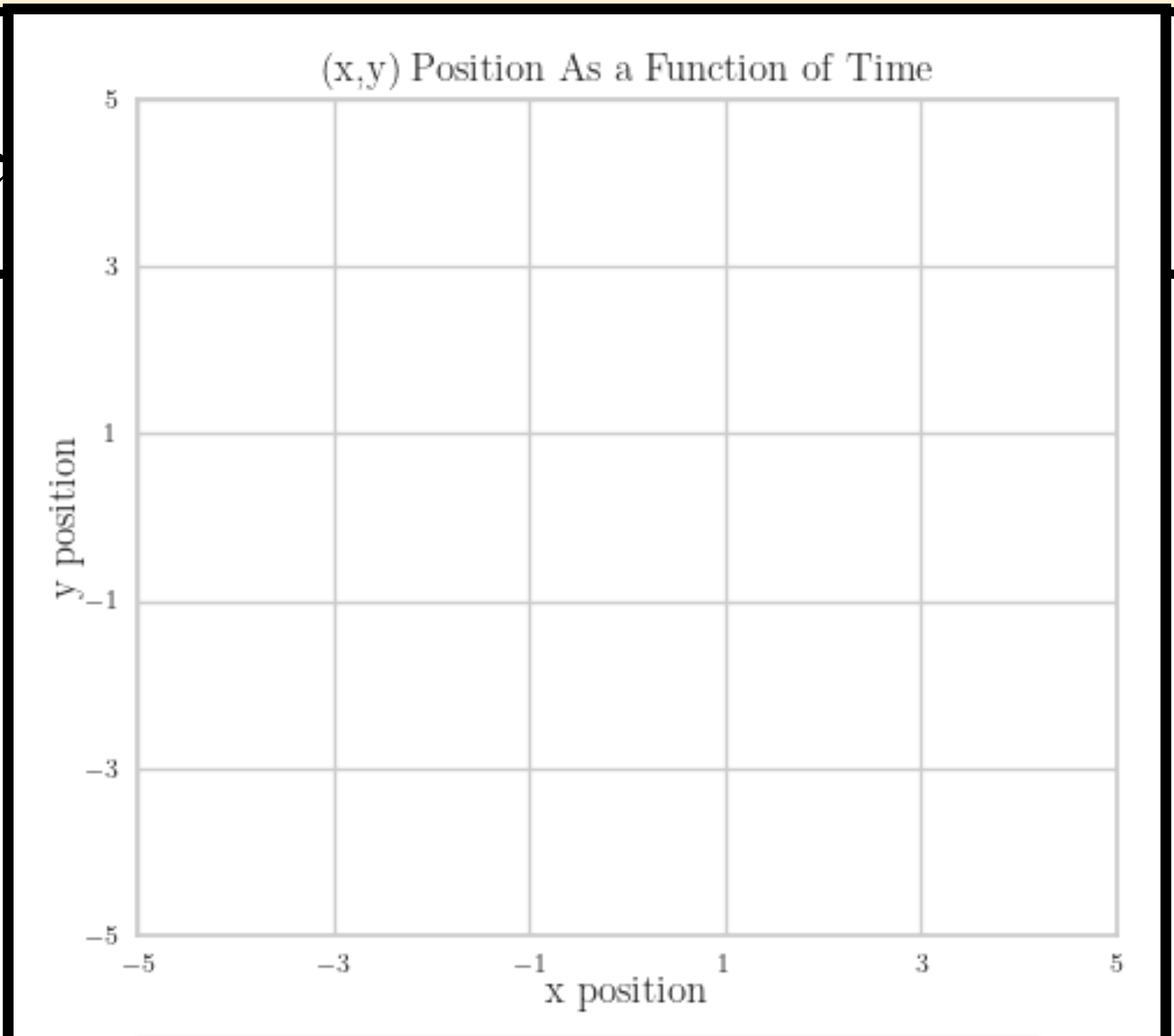
Proof: By updating the policy rarely, you leverage **passive stability of dynamics.**



What We Get from Action Chunking

Theorem (ZPMS): Given an open-loop stable system, there exists a fixed **k** such that (independent of data amount **n**), s.t. **k**-action chunking gives

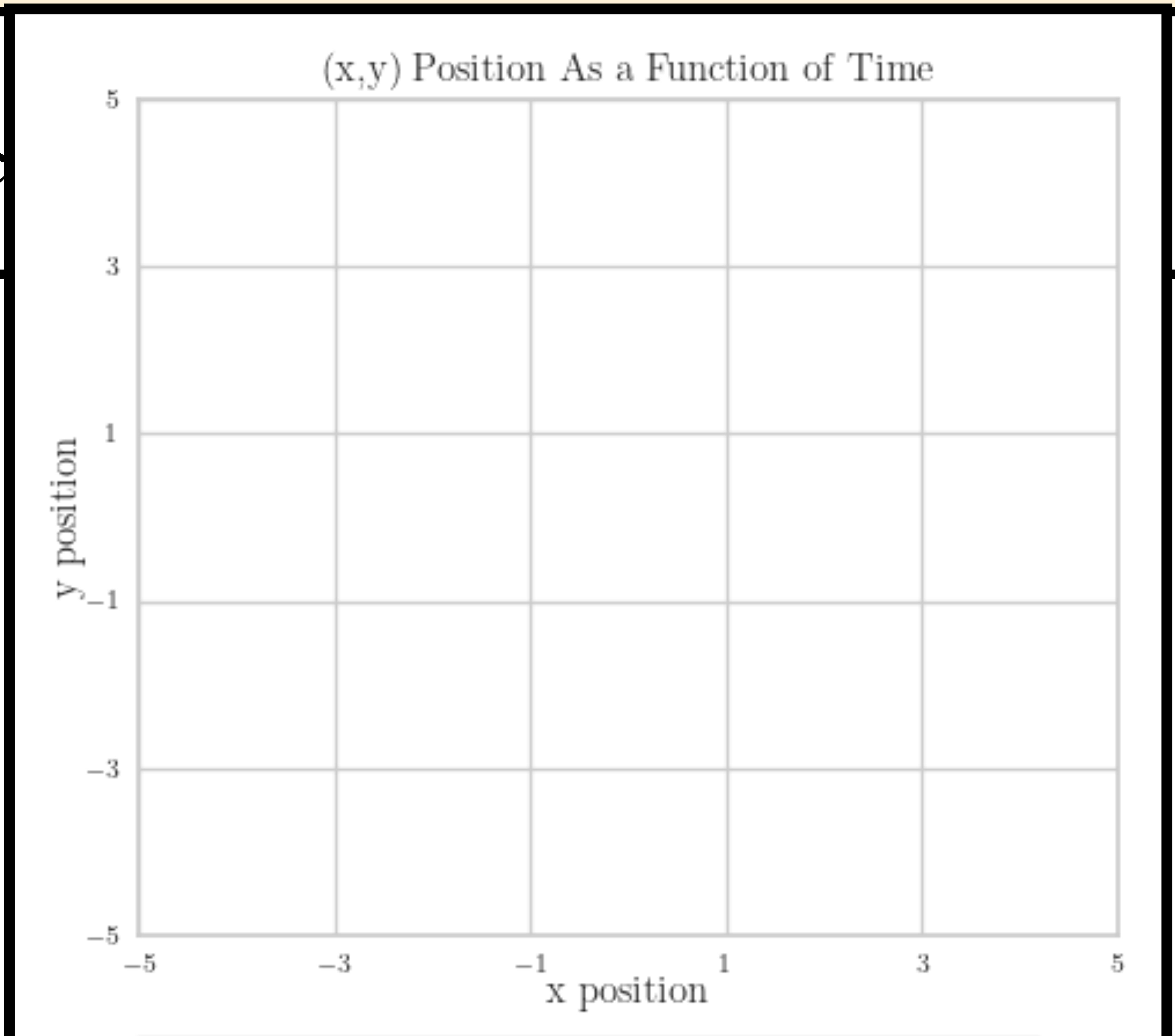
$\mathscr{R}_C$



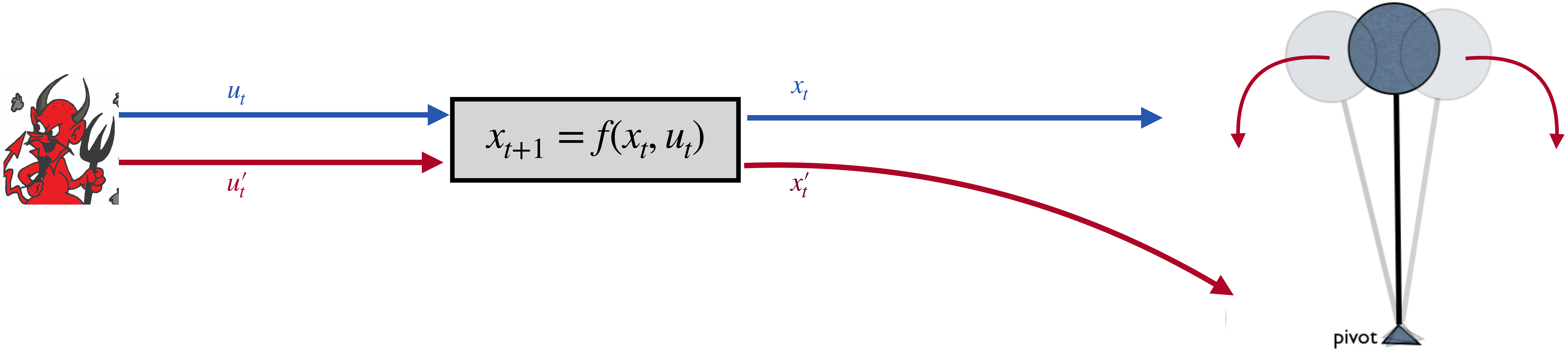
What We Get from Action Chunking

Theorem (ZPMS): Given an open-loop stable system, there exists a fixed **k** such that (independent of data amount **n**), s.t. **k**-action chunking gives

$\mathcal{R}_{\mathrm{c}}$



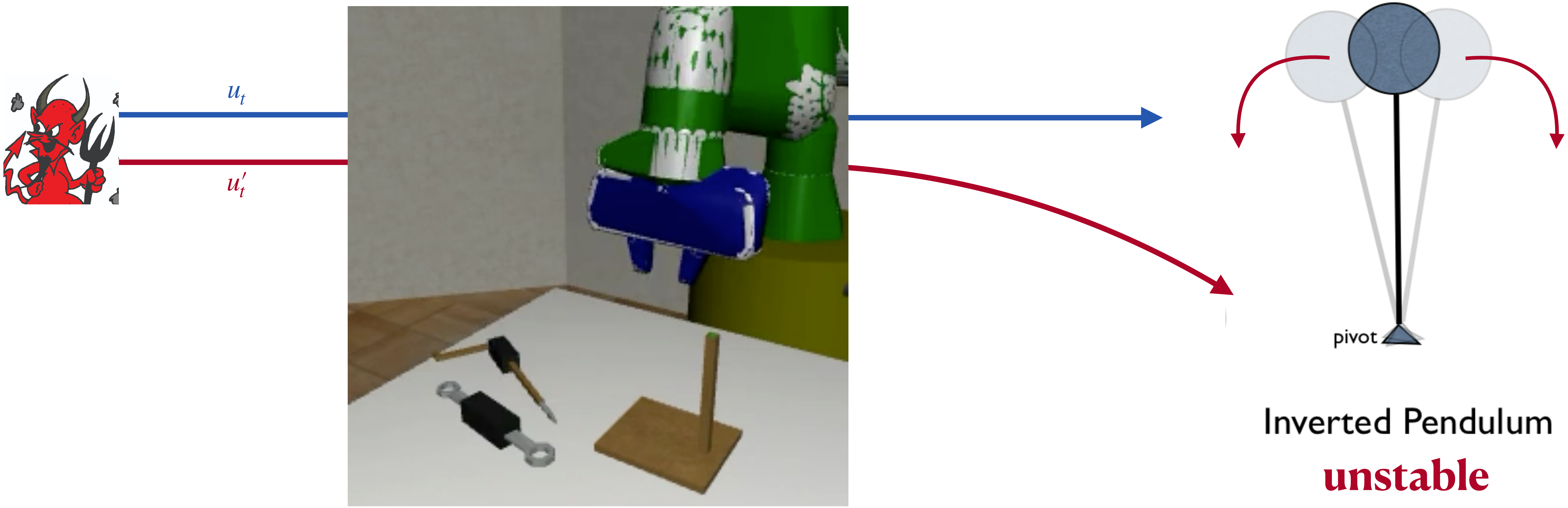
But what about unstable dynamics?



Inverted Pendulum

unstable

But what about unstable dynamics?

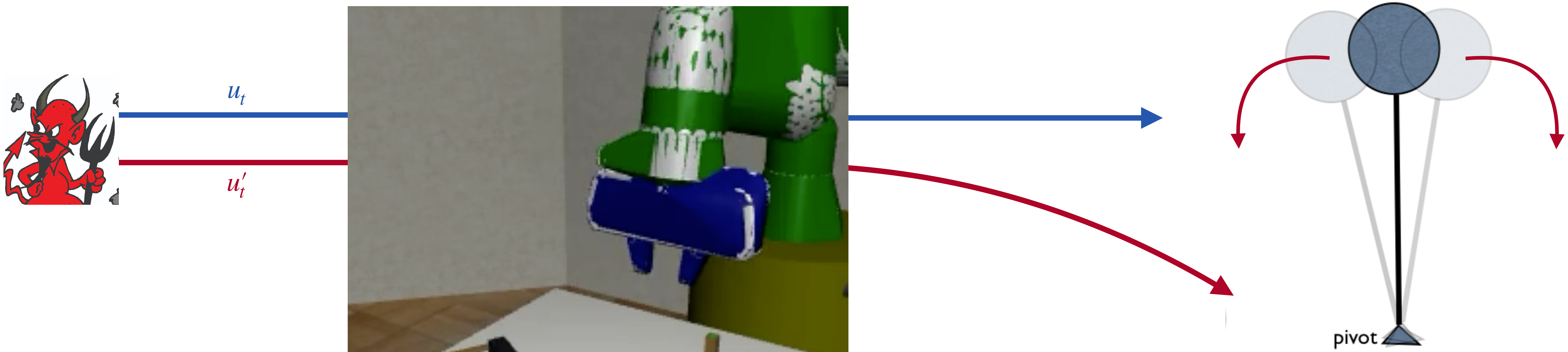


“critical moment”

Inverted Pendulum

unstable

But what about unstable dynamics?



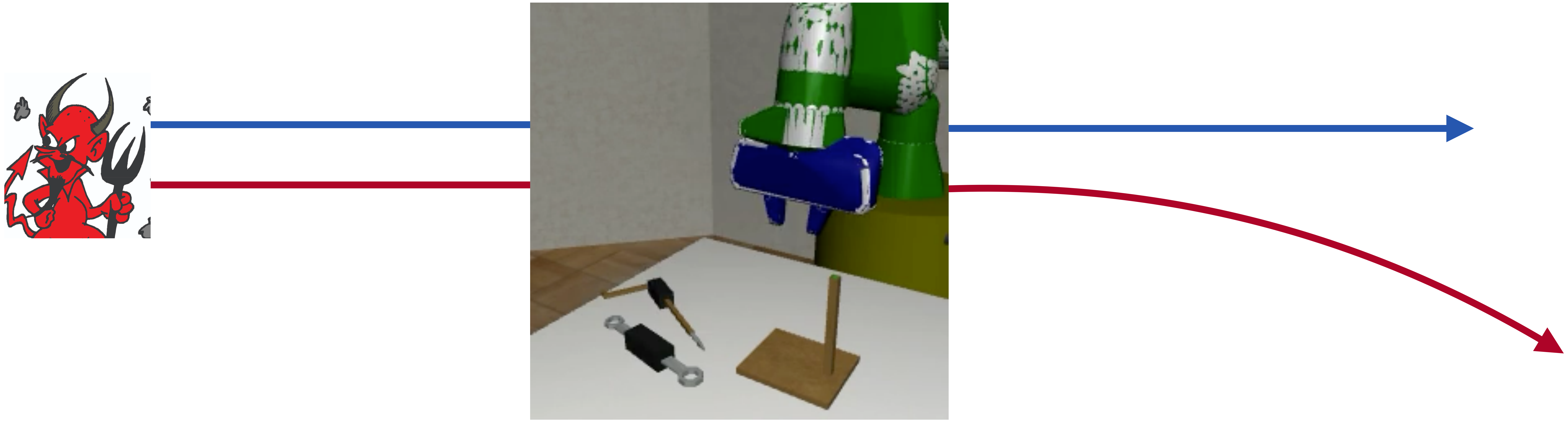
Inverted Pendulum

unstable

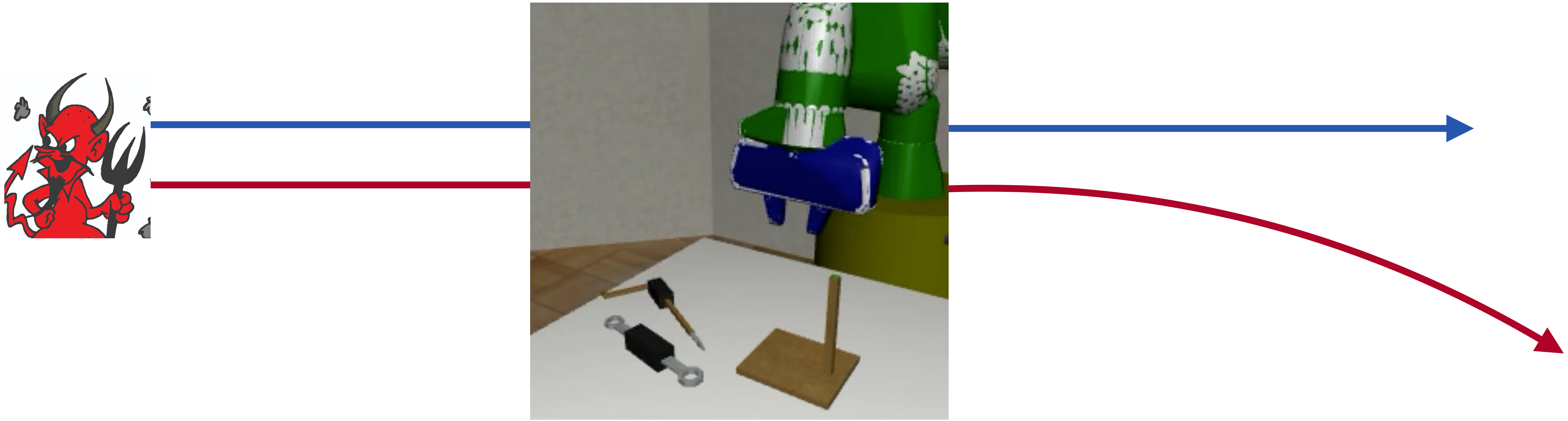
"critical moment"

Theorem (SPJ): Given **only** expert demonstration **data, no algorithm** (no matter how clever!) can imitate without **exponential compounding error.**

The power of data augmentation.

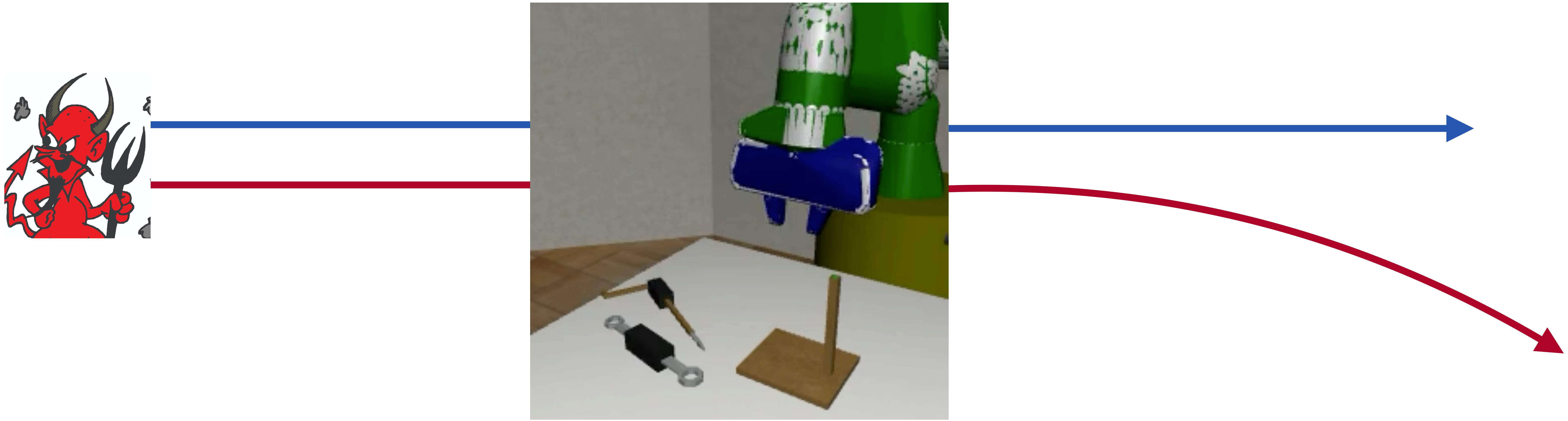


The power of data augmentation.



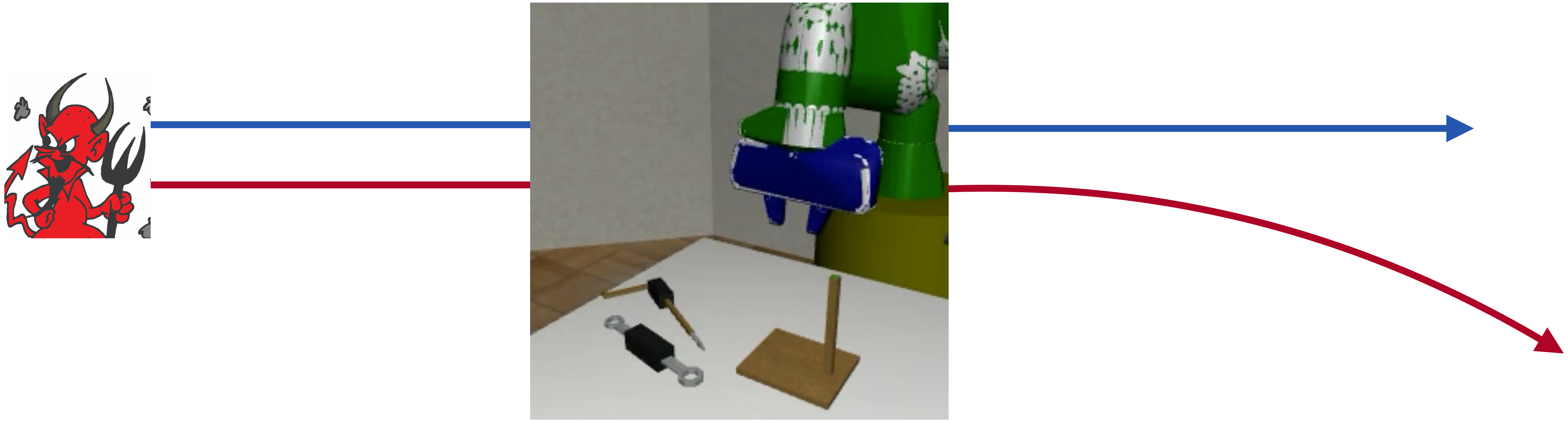
The power of data augmentation.

Theorem: If the expert policies collects trajectories as $u = \pi^\star(x) + \text{noise}$, but provides $(u^{\text{clean}}, x) = (\pi^\star(x), u)$ as training data, we can efficiently learning in **unstable dynamics.**



The power of data augmentation.

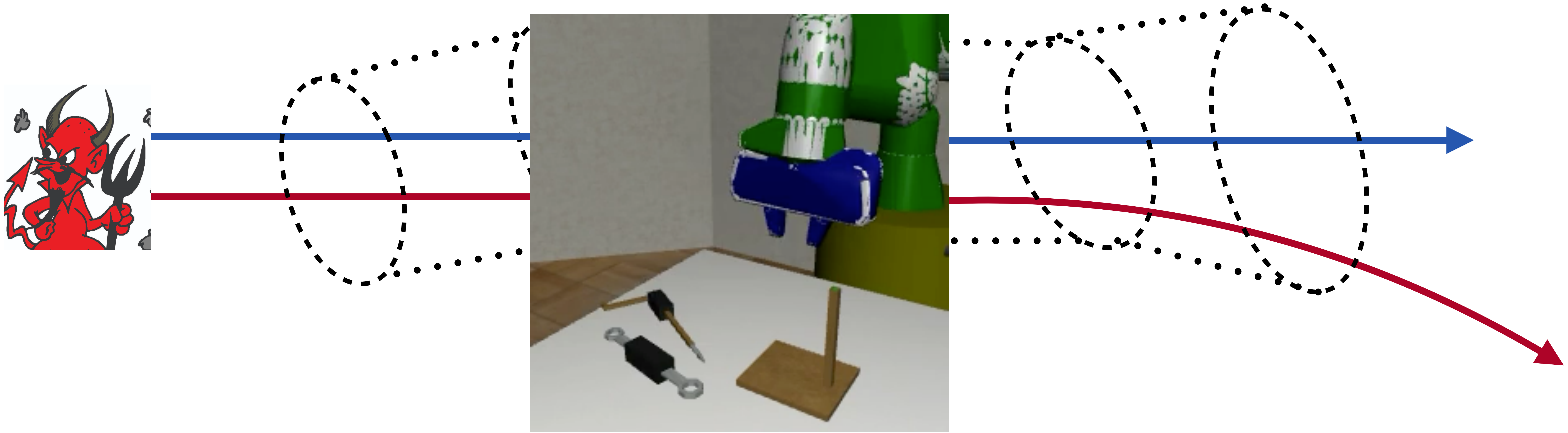
Theorem: If the expert policies collects trajectories as $u = \pi^\star(x) + \text{noise}$, but provides $(u^{\text{clean}}, x) = (\pi^\star(x), u)$ as training data, we can efficiently learning in **unstable dynamics.**



*DART algorithm, Laskey et. 2017

The power of data augmentation.

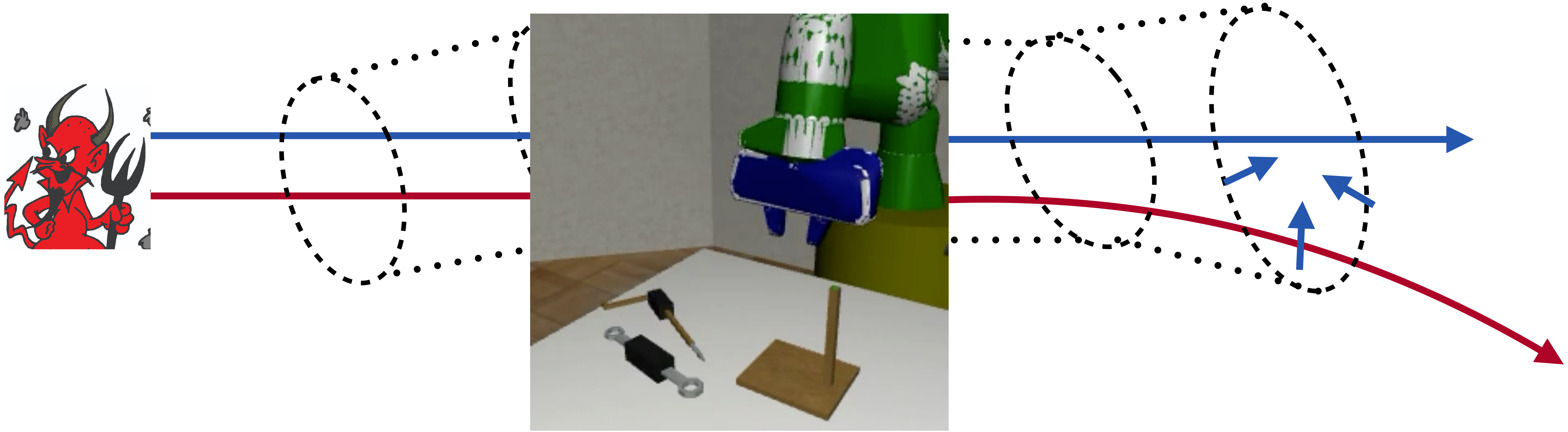
Theorem: If the expert policies collects trajectories as $u = \pi^\star(x) + \text{noise}$, but provides $(u^{\text{clean}}, x) = (\pi^\star(x), u)$ as training data, we can efficiently learning in **unstable dynamics.**



*DART algorithm, Laskey et. 2017

The power of data augmentation.

Theorem: If the expert policies collects trajectories as $u = \pi^\star(x) + \text{noise}$, but provides $(u^{\text{clean}}, x) = (\pi^\star(x), u)$ as training data, we can efficiently learning in **unstable dynamics.**



*DART algorithm, Laskey et. 2017

In summary

In summary

Unlike language pertaining, naive imitation does not work.
However, **better policy representation** + **better data** can overcome the challenges of **physical world learning** 🤖**.**

In summary

Unlike language pertaining, naive imitation does not work. However, **better policy representation** + **better data** can overcome the challenges of **physical world learning** 🤖**.**

Many pathologies in the Physical World 🤖 come from incomplete knowledge of system dynamics.

Conclusion: Where next for Physical AI?

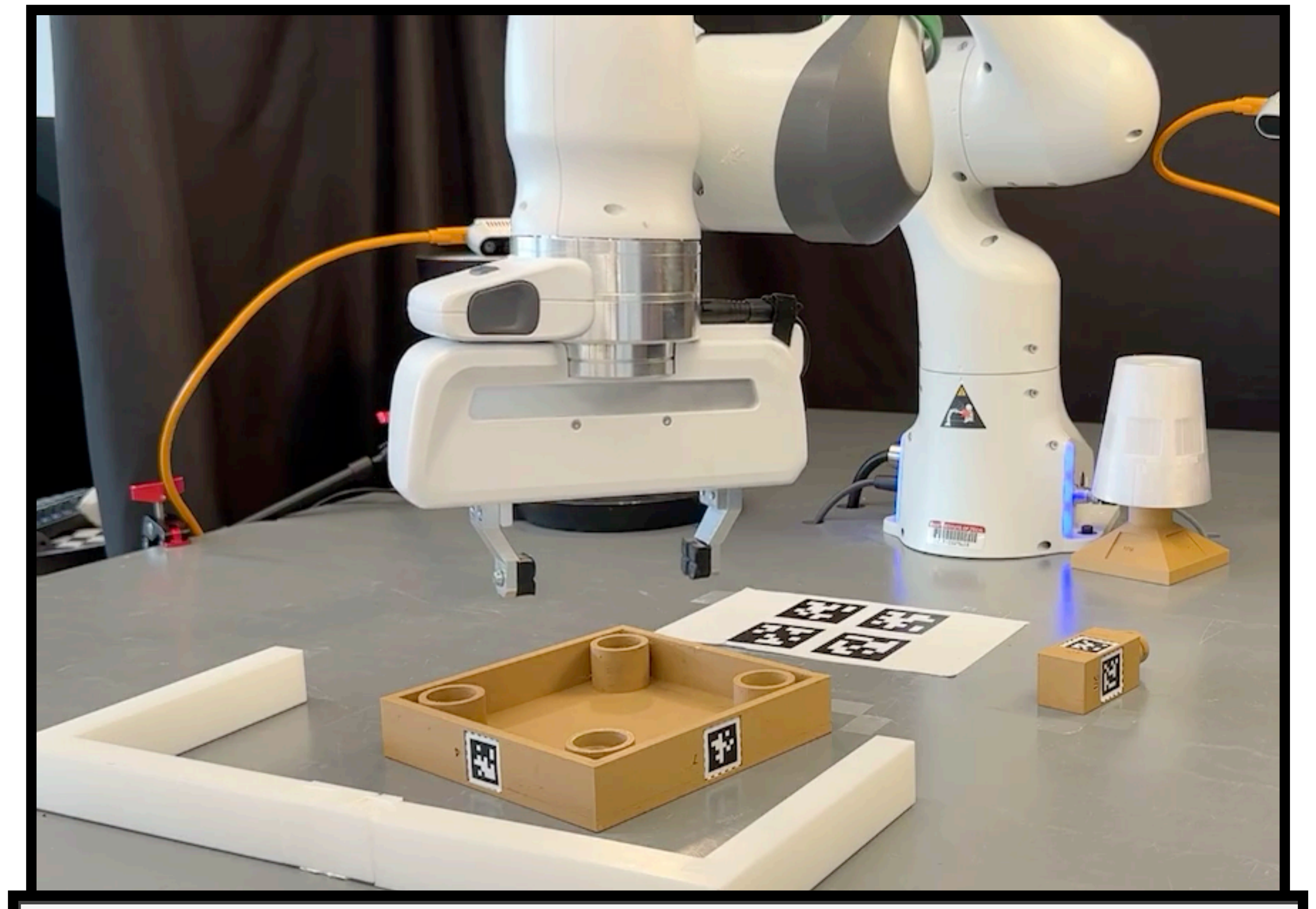
Exploration + World Modeling

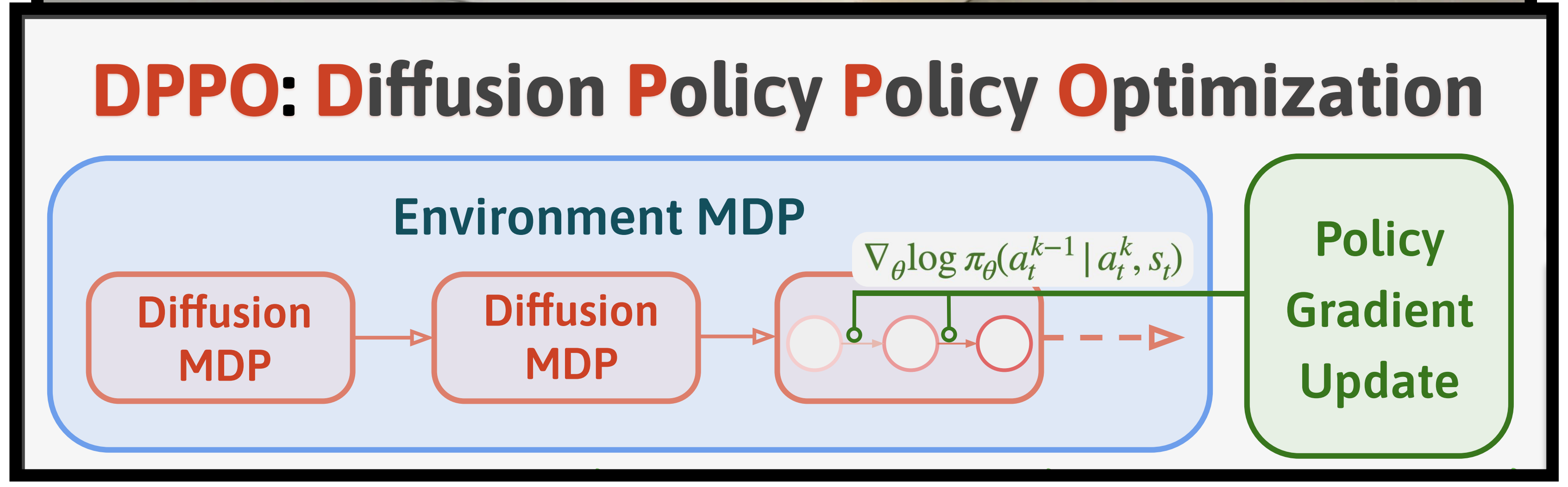
Exploration + World Modeling

Many pathologies in the Physical World 🤖 come from incomplete knowledge of system dynamics.

Exploration + World Modeling

Many pathologies in the Physical World 🤖 come from incomplete knowledge of system dynamics.

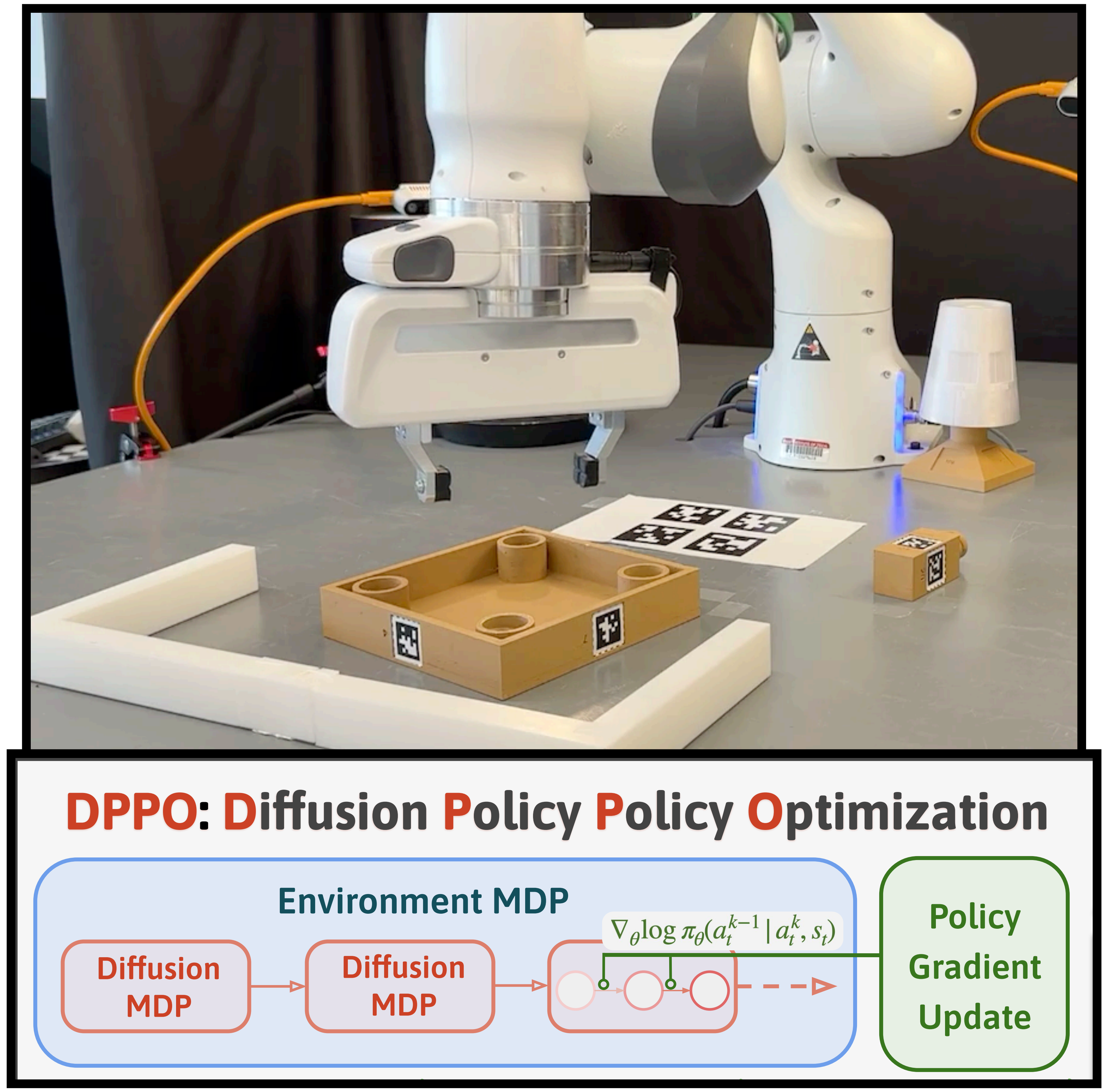




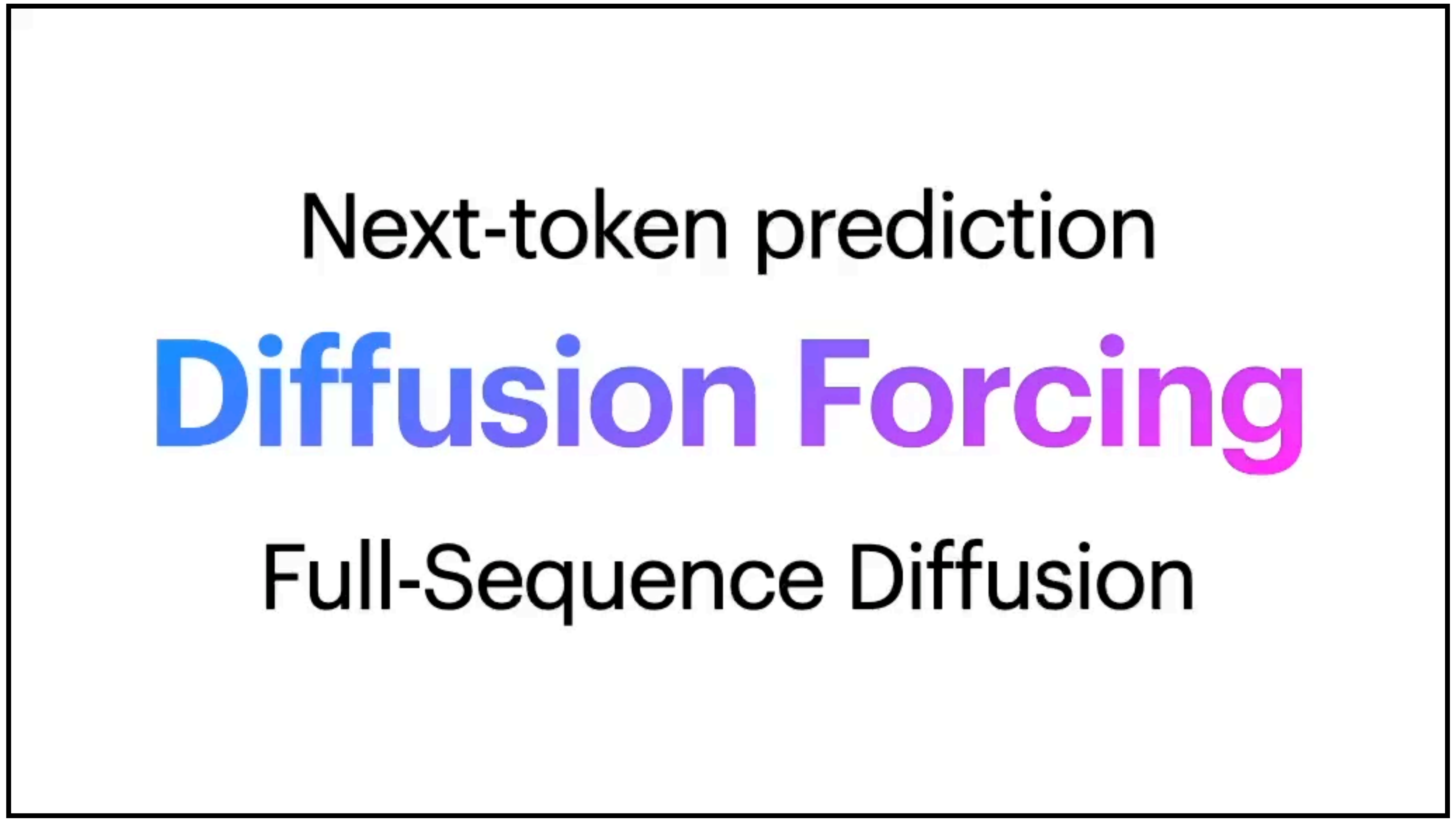
Allen Ren et al. '24

Exploration + World Modeling

Many pathologies in the Physical World 🤖 come from incomplete knowledge of system dynamics.

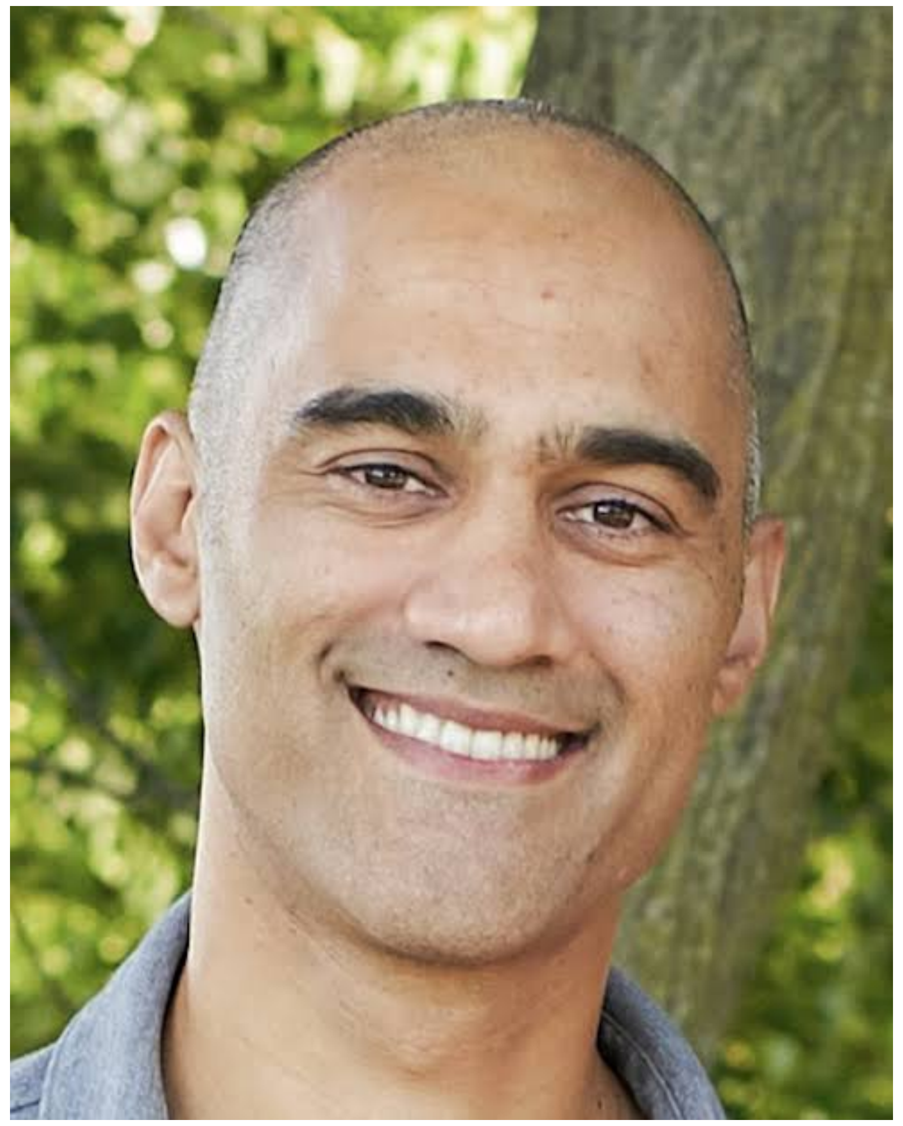


Allen Ren et al. '24

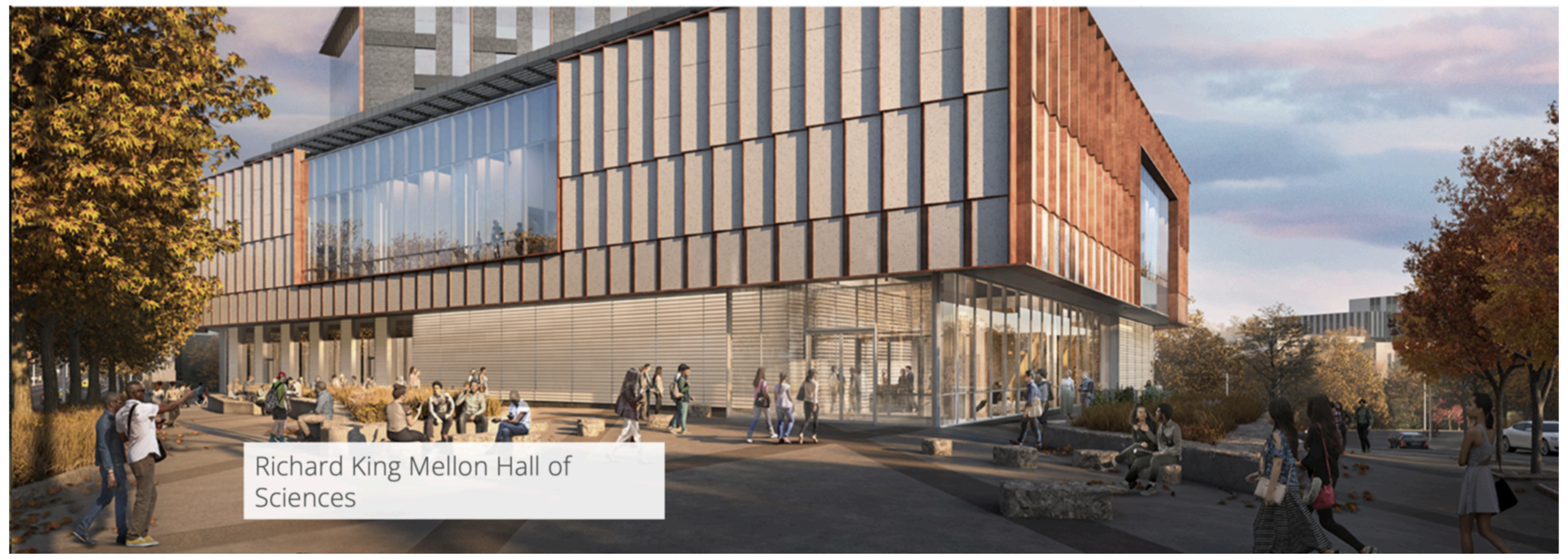


Boyuan Chen et al '24

Generative Engineering, Mathematics, Science (💎s)



Ameet Talwalkar **Nick Boffi** **Andrej Risteski**



@CMU