Multi-Task Learning for Control: From LTI to JEPA

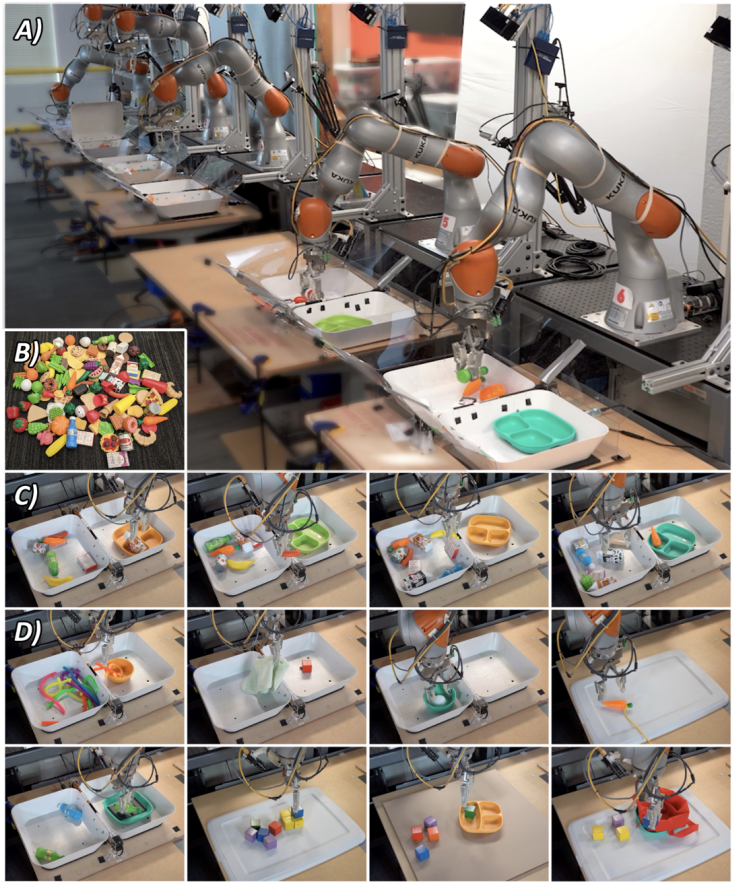
James Anderson

Department of Electrical Engineering
Columbia University

UCSD: Center for Control Systems and Dynamics

March 13th, 2026

Multi-task Control



Images: [Kalashnikov et al., 2021], [Boston Dynamics]

Why Multi-Task Learning

1. **Inductive bias and generalization**

   Single-task learning can overfit to spurious correlations

   Multi-task learning forces the model to find features that are:

   - useful across all tasks
   - more invariant/robust

   providing out-of-distribution robustness and transfer performance

Why Multi-Task Learning

1. **Inductive bias and generalization**

   Single-task learning can overfit to spurious correlations

   Multi-task learning forces the model to find features that are:

   - useful across all tasks
   - more invariant/robust

   providing out-of-distribution robustness and transfer performance

2. **Computational and deployment efficiency**

   Instead of deploying 10 models, deploy 1 backbone & 10 small task heads

   - lower memory
   - faster inference
   - easier maintenance particularly useful for edge deployment

Why Multi-Task Learning

1. **Inductive bias and generalization**

   Single-task learning can overfit to spurious correlations

   Multi-task learning forces the model to find features that are:

   - useful across all tasks
   - more invariant/robust

   providing out-of-distribution robustness and transfer performance

2. **Computational and deployment efficiency**

   Instead of deploying 10 models, deploy 1 backbone & 10 small task heads

   - lower memory
   - faster inference
   - easier maintenance particularly useful for edge deployment

3. **Statistical bootstrapping:**

   If: Task A has little data and Task B is related and has lots of data

   Then: MTL allows Task A to borrow statistical strength from Task B

Outline

- **Multi-Task LQR**
  - policy gradient bounds
  - defining heterogeneity the "right way"

- **Planning and Control with a World Model**
  - world models and JEPAs
  - planning and control

Published in Transactions on Machine Learning Research (02/2026)

Model-Free Learning with Heterogeneous Dynamical Systems: A Federated LQR Approach

Han Wang Leonardo F. Toso Aritra Mitra James Anderson

Linear Quadratic Control

System

consider the discrete-time dynamical system

$$x_{t+1} = Ax_t + Bu_t, \quad x_0 \sim \mathcal{D} \quad t = 0, 1, 2, \dots \quad \text{(dynamics)}$$

with

- state $x_t \in \mathbb{R}^n$, input $u_t \in \mathbb{R}^m$
- initial condition $\mathbb{E}x_0 = 0$, and $\mathbb{E}x_0 x_0^T \succeq \mu I$

Linear Quadratic Control

System

consider the discrete-time dynamical system

$$x_{t+1} = Ax_t + Bu_t, \quad x_0 \sim \mathcal{D} \quad t = 0, 1, 2, \ldots \quad \text{(dynamics)}$$

with

- state $x_t \in \mathbb{R}^n$, input $u_t \in \mathbb{R}^m$
- initial condition $\mathbb{E}x_0 = 0$, and $\mathbb{E}x_0 x_0^T \succeq \mu I$

Objective

design a static linear control policy $u_t = -Kx_t$ such that:

$$K \in \mathcal{K} \triangleq \{K \mid \rho(A - BK) < 1\} \quad \text{(stability)} \qquad \texttt{// non-convex}$$

that minimizes

$$C(K) \triangleq \mathbb{E}_{x_0 \sim \mathcal{D}} \left[\sum_{t=0}^{\infty} \left(x_t^T Q x_t + u_t^\top R u_t \right) \right] \quad \text{s.t. (dynamics)+(stability)}$$

LQR via Policy Gradient

Policy Iteration

instead of solving an algebraic Riccati equation, directly solve

$$\begin{aligned} \underset{K}{\text{minimize}} \quad & C(K) \\ \text{s.t.} \quad & \text{(dynamics)} + \text{(stability)} \end{aligned}$$

via gradient descent:

$$K \leftarrow K - \eta \nabla C(K)$$

Global Convergence

after $N = \mathcal{O}(\log(1/\epsilon))$ iterations, produces a controller that satisfies

$$C(K_N) - C(K^\star) \leq \epsilon.$$

[Fazel, Ke, Kakade & Mesbahi 2018]

Notes

Policy Gradient LQR

- neither feasible set nor objective are convex
- benign landscape and approximate smoothness $\longrightarrow$ global convergence
- model-free global convergence as well
- other descent algorithms converge

Policy Gradient LQG

- set of stabilizing controllers is disjoint:

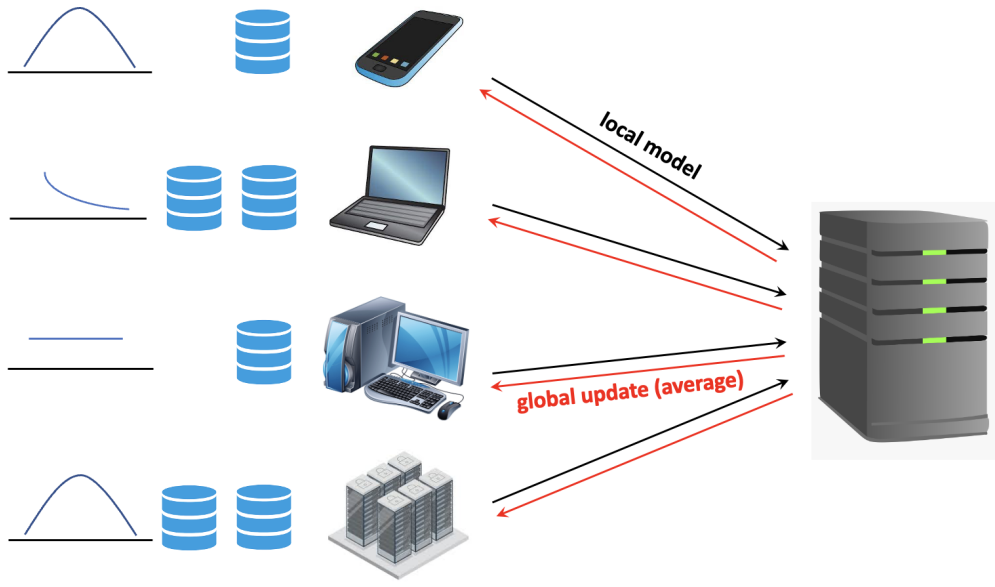
$$\mathcal{K}_{\text{dyn}} = \left\{ \left[\begin{array}{c|c} A_K & B_K \\ \hline C_K & \end{array}\right] \;:\; \text{stabilizes } (A, B, C) \right\}$$

- landscape is not gradient dominated [Zheng, Tang & Li 2021]
- can lift LQG problem to LQR with history [Fallah, Toso & Anderson 2025]

Federated Learning

a framework for distributed learning that accounts for:

- device and data **heterogeneity**
- data **locality** (privacy)
- communication efficiency



- `FedAvg` canonical federated algorithm [McMahan et al. 2016]

Multi-Task LQR

- given $i = 1, \ldots, M$ (stabilzable) LTI systems

$$x_{t+1}^{(i)} = A^{(i)} x_t^{(i)} + B^{(i)} u_t^{(i)}, \quad x_0^{(i)} \sim \mathcal{D}, \quad \text{(dynamics}_i\text{)}$$

Multi-Task LQR

- given $i = 1, \ldots, M$ (stabilzable) LTI systems

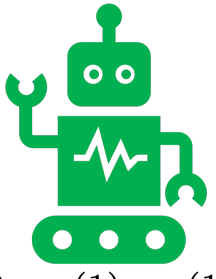
$$x_{t+1}^{(i)} = A^{(i)} x_t^{(i)} + B^{(i)} u_t^{(i)}, \quad x_0^{(i)} \sim \mathcal{D}, \quad \text{(dynamics}_i\text{)}$$

- construct a **common** state feedback controller, $u_t^{(i)} = K x_t^{(i)}$, that solves

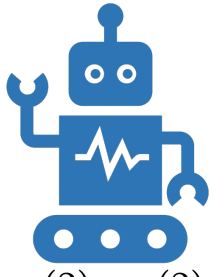
$$K^* = \underset{K}{\operatorname{argmin}} \left\{ C_{\text{avg}}(K) \triangleq \frac{1}{M} \sum_{i=1}^{M} \overbrace{\mathbb{E}\left[\sum_{t=0}^{\infty} x_t^{(i)\top} Q x_t^{(i)} + u_t^{(i)\top} R u_t^{(i)} \right]}^{C^{(i)}(K)} \right\}$$

s.t. $\{\text{(dynamics}_i\text{)}\}_{i=1}^{M} + \{\text{(stability}_i\text{)}\}_{i=1}^{M}$

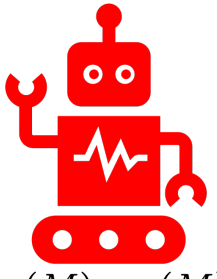
FedLQR



$(A^{(1)}, B^{(1)}, Q^{(1)}, R^{(1)})$

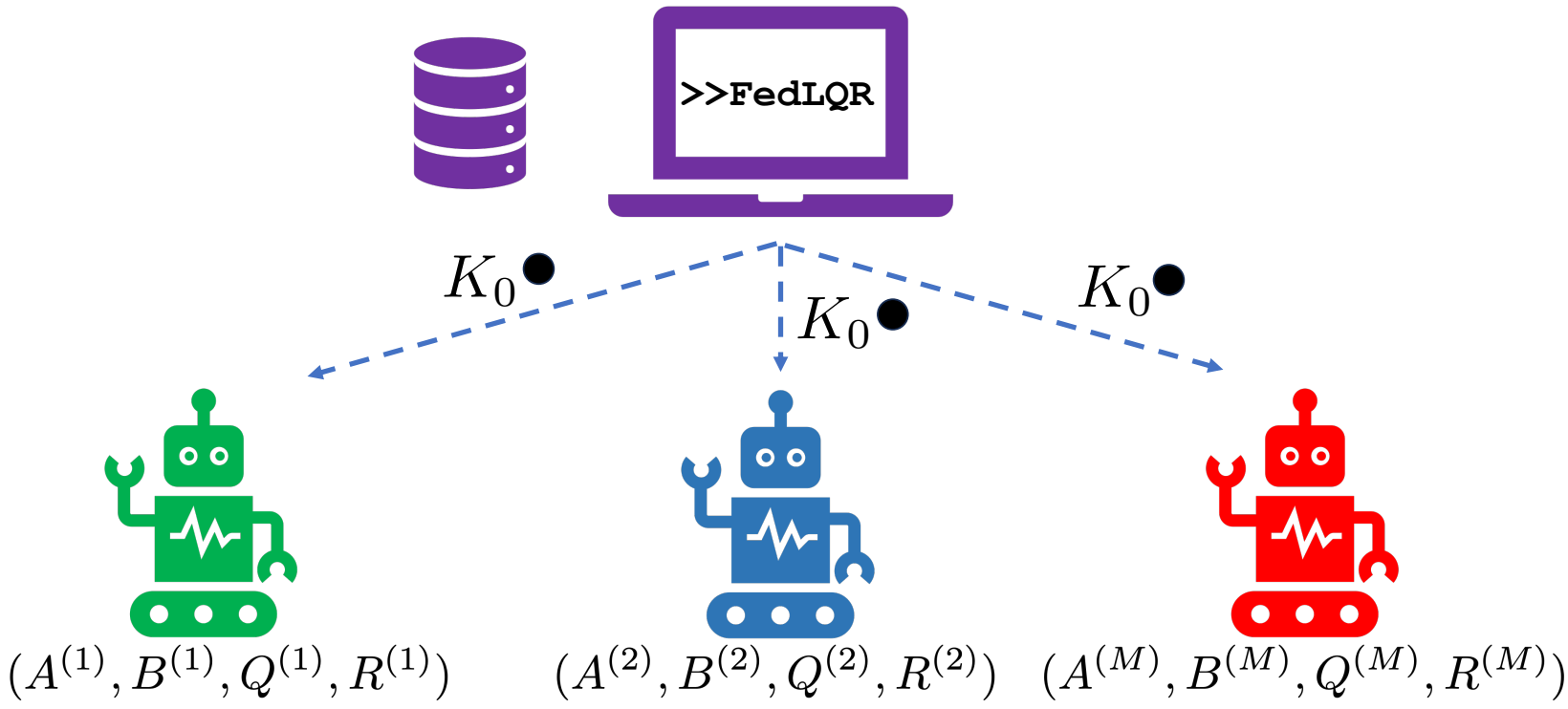


$(A^{(2)}, B^{(2)}, Q^{(2)}, R^{(2)})$

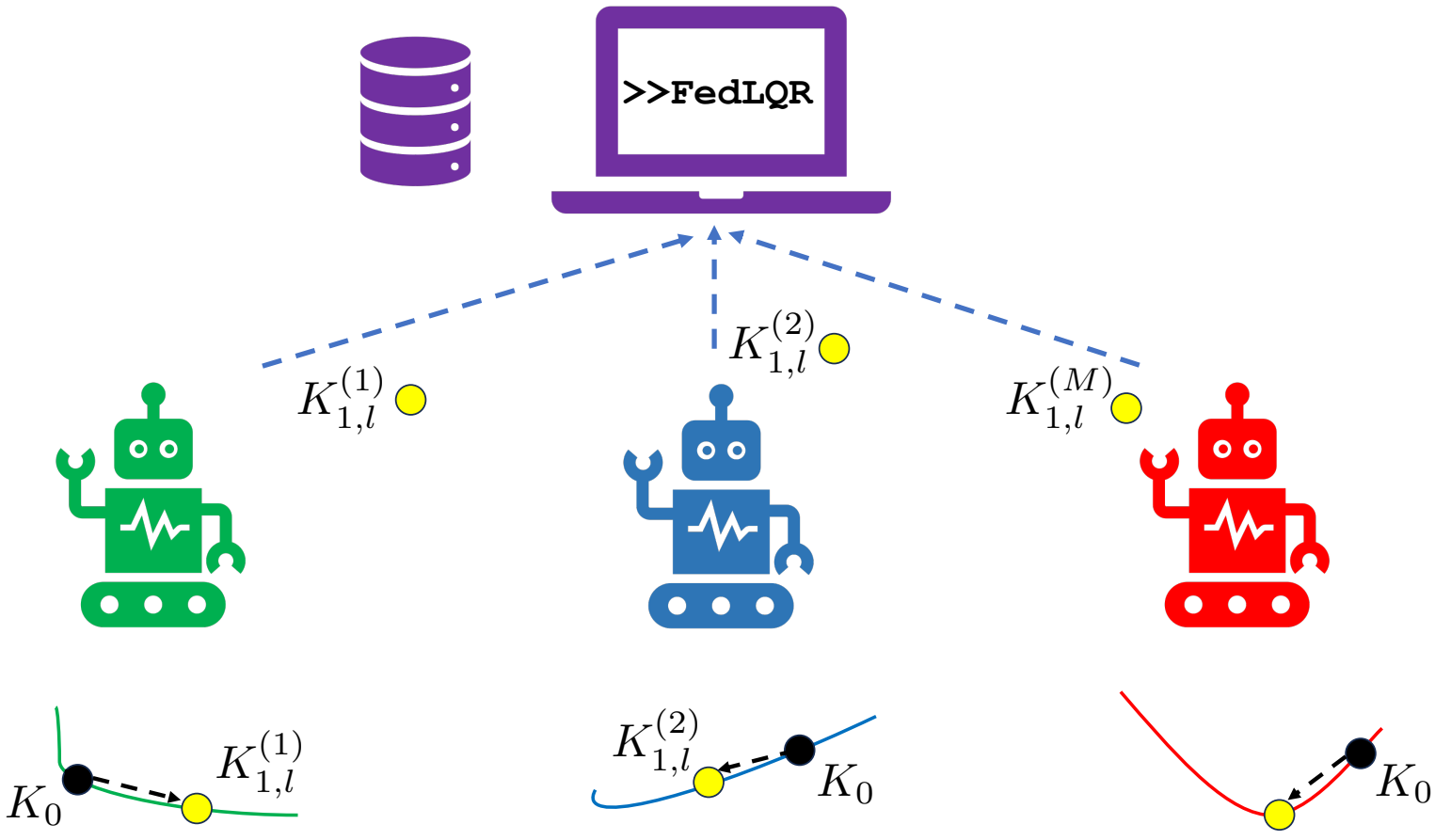


$(A^{(M)}, B^{(M)}, Q^{(M)}, R^{(M)})$

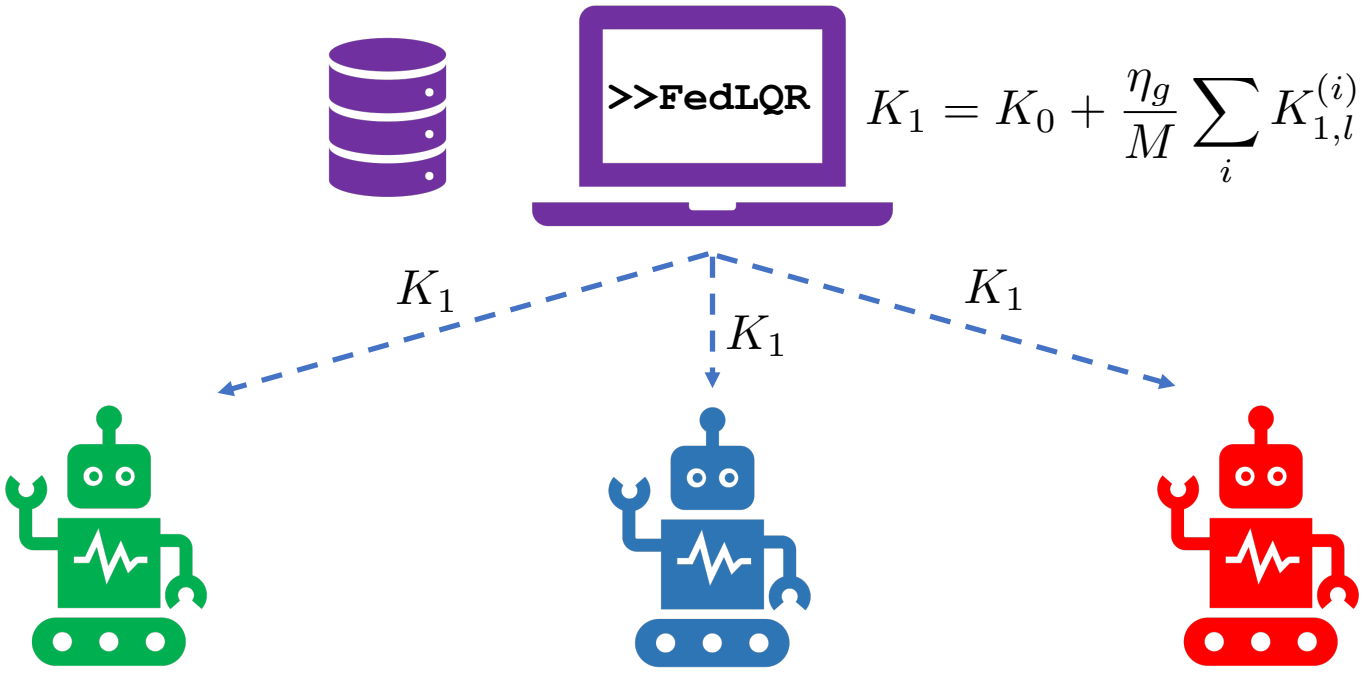
FedLQR



FedLQR



FedLQR



Questions & Challenges

1. **Is this common policy stabilizing for all the systems? If so, under what conditions?**
2. **How far is the learned common policy from each agent's locally optimal policy?**
3. **What is the (sample complexity) benefit to each participating agent?**
4. **Can the optimal controller be applied and fine-tuned on unseen systems? [meta-learning]**

Quantifying Heterogeneity: First Attempt

given tasks $1, \ldots, M$ defined by $\mathcal{T}^{(i)} \triangleq (A^{(i)}, B^{(i)})$, assume there exist constants ϵ_A and ϵ_B such that

$$\max_{i,j} \|A^{(i)} - A^{(j)}\| \leq \epsilon_A, \quad \text{and} \quad \max_{i,j} \|B^{(i)} - B^{(j)}\| \leq \epsilon_B, \quad \text{for all } i, j$$

- quantifies how similar a set of systems are
- don't need to know ϵ_A, ϵ_B
- ϵ_Q, ϵ_R defined in the same way if needed

How do ϵ_A, ϵ_B impact performance and convergence?

Heterogeneity Bound is Necessary

Scalar Example

consider a simple 2 system setting, with

$$x_{t+1}^{(1)} = \alpha x_t^{(1)} + u_t, \qquad x_{t+1}^{(2)} = -\alpha x_t^{(2)} + u_t$$

and controller $u_t^{(i)} = K x_t^{(i)}$ for $i = 1, 2$

Heterogeneity Bound is Necessary

Scalar Example

consider a simple 2 system setting, with

$$x_{t+1}^{(1)} = \alpha x_t^{(1)} + u_t, \qquad x_{t+1}^{(2)} = -\alpha x_t^{(2)} + u_t$$

and controller $u_t^{(i)} = K x_t^{(i)}$ for $i = 1, 2$

- $\epsilon_A = 2\alpha$ and $\epsilon_B = 0$ `// exact measures`
- $\epsilon_A > 2 \implies \alpha > 1 \implies$ both systems unstable
- for stability require $|\alpha - K| < 1$ **and** $|\alpha + K| < 1$ `// impossible`

Takeaway

need to impose some bound on the degree of heterogeneity

Do ϵ_A, ϵ_B capture what we need?

recall our definition:

$$\max_{i,j} \|A^{(i)} - A^{(j)}\| \leq \epsilon_A, \quad \text{and} \quad \max_{i,j} \|B^{(i)} - B^{(j)}\| \leq \epsilon_B, \quad \text{for all } i, j$$

are these systems really similar?

$$x_{t+1}^{(1)} = 0.99 x_t^{(1)} + 0.1 u_t \quad \text{and} \quad x_{t+1}^{(2)} = 1.01 x_t^{(2)} + 0.01 u_t$$

Issues

- fails to capture stability/dynamics ✗
- ...and system theoretic properties ✗

LQR Gradient

consider the single client setting, i.e., $M = 1$, with cost:

$$C(K) \triangleq \mathbb{E}_{x_0 \sim \mathcal{D}} \left[\sum_{t=0}^{\infty} \left(x_t^T Q x_t + u_t^\top R u_t \right) \right], \quad u_t = -K x_t$$

the **policy gradient** is

$$\nabla C(K) = 2 E_K \Sigma_K = 2 \underbrace{\left((R + B^T P_K B) K - B^T P_K A \right)}_{E_K} \Sigma_K$$

where

- $\Sigma_K \triangleq \mathbb{E}_{x_0 \sim \mathcal{D}} [\sum_{t=0}^{\infty} x_t x_t^T]$, $x_{t+1} = (A - BK) x_t$ `// cl-covariance`
- $P_K \succ 0$ solves the Lyapunov equation

$$P_K = Q + K^T R K + (A - BK)^T P_K (A - BK)$$

Performance Guarantees: Bounded Policy Gradients

`FedLQR` produces a controller K_n via the iterates:

$$K_{n+1} = \overbrace{K_n - \frac{\eta}{ML} \sum_{i=1}^{M} \sum_{l=0}^{L-1} \underbrace{\nabla C(K_{n,l}^{(i)})}_{\text{locally computed}}}^{\text{global averaging}}, \quad \eta \triangleq L\eta_l\eta_g$$

Performance Guarantees: Bounded Policy Gradients

`FedLQR` produces a controller K_n via the iterates:

$$K_{n+1} = \overbrace{K_n - \frac{\eta}{ML}\sum_{i=1}^{M}\sum_{l=0}^{L-1}\underbrace{\nabla C(K_{n,l}^{(i)})}_{\text{locally computed}}}^{\text{global averaging}}, \quad \eta \triangleq L\eta_l\eta_g$$

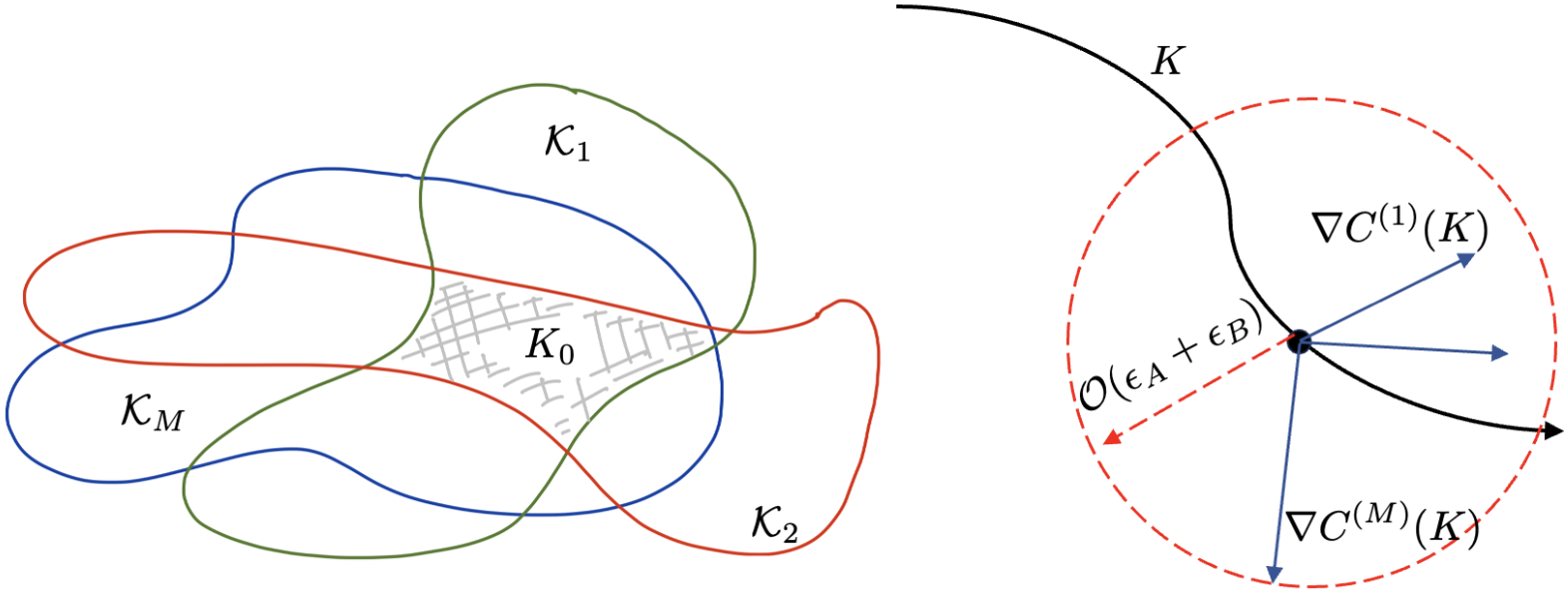
- if ϵ_A, ϵ_B are small then their policy gradient directions should be close
- for any i, j, we have

  $$g_{ij}(K) \triangleq \|\nabla C^{(i)}(K) - \nabla C^{(j)}(K)\| \leq \underbrace{\epsilon_A h_1(K) + \epsilon_B h_2(K)}_{\mathcal{O}(\epsilon_A+\epsilon_B)} \triangleq f_{\text{het}}(\epsilon_A, \epsilon_B)$$

  where h_1, h_2 are bounded polynomials of the problem data
- gradient of agent i can be approximated by gradient of agent j

Bounded Policy Gradients

$$\|\nabla C^{(i)}(K) - \nabla C^{(j)}(K)\| \leq \underbrace{\epsilon_A h_1(K) + \epsilon_B h_2(K)}_{\mathcal{O}(\epsilon_A + \epsilon_B)}$$



- initial stabilizing controller [Fujimani, Lee, Matni & Pappas 2024]

Performance Guarantees: Convergence

Distance between K_N and $K_\star^{(i)}$: [informal]

For each agent, after N rounds, if $\underbrace{(\epsilon_A g_1 + \epsilon_B g_2)^2 < g_3}_{\text{low heterogeneityregime}}$, then

$$C^{(i)}(K_N) - C^{(i)}(K_\star^{(i)}) \leq \underbrace{\left(1 - \eta\mu^2 C_1\right)^N}_{<1} \underbrace{\left(C^{(i)}(K_0) - C^{(i)}(K_\star^{(i)})\right)}_{\text{initial optimaility gap}} + \underbrace{C_u \mathcal{B}(\epsilon_A, \epsilon_B)}_{\text{bias}}$$

moreover K_N is stabilizing for all N.

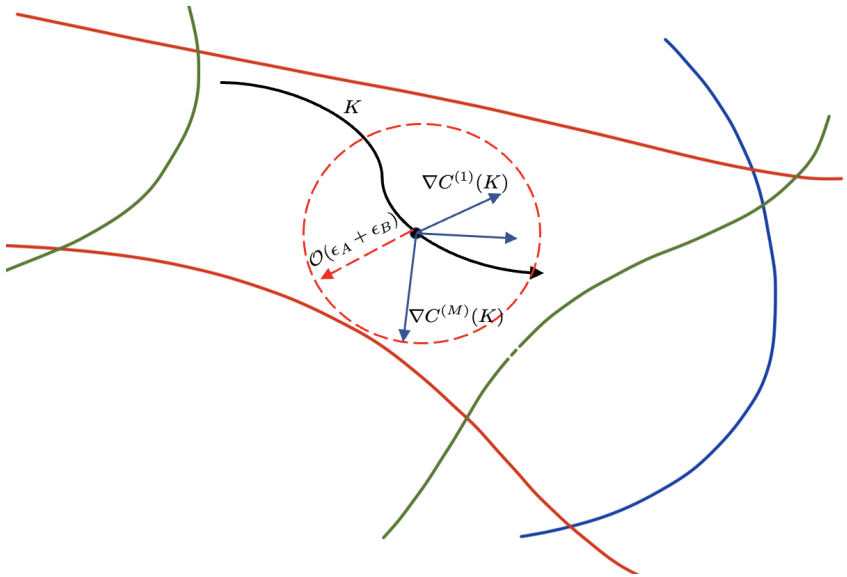
Performance Guarantees: Convergence

Distance between K_N and $K_\star^{(i)}$: [informal]

For each agent, after N rounds, if $\underbrace{(\epsilon_A g_1 + \epsilon_B g_2)^2 < g_3}_{\text{low heterogeneityregime}}$, then

$$C^{(i)}(K_N) - C^{(i)}(K_\star^{(i)}) \leq \underbrace{\left(1 - \eta\mu^2 C_1\right)^N}_{<1} \underbrace{\left(C^{(i)}(K_0) - C^{(i)}(K_\star^{(i)})\right)}_{\text{initial optimaility gap}} + \underbrace{C_u \mathcal{B}(\epsilon_A, \epsilon_B)}_{\text{bias}}$$

moreover K_N is stabilizing for all N.



Performance Guarantees: Agent Optimality

Distance between K_N and $K_i^\star$: [informal]

For each agent, after N rounds, if $(\epsilon_A g_1 + \epsilon_B g_2)^2 < g_3$, then

$$C^{(i)}(K_N) - C^{(i)}(K_i^\star) \leq \underbrace{\left(1 - \eta\mu^2 C_1\right)^N}_{<1} \underbrace{C^{(i)}(K_0) - C^{(i)}(K_i^{(\star)})}_{\text{initial gap}} + \underbrace{C_u \mathcal{B}(\epsilon_A, \epsilon_B)}_{\text{bias}}$$

moreover K_N is stabilizing for all N.

Distance between $K^\star$ and $K_\star^{(i)}$: [informal]

For all agents

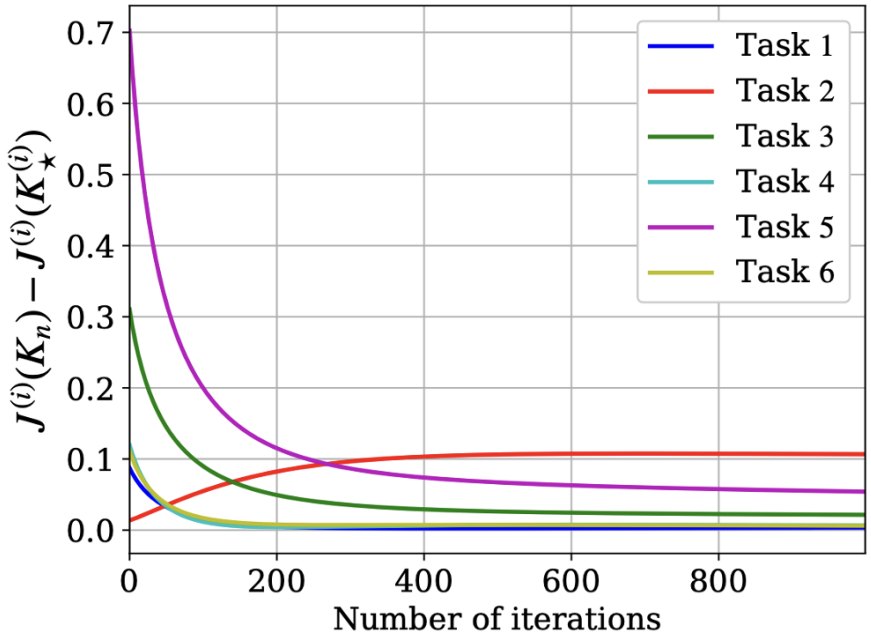
$$C^{(i)}(K^\star) - C^{(i)}(K_\star^{(i)}) = \mathcal{O}((\epsilon_A + \epsilon_B)^2).$$

Model-Free results: $\uparrow + 1/M$ sample complexity improvement
[Wang, Toso, Mitra, Anderson'26]

Observation

$$x_{t+1}^{(i)} = \begin{bmatrix} 1 & \mathrm{d}t \\ \frac{g}{l_i}\mathrm{d}t & 1 \end{bmatrix} x_t^{(i)} + \begin{bmatrix} 0 \\ \frac{\mathrm{d}t}{m_i l_i^2} \end{bmatrix} u_t^{(i)}, \quad Q = q_i I, \quad R = r_i$$

$$l_i, m_i, q_i, r_i \sim \mathcal{U}([0.5, 1] \times [0.1, 0.5] \times [0.1, 0.5] \times [0.01, 0.05])$$



// optimality gap

- $f_{\mathrm{het}}(\epsilon_A, \epsilon_B) \approx 2.3 \times 10^6$ at convergence ✗

Closed-Loop Systems

Lifted Systems

consider task $\mathcal{T}^{(i)} \triangleq (A^{(i)}, B^{(i)}, Q^{(i)}, R^{(i)})$ and any $K \in \mathcal{K}_{\text{stab}}$, define

$$S_K^{(i)} \triangleq \begin{cases} \Sigma_{K,t+1}^{(i)} & = A_K^{(i)} \Sigma_{K,t}^{(i)} A_K^{(i)T} + \Sigma_0^{(i)} \quad \texttt{// covariance dynamics} \\ Y_{K,t} & = 2 E_K^{(i)} \Sigma_{K,t}^{(i)} \quad \texttt{// gradient} \end{cases}$$

where $A_K^{(i)} = A^{(i)} - B^{(i)} K$

Closed-Loop Systems

Lifted Systems

consider task $\mathcal{T}^{(i)} \triangleq (A^{(i)}, B^{(i)}, Q^{(i)}, R^{(i)})$ and any $K \in \mathcal{K}_{\text{stab}}$, define

$$S_K^{(i)} \triangleq \begin{cases} \Sigma_{K,t+1}^{(i)} & = A_K^{(i)} \Sigma_{K,t}^{(i)} A_K^{(i)T} + \Sigma_0^{(i)} \quad \text{// covariance dynamics} \\ Y_{K,t} & = 2E_K^{(i)} \Sigma_{K,t}^{(i)} \quad \text{// gradient} \end{cases}$$

where $A_K^{(i)} = A^{(i)} - B^{(i)} K$

Idea

it follows that

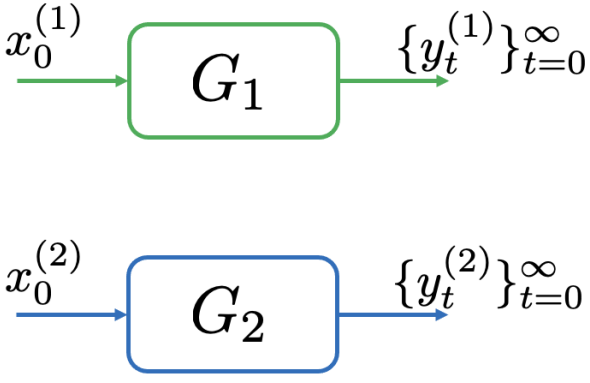
$$\|g_{ij}(K)\|_F \triangleq \|\nabla C^{(i)}(K) - \nabla C^{(j)}(K)\|_F = \lim_{t \to \infty} \|Y_{K,t}^{(i)} - Y_{K,t}^{(j)}\|_F \leq ???$$

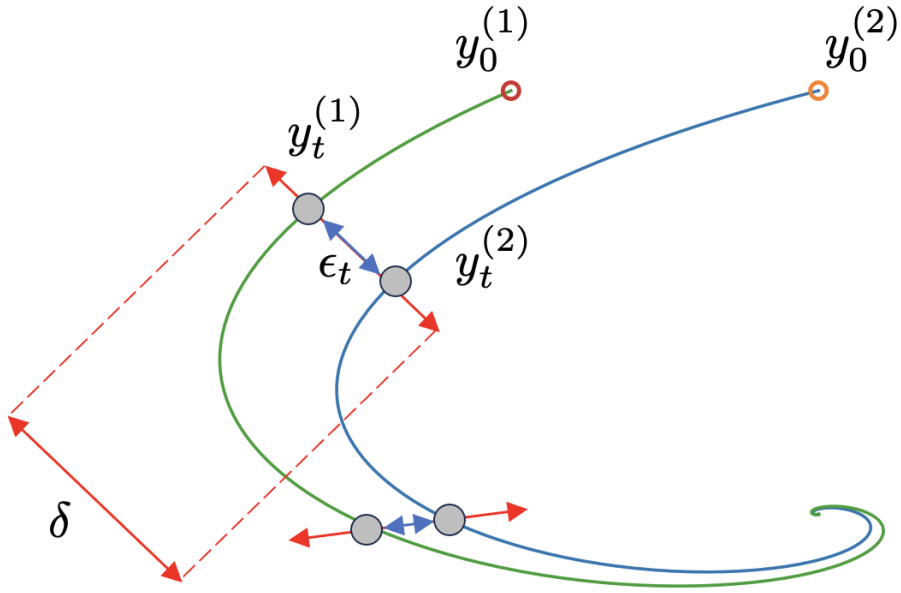
focus on bounding the difference between $S_K^{(i)}$ and $S_K^{(i)}$

Approximate Bisimulation

Stable autonomous systems

characterizes the difference between system traces





G_1 and G_2 are δ-bisimilar if $\underbrace{\|y_t^{(1)} - y_t^{(2)}\|}_{\epsilon_t} \leq \delta$ for all t

- all stable LTI systems are δ-bisimilar [Girard, Pappas, 2005]

Quantifying Heterogeneity: Second Attempt

Gradient-Space Bisimulation Function

A continuous function $\mathcal{V} : \mathbb{S}_+ \times \mathbb{S}_+ \to \mathbb{R}_+$ is a bisimulation function between $S_K^{(i)}$ and $S_K^{(j)}$ if there exits a $\lambda \in (0, 1)$ such that

1. $\underbrace{\mathcal{V}(\Sigma_{K,t}^{(i)}, \Sigma_{K,t}^{(j)})}_{\mathcal{V}_t} \geq \|E_K^{(i)}\Sigma_{K,t}^{(i)} - E_K^{(j)}\Sigma_{K,t}^{(j)}\|_F$
2. $\mathcal{V}_{t+1} - \mathcal{V}_t \leq -\lambda\mathcal{V}_t + \mathcal{V}(\Sigma_0^{(i)}, \Sigma_0^{(j)})$

for all $\left(\Sigma_{K,t}^{(i)}, \Sigma_{K,t}^{(j)}\right)$.

Quantifying Heterogeneity: Second Attempt

Gradient-Space Bisimulation Function

A continuous function $\mathcal{V} : \mathbb{S}_+ \times \mathbb{S}_+ \to \mathbb{R}_+$ is a bisimulation function between $S_K^{(i)}$ and $S_K^{(j)}$ if there exits a $\lambda \in (0, 1)$ such that

1. $\underbrace{\mathcal{V}(\Sigma_{K,t}^{(i)}, \Sigma_{K,t}^{(j)})}_{\mathcal{V}_t} \geq \|E_K^{(i)}\Sigma_{K,t}^{(i)} - E_K^{(j)}\Sigma_{K,t}^{(j)}\|_F$
2. $\mathcal{V}_{t+1} - \mathcal{V}_t \leq -\lambda\mathcal{V}_t + \mathcal{V}(\Sigma_0^{(i)}, \Sigma_0^{(j)})$

for all $\left(\Sigma_{K,t}^{(i)}, \Sigma_{K,t}^{(j)}\right)$.

Interpretation

- condition (1) $\implies$ $\mathcal{V}$ upper-bounds distance between $S_K^{(i)}$ and $S_K^{(j)}$
- condition (2) $\implies$

$$\limsup_{t\to\infty} \quad \mathcal{V}_t \leq \frac{\mathcal{V}(\Sigma_0^{(i)}, \Sigma_0^{(j)})}{\lambda}$$

Putting it Together

Completed Bound

$$\|\nabla C^{(i)}(K) - \nabla C^{(j)}(K)\|_F \quad = \quad \lim_{t\to\infty} \|Y_{K,t}^{(i)} - Y_{K,t}^{(j)}\|_F \quad \leq \quad \frac{\mathcal{V}(\Sigma_0^{(i)}, \Sigma_0^{(j)})}{\lambda}$$

Putting it Together

Completed Bound

$$\|\nabla C^{(i)}(K) - \nabla C^{(j)}(K)\|_F \quad = \quad \lim_{t\to\infty} \|Y^{(i)}_{K,t} - Y^{(j)}_{K,t}\|_F \quad \leq \quad \frac{\mathcal{V}(\Sigma^{(i)}_0, \Sigma^{(j)}_0)}{\lambda}$$

Explicit Bisimulation Function

$$\mathcal{V}(\Sigma^{(i)}_{K,t}, \Sigma^{(j)}_{K,t}) = \frac{\sqrt{2}\mathbf{tr}\left(M\mathbf{diag}(\Sigma^{(i)}_{K,t}, \Sigma^{(j)}_{K,t})\right)}{\sqrt{\underline{\sigma}(M)}}$$

where $M \succeq 0$ is obtained from an LMI

Putting it Together

Completed Bound

$$\|\nabla C^{(i)}(K) - \nabla C^{(j)}(K)\|_F \quad = \quad \lim_{t\to\infty} \|Y^{(i)}_{K,t} - Y^{(j)}_{K,t}\|_F \quad \leq \quad \frac{\mathcal{V}(\Sigma^{(i)}_0, \Sigma^{(j)}_0)}{\lambda}$$

Explicit Bisimulation Function

$$\mathcal{V}(\Sigma^{(i)}_{K,t}, \Sigma^{(j)}_{K,t}) = \frac{\sqrt{2}\mathbf{tr}\left(M\mathbf{diag}(\Sigma^{(i)}_{K,t}, \Sigma^{(j)}_{K,t})\right)}{\sqrt{\underline{\sigma}(M)}}$$

where $M \succeq 0$ is obtained from an LMI

Gradient Bound

$$g_{ij}(K) \;\leq\; \frac{\sqrt{2}\mathbf{tr}\left(M\mathbf{diag}(\Sigma^{(i)}_0, \Sigma^{(j)}_0)\right)}{\lambda\sqrt{\underline{\sigma}(M)}} \;\overset{\text{CVX}}{\simeq}\; \overbrace{\frac{\sqrt{2}\mathbf{tr}\left(M^{(ij)}_K\mathbf{diag}(\Sigma^{(i)}_0, \Sigma^{(j)}_0)\right)}{\lambda^{(ij)}_K\sqrt{\underline{\sigma}(M^{(ij)}_K)}}}^{b_{ij}(K)}$$

- $\overset{\text{CVX}}{\simeq}$ minimize via convex optimization

Quantifying Heterogeneity via Bisimulation

Given a controller K that stabilizes all systems:

- for every pair of tasks, we have a gradient bound

$$g_{ij}(K) \quad \leq \quad b_{ij}(K) \quad \triangleq \quad \frac{\sqrt{2}\mathbf{tr}\left(M_K^{(ij)}\mathbf{diag}(\Sigma_0^{(i)}, \Sigma_0^{(j)})\right)}{\lambda_K^{(ij)}\sqrt{\underline{\sigma}(M_K^{(ij)})}}$$

- for every task $i = 1, \ldots, M$, define

$$b_i(K) = \frac{1}{M}\sum_{j \neq i}^{M} b_{ij}$$

Performance Bound: General

Task specific optimality gap

error between optimal federated controller and locally optimal controller:

Performance Bound: General

Task specific optimality gap

error between optimal federated controller and locally optimal controller:

Let $K_\star \in \mathcal{K}_{\mathrm{stab}}$ minimize

$$C_{\mathrm{avg}}(K) \triangleq \frac{1}{M}\sum_{i=1}^{M} C^{(i)}(K),$$

then for all tasks $i = 1, \ldots, M$,

$$C^{(i)}(K_\star) - C^{(i)}(K_\star^{(i)}) \quad \leq \quad \frac{2\|\Sigma^{(i)}_{K_\star^{(i)}}\| b_i(K_\star)^2}{\underline{\lambda}(\Sigma_0^{(i)})^2 \underline{\sigma}(R^{(i)})} \quad \sim \quad \mathcal{O}(b_i(K_\star)^2)$$

- bound is algorithm independent

Performance Bound: Policy Gradient

Given $K_0 \in \mathcal{K}_{\mathrm{stab}}$ and $\alpha > 0$ sufficiently small, compute the sequence

$$K_{n+1} = K_n - \alpha \nabla C_{\mathrm{avg}}(K_n), \quad n = 0, 1, 2, \ldots$$

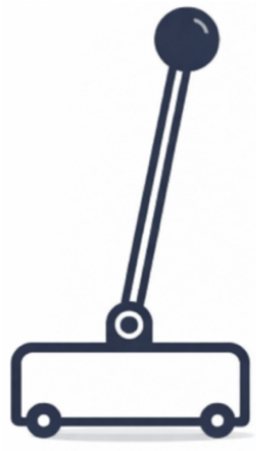
then for all tasks $i = 1, \ldots, M$

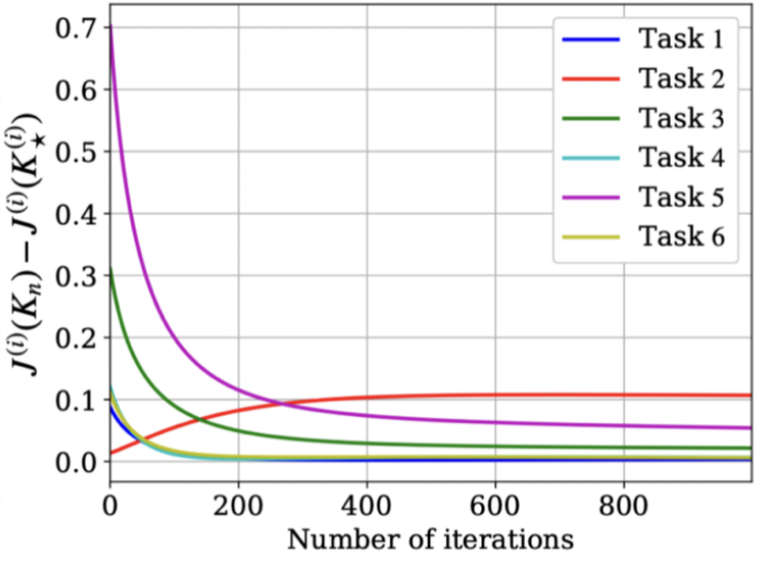
1. K_n is stabilizing for all n

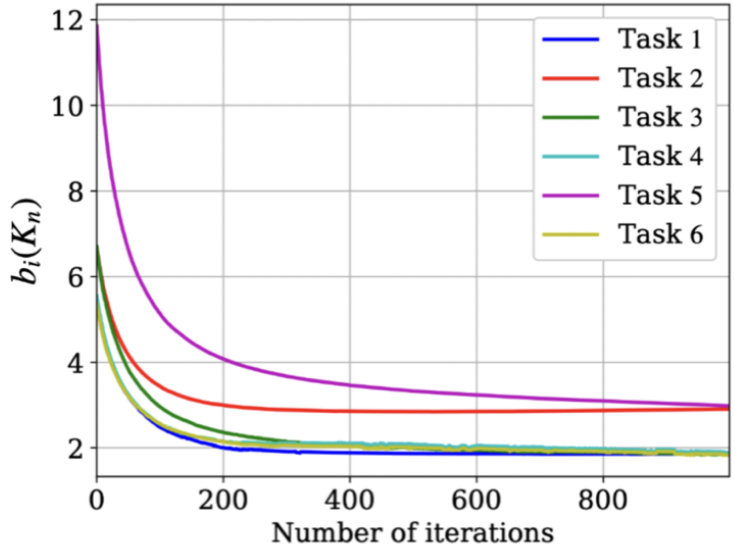
2. we have the asymptotic bound

$$C^{(i)}(K_\infty) - C^{(i)}(K_\star^{(i)}) \leq \frac{3\|\Sigma^{(i)}_{K_\star^{(i)}}\| b_i(K_\infty)^2}{4\underline{\lambda}(\Sigma_0^{(i)})^2 \underline{\sigma}(R^{(i)})} \sim \mathcal{O}(b_i(K_\infty)^2)$$

Inverted Pendulum







Unicycle

Kinematic dynamics:

$$\dot{p}_x = v\cos\theta, \quad \dot{p}_y = v\sin\theta, \quad \text{and} \quad \dot{\theta} = \omega$$

- linearized at $(v_{0,i}, \theta_{0,i})$, discretized with step size $\mathrm{d}t = 0.05$

$$x^{(i)} = \begin{bmatrix} p_x^{(i)} \\ p_y^{(i)} \\ \theta^{(i)} \end{bmatrix} \quad A^{(i)} = \begin{bmatrix} 1 & 0 & -\mathrm{d}t v_{0,i}\sin\theta_{0,i} \\ 0 & 1 & \mathrm{d}t v_{0,i}\cos\theta_{0,i} \\ 0 & 0 & 1 \end{bmatrix}, \quad B^{(i)} = \begin{bmatrix} \mathrm{d}t\cos\theta_{0,i} & 0 \\ \mathrm{d}t\sin\theta_{0,i} & 0 \\ 0 & \mathrm{d}t \end{bmatrix}$$

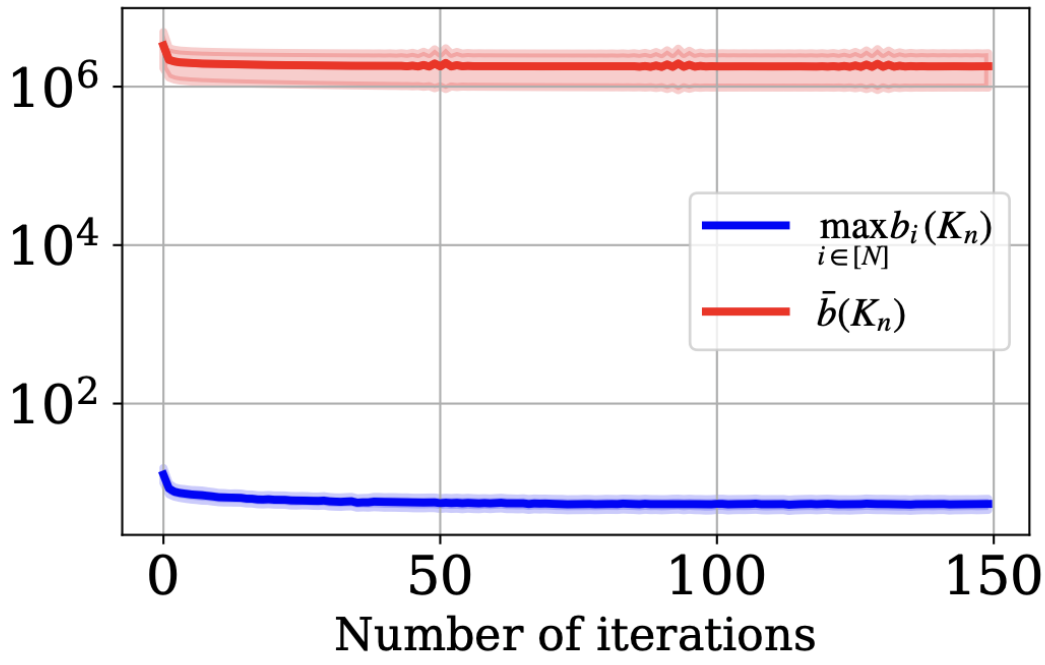
- consider $M = 6$ systems by sampling

$$v_{0,i}, \theta_{0,i}, q_i, r_i \sim \mathcal{U}([0.1, 1.75] \times [0, \pi/2] \times [0.1, 0.5] \times [0.01, 0.05])$$

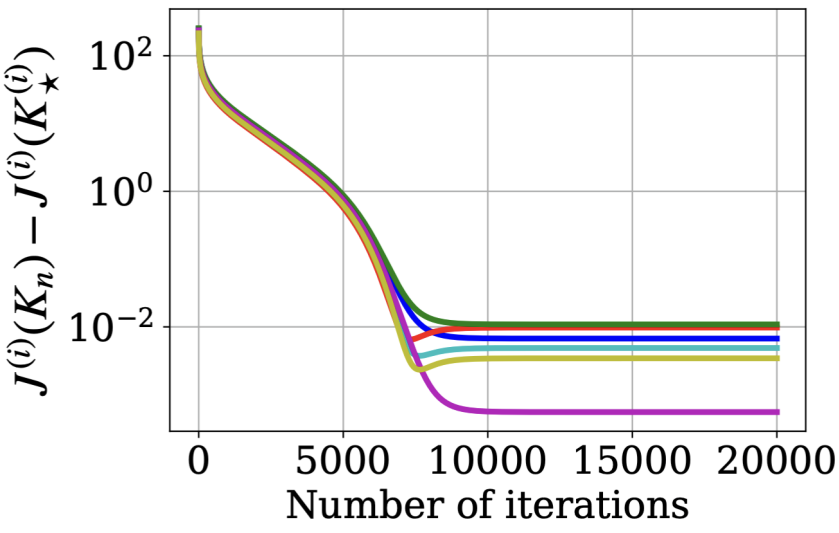
- with performance matrices given by

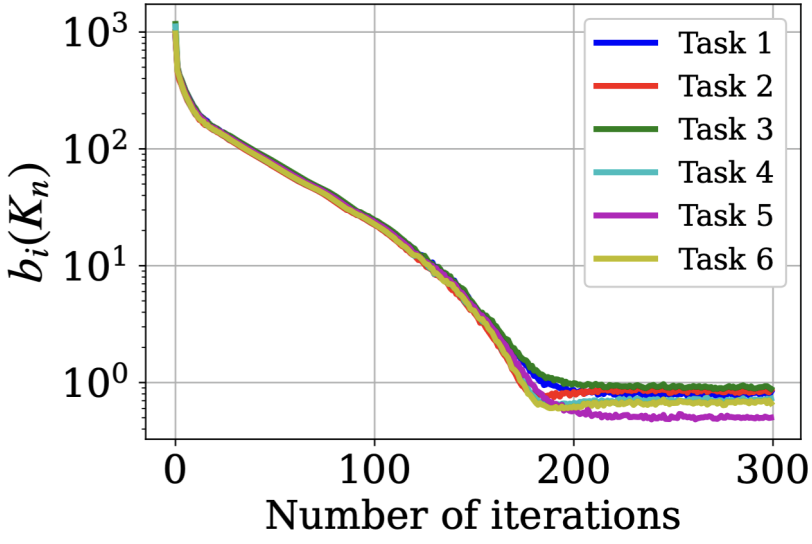
$$Q^{(i)} = q_i I, \quad R^{(i)} = r_i I$$

Unicycle: Heterogeneity Metrics



Unicycle Performance





Multi-Task LQR: Take Aways

- **Towards Foundation Control Policies:**

  Formalized multitask LQR as a step toward learning controllers that generalize across **heterogeneous** dynamical systems

- **System Theoretic Notion of Task Similarity:**

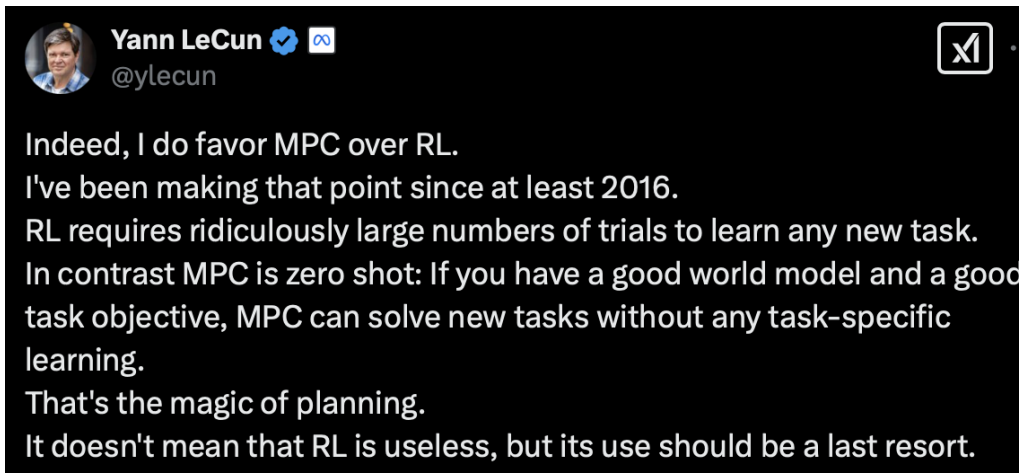
  Approximate bisimulation metrics in gradient-space capture closed-loop task similarity

- **Guarantees:**

  Established multi-task LQR performance bounds & algorithm specific results

Learning Invariant Visual Representations for Planning with Joint-Embedding Predictive World Models

Leonardo F. Toso [*1] **Davit Shadunts** [*1] **Yunyang Lu** [*1] **Nihal Sharma** [2] **Donglin Zhan** [1]
Nam H. Nguyen [2] **James Anderson** [1]



HPCwire

Yann LeCun's AMI Secures $1B Seed to Develop AI World Models

18 hours ago

WIRED

Yann LeCun Raises $1 Billion to Build AI That Understands the Physical World

2 days ago

Fortune

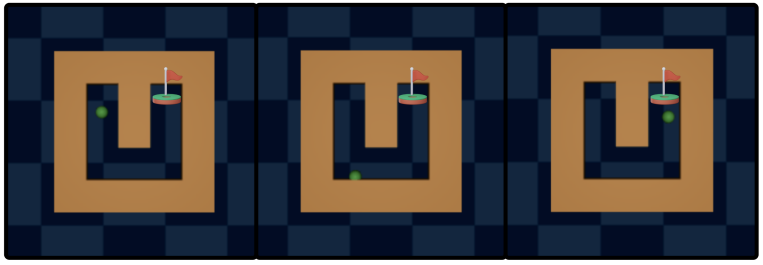
Yann LeCun's raises $1 billion in bet against LLMs

1 day ago

Visual Control: Setup

Reinforcement Learning

- **Observation** space $o_t \in \mathcal{O}$ (RGB images)
- **Action** space $a_t \in \mathcal{A}$
- **Policy** $\pi : \mathcal{O} \to \mathcal{A}$
- **Reward** function r_π
- **Trajectories** $(o_t, a_t, o_{t+1}) \sim \mathcal{D}_\pi$



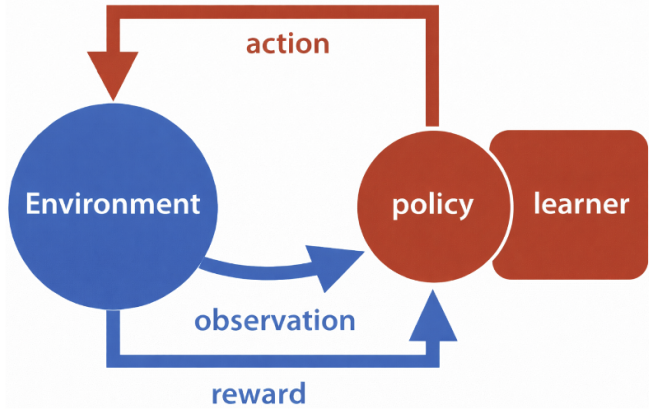
Objectives

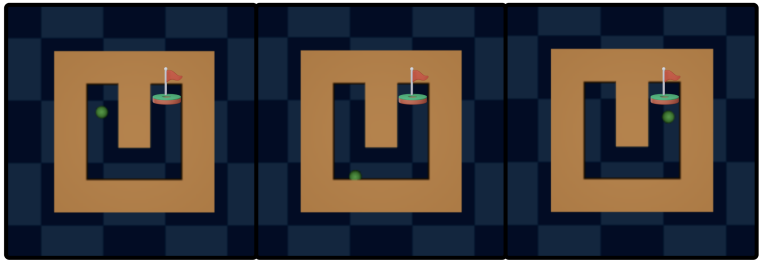
- learn the dynamics of the environment from the observations
- find a policy to move the ball to the hole

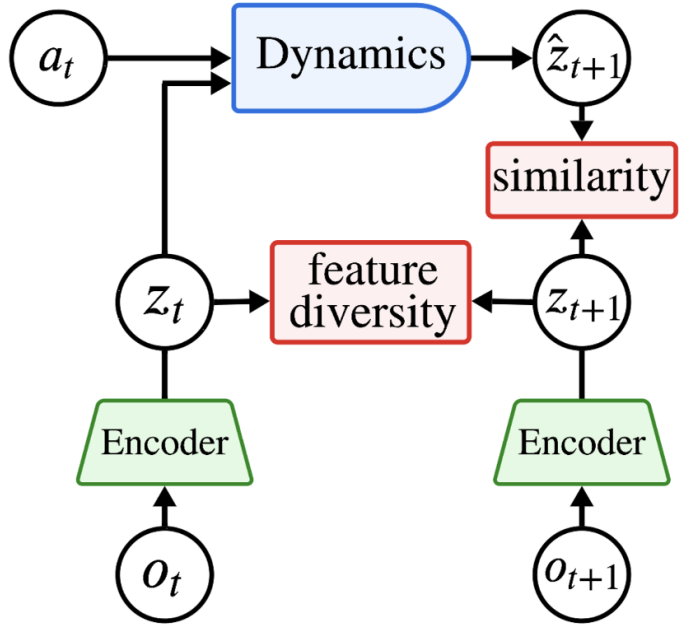
Visual Control: Setup

Reinforcement Learning

- **Observation** space $o_t \in \mathcal{O}$ (RGB images)
- **Action** space $a_t \in \mathcal{A}$
- **Policy** $\pi : \mathcal{O} \to \mathcal{A}$
- **Reward** function r_π
- **Trajectories** $(o_t, a_t, o_{t+1}) \sim \mathcal{D}_\pi$







Objectives

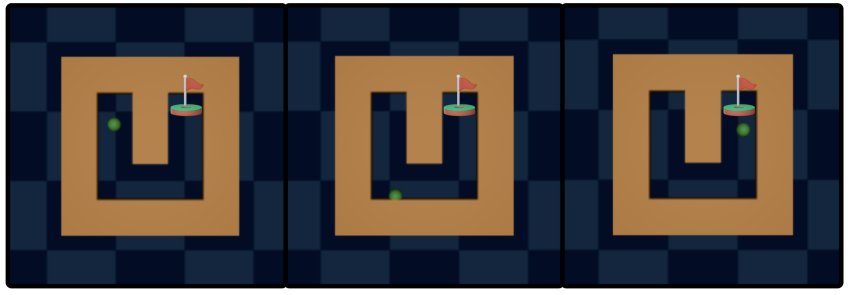
- learn the dynamics of the environment from the observations
- find a policy to move the ball to the hole

Slow Features

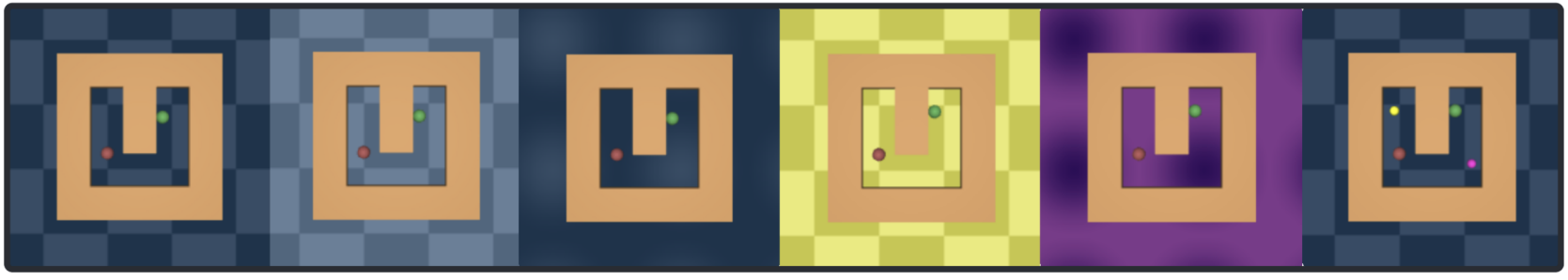
in observation space we have lots of task irrelevant information

- background color, texture, patterns
- lighting, shadows, image blurring
- camera position

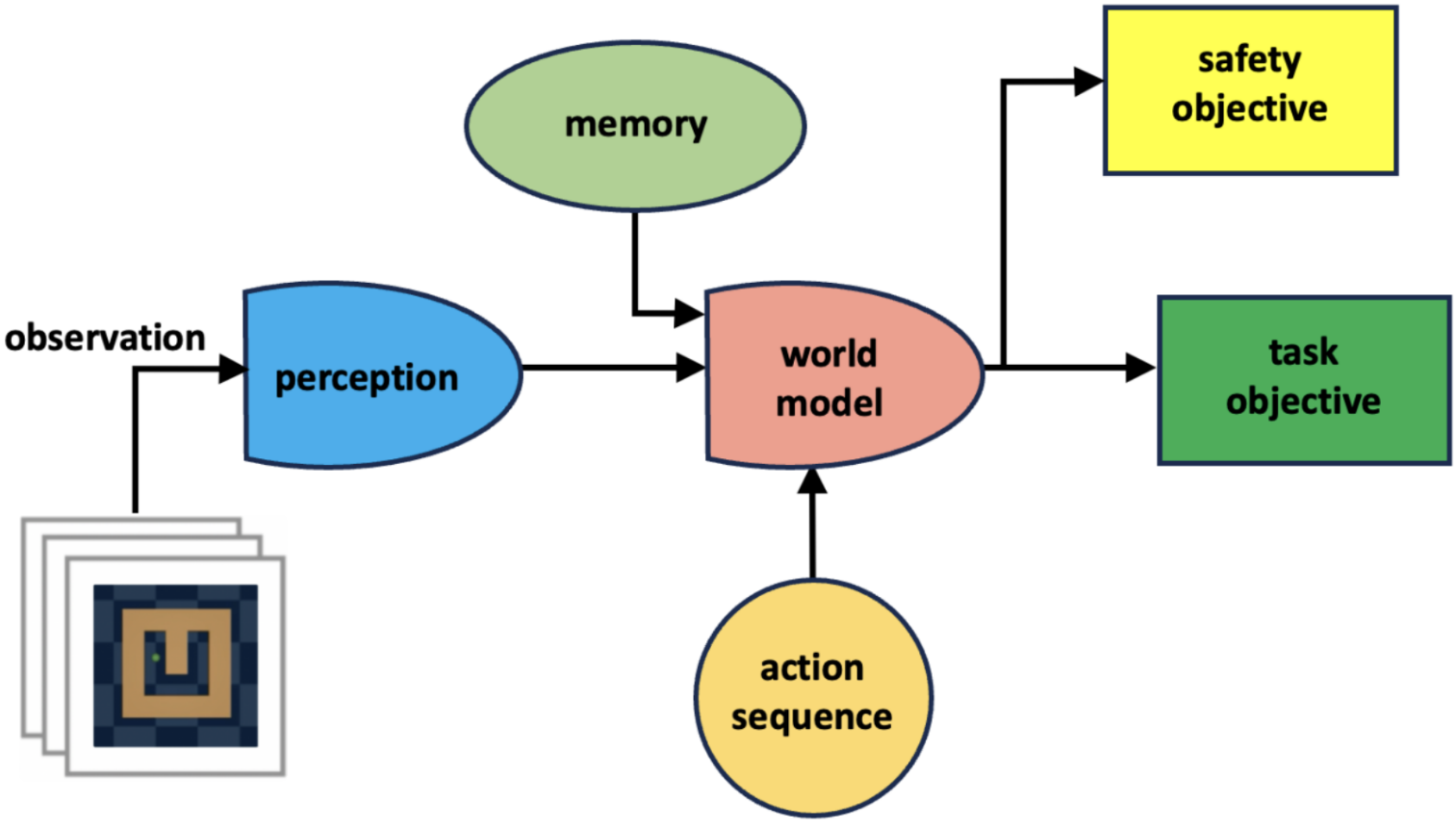
train with observations of the form



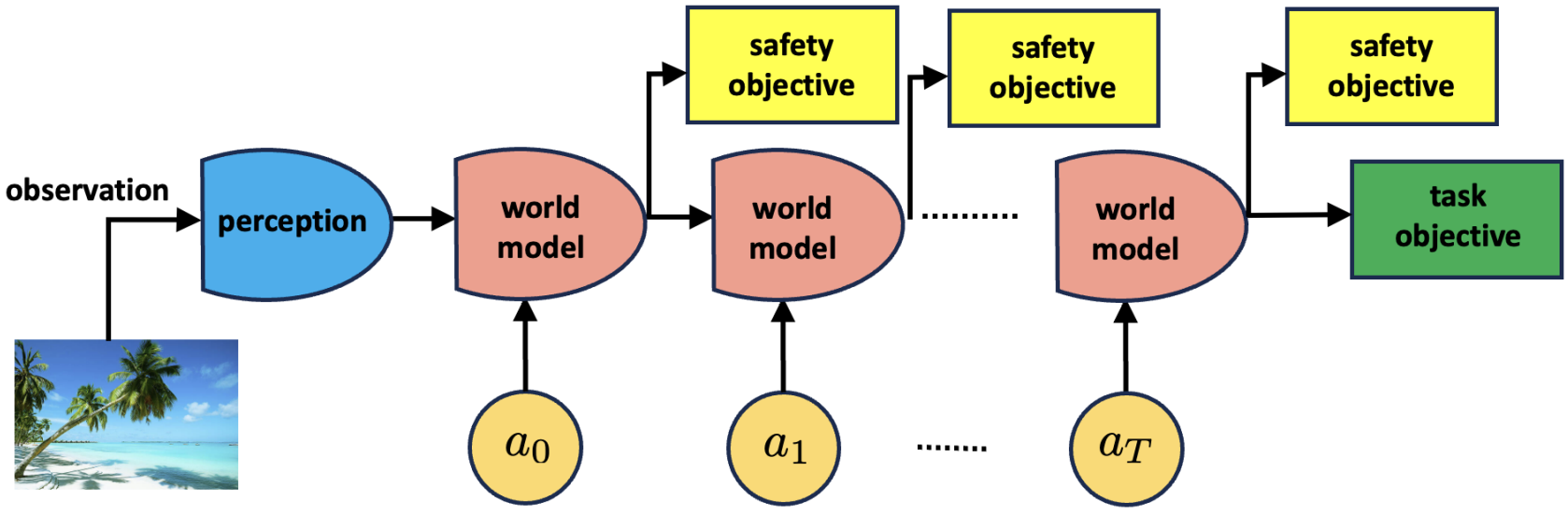
deploy to tasks with varying slow features



Planning with a World Model

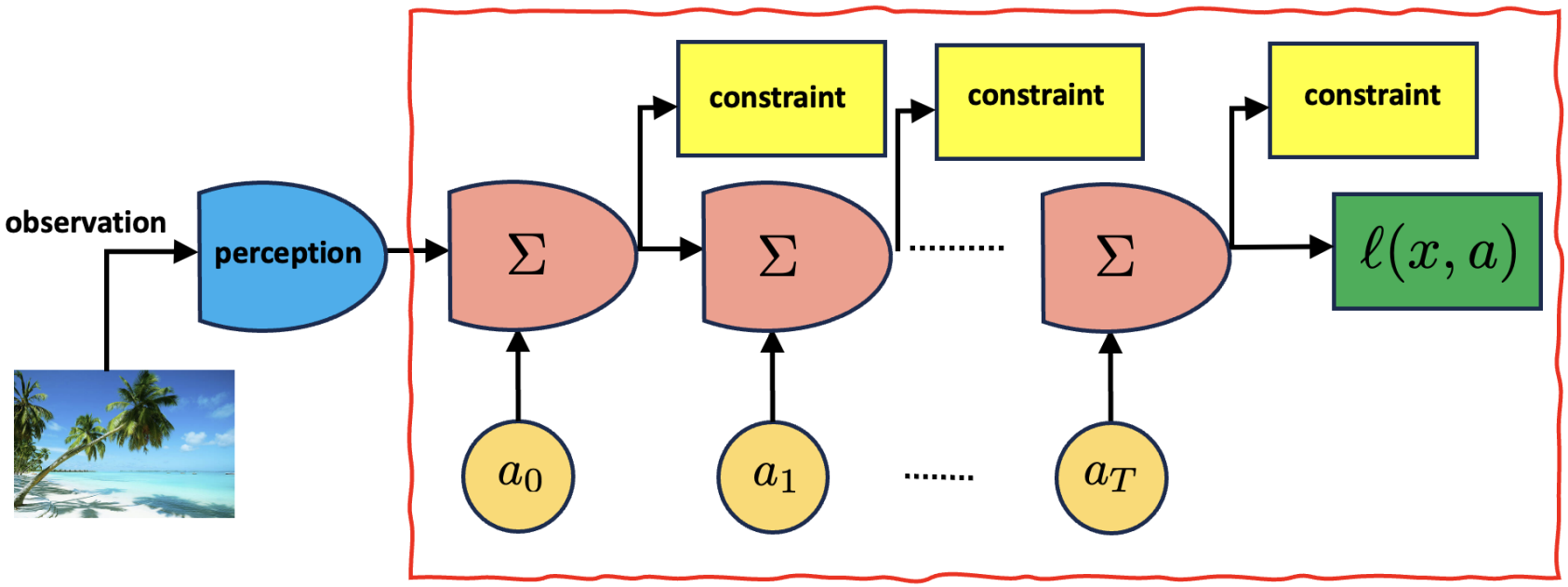


Sequential Planning



Model Predictive Control

MPC is a special case of sequential planning with a world model!



Most simple realization

- Σ is an LTI system

$$s_{t+1} = As_t + Ba_t + w_t$$

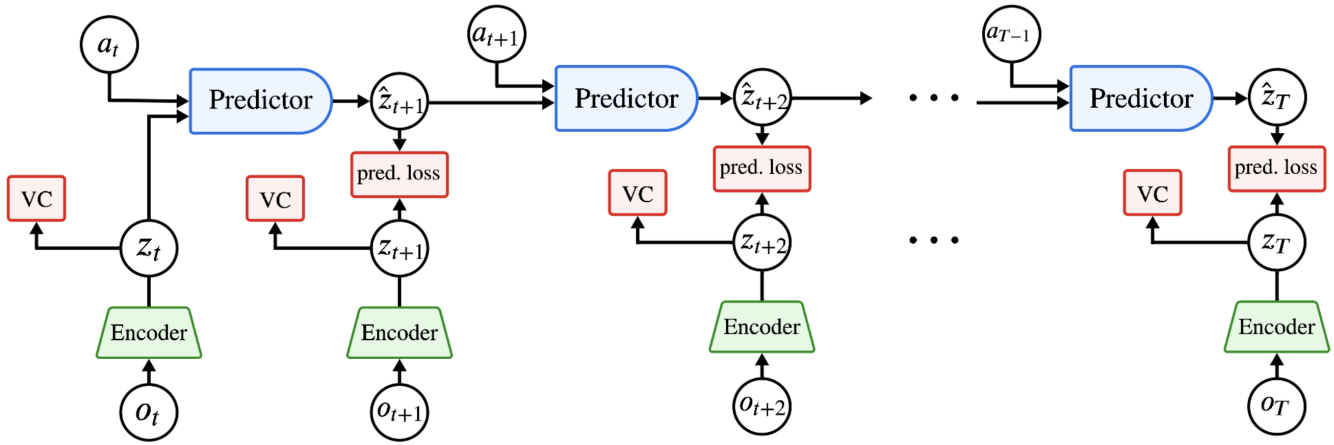
- task objective is quadratic in state and action

$$l(s, a) = \sum_{\tau=0}^{T-1} (s_\tau^T Q s_\tau + a_\tau^T R a_\tau) + s_T^T Q_f s_t$$

Joint-Embedding Predictive Architectures (JEPAs)

Objective

simultaneously learn the encoder and dynamics that capture task relevant visual features such that we can predict and plan



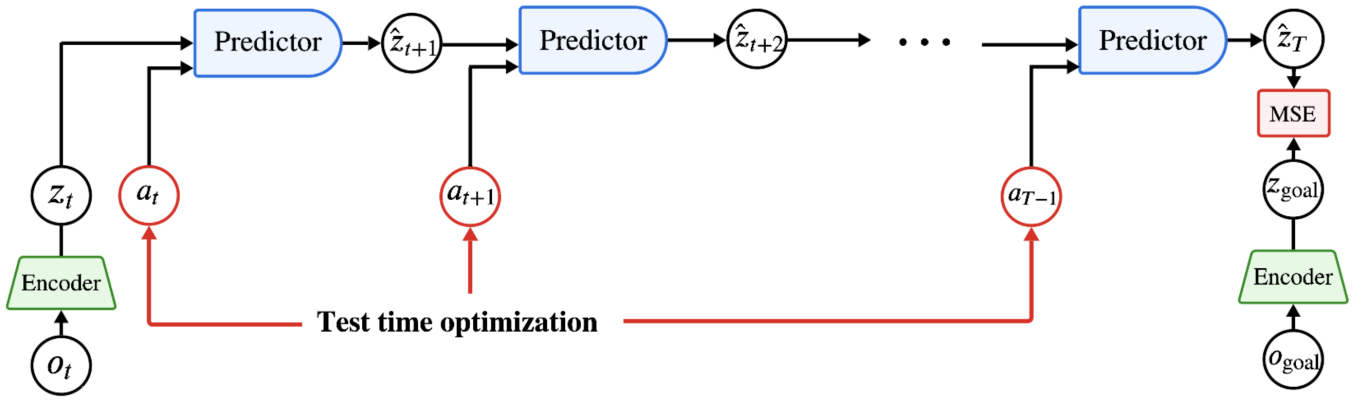
- **Encoder:** $f_\theta : \mathcal{O} \to \mathbb{R}^{d_z}$ – maps observations, o_t, to latent embeddings
- **Predictor:** $T_\phi : \mathbb{R}^{d_z} \times \mathcal{A} \to \mathbb{R}^{d_z}$ – encodes dynamics

$$\underbrace{\mathcal{L}_{\text{pred}}(\theta, \phi) = \mathbb{E}_{\mathcal{D}_\pi} \left[l(\overbrace{T_\phi(f_\theta(o_t), a_t)}^{\hat{z}_{t+1}}, \overbrace{f_\theta(o_{t+1})}^{z_{t+1}}) \right]}_{\text{prediction loss}}$$

- Feature diversity promoted through Variance–Invariance–Covariance Regularization (VICReg) in PCA-space

Test-Time Planning

use the pre-trained encoder f_θ (DINOv2, iBOT, SimDinov2, etc)



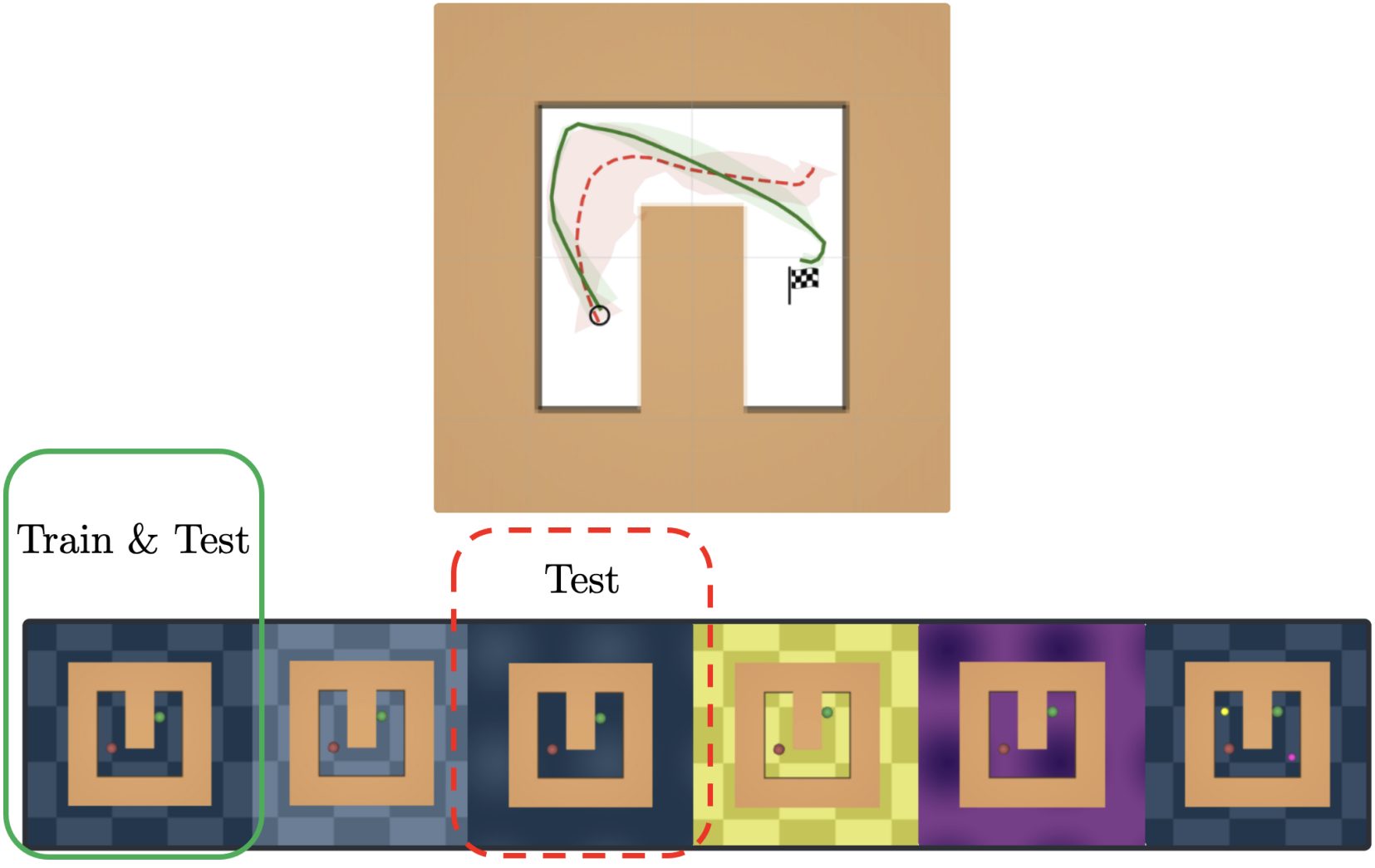
MPC planning in latent space using a cross entropy method, with terminal cost

$$c(z_T, \{a_t\}_t) = \|\hat{z}_T(\{a_t\}_t) - z_g\|^2$$

Problem

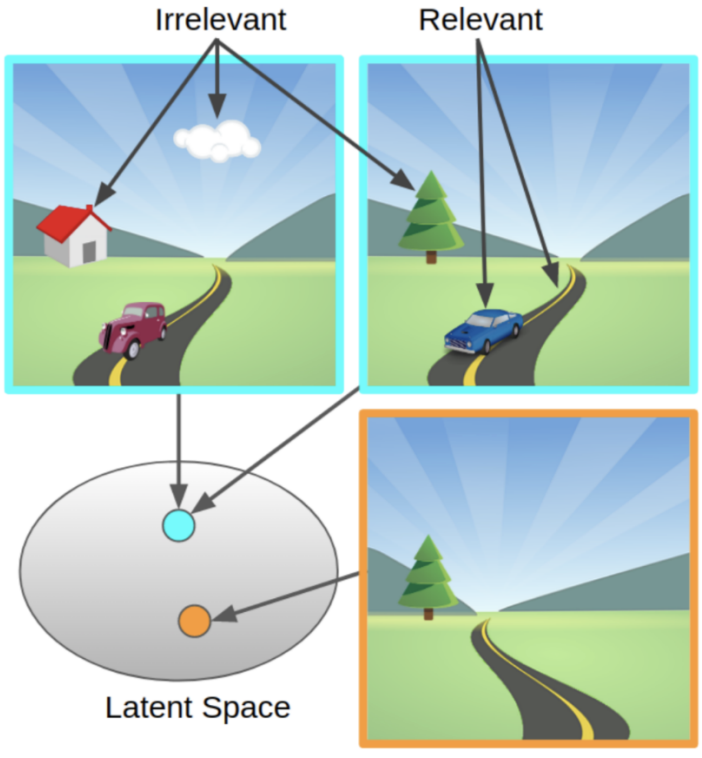
- slow features are captured in the latent space
- easy to predict, but not useful for planning

Slow Features in Latent Space



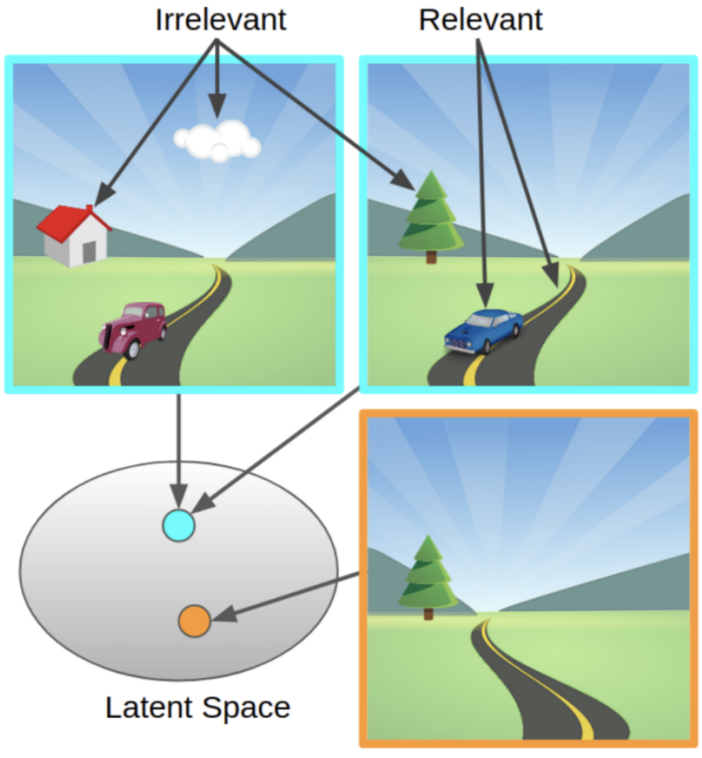
Fails because we are planning in the wrong space!

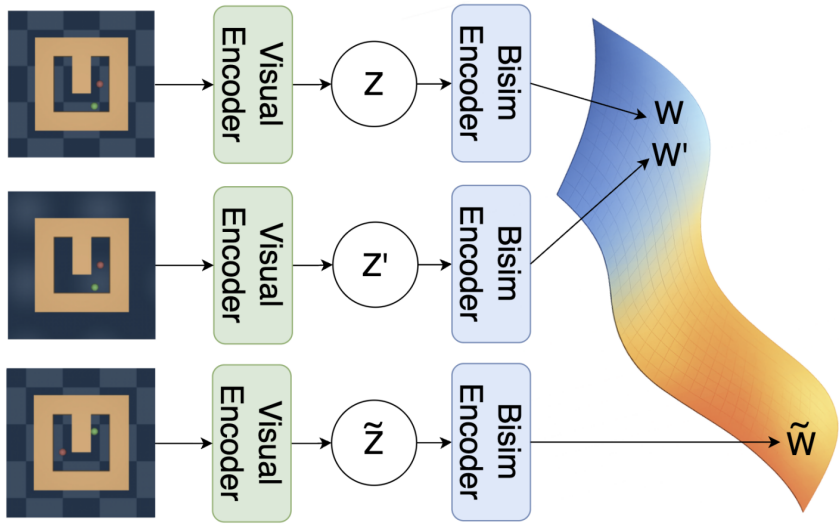
Bisimulation Metric



[Zhang, McAllister, Calandra, Gal, and Levine, ICLR, 2021]

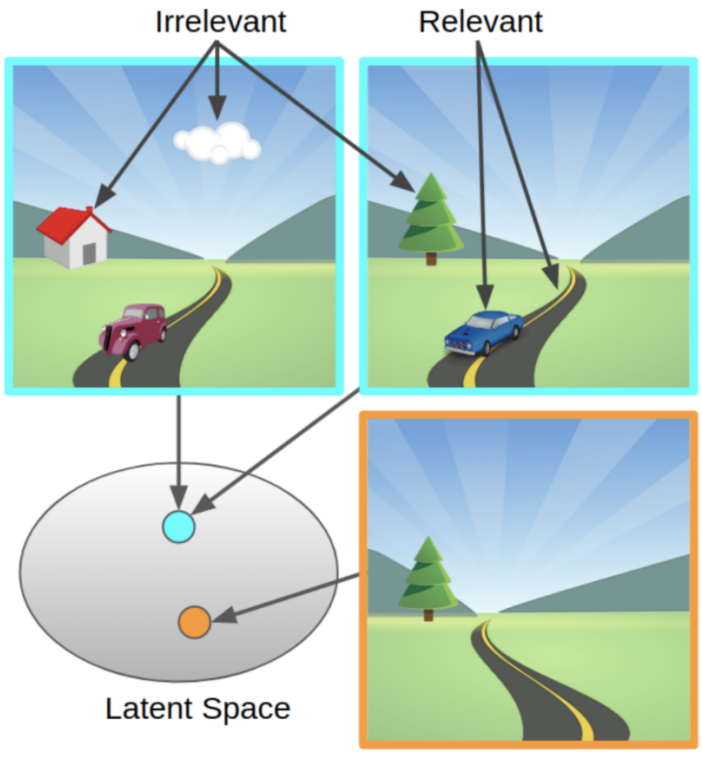
Bisimulation Metric

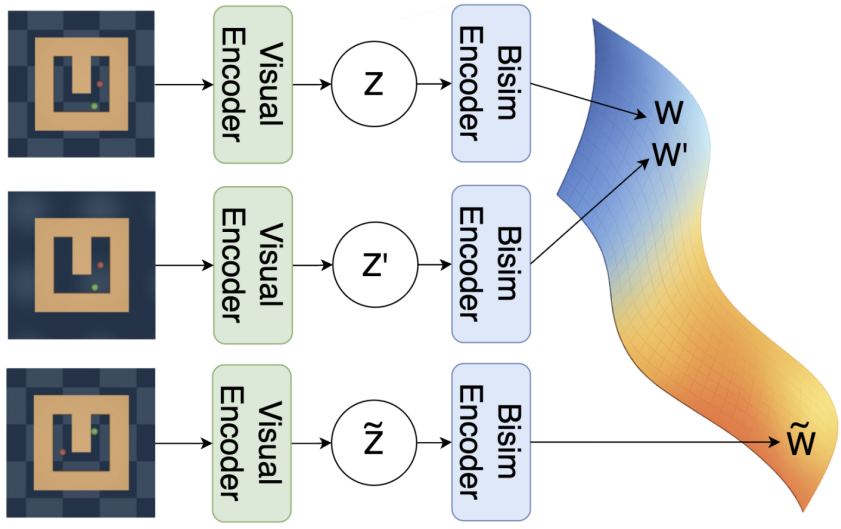




[Zhang, McAllister, Calandra, Gal, and Levine, ICLR, 2021]

Bisimulation Metric





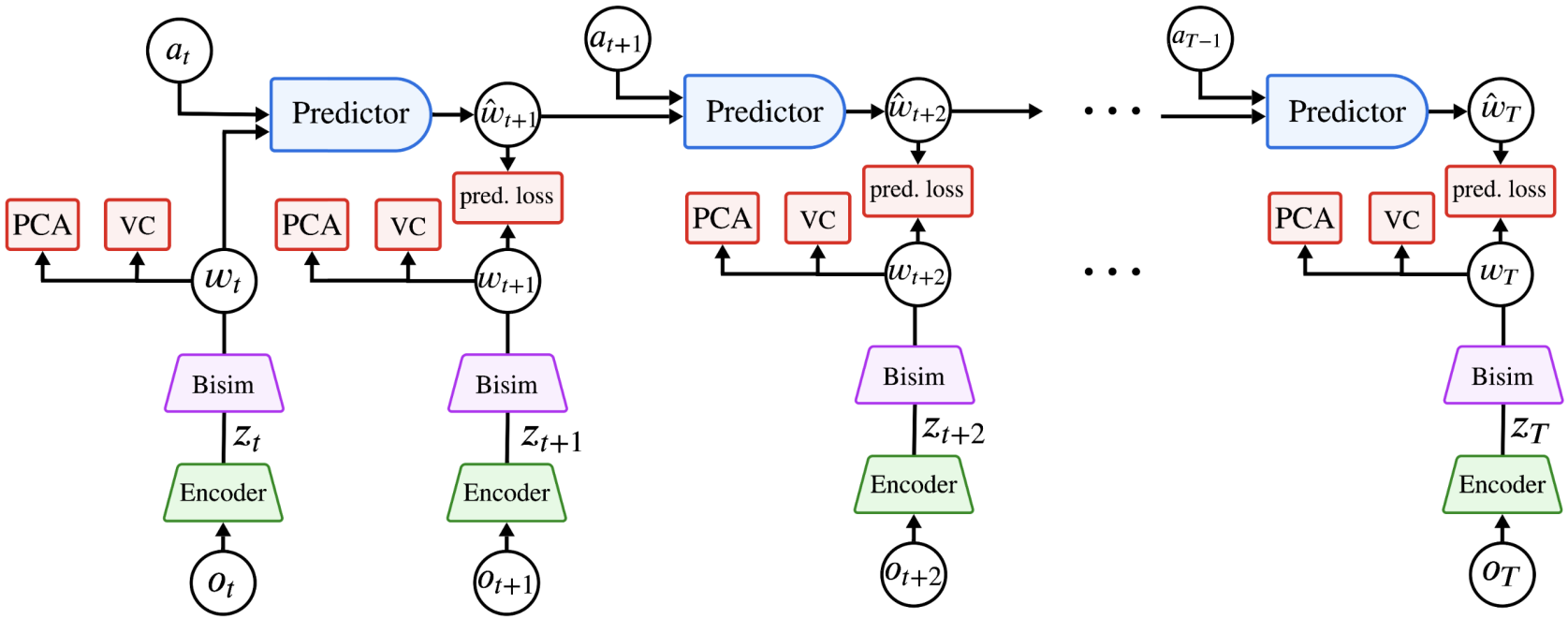
On-Policy Bisimulation Metric [Castro, 2020]

given a reward function $r(z, a)$, and a transition kernel $P_\pi(\cdot \mid z)$ induced by π

$$d_\pi(z, z') = \underbrace{\left|r_\pi(z) - r_\pi(z')\right|}_{\text{reward similarity}} + \gamma \underbrace{W_1\left(P_\pi(\cdot \mid z), P_\pi(\cdot \mid z')\right)}_{\text{dynamics similarity}} \quad \text{(bisim-metric)}$$

Proposed Architecture

- **Encoder:** $f_\theta : \mathcal{O} \to \mathbb{R}^{d_z}$ – pre-trained DINOv2 [Meta AI]
- **Bisim Encoder:** $h_\eta : \mathbb{R}^{d_z} \to \mathbb{R}^{d_w}$ – map to invariant latent space $d_w < d_z$
- **Predictor:** $T_\phi : \mathbb{R}^{d_w} \times \mathcal{A} \to \mathbb{R}^{d_w}$ – encodes dynamics

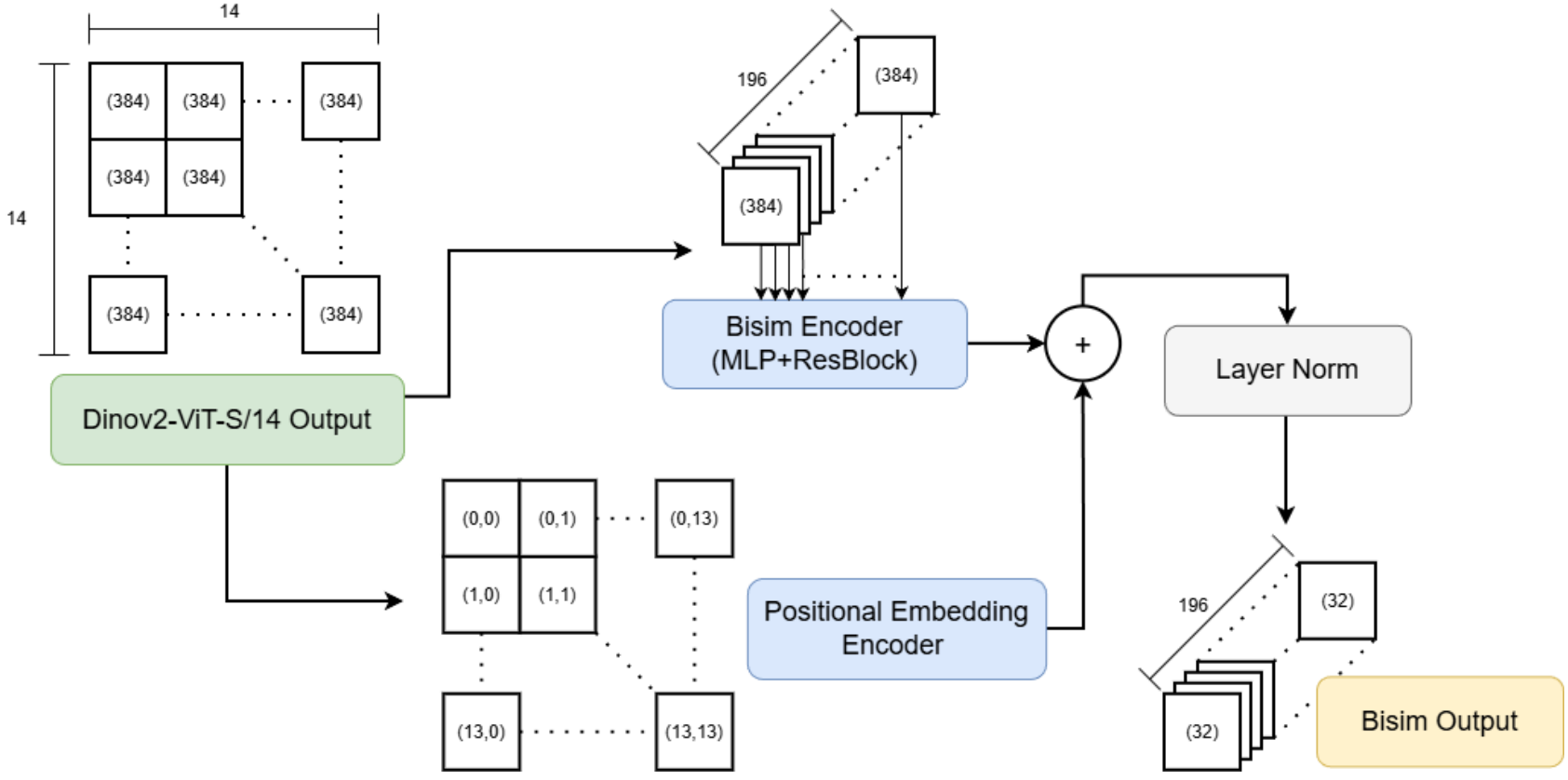


$$\mathcal{L}_{\text{bisim}}(\eta, \phi) \triangleq \mathbb{E}\left[\left(\|w_t - w'_t\| - \Delta_{\text{bisim}}(\eta, \phi)\right)^2\right],$$

where

$$\Delta_{\text{bisim}}(\eta, \phi) \triangleq \gamma \left\|T_\phi(w_t, a_t) - T_\phi\left(w'_t, a'_t\right)\right\|$$

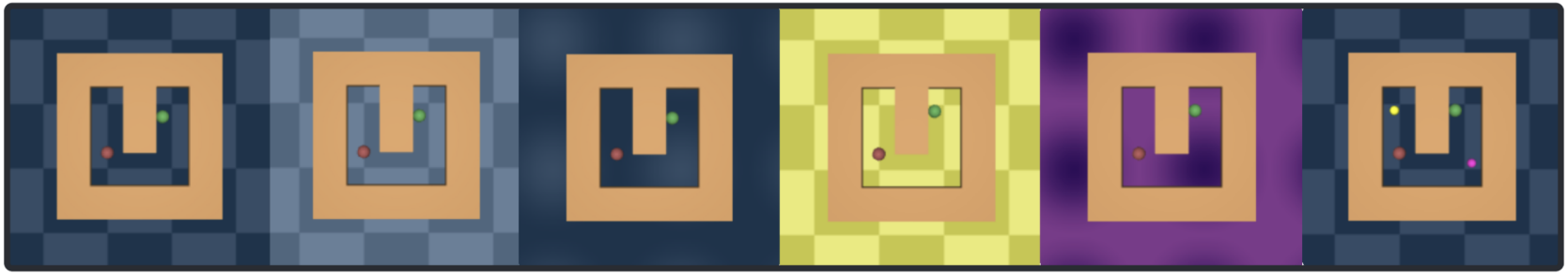
Detailed Network Architecture



- Bisim latent space is $10\times$ smaller than DINO WM
- DINO 4–5 days to train on 1 GPU
- DINO-Bisim 3 days

Experiments

Task: MuJoCo PointMaze



Experiments

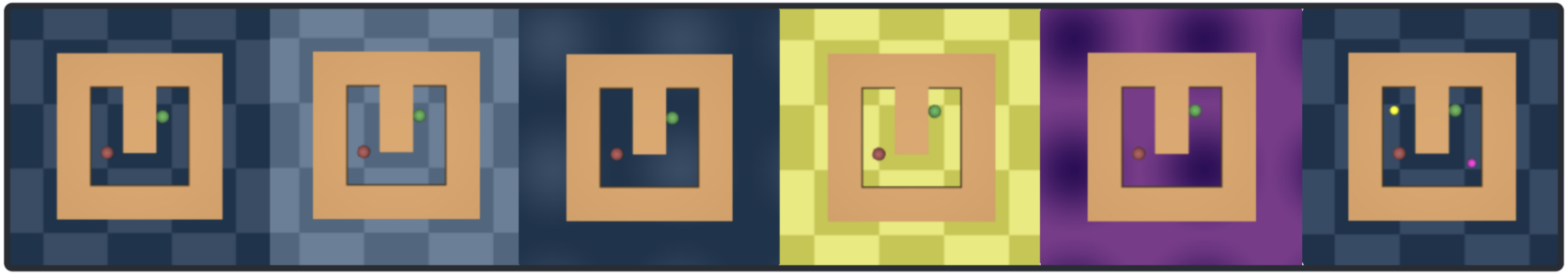
Task: MuJoCo PointMaze

DINO Baselines

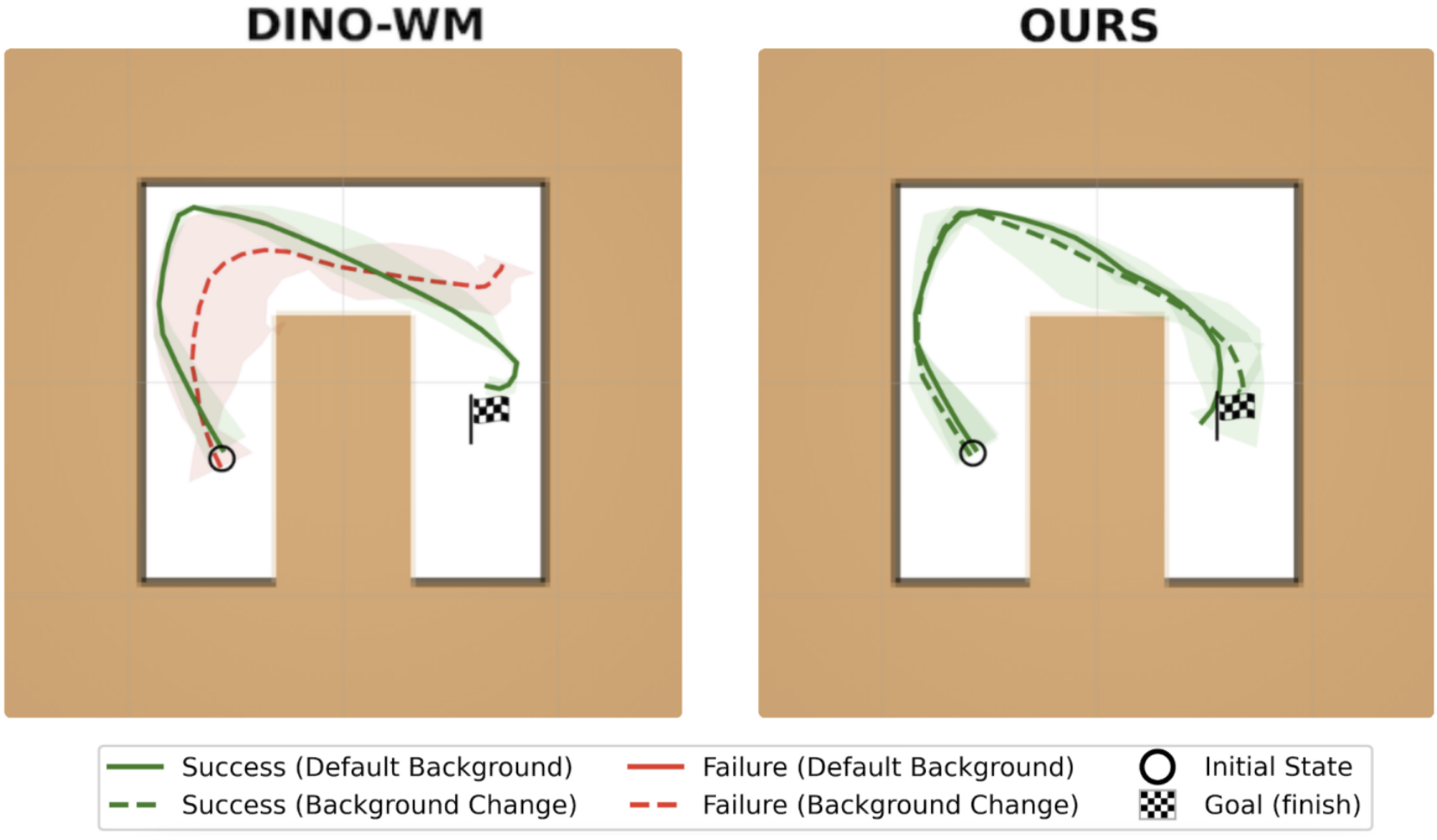
- DINO-WM
  [Zhou et al., 2025]
- Domain Randomization (40%)
  [Tobin et al., 2017]

Other JEPA Baselines

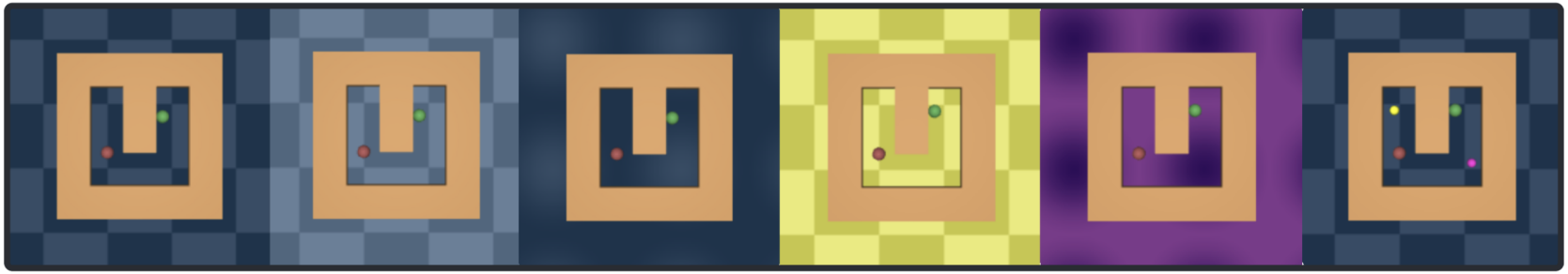
- IBOT
  [Zhou et al., 2017]
- SimDINOv2
  [Wu et al., 2025]



Results: Sample Trajectories

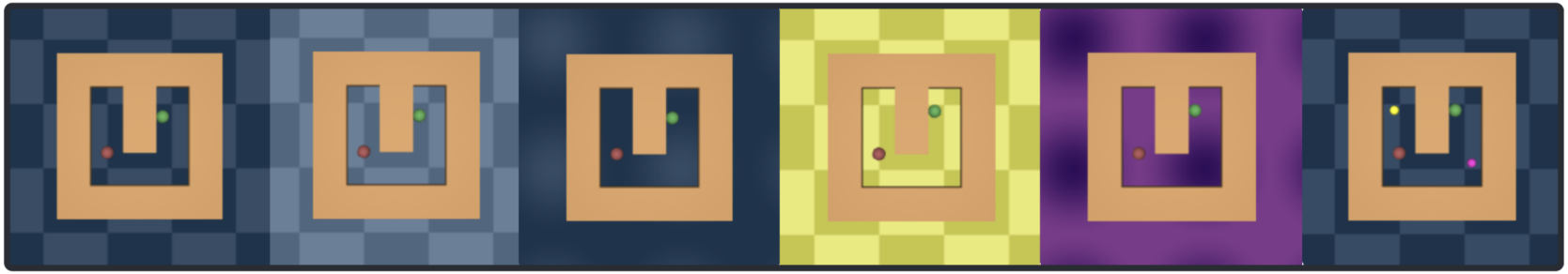


Results: Success Rate



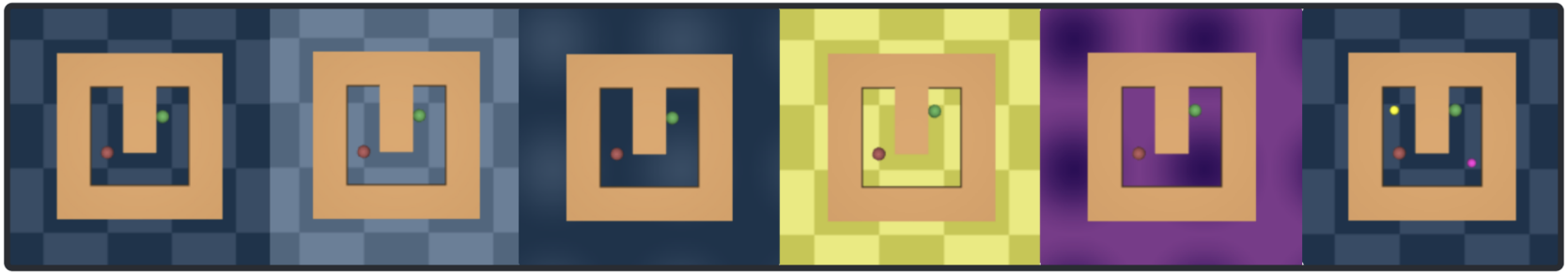
- **NC:** No Change
- **SC:** Slight color change
- **C:** Color gradient change
- **LC:** Large color change
- **LCG:** Large color and gradient change
- **D:** moving distractors

Results: Success Rate



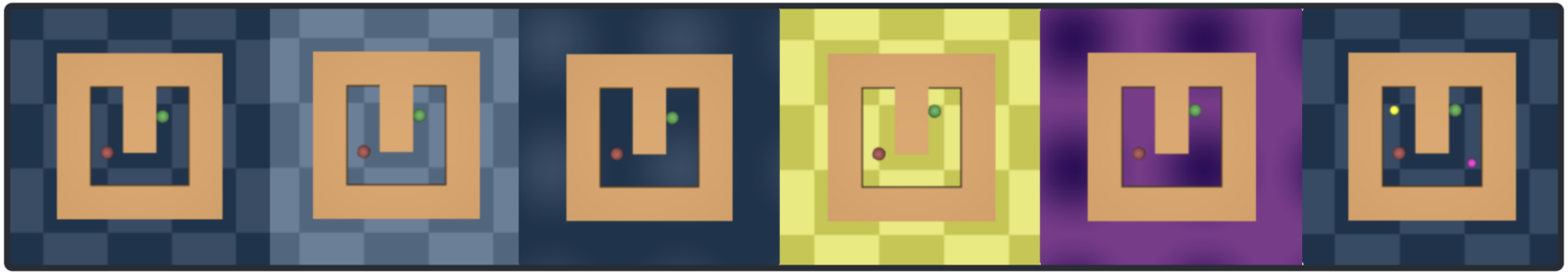
Model	NC	SC	C	LC	LCG	D
DINO-WM	0.80	0.72	0.60	0.56	0.48	0.78
w/DR	0.82	0.82	0.82	0.68	0.64	0.82
Ours	0.78	0.80	0.76	**0.86**	**0.78**	**0.82**

Results: Success Rate



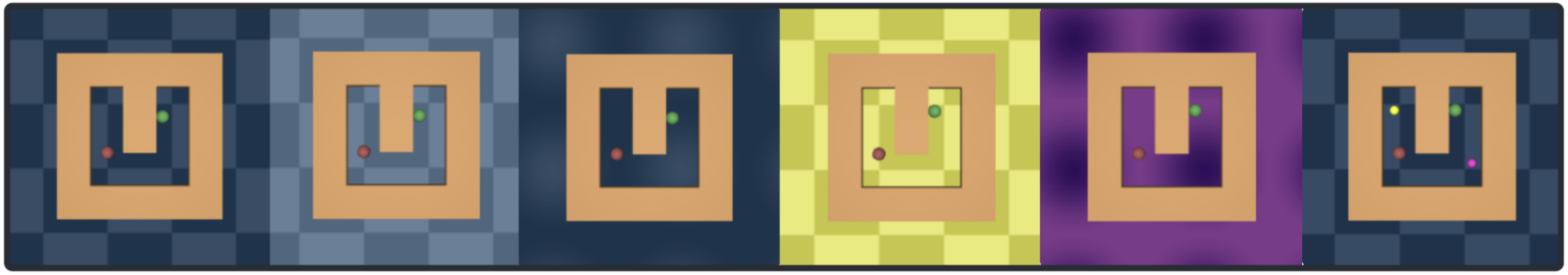
- **NC:** No Change
- **SC:** Slight color change
- **C:** Color gradient change
- **LC:** Large color change
- **LCG:** Large color and gradient change
- **D:** moving distractors

Results: Success Rate



Model	NC	SC	C	LC	LCG	D
DINO-WM	0.80	0.72	0.60	0.56	0.48	0.78
w/DR	0.82	0.82	0.82	0.68	0.64	0.82
Ours	0.78	0.80	0.76	**0.86**	**0.78**	**0.82**

Results: Success Rate



Model	NC	SC	C	LC	LCG	D
DINO-WM	0.80	0.72	0.60	0.56	0.48	0.78
w/DR	0.82	0.82	0.82	0.68	0.64	0.82
Ours	0.78	0.80	0.76	**0.86**	**0.78**	**0.82**

No Encoder	0.68	0.44	0.70	0.26	0.36	0.64
SimDINOv2	0.40	0.38	0.36	0.42	0.42	0.36
iBOT	0.72	0.70	0.74	0.72	0.72	0.72

Conclusions

1. **Theoretical performance bounds with LQR**
2. **Towards real problem instances with JEPAs**
3. **Plan with MPC not RL**
4. **Task irrelevant features inhibit planning with JEPAs**
5. **Plan in latent space**... but make sure it's the right latent space

   – *approximate bisimilarity looks promising for constructing the right space*

Acknowledgments


Leonardo Toso


Donglin Zhan


Han Wang


Davit Shadunts


Yunyang Lu


Charis Stamouli


George Pappas


Anastasios Tsiamis


Nam Nguyen


Nihal Sharma

COLUMBIA | ENGINEERING
The Fu Foundation School of Engineering and Applied Science

Hyperparamters

Name	Value
Image size	224
Optimizer	AdamW
Decoder learning rate	3e−4
Predictor learning rate	1e−5
Action encoder learning rate	1e−4
Bisimulation learning rate	5e−7
Action embedding dim.	10
Proprio embedding dim.	10
Epochs	50
Batch size	20

Name	Value
Bisimulation memory buffer size	1000
Bisimulation state comparison size	200
Action dim.	10
Num patches	196
Latent dim.	32
Patch dim.	32
Patch embedding dim.	384