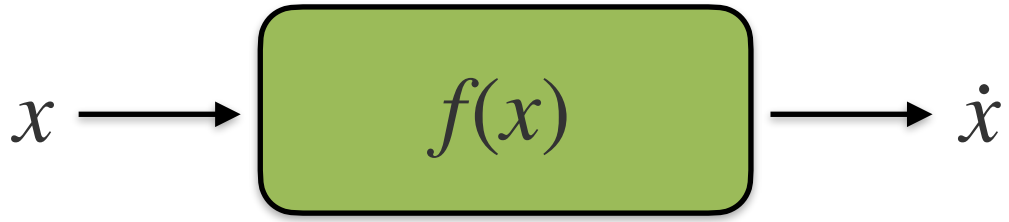
USC University of Southern California

USC Viterbi School of Engineering

Actuator-State-Aware Koopman MPC

Zacharias Brown

Nonlinear Plant Model



$$\dot{x} = f(x)$$

OR

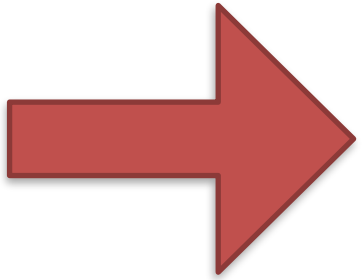
$$x_{k+1} = f(x_k)$$

A nonlinear f(x) can be very unwieldy and thus make the future states hard to predict

A Different Approach

Instead of only watching x:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \end{bmatrix}$$



watch functions of x:

$$\begin{bmatrix} g_1(x) \\ g_2(x) \\ g_3(x) \\ \vdots \end{bmatrix}$$

$$x_{k+1} = f(x_k)$$

$$g(x_{k+1}) = g(f(x_k))$$

USC Viterbi School of Engineering

Observables Become A Lifted State

$$z_k = \psi(x_k) = \begin{bmatrix} g_1(x_k) \\ g_2(x_k) \\ g_3(x_k) \\ \vdots \end{bmatrix}$$

Example Dictionary

$$\psi(x) = \begin{bmatrix} x \\ x^2 \\ \sin(x) \\ \cos(x) \\ \vdots \end{bmatrix}$$

The lifted state is a vector of chosen observables.

USC Viterbi School of Engineering

Polynomial Example

Take the plant

$$x_{k+1} = x_k^2$$

And lifted state

$$\psi(x) = \begin{bmatrix} x \\ x^2 \\ x^4 \end{bmatrix}$$

$$z_k = \begin{bmatrix} x_k \\ x_k^2 \\ x_k^4 \end{bmatrix} \Rightarrow z_{k+1} = \begin{bmatrix} x_k^2 \\ x_k^4 \\ x_k^8 \end{bmatrix}$$

Two entries shift cleanly. One new entry appears.

Infinite Lift

With enough basis functions…

$$z_k = \begin{bmatrix} x_k \\ x_k^2 \\ x_k^4 \\ x_k^8 \\ x_k^{16} \\ \vdots \end{bmatrix} \quad z_{k+1} = \begin{bmatrix} x_k^2 \\ x_k^4 \\ x_k^8 \\ x_k^{16} \\ x_k^{32} \\ \vdots \end{bmatrix}$$

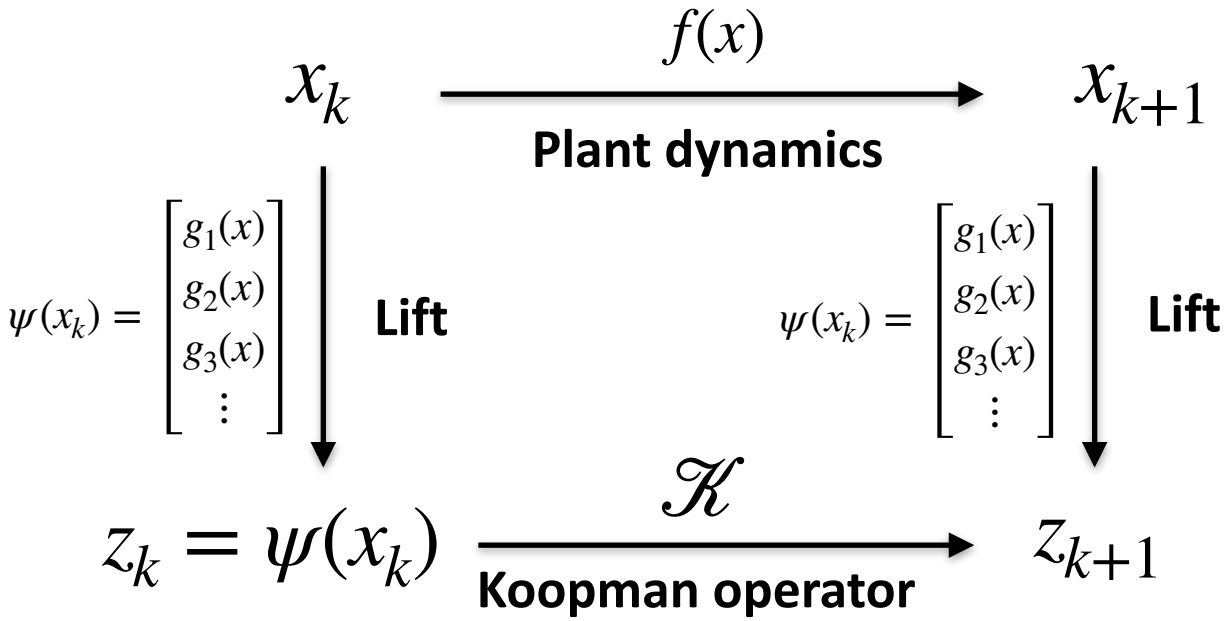
We approach a linear map from $z_k -> z_{k+1}$

$$z_{k+1} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 1 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 1 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 1 & \cdots \\ 0 & 0 & 0 & 0 & 0 & \ddots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} z_k$$

In fact, for a infinitely large lifted state, there is a liner mapping from $z_k -> z_{k+1}$

USC Viterbi School of Engineering

The Koopman Operator



$$(\mathcal{K}g)(x) = g(f(x))$$

The Koopman operator evolves the observables

Background: Koopman operator framing; see Otto and Rowley (2021).

USC Viterbi School of Engineering

EDMD Finite Approximation

An infinitely large predictor isn't very helpful for control

$$z_k^{(\infty)} = \begin{bmatrix} g_1(x_k) \\ g_2(x_k) \\ g_3(x_k) \\ \vdots \end{bmatrix}$$

$$z_{k+1}^{(\infty)} = \mathcal{K} z_k^{(\infty)}$$

Infinite dimensional exact predictor.

$$z_k = \psi_N(x_k) = \begin{bmatrix} g_1(x_k) \\ g_2(x_k) \\ \vdots \\ g_N(x_k) \end{bmatrix}$$

$$z_{k+1} \approx A z_k$$

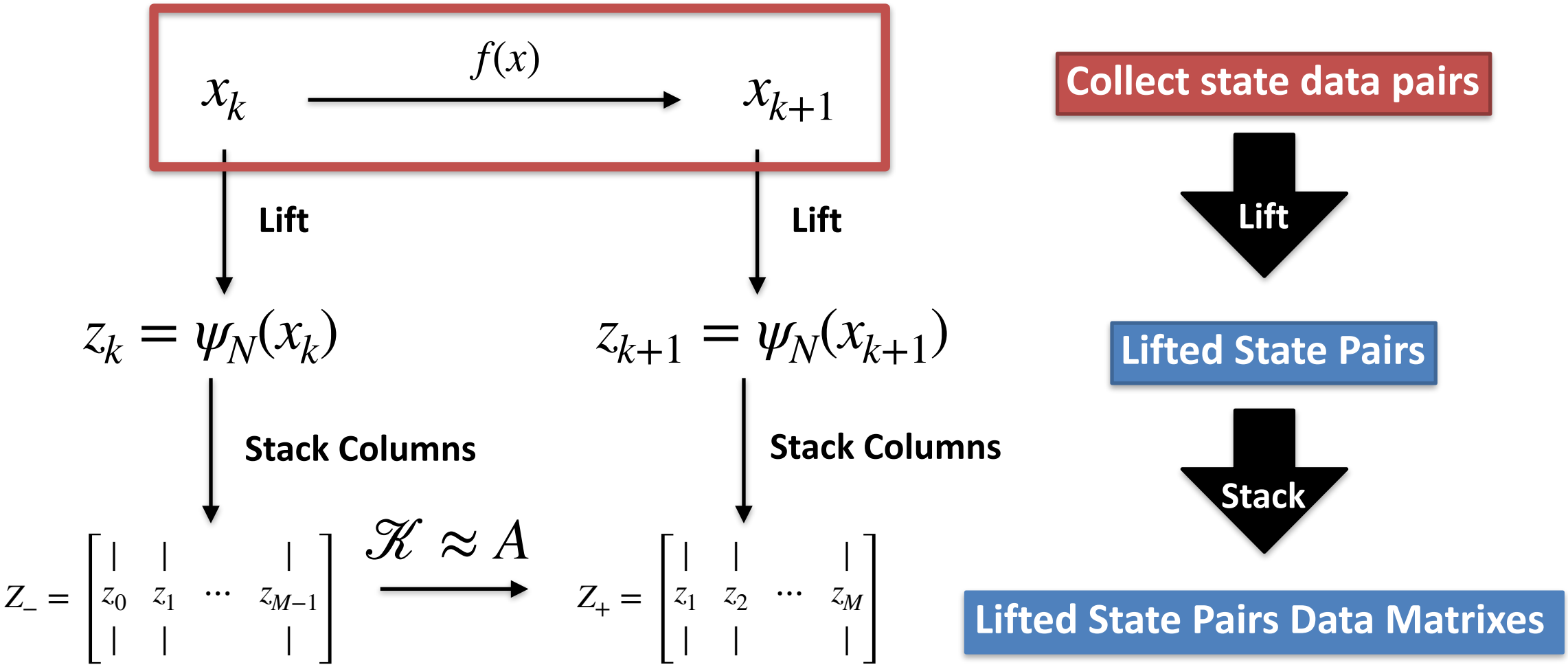
$$\hat{x}_k = C z_k$$

Finite approximate predictor

Practical approximation: Extended Dynamic Mode Decomposition (EDMD).

USC Viterbi School of Engineering

EDMD Finite Approximation



Collect state data pairs

Lift

Lifted State Pairs

Stack

Lifted State Pairs Data Matrixes

Practical approximation: Extended Dynamic Mode Decomposition (EDMD).

EDMD linear regression fit

$$Z_+ \approx AZ_-$$

$$A^\star = \arg\min_A \|Z_+ - AZ_-\|_F^2$$

$$A^\star = Z_+ Z_-^\dagger \quad \text{or} \quad A^\star = Z_+ Z_-^T \left(Z_- Z_-^T + \lambda I\right)^{-1}$$

EDMD = finite lifted least-squares Koopman approximation.

USC Viterbi School of Engineering

Add Control: EDMDc / Koopman-MPC Form

$$x_{k+1} = f(x_k, u_k)$$

$$z_{k+1} \approx A z_k + B u_k$$

$$U_- = \begin{bmatrix} | & | & & | \\ u_0 & u_1 & \cdots & u_{M-1} \\ | & | & & | \end{bmatrix}$$

$$Z_+ \approx AZ_- + BU_-$$

$$Z_+ \approx \begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} Z_- \\ U_- \end{bmatrix}$$

$$[A^\star \quad B^\star] = \arg\min_{A,B} \left\| Z_+ - AZ_- - BU_- \right\|_F^2$$

$$[A^\star \quad B^\star] = Z_+ \begin{bmatrix} Z_- \\ U_- \end{bmatrix}^\dagger$$

Control term enters the lifted state **linearly**

Baseline Koopman-MPC form: Korda and Mezic (2018).

USC Viterbi School of Engineering

The Input Channel Is Delicate

$$x_{k+1} = f(x_k)$$

$$\downarrow$$

$$z_{k+1} = \mathcal{K} z_k$$

$$x_{k+1} = f(x_k, u_k)$$

$$\downarrow$$

$$(\mathcal{K}_u g)(x) = g(f(x, u))$$

$$\downarrow$$

$$z_{k+1} = \mathcal{K}_{u_k} z_k$$

$$\downarrow$$

$$z_{k+1} \approx A z_k + B u_k$$

Example

$$x_{k+1} = x_k^2 + x_k u_k$$

$$z_k = \begin{bmatrix} x_k \\ x_k^2 \\ x_k^4 \\ \vdots \end{bmatrix}$$

$$x_{k+1} = x_k^2 + x_k u_k$$

Nonlinear Control term

A constant B says the command has a fixed additive effect in lifted space.

That can be wrong when command input is not the force the plant actually feels.

Input-structure warning: Iacob, Toth, and Schoukens (2024).

USC Viterbi School of Engineering

Three Modeling Choices

1. Direct input

$$z_k = \psi(x_k)$$

$$z_{k+1} \approx Az_k + Bu_k$$

Simple, convex, but may hide actuator dynamics

2. Input features

$$\eta_k = \phi(u_k)$$

$$z_{k+1} \approx Az_k + B_\eta \eta_k$$

Richer input map, but more complex controller formulation

3. Add actuator state

$$x_{k+1} = f(x_k, a_k)$$

$$a_{k+1} = g(a_k, u_k)$$

$$z_k = \psi(x_k, a_k)$$

$$z_{k+1} \approx Az_k + Bu_k$$

Make the state more complete

Project choice: keep the command input simple, but expose the missing actuator state.

Near-neighbors: Li et al. (2024) input augmentation; Asada and Solano-Castellanos (2024) physical input structure.

USC Viterbi
School of Engineering

Actuator State Belongs In The Lift

Baseline Model

$$z_k^{\text{direct}} = \psi_x(x_k)$$

$$z_{k+1}^{\text{direct}} \approx A_d z_k^{\text{direct}} + B_d u_k$$

Actuator Aware Model

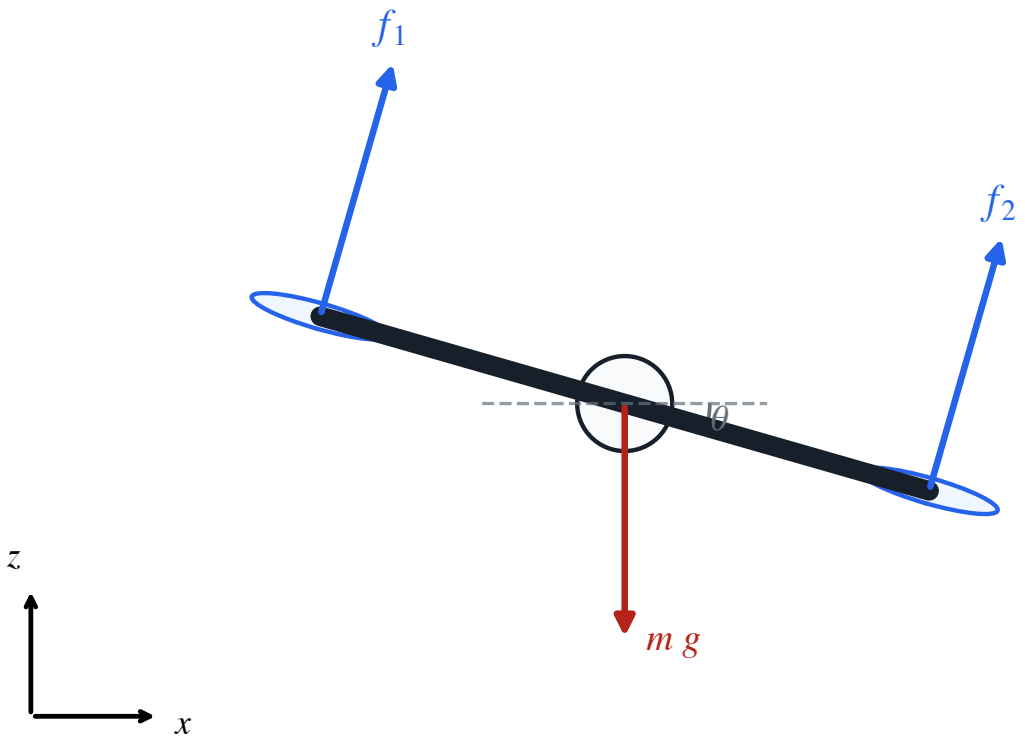
$$x_k^{\text{aug}} = \begin{bmatrix} x_k \\ a_k \end{bmatrix}$$

$$z_k^{\text{act}} = \psi_{xa}(x_k, a_k)$$

$$z_{k+1}^{\text{act}} \approx A_a z_k^{\text{act}} + B_a u_k$$

Does actuator-state lifting improve prediction and control under lag/saturation?

Plant Model



$$x = \begin{bmatrix} p_x & \dot{p}_x & p_z & \dot{p}_z & \theta & \dot{\theta} \end{bmatrix}^T$$

Plant dynamics

$$\ddot{p}_x = \frac{\sin\theta}{m}(f_1 + f_2)$$

$$\ddot{p}_z = \frac{\cos\theta}{m}(f_1 + f_2) - g$$

$$\ddot{\theta} = \frac{d}{I_{yy}}(f_2 - f_1)$$

Actuator dynamics

$$\tau_m \dot{f}_1 = \text{sat}(u_1) - f_1$$

$$\tau_m \dot{f}_2 = \text{sat}(u_2) - f_2$$

$$u_i \longrightarrow \text{sat}(u_i) \longrightarrow f_i$$

USC Viterbi School of Engineering

Model Comparison

Direct Input

$$z_k^{\text{direct}} = \psi_x(x_k)$$

$$z_{k+1}^{\text{direct}} \approx A_d z_k^{\text{direct}} + B_d u_k$$

- Sees rigid body state only
- Commands u directly
- Does not know realized thrust f

Actuator-aware

$$z_k^{\text{act}} = \psi_{xf}(x_k, f_k)$$

$$z_{k+1}^{\text{act}} \approx A_a z_k^{\text{act}} + B_a u_k$$

- Sees rigid body state and realized thrust
- Commands u, but predicts F

USC Viterbi School of Engineering

Model Comparison: Function Basis

Direct Small

$$\psi_s(x_k) = \begin{bmatrix} p_{x,k} \\ \dot{p}_{x,k} \\ p_{z,k} \\ \dot{p}_{z,k} \\ \theta_k \\ \dot{\theta}_k \\ \sin(\theta_k) \\ \cos(\theta_k) \\ \dot{p}_{x,k}^2 \\ \dot{p}_{z,k}^2 \\ \dot{\theta}_k^2 \\ 1 \end{bmatrix} \in \mathbb{R}^{12}$$

Direct Matched

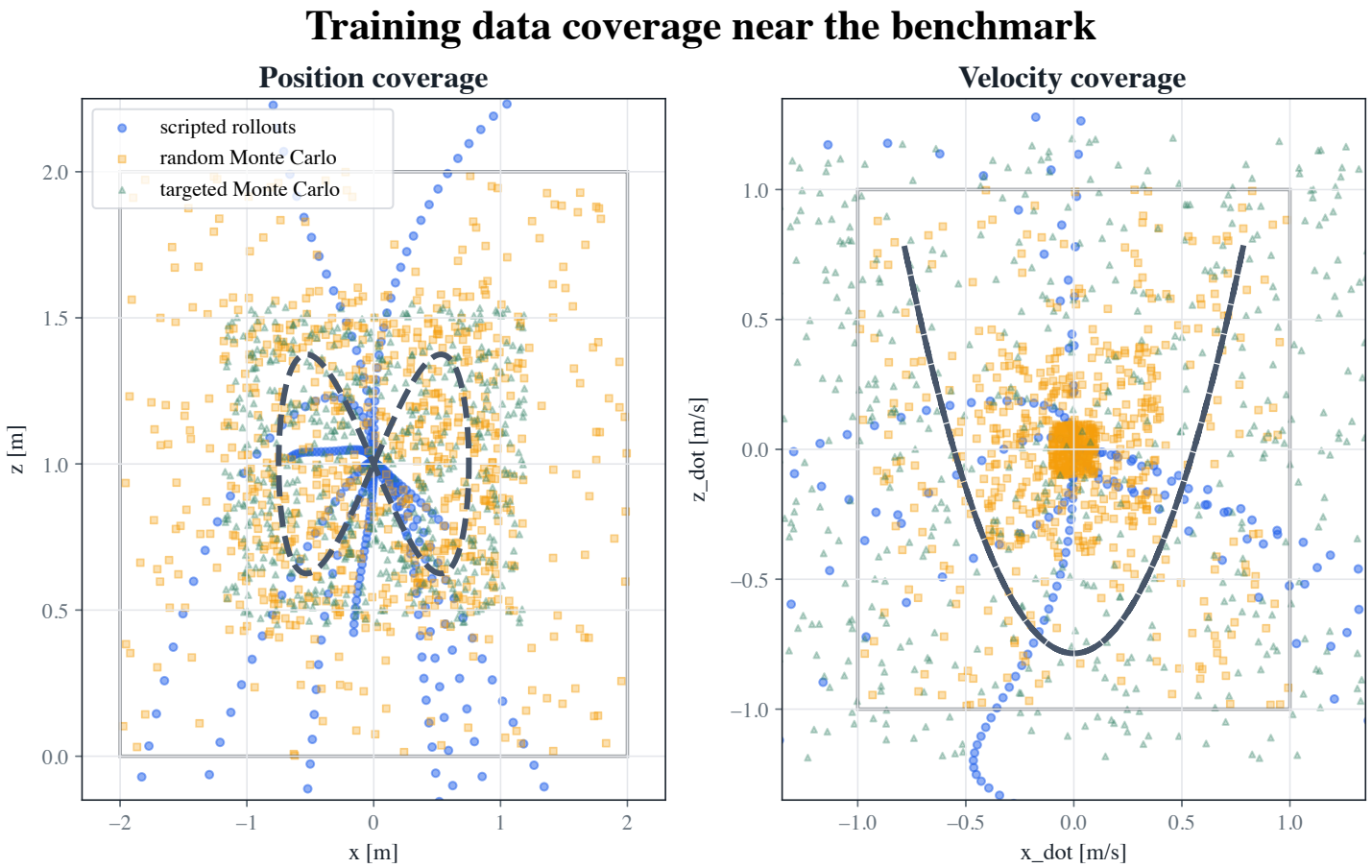
$$\psi_m(x_k) = \begin{bmatrix} p_{x,k} \\ \dot{p}_{x,k} \\ p_{z,k} \\ \dot{p}_{z,k} \\ \theta_k \\ \dot{\theta}_k \\ \sin(\theta_k) \\ \cos(\theta_k) \\ \dot{p}_{x,k}^2 \\ \dot{p}_{z,k}^2 \\ \dot{\theta}_k^2 \\ p_{x,k}^2 \\ p_{z,k}^2 \\ \theta_k^2 \\ p_{x,k}\theta_k \\ p_{z,k}\theta_k \\ \dot{p}_{x,k}\dot{\theta}_k \\ \dot{p}_{z,k}\dot{\theta}_k \\ 1 \end{bmatrix} \in \mathbb{R}^{19}$$

Actuator-aware

$$\psi_a(x_k, f_k) = \begin{bmatrix} p_{x,k} \\ \dot{p}_{x,k} \\ p_{z,k} \\ \dot{p}_{z,k} \\ \theta_k \\ \dot{\theta}_k \\ f_{1,k} \\ f_{2,k} \\ \sin(\theta_k) \\ \cos(\theta_k) \\ \dot{p}_{x,k}^2 \\ \dot{p}_{z,k}^2 \\ \dot{\theta}_k^2 \\ f_{1,k} + f_{2,k} \\ f_{2,k} - f_{1,k} \\ (f_{1,k} + f_{2,k})\sin(\theta_k) \\ (f_{1,k} + f_{2,k})\cos(\theta_k) \\ f_{1,k} f_{2,k} \\ 1 \end{bmatrix} \in \mathbb{R}^{19}$$

USC Viterbi
School of Engineering

Data Collection



"Random" Open-loop commands: hover steps, smooth sines, attitude pulses, near saturation.

Monte Carlo one-step samples cover the state/input box.

10,000 targeted figure-eight samples improve task-region accuracy.

USC Viterbi School of Engineering

MPC Controller

$$\min_{u_0,\ldots,u_{N-1}} \sum_{j=0}^{N} \left\| Cz_j - r_j \right\|_Q^2 + \sum_{j=0}^{N-1} \left\| u_j - u_{\text{hover}} \right\|_R^2$$

With

$$z_{j+1} = Az_j + Bu_j \qquad u_{\min} \le u_j \le u_{\max}$$

$$C_{\text{aug}} = \begin{bmatrix} I_8 & 0_{8\times 11} \end{bmatrix} \qquad z_k = \psi(x_k, f_k) = \begin{bmatrix} p_{x,k} \\ \dot{p}_{x,k} \\ p_{z,k} \\ \dot{p}_{z,k} \\ \theta_k \\ \dot{\theta}_k \\ f_{1,k} \\ f_{2,k} \\ \sin(\theta_k) \\ \cos(\theta_k) \\ \dot{p}_{x,k}^2 \\ \dot{p}_{z,k}^2 \\ \dot{\theta}_k^2 \\ f_{1,k} + f_{2,k} \\ f_{2,k} - f_{1,k} \\ (f_{1,k} + f_{2,k})\sin(\theta_k) \\ (f_{1,k} + f_{2,k})\cos(\theta_k) \\ f_{1,k} f_{2,k} \\ 1 \end{bmatrix} \in \mathbb{R}^{19}$$

$$Q = \text{diag}\,(20,\ 0.1,\ 20,\ 0.1,\ 0.01,\ 0.01)$$

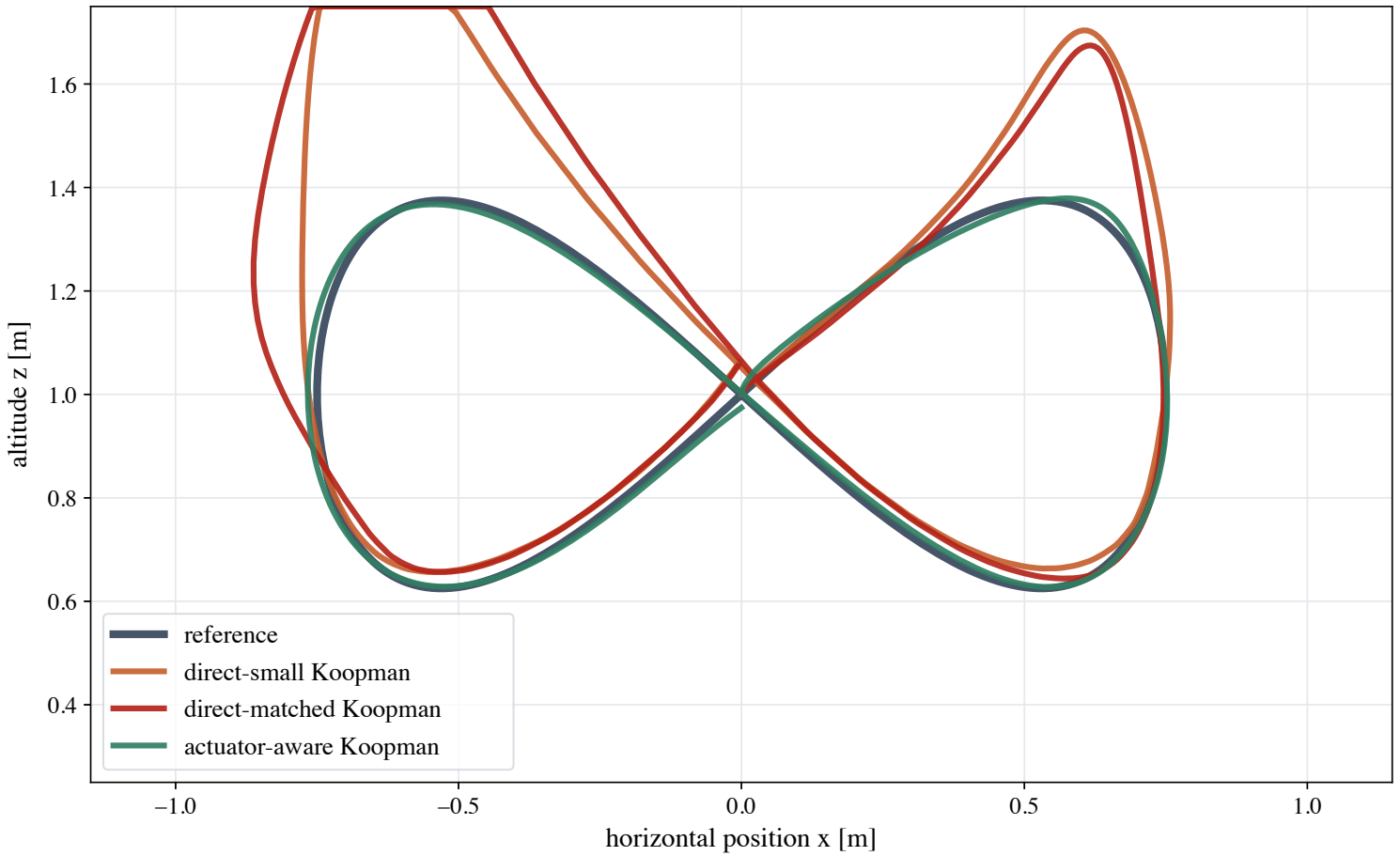
$$R = \text{diag}\,(0.03,\ 0.03)$$

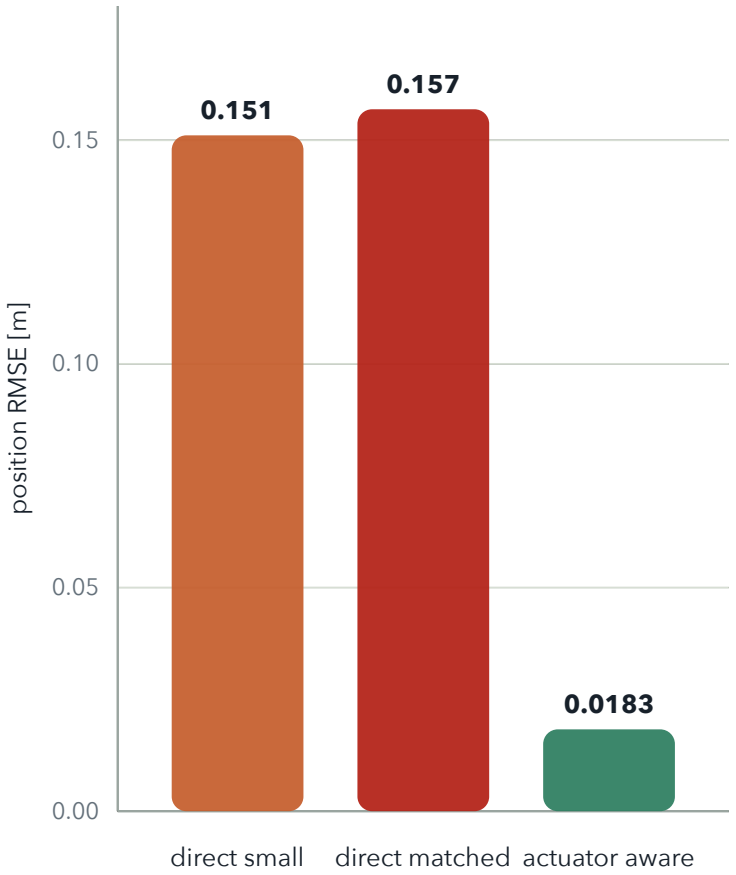
$$N = 15$$

USC Viterbi School of Engineering

Tracking Results

Figure 8 tracking with tau 0.02 seconds

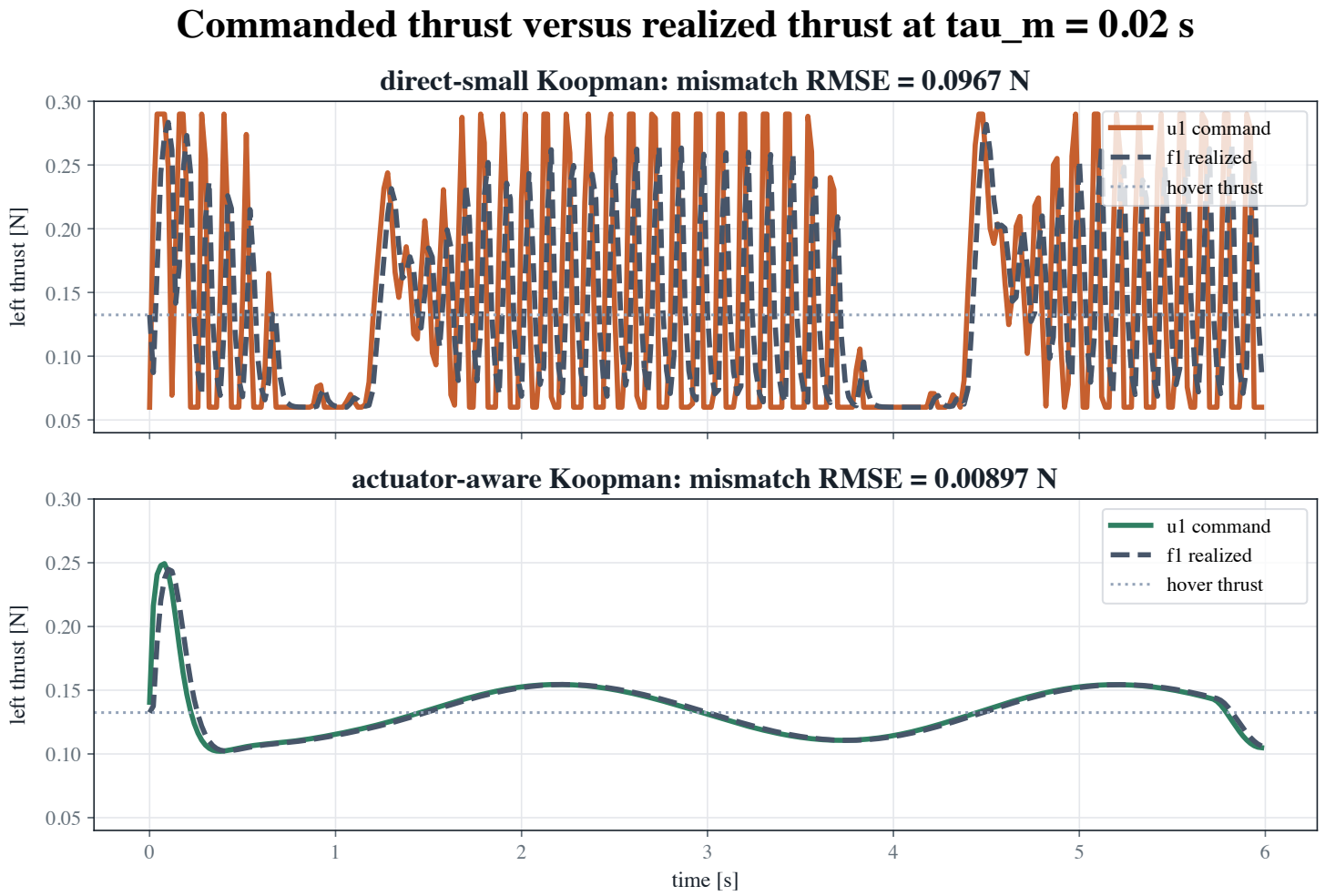




$$\tau_m = 0.02 \text{ s}$$

USC Viterbi
School of Engineering

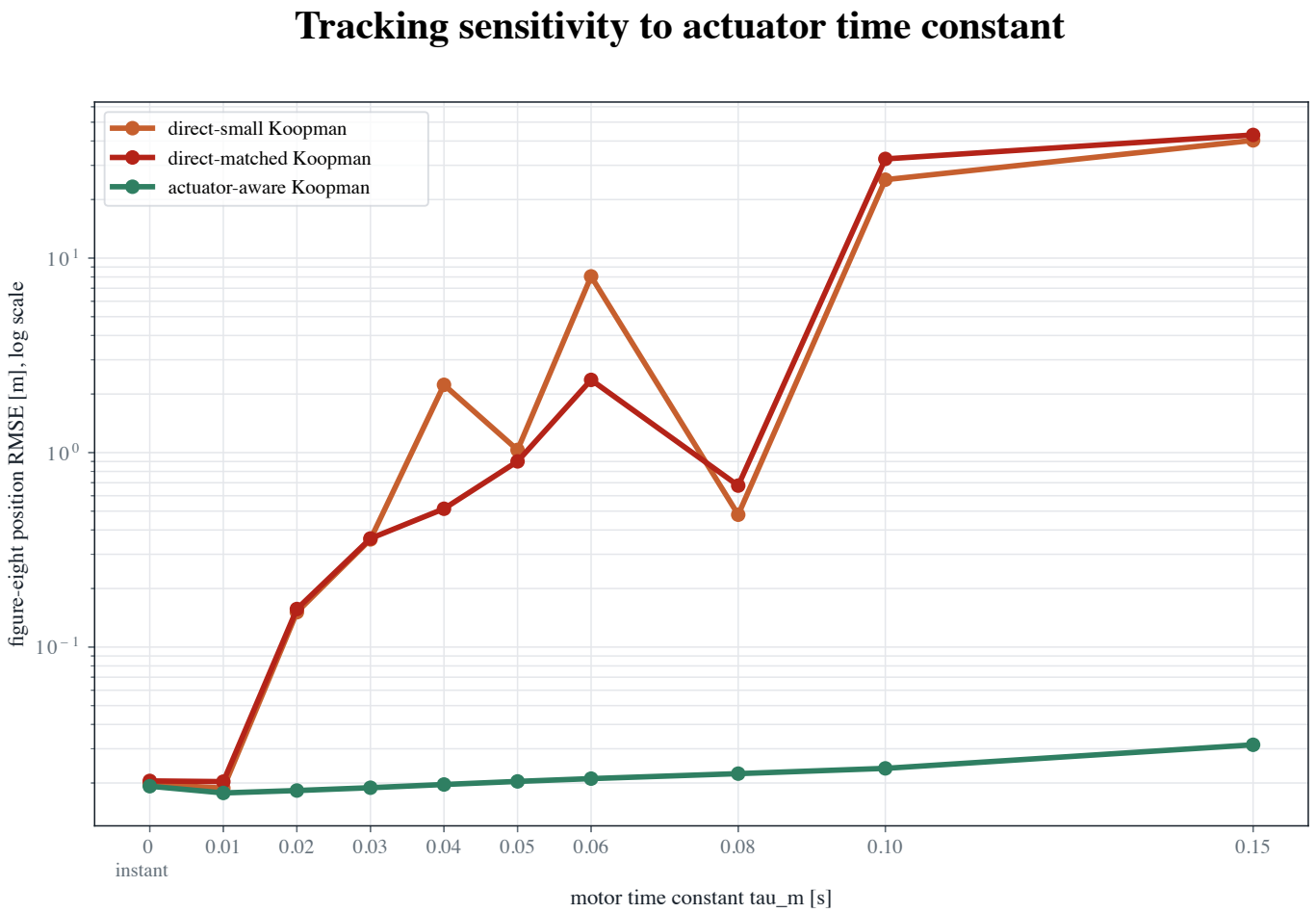
Actuator States

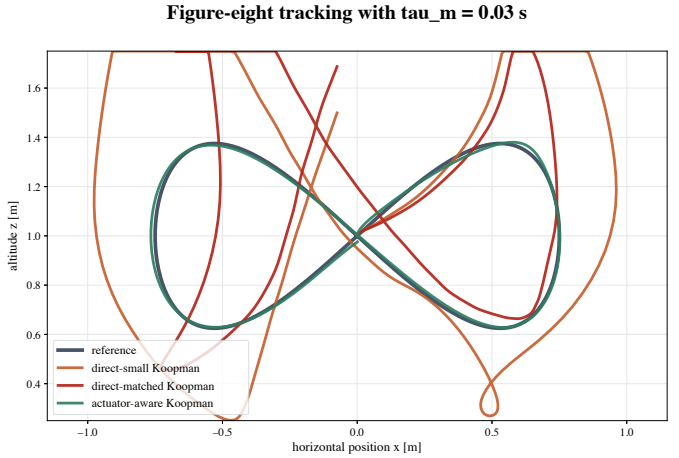


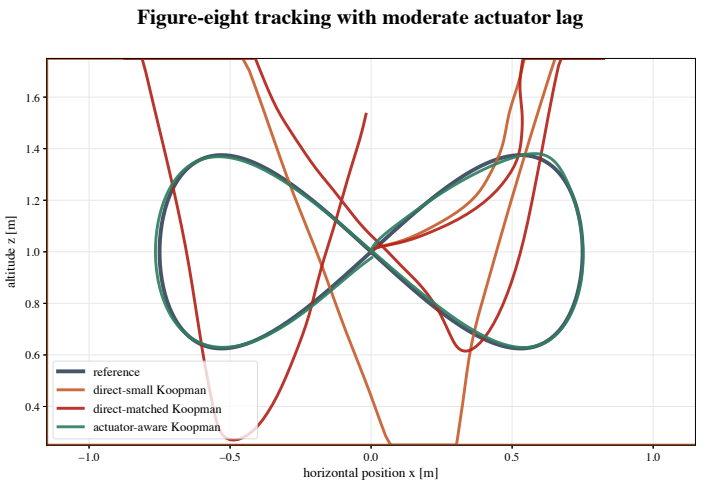
The actuator-aware model **predicts the realized thrust** state, so MPC can plan around the lag instead of fighting it

USC Viterbi School of Engineering

Lag Sensitivity



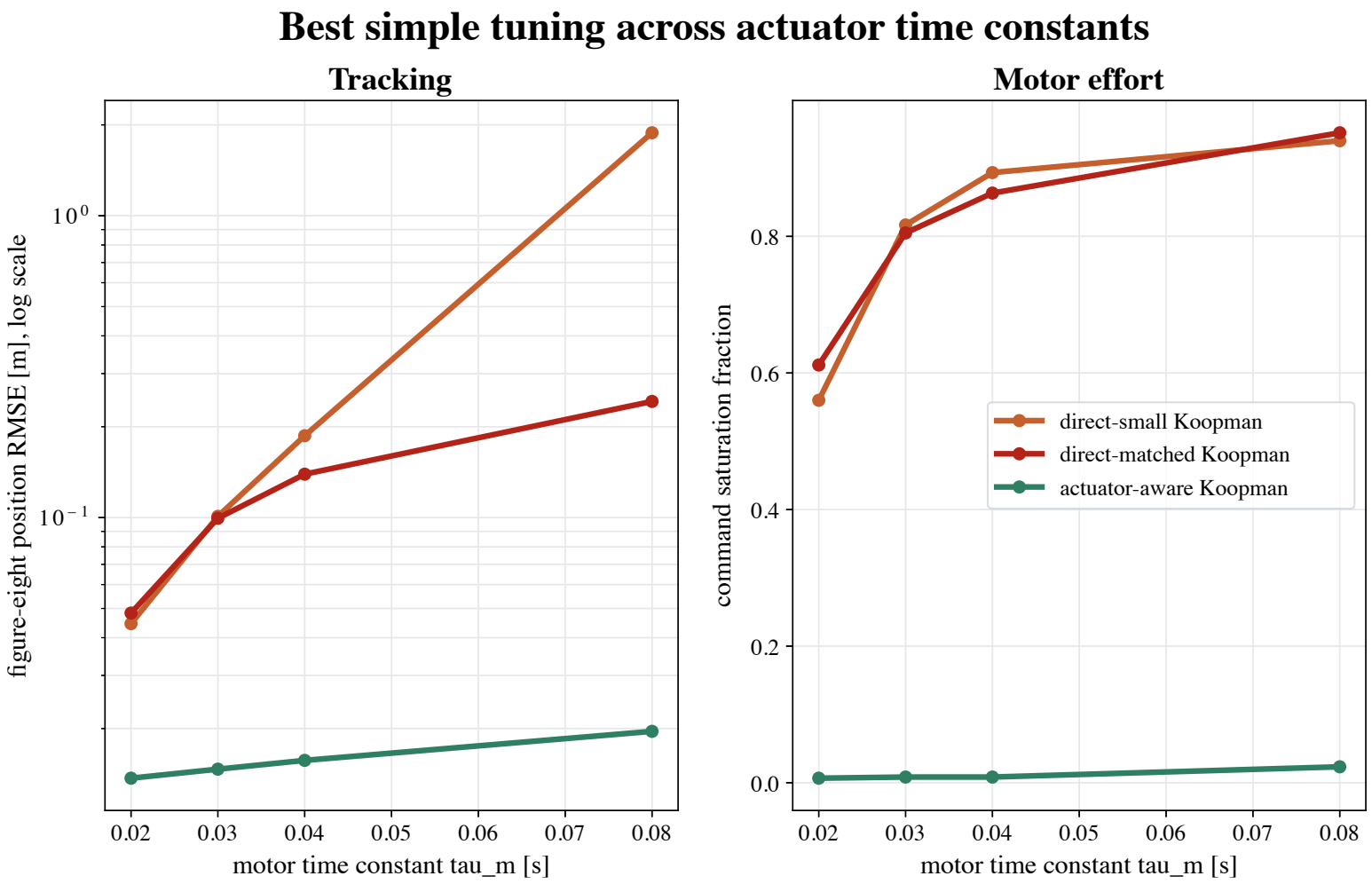




When an actuator lag is very small, direct input approximation is less harmful, but as the actuator lag grows, modeling the actuator becomes exponentially more important.

USC Viterbi
School of Engineering

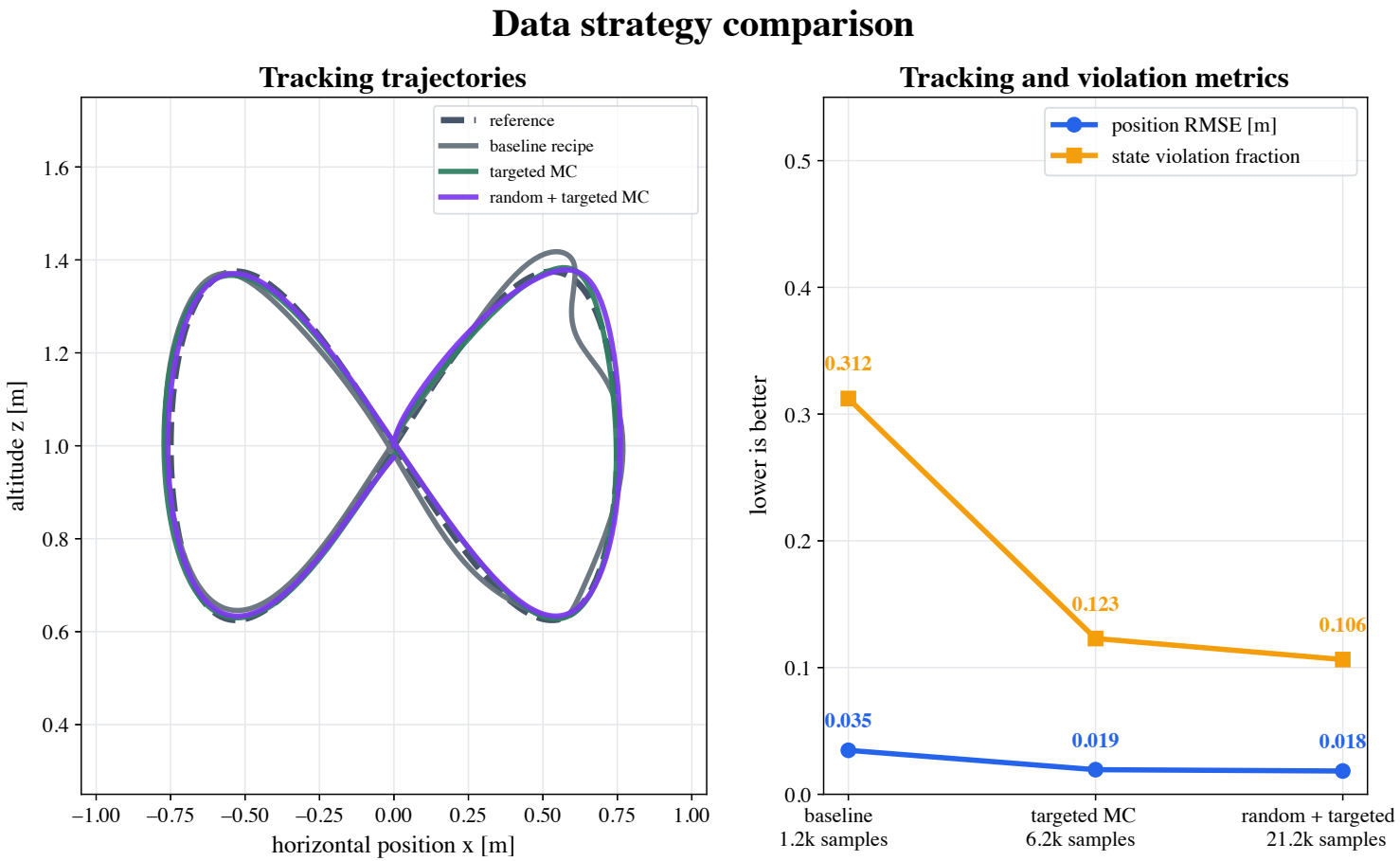
Lag Sensitivity



Using tuning, we can make the larger basis non-actuator where it performs slightly better.
More basis functions seems to help.

USC Viterbi School of Engineering

Data Matters



The same Koopman-MPC architecture changes behavior depending on what data the least-squares fit sees.

- More data is better
- Data closer to your working region is better

USC Viterbi School of Engineering

Refrences

Otto, S. E., & Rowley, C. W. (2021). Koopman operators for estimation and control of dynamical systems. Annual Review of Control, Robotics, and Autonomous Systems. https://doi.org/10.1146/annurev-control-071020-010108

Korda, M., & Mezić, I. (2018). Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. Automatica. https://doi.org/10.1016/j.automatica.2018.03.046

Iacob, L. C., Tóth, R., & Schoukens, M. (2024). On the Koopman form of nonlinear systems with inputs. Automatica. https://doi.org/10.1016/j.automatica.2024.111525

Li, Z., Han, M., Vo, D.-N., & Yin, X. (2024). Machine learning-based input-augmented Koopman modeling and predictive control of nonlinear processes. Computers & Chemical Engineering. https://doi.org/10.1016/j.compchemeng.2024.108854

Asada, H. H., & Solano-Castellanos, J. A. (2024). Control-Coherent Koopman Modeling: A Physical Modeling Approach. IEEE Conference on Decision and Control.

Yuan, Z., Hall, A. W., Zhou, S., Brunke, L., Greeff, M., Panerati, J., & Schoellig, A. P. (2022). Safe-Control-Gym: A unified benchmark suite for safe learning-based control and reinforcement learning in robotics. IEEE Robotics and Automation Letters. https://doi.org/10.1109/LRA.2022.3196132

Eschmann, J., Albani, D., & Loianno, G. (2024). Data-driven system identification of quadrotors subject to motor delays. IEEE/RSJ International Conference on Intelligent Robots and Systems. https://doi.org/10.1109/IROS58592.2024.10801441