

# Closing the loop between optimal nonlinear control and learning-based optimization

Luca Furieri



**EPFL**



# Neural network control

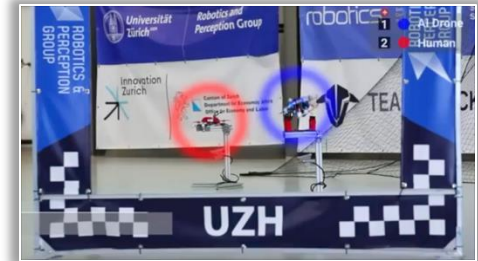
## Success stories in robotics



[Kalashnikov *et al.*, '18]



[Youssef *et al.*, '20]

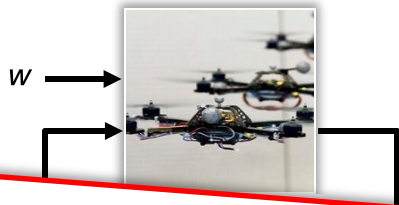


[Kaufmann *et al.*, '23]

- Flexibility of NN controllers, optimization of complex objective functions

# Two challenges

1. Certify closed-loop stability during the learning?

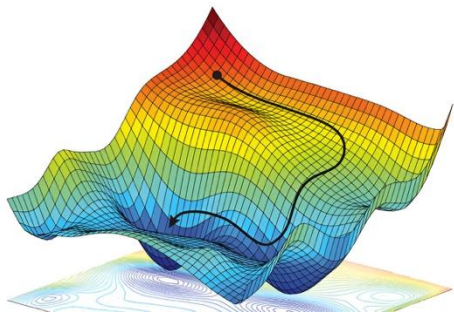


**Goal:** stability for all choices of  $\theta$

**Unified methodology**

From designing **stabilizing** controllers... to designing **stable** closed-loop maps

2. Navigate the highly nonconvex optimization landscape



**Goal:** converge, fast, to good (local) solution

# Part 1

## Stable NN closed-loop maps for nonlinear optimal control

[1] Furieri L., Galimberti C., Ferrari-Trecate G., “*Neural system level synthesis: learning over all stabilizing policies for nonlinear systems*”, [CDC 2022]

[2] Furieri L., Galimberti C., Ferrari-Trecate G., “*Learning to boost the performance of stable nonlinear systems*”, [OJ-CSYS, 2024]

# Common scenario in engineering

- ... frequent availability of stabilizing controllers around equilibrium or a reference

... however, stability is not enough

The image compares two types of robots. On the left is a modular 'origami' robot from EPFL, which is a small, multi-legged robot with a purple body and thin legs. It is shown in a close-up view. On the right is an all-terrain legged robot from the University of Oxford, which is a larger, more complex robot with a black body and thick legs. It is shown in a wider view, crossing a 25cm barrier. A red arrow points from the text '... however, stability is not enough' to the comparison. A red box at the bottom contains the text 'Boost performance without compromising stability?'. A diagram labeled 'Distributed PID control' shows three triangles connected by arrows, representing a control system. The text '8x speed' is written in red next to the origami robot, and '>4x speed' is written in red next to the all-terrain robot.

Modular "origami" robot<sup>[1]</sup> EPFL

8x speed

All-terrain legged-robots<sup>[2]</sup> UNIVERSITY OF OXFORD

>4x speed

CROSSING 25CM BARRIERS BY INVERTING ITS LEGS

ORI

Distributed PID control

Boost performance without compromising stability?

[1] Wisth, Camurri, Fallon, "VILENS: Visual, inertial, lidar, and leg odometry for all-terrain legged robots." IEEE Transactions on Robotics, 2022

[2] Belke, Holdcroft, Sigrist, Paik, "Morphological flexibility in robotic systems through physical polygon meshing." Nature Machine Intelligence, 2023

# Performance boosting

## System

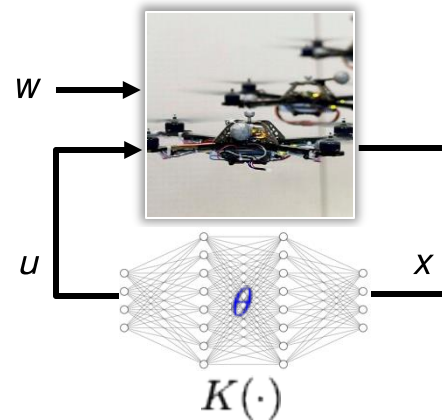
- Nonlinear, **stable/pre-stabilized**

## Performance-boosting controller

- **Stability-preserving**, optimizing complex costs
- Performance = task execution, safety, robustness, ...

## Goals

- Leverage NNs flexibility
- Harness open-loop stability for control design



## Nonlinear Optimal Control

$$K(\cdot) \in \operatorname{argmin} \frac{1}{T} \mathbb{E}_w [\mathcal{L}(x_{0:T}, u_{0:T})]$$

s.t. **CLOSED-LOOP STABILITY**

# Setup and notation

**Time-varying, nonlinear, controlled system**

$$\begin{cases} x_t = f_t(x_{t-1}, u_{t-1}) + w_t \\ u_t = K_t(x_{t:0}) \end{cases}$$

# Setup and notation

Time-varying, nonlinear, controlled system

$$\begin{cases} x_t = f_t(x_{t-1}, u_{t-1}) + w_t \\ u_t = K_t(x_{t:0}) \end{cases}$$

Dynamic controller

Process noise

# Setup and notation

## Time-varying, nonlinear, controlled system

$$\begin{cases} x_t = f_t(x_{t-1}, u_{t-1}) + w_t \\ u_t = K_t(x_{t:0}) \end{cases}$$

$$\mathbf{K}(\mathbf{x}) = (K_0(x_0), K_1(x_{1:0}), \dots)$$

$\xrightarrow{\mathbf{x} = (x_0, x_1, \dots)}$

## Operator model

$$\begin{cases} \mathbf{x} = \mathbf{F}(\mathbf{x}, \mathbf{u}) + \mathbf{w} \\ \mathbf{u} = \mathbf{K}(\mathbf{x}) \end{cases}$$

# Setup and notation

## Time-varying, nonlinear, controlled system

$$\begin{cases} x_t = f_t(x_{t-1}, u_{t-1}) + w_t \\ u_t = K_t(x_{t:0}) \end{cases} \quad \begin{array}{c} \mathbf{K}(\mathbf{x}) = (K_0(x_0), K_1(x_{1:0}), \dots) \\ \xrightarrow{\mathbf{x} = (x_0, x_1, \dots)} \end{array}$$

## Operator model

$$\begin{cases} \mathbf{x} = \mathbf{F}(\mathbf{x}, \mathbf{u}) + \mathbf{w} \\ \mathbf{u} = \mathbf{K}(\mathbf{x}) \end{cases}$$

LTI system:  $x_t = Ax_{t-1} + Bu_{t-1} + w_t$

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & \dots \\ A & 0 & 0 & \dots \\ 0 & A & 0 & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & \dots \\ B & 0 & 0 & \dots \\ 0 & B & 0 & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ \vdots \end{bmatrix} + \begin{bmatrix} x_0 \\ w_1 \\ w_2 \\ \vdots \end{bmatrix}$$

# Setup and notation

## Time-varying, nonlinear, controlled system

$$\begin{cases} x_t = f_t(x_{t-1}, u_{t-1}) + w_t \\ u_t = K_t(x_{t:0}) \end{cases} \quad \begin{array}{l} \mathbf{K}(\mathbf{x}) = (K_0(x_0), K_1(x_{1:0}), \dots) \\ \xrightarrow{\mathbf{x} = (x_0, x_1, \dots)} \end{array}$$

## Operator model

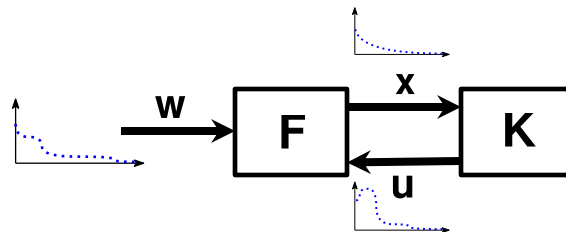
$$\begin{cases} \mathbf{x} = \mathbf{F}(\mathbf{x}, \mathbf{u}) + \mathbf{w} \\ \mathbf{u} = \mathbf{K}(\mathbf{x}) \end{cases}$$

## Stability in the $\mathcal{L}_2$ sense

- **A is a stable operator** if it is causal and  $\mathbf{A}(\mathbf{x}) \in \ell_2, \forall \mathbf{x} \in \ell_2$ 
  - For short:  $\mathbf{A} \in \mathcal{L}_2$

← ---  $\mathbf{x} \in \ell_2$  if  $\sum_{t=0}^{\infty} \|x_t\|^2 < \infty$

- **Stabilizing controller:** the closed-loop maps  $\mathbf{w} \rightarrow \mathbf{x}$  and  $\mathbf{w} \rightarrow \mathbf{u}$  are stable operators



# Setup and notation

## Time-varying, nonlinear, controlled system

$$\begin{cases} x_t = f_t(x_{t-1}, u_{t-1}) + w_t \\ u_t = K_t(x_{t:0}) \end{cases} \quad \begin{array}{c} \mathbf{K}(\mathbf{x}) = (K_0(x_0), K_1(x_{1:0}), \dots) \\ \xrightarrow{\mathbf{x} = (x_0, x_1, \dots)} \end{array}$$

## Operator model

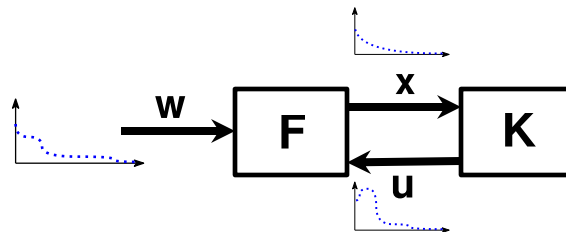
$$\begin{cases} \mathbf{x} = \mathbf{F}(\mathbf{x}, \mathbf{u}) + \mathbf{w} \\ \mathbf{u} = \mathbf{K}(\mathbf{x}) \end{cases}$$

## Stability in the $\mathcal{L}_2$ sense

- **A is a stable operator** if it is causal and  $\mathbf{A}(\mathbf{x}) \in \ell_2, \forall \mathbf{x} \in \ell_2$ 
  - For short:  $\mathbf{A} \in \mathcal{L}_2$

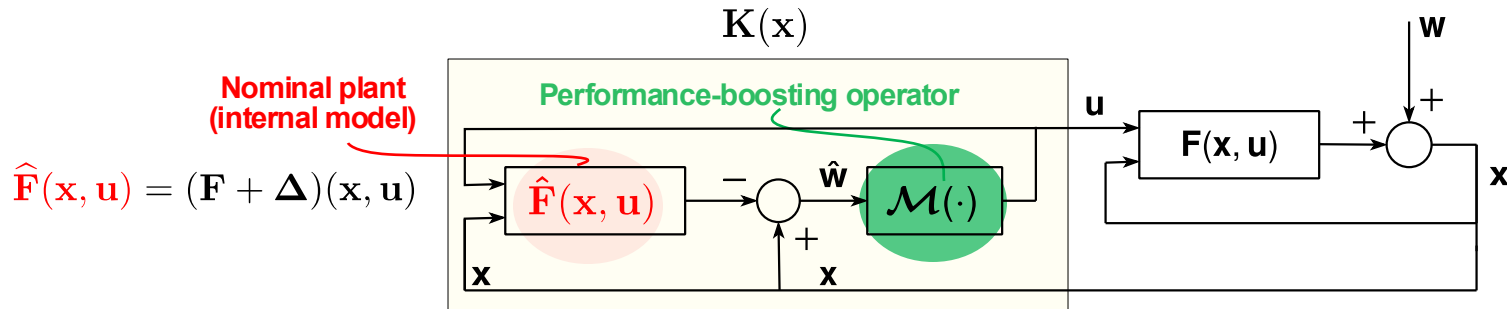
← ---  $\mathbf{x} \in \ell_2$  if  $\sum_{t=0}^{\infty} \|x_t\|^2 < \infty$

- **Stabilizing controller:** the closed-loop maps  $\mathbf{w} \rightarrow \mathbf{x}$  and  $\mathbf{w} \rightarrow \mathbf{u}$  are stable operators



**Assumption:** the open-loop plant  $\mathbf{x} = \mathcal{F}(\mathbf{u}, \mathbf{w})$  is a stable operator

# Parametrization of all nonlinear stabilizing controllers



- Sufficiency:** the controller  $K(x)$  as above stabilizes the real system  $F(x, u)$  if

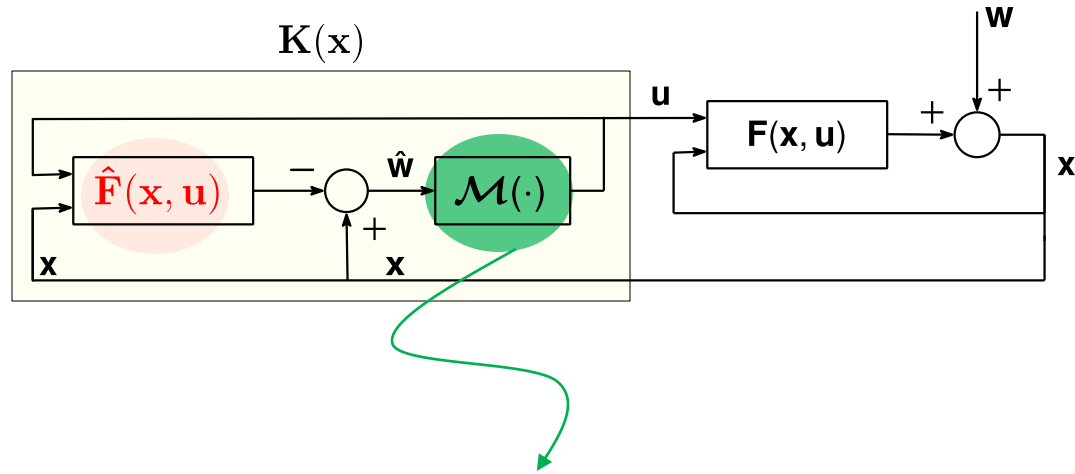
$$\mathcal{M} \in \mathcal{L}_2 \quad \text{and} \quad \text{gain}(\mathcal{M}) < \frac{1}{\text{gain}(\Delta)(\text{gain}(\mathcal{F}) + 1)}$$

- Necessity:** if the real model is known ( $\Delta = 0$ ), then  $\mathcal{M}$  is the closed-loop map  $w \rightarrow u$   
 → Therefore, any stable closed-loop behavior can be obtained by selecting  $\mathcal{M} \in \mathcal{L}_2$

[1] L. Furieri, C. L. Galimberti, and GFT, "Neural System Level Synthesis: Learning over All Stabilizing Policies for Nonlinear Systems," IEEE CDC 2022

[2] L. Furieri, C. L. Galimberti, and GFT, "Learning to Boost the Performance of Stable Nonlinear Systems," OJ-CSYS 2024

## Next question...



How to implement *neural-network* stable operators?

# Models of stable operators

Finite-dimensional parametrization of  $\mathcal{M}^\theta \in \mathcal{L}_2$

- Linear operators  $\mathcal{M}^\theta = \sum_{h=0}^N \frac{M_h}{z^h}$  (Finite Impulse Response models)
- *Nonlinear operators?*

# Models of stable operators

Finite-dimensional parametrization of  $\mathcal{M}^\theta \in \mathcal{L}_2$

- Linear operators  $\mathcal{M}^\theta = \sum_{h=0}^N \frac{M_h}{z^h}$  (Finite Impulse Response models)
- Nonlinear operators?*

EFFICIENTLY MODELING LONG SEQUENCES WITH  
STRUCTURED STATE SPACES

Albert Gu & Karan Goel & Christopher Ré  
Department of Computer Science, Stanford University  
{albertgu, krng}@stanford.edu, chrismre@cs.stanford.edu

'22

Standard representation and unified stability analysis for dynamic  
artificial neural network models<sup>☆</sup>

Kwang-Ki K. Kim<sup>a,\*</sup>, Ernesto Ríos Patrón<sup>b</sup>, Richard D. Braatz<sup>c</sup>

<sup>a</sup> Department of Electrical Engineering, Inha University, Incheon, Republic of Korea

<sup>b</sup> Petroleum Inst of Mexico, Mexico City, Mexico

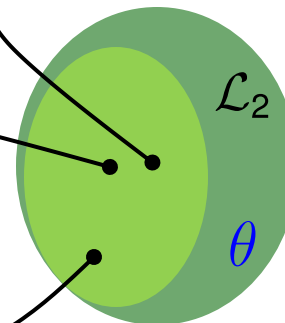
<sup>c</sup> Massachusetts Institute of Technology, Cambridge, MA, United States

'18, '23

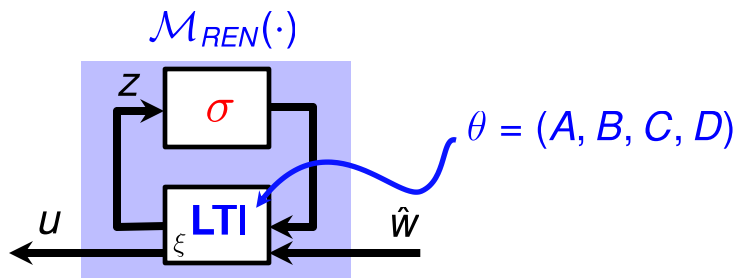
Recurrent Equilibrium Networks:  
Flexible Dynamic Models with Guaranteed Stability  
and Robustness

Max Revay, Ruigang Wang, Ian R. Manchester

'21, '23



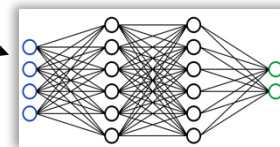
# Recurrent Equilibrium Networks (RENs)<sup>[1,2]</sup>



- Expressive models including

$$\xi_t = \hat{A}\xi_{t-1} + \hat{B} \text{NN}^\xi(\xi_{t-1}, \hat{w}_t)$$

$$u_t = \hat{C}\xi_t + \hat{D} \text{NN}^u(\xi_{t-1}, \hat{w}_t)$$



- $\mathcal{M}_{REN} \in \mathcal{L}_2$  if there is a storage function  $V(\xi) = \xi^T P \xi$  verifying

$$V(\xi_{t+1}) - V(\xi_t) \leq \gamma^2 \|\hat{w}_t\| - \|u_t\|$$

- Free parametrization**<sup>[2]</sup>: explicit map  $\Theta \mapsto (\theta, P)$  such that  $\mathcal{M}_{REN} \in \mathcal{L}_2$  for any  $\Theta \in \mathbb{R}^d$   
 – **Limitation**: contractive models

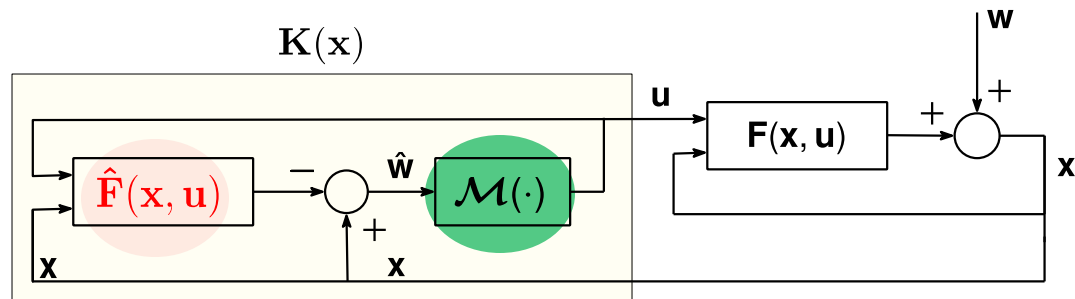
[1] Kim, K. K., E. Ríos Patrón, and R. D. Braatz. "Standard representation and unified stability analysis for dynamic artificial neural network models." *Neural Networks* 2018

[2] Revay, M, R. Wang, and I.R. Manchester. "Recurrent equilibrium networks: Flexible dynamic models with guaranteed stability and robustness." *IEEE TAC* 2023

# Deep learning formulation

$$K(\cdot) \in \operatorname{argmin} \frac{1}{T} \mathbf{E}_w [\mathcal{L}(x_{0:T}, u_{0:T})]$$

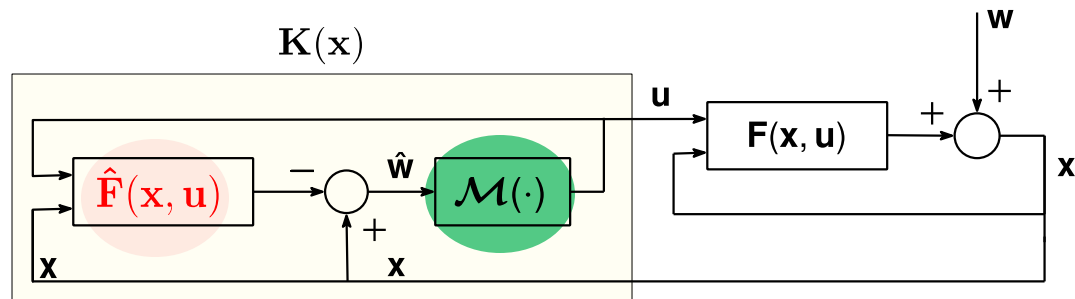
s. t. **CLOSED-LOOP STABILITY**



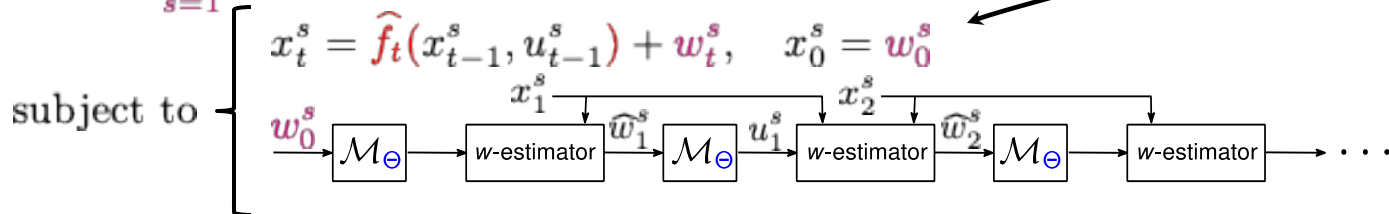
# Deep learning formulation

$$K(\cdot) \in \operatorname{argmin} \frac{1}{T} \mathbb{E}_w [\mathcal{L}(x_{0:T}, u_{0:T})]$$

s. t. **CLOSED-LOOP STABILITY**



$$\min_{\Theta \in \mathbb{R}^d} \frac{1}{S} \sum_{s=1}^S \mathcal{L}(x_{0:T}^s, u_{0:T}^s)$$

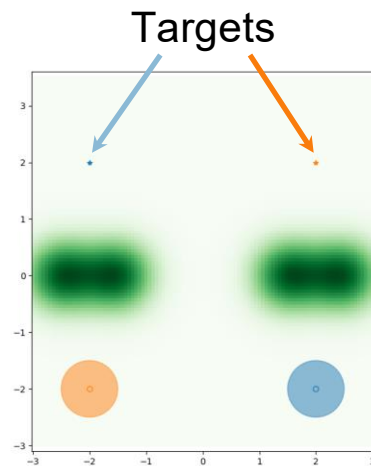


- Free parametrization of  $\mathcal{M}$   $\rightarrow$  unconstrained optimization  $\rightarrow$  backprop
- CL stability guaranteed even if optimization stops early



# Numerical example: the corridor problem

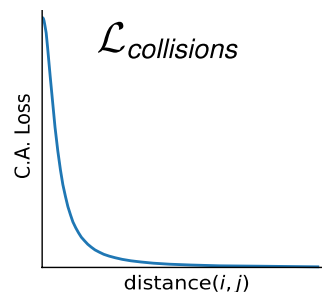
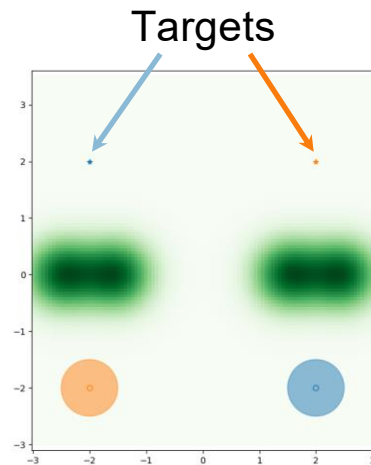
- 2 robots: point-mass dynamics, nonlinear drag
- **Goal:** CL stability on targets, avoid collisions & obstacles



# Numerical example: the corridor problem

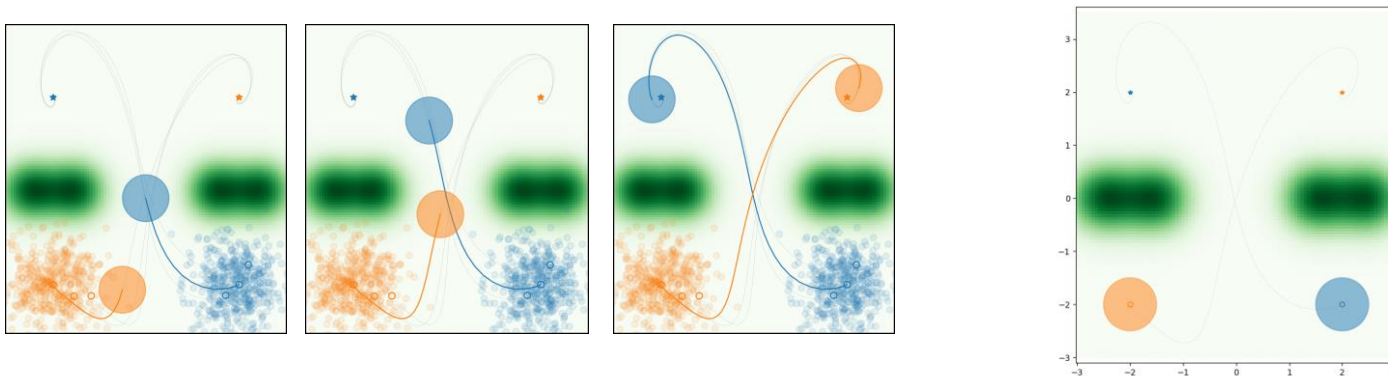
- 2 robots: point-mass dynamics, nonlinear drag
- **Goal:** CL stability on targets, avoid collisions & obstacles
- **Separation of concerns:**
  1. Design a simple stabilizing base controller
    - Linear spring at rest on target (overshoot, collisions....)
  2. Performance-boosting controller minimizing

$$\mathcal{L}(\cdot) = \mathcal{L}_{target}(\cdot) + \mathcal{L}_{collisions}(\cdot) + \mathcal{L}_{obstacles}(\cdot)$$

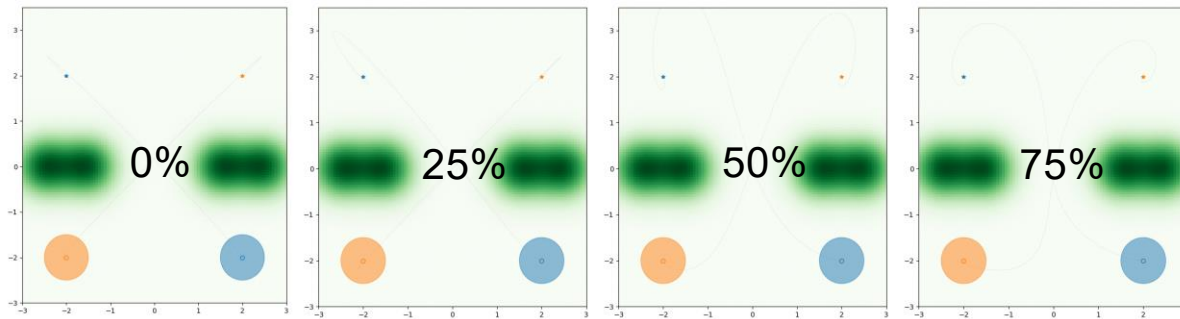


# Numerical example: the corridor problem

- Upon training over a dataset 500 different initial conditions



- CL stability guaranteed even with early stopping of training



# The power of the cost: lessons from RL

- «Reward» shaping does the magic in RL
- Our result: *decoupling* reward from stability

## Performance-boosting problem

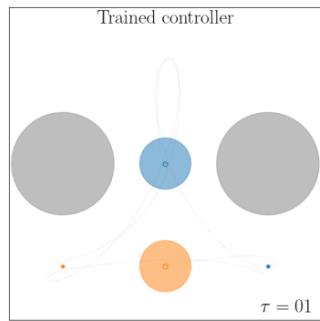
$$\min_{\Theta \in \mathbb{R}^d} \frac{1}{S} \sum_{s=1}^S \mathcal{L}(x_{T:0}^s, u_{T:0}^s)$$

$$\text{s. t. } x_t^s = f_t(x_{t-1}^s, u_{t-1}^s) + w_t^s, \quad x_0^s = w_0^s$$

$$u_t^s = \mathcal{M}_t^\Theta(x_t^s - f_t(x_{t-1}^s, u_{t-1}^s))$$

# The power of the cost: lessons from RL

## Waypoint tracking



- «Reward» shaping does the magic in RL
- Our result: *decoupling* reward from stability

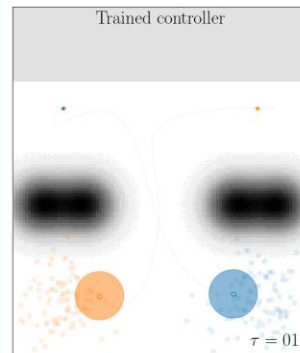
## Performance-boosting problem

$$\min_{\Theta \in \mathbb{R}^d} \frac{1}{S} \sum_{s=1}^S [\mathcal{L}(x_{T:0}^s, u_{T:0}^s)]$$

$$\text{s. t. } x_t^s = f_t(x_{t-1}^s, u_{t-1}^s) + w_t^s, \quad x_0^s = w_0^s$$

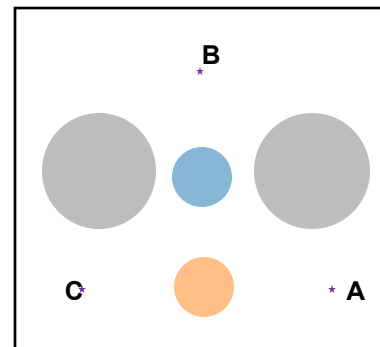
$$u_t^s = \mathcal{M}_t^\Theta (x_t^s - f_t(x_{t-1}^s, u_{t-1}^s))$$

## Safety via invariance



# Waypoints tracking

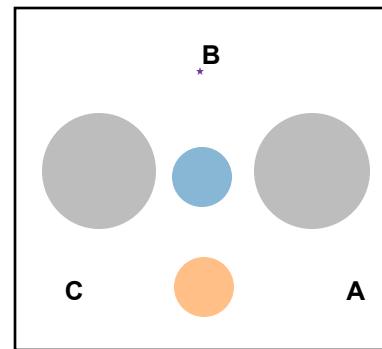
- **Task specs:**
  - No collisions
  - **Blue robot:**  $A \rightarrow B \rightarrow C$ , stabilizing around  $C$
  - **Orange robot:**  $C \rightarrow A \rightarrow B$ , stabilizing around  $B$



# Numerical example: waypoints tracking

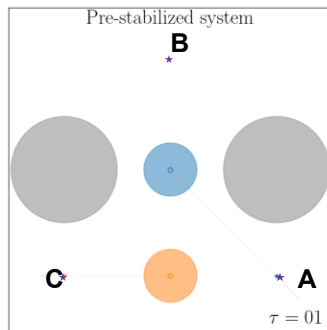
- **Task specs:**

- No collisions
- **Blue robot:**  $B \rightarrow C \rightarrow A$ , stabilizing around  $A$
- **Orange robot:**  $A \rightarrow B \rightarrow C$ , stabilizing around  $C$

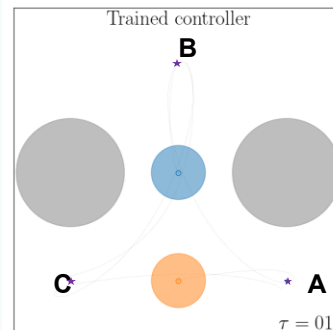


- Waypoints  $\rightarrow$  **Linear Temporal Logic** formulae<sup>[1]</sup>  $\rightarrow$  cost  $\mathcal{L}_{way}$

## Base controller

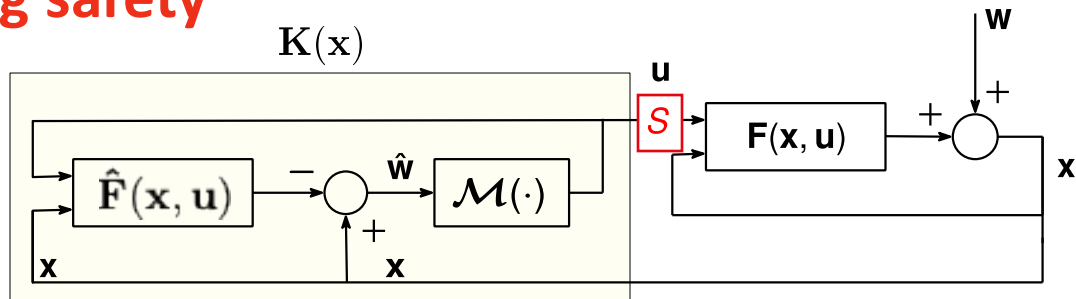


## Performance boosting



[1] Li, X., C.-I. Vasile, and C. Belta. "Reinforcement learning with temporal logic rewards.", IEEE IROS, 2017

# Embedding safety

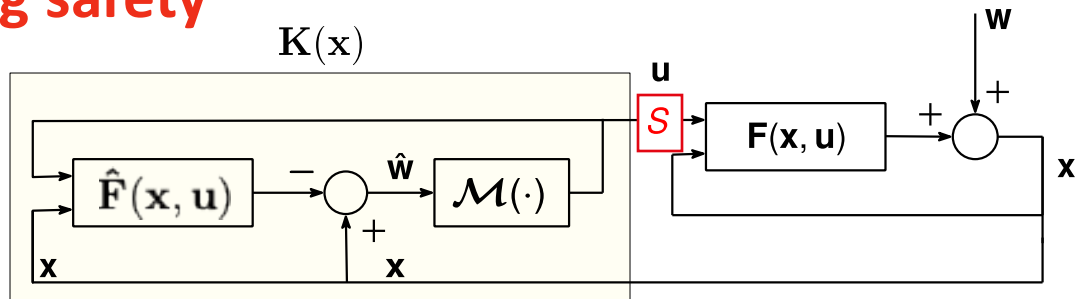


- Add a **safety filter**<sup>[1]</sup> guaranteeing  $(x_t, u_t) \in \mathcal{C}, \forall t > 0$ 
  - Requires online optimization
  - Tweaks  $u$  only if needed

[1] Hewing, L., *et al.* "Learning-based model predictive control: Toward safe learning in control." Annual Review of Control, Robotics, and Autonomous Systems, 2020

[2] Agrawal, A., and K. Sreenath. "Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation." *Robotics: Science and Systems*. 2017

# Embedding safety



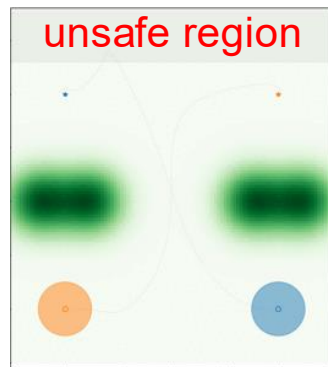
- Add a **safety filter**<sup>[1]</sup> guaranteeing  $(x_t, u_t) \in \mathcal{C}, \forall t > 0$ 
  - Requires online optimization
  - Tweaks  $u$  only if needed
- **Reduce filter activation** embedding **soft safety specs** in the cost
  - Promote constraint fulfillment  $\rightarrow \mathcal{L}_{safe} = \max_{t < T} \text{Barrier}_{\mathcal{C}}(x_t, u_t)$
  - Promote invariance<sup>[2]</sup> of  $\mathcal{X} = \{x : h(x) \leq 0\}$

$$\mathcal{L}_{inv} = \max_{t < T} \text{ReLU}(h(x_t) - h(x_{t+1}) - \gamma h(x_t))$$

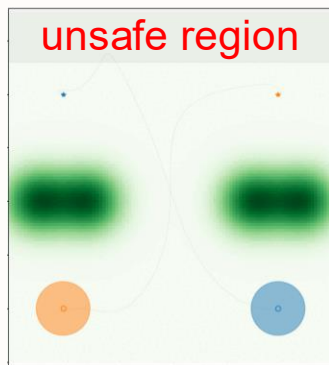
[1] Hewing, L., *et al.* "Learning-based model predictive control: Toward safe learning in control." Annual Review of Control, Robotics, and Autonomous Systems, 2020

[2] Agrawal, A., and K. Sreenath. "Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation." *Robotics: Science and Systems*. 2017

# Numerical example: the safe corridor problem



$\mathcal{L}$  **without** safety-promoting terms  
Average violation: 43%



$\mathcal{L}$  **including**  $\mathcal{L}_{inv}$   
Average violation: 1.4%

