Reliable Deep Neural Networks and Regret Minimization for Optimal Distributed Control

Luca Furieri

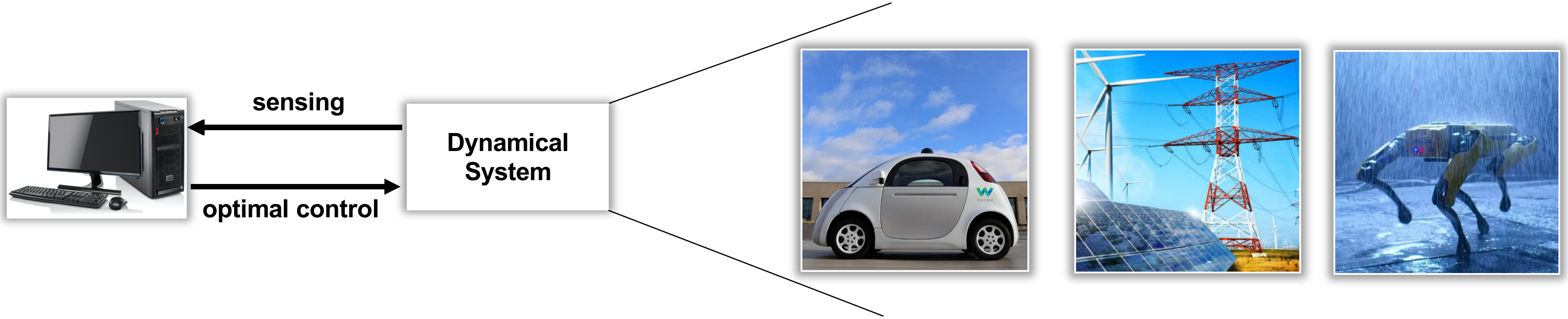
EPFL, SNSF Principal Investigator

TokyoTech, 26.07.2023

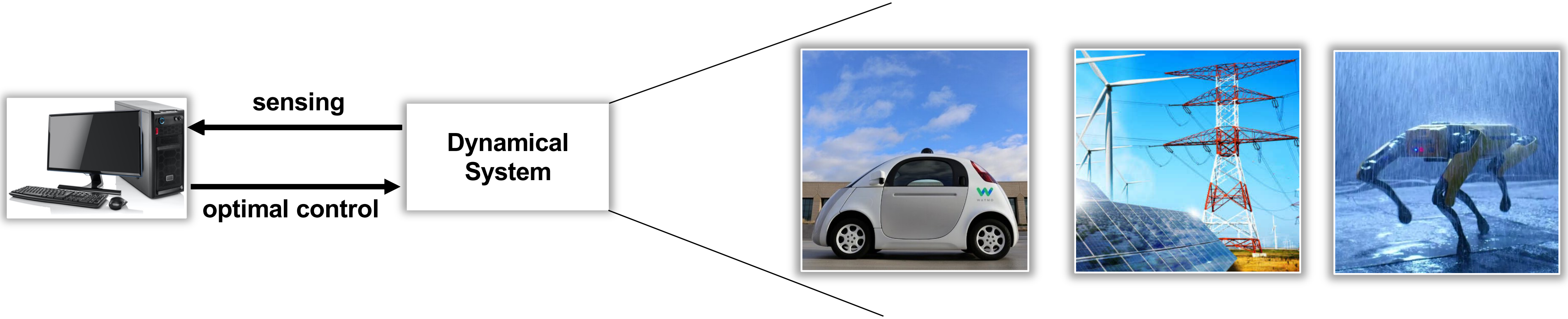
Swiss National Science Foundation

Optimal Control for Complex Dynamical Systems

Luca Furieri

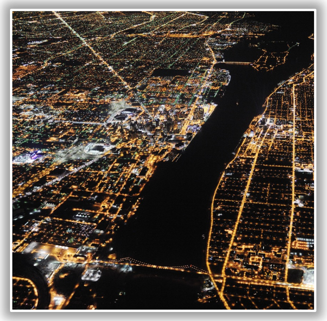


Optimal Control for Complex Dynamical Systems

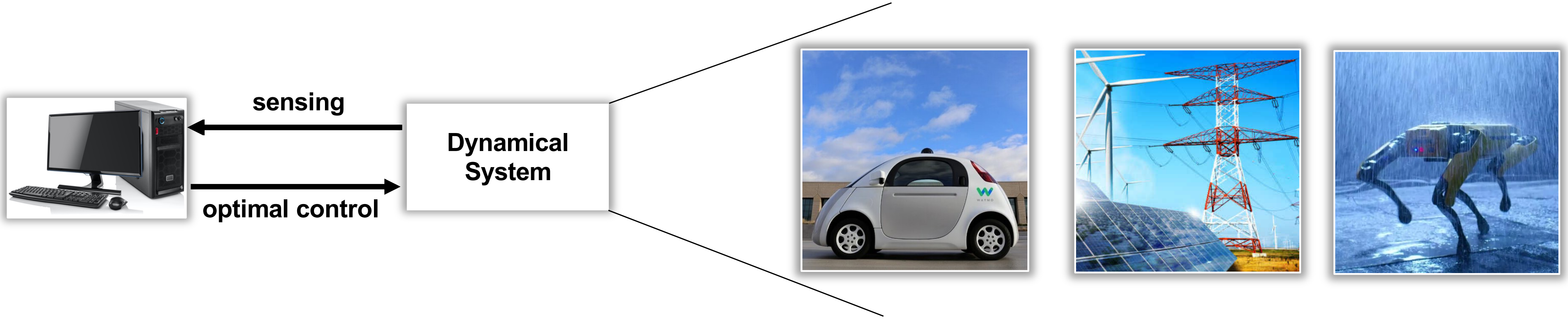


Challenges

I) Optimality in coordinated tasks



Optimal Control for Complex Dynamical Systems



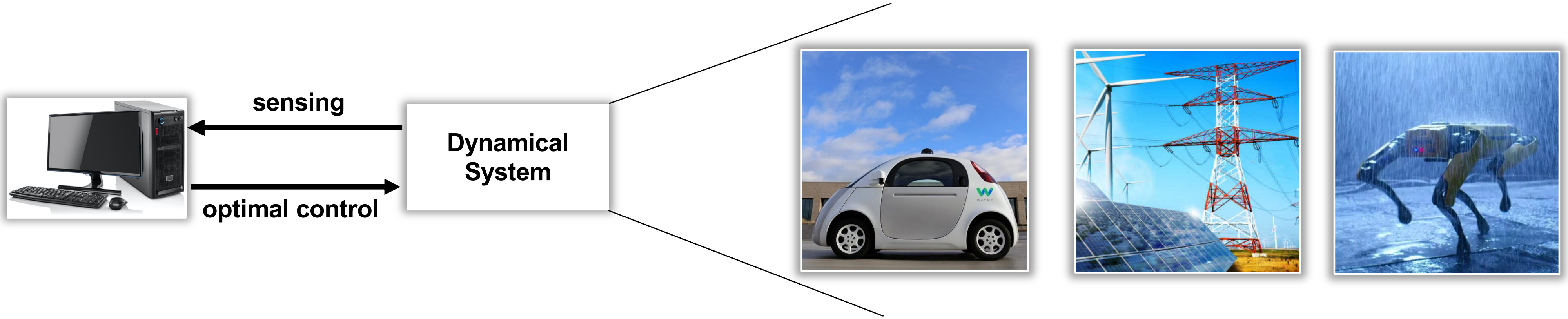
Challenges

I) Optimality in coordinated tasks

- Linear systems with quadratic costs?
  - NL policies needed! *[Witsenhausen, 1969]*
- … NL objectives for NL systems
- Recent attempt: Deep Neural Nets (DNNs)

→ Stability? Safety?

Optimal Control for Complex Dynamical Systems



sensing

Dynamical System

optimal control

Challenges

I) Optimality in coordinated tasks

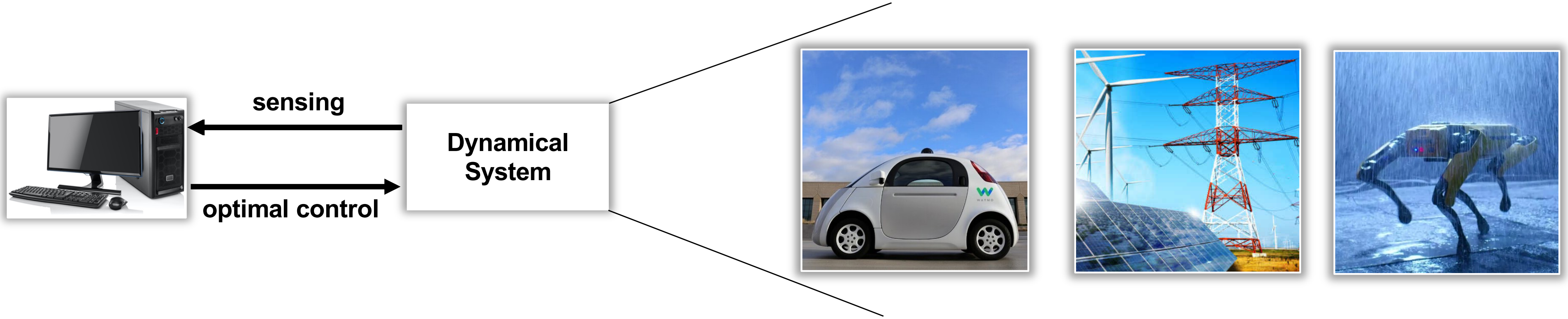
- Linear systems with quadratic costs?
  - NL policies needed! *[Witsenhausen, 1969]*
- … NL objectives for NL systems
- Recent attempt: Deep Neural Nets (DNNs)

→ Stability? Safety?

II) Adaptation to unmodeled disturbances

unsafe, unmodeled

Optimal Control for Complex Dynamical Systems



sensing

Dynamical System

optimal control

Challenges

I) Optimality in coordinated tasks

- Linear systems with quadratic costs?
  - NL policies needed! *[Witsenhausen, 1969]*
- … NL objectives for NL systems
- Recent attempt: Deep Neural Nets (DNNs)

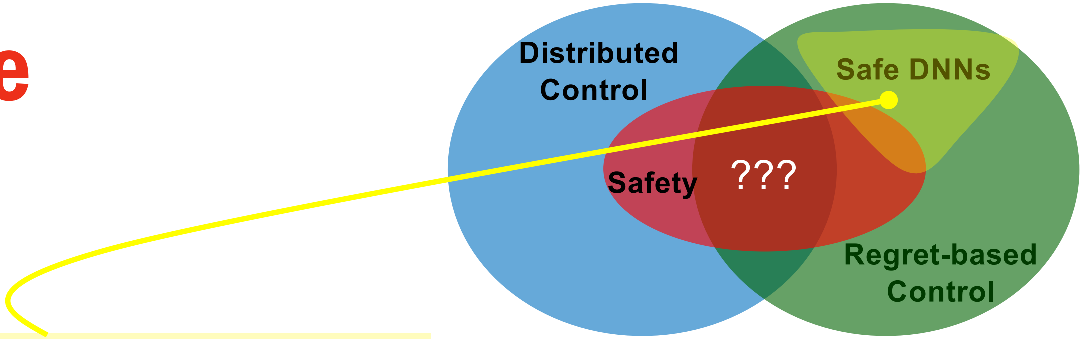
→ Stability? Safety?

II) Adaptation to unmodeled disturbances

- Most ODC approaches so far…
  - Well-modeled disturbances only
  - Safety at the cost of performance

→Regret Minimization to *safely* go beyond?

Presentation Structure



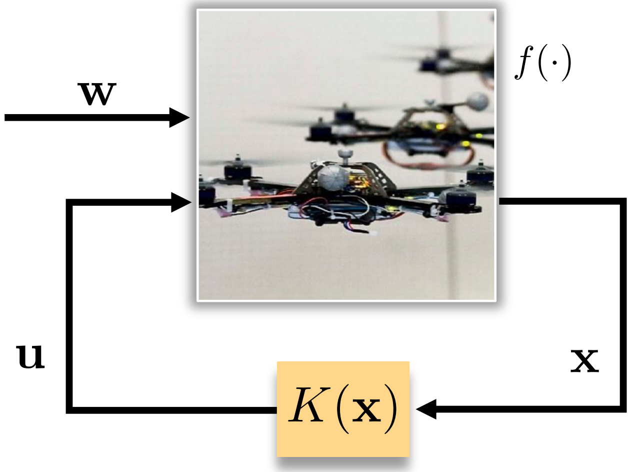
1. Learning over *all and only* stabilizing policies for nonlinear optimal control using DNNs

2. Port-Hamiltonian DNNs for optimal distributed control with built-in stability and non-vanishing gradients

3. Regret minimization for safe adaptive control

"Neural System Level Synthesis: Learning over all and only stabilizing policies for nonlinear systems", Luca Furieri, Clara Galimberti and Giancarlo Ferrari Trecate, CDC 2022

The Nonlinear Optimal Control (NOC) Problem



NOC Problem

$$K(\cdot) \in \operatorname{argmin} \frac{1}{T}\mathbb{E}_w \left[\sum_{t=0}^{T} l(x(t), u(t)) \right]$$

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

Challenges

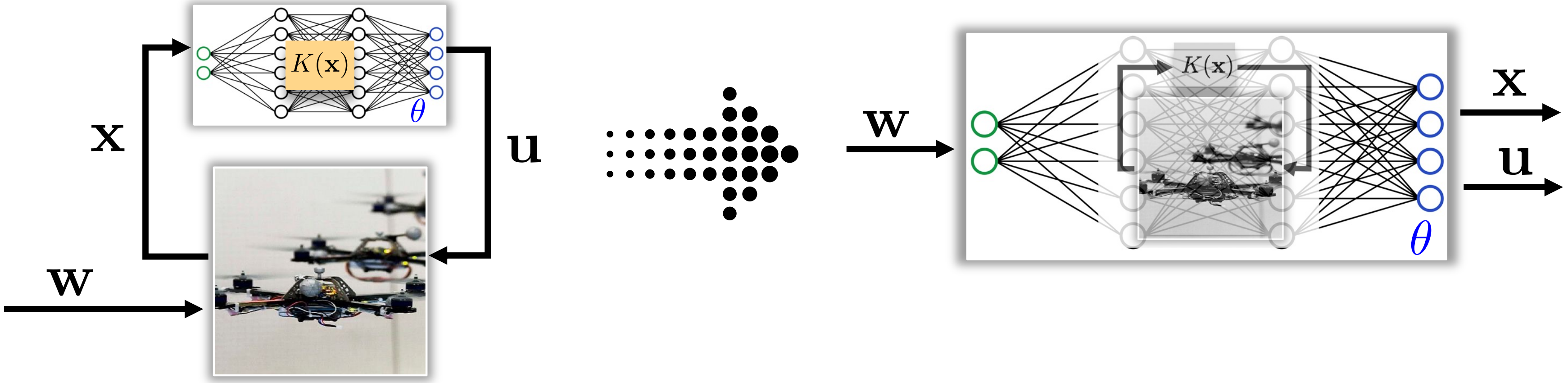
- **Nonlinearities:** system dynamics $f(\cdot)$, loss function $l(\cdot)$, control policy $K(\cdot)$
  - ~~(Tractable) optimization~~
  - ~~Global Optimality~~
- ***Dependability***: stability **during** the optimization

Our Contribution

System Level Synthesis (SLS) philosophy

From designing ***stabilizing*** policies….

To designing ***stable*** closed-loop operators



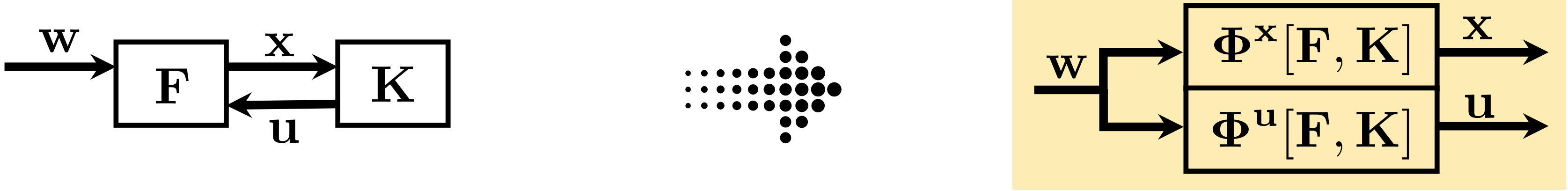
Setup and Notation

- General, non-Markovian, time-varying controlled systems

signal space

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

- Closed-loop (CL) maps induced by interconnection of **F** and **K**



- Stability notions
  - Stable signals: $\sum_{t=0}^{\infty} |x_t|^p \in \ell_p < \infty \implies \mathbf{x} \in \ell_p$
  - Stable operators: $\mathbf{A}(\mathbf{x}) \in \ell_p, \forall \mathbf{x} \in \ell_p \implies \mathbf{A} \in \mathcal{L}_p$

CL stability :=$(\mathbf{\Phi^x}, \mathbf{\Phi^u}) \in \mathcal{L}_p$

System Level Synthesis (SLS) for NOC

NOC

$$\min_{\mathbf{K}(\cdot)} \quad \mathbb{E}_{w_{T:0}}\left[\sum_{t=0}^{T} l(x_t,u_t)\right]$$

$$\text{s. t.} \quad \mathbf{x} = \mathbf{F}(\mathbf{x},\mathbf{u}) + \mathbf{w}\,, \quad \mathbf{u} = \mathbf{K}(\mathbf{x})$$

$$(\mathbf{\Phi}^{\mathbf{x}}[\mathbf{F},\mathbf{K}], \mathbf{\Phi}^{\mathbf{u}}[\mathbf{F},\mathbf{K}]) \in \mathcal{L}_p\,.$$

SLS

$$\min_{(\mathbf{\Psi}^{\mathbf{x}},\mathbf{\Psi}^{\mathbf{u}})} \quad \mathbb{E}_{w_{T:0}}\left[\sum_{t=0}^{T} l(\Psi_t^x(w_{t:0}), \Psi_t^u(w_{t:0}))\right]$$

$$\text{s. t.} \quad \boxed{\mathbf{\Psi}^{\mathbf{x}} = \mathbf{F}(\mathbf{\Psi}^{\mathbf{x}}, \mathbf{\Psi}^{\mathbf{u}}) + I} \quad \text{«Achievability»}$$

$$(\mathbf{\Psi}^{\mathbf{x}}, \mathbf{\Psi}^{\mathbf{u}}) \in \mathcal{L}_p$$

- **Challenge:** achievability constraints
  - **…i.e., nonlinear functional equalities ☹**

If linear system… *[Wang, Matni, Doyle, 2019]*

$$x_t = Ax_{t-1} + Bu_{t-1}$$

$$(zI - A)\mathbf{\Psi}^{\mathbf{x}}(z) = B\mathbf{\Psi}^{\mathbf{u}}(z) + I$$

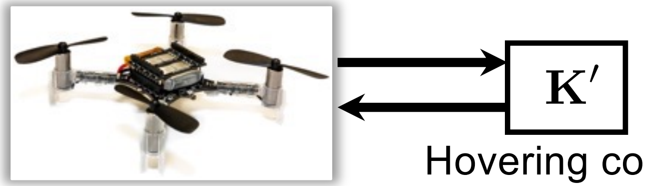
Get rid of *achievability*?

Main Result

$$\mathbf{u} = \underbrace{\mathbf{K}'(\mathbf{x})}_{\textbf{I) Base controller}\text{: stabilize}} + \underbrace{\mathbf{v}(\mathbf{x})}_{\textbf{II) Additional input}\text{: optimize performance}}$$

I. **Assumption**: $\mathbf{K}'(\cdot)$ is Input-to-State (IS) stabilizing

- i.e., leads to CL maps $(\mathbf{w}, \mathbf{v}) \rightarrow (\mathbf{x}, \mathbf{u})$ in $\mathcal{L}_p$



Hovering controller

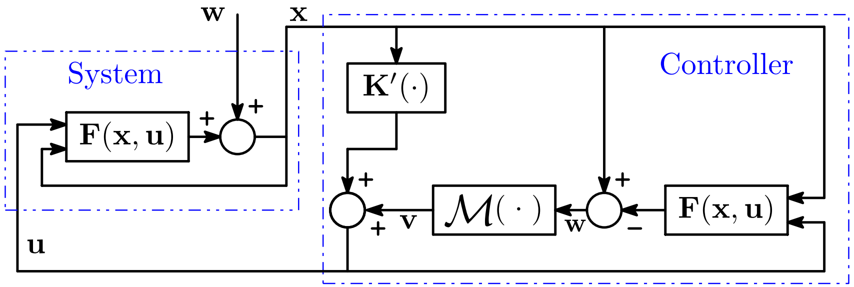
E.g.

- Feedback linearization…
- Stabilizing NMPC…

Main Result

II. Parametrize **v(x)** as follows

$$\mathbf{v} = \mathcal{M}(\mathbf{w}) = \mathcal{M}(\mathbf{x} - \mathbf{F}(\mathbf{x}, \mathbf{u}))$$


w
x
System
Controller
$\mathbf{K}'(\cdot)$
$\mathbf{F}(\mathbf{x},\mathbf{u})$
+
+
+
v
$\mathcal{M}(\cdot)$
w
+
−
+
$\mathbf{F}(\mathbf{x},\mathbf{u})$
u

Result part 1 (sufficiency)

The CL maps $(\boldsymbol{\Phi}^x, \boldsymbol{\Phi}^u)$ achieved by the control scheme above are stable for any $\mathcal{M}(\cdot) \in \mathcal{L}_p$

Proof

- By hypothesis, disturbance sequence $\mathbf{w} \in \ell_p$
- Since $\mathcal{M}(\cdot) \in \mathcal{L}_p$, then $\mathbf{v} = \mathcal{M}(\mathbf{w}) \in \ell_p$
- By hypothesis, base controller $\mathbf{K}'(\cdot)$ such that $(\mathbf{w}, \mathbf{v}) \in \ell_p \implies (\mathbf{x}, \mathbf{u}) \in \ell_p$

Main Result

Result part 2 (necessity)

If $\mathbf{K}' \in \mathcal{L}_p$, we can obtain any achievable CL maps $(\mathbf{\Psi^x}, \mathbf{\Psi^u}) \in \mathcal{L}_p$ by searching over the space of stable operators $\boldsymbol{\mathcal{M}} \in \mathcal{L}_p$.

- $\Longrightarrow$ **Globally optimal** CL maps by searching over $\boldsymbol{\mathcal{M}} \in \mathcal{L}_p$!

Proof

- Select $\boldsymbol{\mathcal{M}} = -\mathbf{K}'(\mathbf{\Psi^x}) + \mathbf{\Psi^u}$. Then, $(\mathbf{K}', \mathbf{\Psi^x}, \mathbf{\Psi^u}) \in \mathcal{L}_p \implies \boldsymbol{\mathcal{M}} \in \mathcal{L}_p$
- So, the corresponding policy $\mathbf{u} = \mathbf{K}'(\mathbf{x}) + \boldsymbol{\mathcal{M}}(\mathbf{x} - \mathbf{F}(\mathbf{x}, \mathbf{u}))$ is within our search space ☺

What closed-loop maps do we achieve?

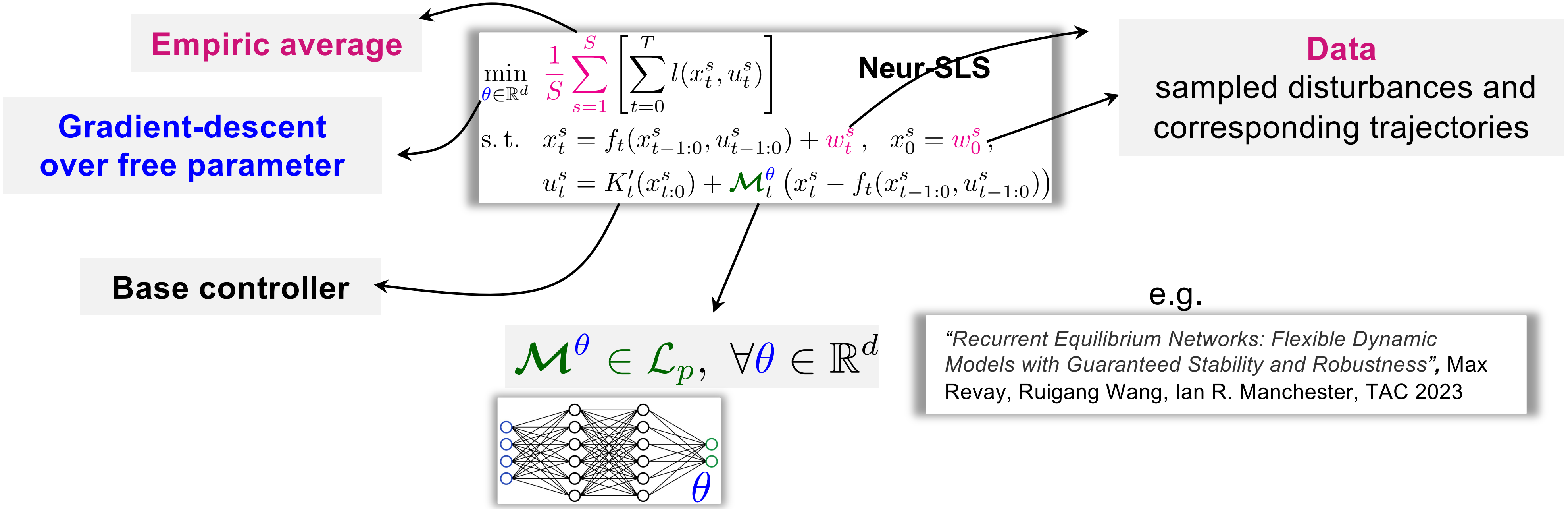
- We prove <u>by induction</u> that $(\mathbf{\Phi^x}, \mathbf{\Phi^u}) = (\mathbf{\Psi^x}, \mathbf{\Psi^u})$, i.e., we achieve the desired CL maps.
- <u>Inductive Step</u>: assume $(\Phi^x_{j:0}, \Phi^u_{j:0}) = (\Psi^x_{j:0}, \Psi^u_{j:0})$. Then

$$\Phi^u_{j+1} = \cancel{K'_{j+1}(\underbrace{F_{j+1:0}(\Phi^x_{j:0}, \Phi^u_{j:0}) + I}_{=\Phi^x_{j+1}})} - \cancel{K'_{j+1}(\underbrace{F_{j+1:0}(\Psi^x_{j:0}, \Psi^u_{j:0}) + I}_{=\Psi^x_{j+1}})} + \Psi^u_{j+1} = \Psi^u_{j+1}$$ ☺

- <u>Base Step</u>: $\Phi^x_0 = \Psi^x_0 = I$ … *(the initial state is the initial state)*

The Proposed Neur-SLS

- We establish a *deep learning* procedure to tackle NOC in a dependable way



Empiric average

Gradient-descent over free parameter

Neur-SLS

$$\min_{\theta\in\mathbb{R}^d} \frac{1}{S}\sum_{s=1}^{S}\left[\sum_{t=0}^{T} l(x_t^s,u_t^s)\right]$$

$$\text{s.t.}\quad x_t^s = f_t(x_{t-1:0}^s, u_{t-1:0}^s) + w_t^s\,, \quad x_0^s = w_0^s\,,$$

$$u_t^s = K_t'(x_{t:0}^s) + \mathcal{M}_t^\theta\left(x_t^s - f_t(x_{t-1:0}^s, u_{t-1:0}^s)\right)$$

Data
sampled disturbances and corresponding trajectories

Base controller

$$\mathcal{M}^\theta \in \mathcal{L}_p, \quad \forall\theta\in\mathbb{R}^d$$

e.g.

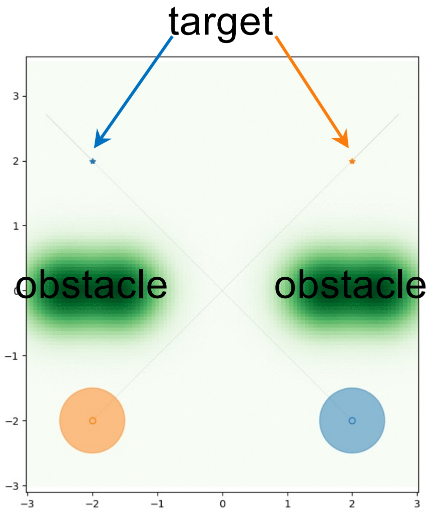
"Recurrent Equilibrium Networks: Flexible Dynamic Models with Guaranteed Stability and Robustness", Max Revay, Ruigang Wang, Ian R. Manchester, TAC 2023

- Neur-SLS offers the following guarantees:
  1. CL stability for any θ
  2. *Representation power* only limited by approximation of $\mathcal{L}_p$

The Corridor Problem

- Point-mass vehicles, nonlinear drag forces, force input

$$\begin{bmatrix} x_t \\ \dot{x}_t \end{bmatrix} = \begin{bmatrix} x_{t-1} \\ \dot{x}_{t-1} \end{bmatrix} + T_s \begin{bmatrix} \dot{x}_{t-1} \\ -\left\|\dot{x}_{t-1}\right\|^2 \dot{x}_{t-1} + u_{t-1} \end{bmatrix}$$



- **Goal**: CL stability on target, avoid collisions & obstacles

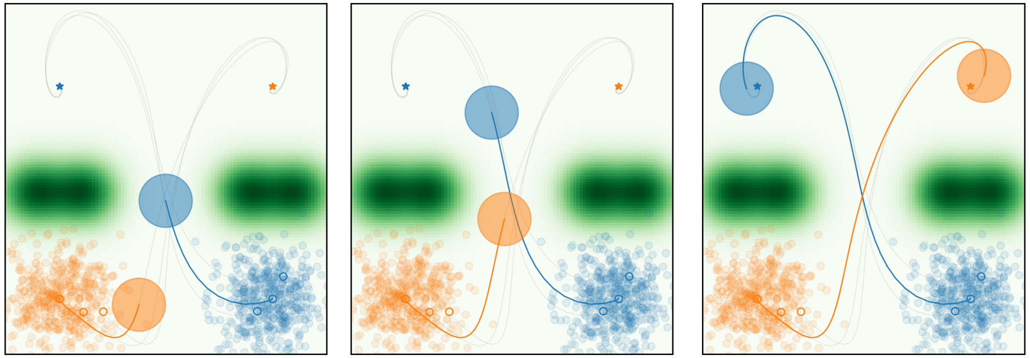
$$l(\cdot) = l_{target}(\cdot) + l_{collisions}(\cdot) + l_{obstacles}(\cdot)$$

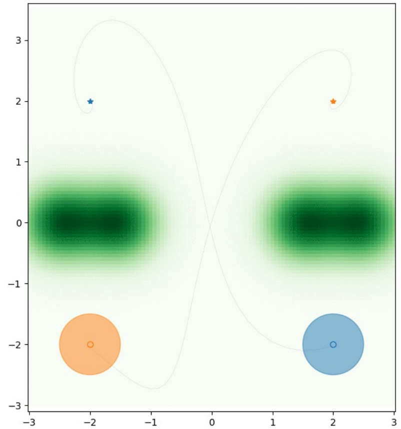
- **Base controller $\mathbf{K}'$**: linear spring at rest on target
  - Overshoot, collisions…. But stabilizing

- **Approach**: train the corresponding Neur-SLS with standard GD!

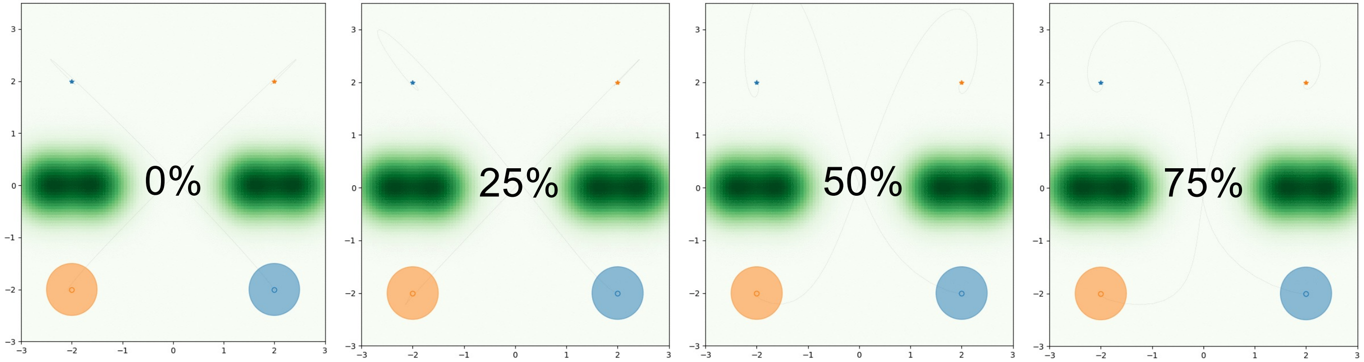
The Corridor Problem

- Upon training over a dataset 500 different initial conditions…

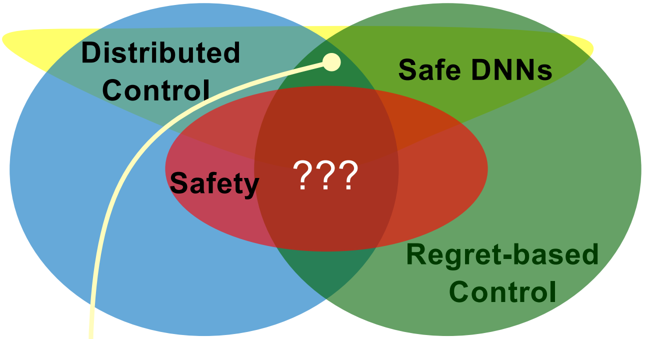




- …robots learn the “corridor behavior” (robustly).
- CL stability guaranteed by design! Even with early stopping of training



Presentation Structure



1. Learning over *all and only* stabilizing policies for nonlinear optimal control using DNNs

2. Port-Hamiltonian DNNs for optimal distributed control with built-in stability and non-vanishing gradients

3. Regret minimization for safe adaptive control

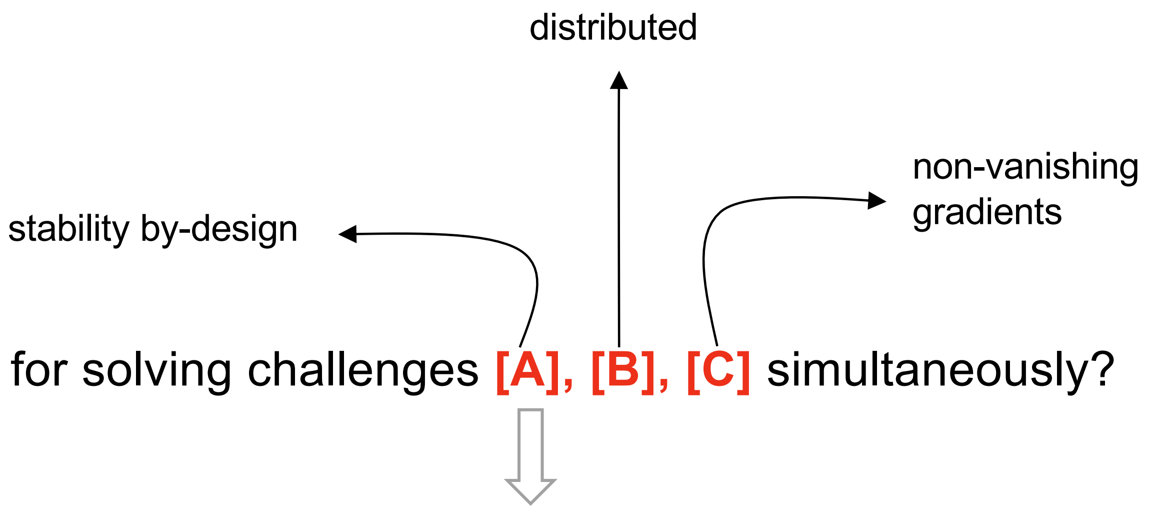
"Distributed neural network control with dependability guarantees: a compositional port-Hamiltonian approach", Luca Furieri, Clara Galimberti, Muhammad Zakwan, and Giancarlo Ferrari Trecate, L4DC 2022 (Spotlight Oral)

Challenges of Using DNN Policies… at Large Scale

A. Closed-loop stability
- Neural SLS to parametrize all stabilizing NL policies

B. … Even in a distributed setup for networked control
- Sparse NN matrices? → Instability!

C. Vanishing gradients during optimization
- Training stops prematurely because gradients are small…
- … Despite being far from stationary point.



distributed

stability by-design

non-vanishing gradients

A framework for solving challenges [A], [B], [C] simultaneously?

port-Hamiltonian systems

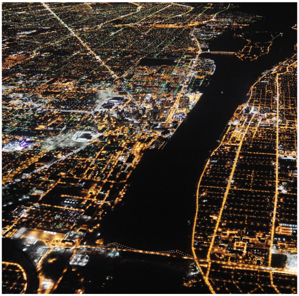
Port-Hamiltonian (pH) systems[1]

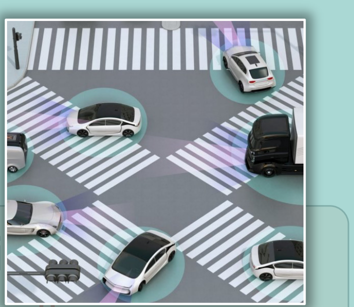
$$\dot{\mathbf{x}}(t) = (\mathbf{J} - \mathbf{R})\frac{\partial V(\mathbf{x}(t))}{\partial \mathbf{x}} + \mathbf{G}^{\top}\mathbf{u}(t)$$

$$\mathbf{y}(t) = \mathbf{G}\frac{\partial V(\mathbf{x}(t))}{\partial \mathbf{x}}$$

- **J** skew-symmetric
- **R** $\succeq 0$

- *V*: Hamiltonian function (internal system energy)





- n: (i.e. irrespectively o *V*)

of the origin if **R** $\succ 0$ ipation)

P2: Composition

$\sum pH$ $\equiv$ $\sum pH$

ploiti

⇩

NN controllers

nteeing stability (**A**)

[1] A. van der Schaft and D. Jeltsema. "Port-Hamiltonian systems theory: An introductory overview." *Foundations and Trends in Systems and Control* 1.2-3 (2014): 173-378.

Main Result

For a (nonlinear) pH system, consider a dynamic controller in pH form

Theorem (B)

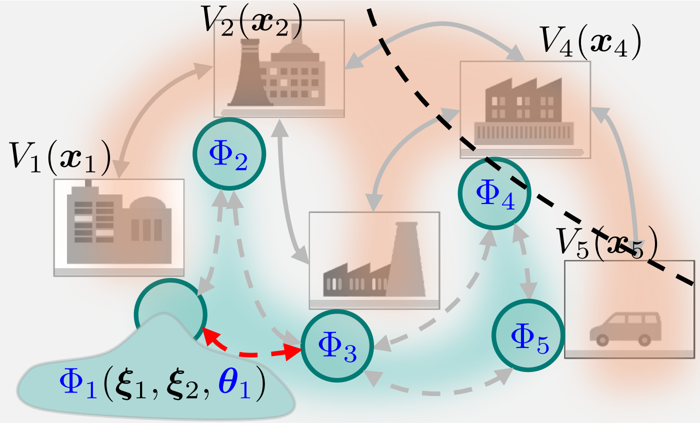
$\mathcal{G}_\Phi$: describe which local energy depends on which local controller states.

Then, the control policy is *distributed* according to $\mathcal{G}_\Phi^2$ (paths of length 2).

ble parameters

where $\Phi(\xi(t), \theta)$ is

- **Closed-loop**
- **Distributed implementations (B) using** $\Phi(\xi(t), \theta) = \sum_{i=1}^{N} \Phi_i(\xi_{\mathcal{N}_i}, \theta_i)$



- Total energy: $P = \sum_{i=1}^{5} V_i(\cdot) + \sum_{i=1}^{5} \Phi_i(\cdot)$
- Closed-loop is pH: $\dot{P}(\cdot) \leq 0$ **for any θ ! (A)**
- Take care of $\frac{\partial \Phi}{\partial \xi_1} = \frac{\partial \Phi_1(\xi_1, \xi_2)}{\partial \xi_1} + \frac{\partial \Phi_2(\xi_1, \xi_2, \xi_3)}{\partial \xi_1}$

Main Result

For a (nonlinear) pH system, consider a dynamic controller in pH form

$$\dot{\boldsymbol{\xi}} = \mathbf{J}_c \frac{\partial \Phi(\boldsymbol{\xi}(t), \boldsymbol{\theta})}{\partial \mathbf{x}} + \mathbf{G}_c{}^{\top} \mathbf{y}(t)$$

$$\mathbf{u}(t) = \mathbf{G}_c \frac{\partial \Phi(\boldsymbol{\xi}(t), \boldsymbol{\theta})}{\partial \mathbf{x}}$$

blue = trainable parameters

where $\Phi(\boldsymbol{\xi}(t), \boldsymbol{\theta})$ is *a Deep Neural Network* energy function. Then

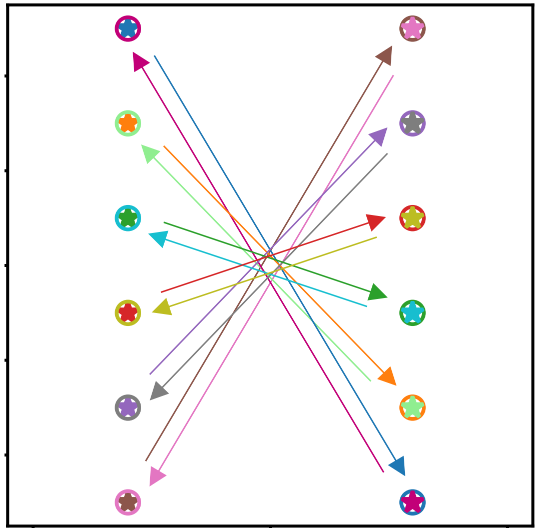
- **Closed-loop stability (A) holds by design (for any θ)**
- **Distributed implementations (B) using** $\Phi(\boldsymbol{\xi}(t), \boldsymbol{\theta}) = \sum_{i=1}^{N} \Phi_i(\xi_{\mathcal{N}_i}, \boldsymbol{\theta}_i)$
- **Non-vanishing gradients (C)**

pH systems preserve *symplecticity:* calling $\zeta = \begin{bmatrix} \text{system state} \\ \text{controller state} \end{bmatrix}$ we have

$$\left(\frac{\partial \zeta(T)}{\partial \zeta(T-t)}\right)^{\top} \begin{bmatrix} J & 0 \\ 0 & J_c \end{bmatrix} \frac{\partial \zeta(T)}{\partial \zeta(T-t)} = \begin{bmatrix} J & 0 \\ 0 & J_c \end{bmatrix} \implies \left\| \frac{\partial \zeta(T)}{\partial \zeta(T-t)} \right\| \geq 1$$

Navigation task using pH-DNN distributed controllers

- Position swapping of 12 mobile robots
  - Modelled as pH systems
  - Local controllers with ring communication topology



- Objective:

Stable closed-loop system + collision avoidance

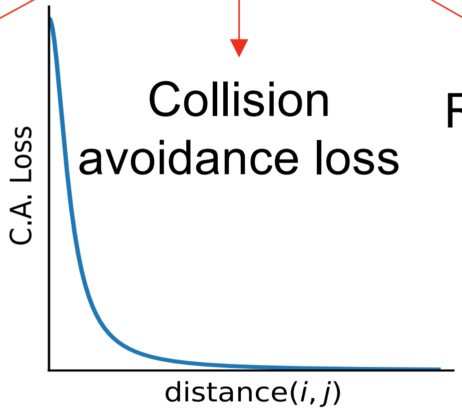
- Control cost →

$$\mathcal{L} = \int_0^T (\ell_Q + \ell_{CA} + \ell_R)\, \mathrm{d}t$$

Quadratic loss penalizing:

- Distance to target point
- Non zero velocity
- Input magnitude

Collision avoidance loss



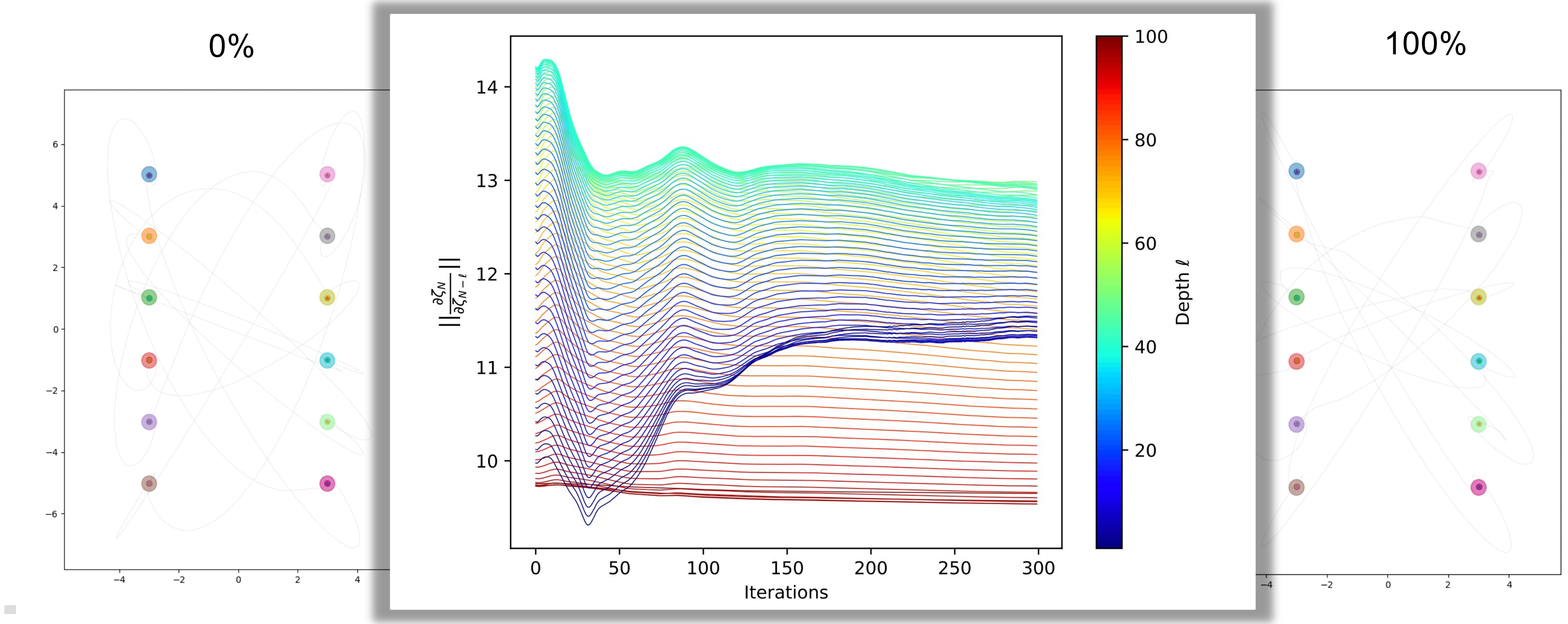
Regularization loss

↓

Penalizes parameter variations across layers

Numerical Experiments

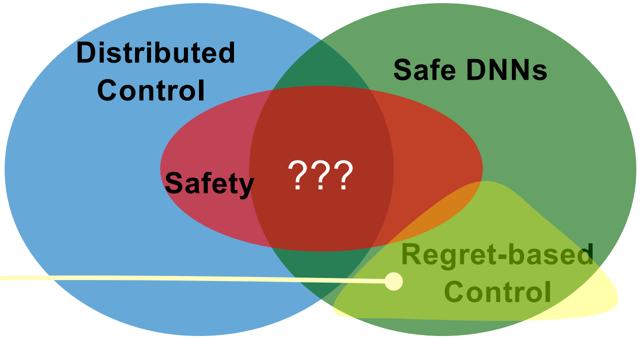
- Closed-loop stability during training (**A**)
- Distributed controllers (ring topology) (**B**)
- Non-Vanishing gradients (**C**)



DNN controllers → optimality in coordinated tasks…

Adapt the task to unmodeled environments?

Presentation Structure



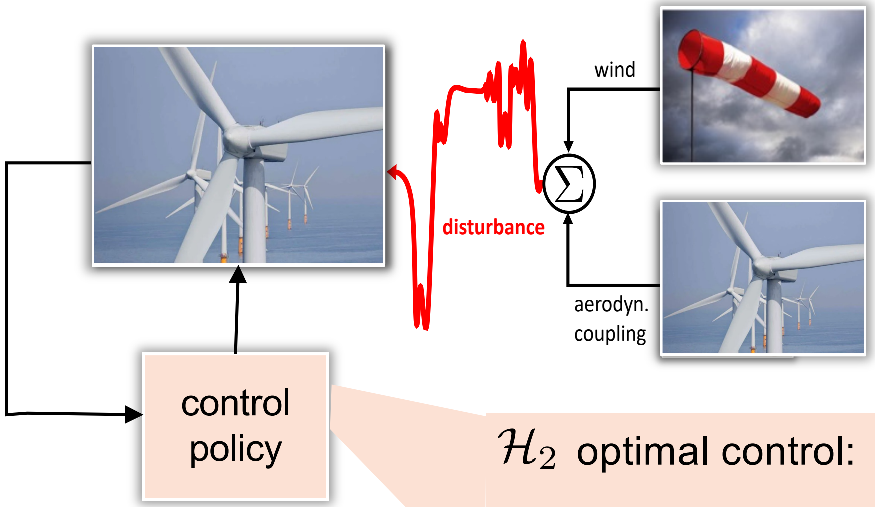
1. Learning over *all and only* stabilizing policies for nonlinear optimal control using DNNs

2. Port-Hamiltonian DNNs for optimal distributed control with built-in stability and non-vanishing gradients

3. Regret minimization for safe adaptive control

"Safe Control with Minimal Regret", Andrea Martin, Luca Furieri, Florian Dorfler, John Lygeros and Giancarlo Ferrari-Trecate, L4DC 2022

Regret-optimal Control



- Stochastic & time-varying disturbances
- Exacerbated in networked control system

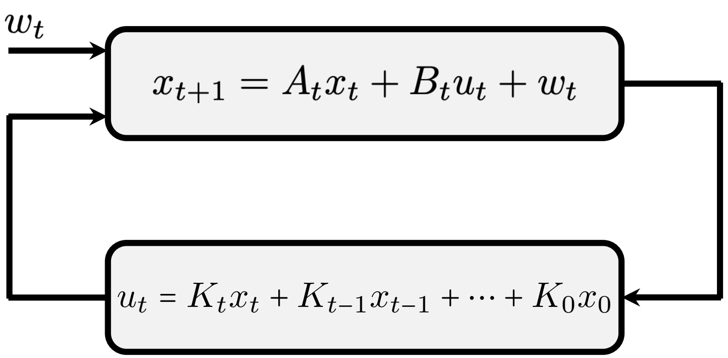
$\mathcal{H}_2$ optimal control:	✓	optimal for Gaussian w(t)	✗	lack of robustness
$\mathcal{H}_\infty$ optimal control:	✓	optimal for worst-case w(t)	✗	overly conservative

Idea

<u>Regret minimization</u> for optimal adaptation to unmodeled disturbances

- Learn the best behavior *in hindsight*
- Literature on *regret in control*: no safety, suboptimal [Agarwal et al., 2019], [Cohen et al., 2019], [Sabag et al., 2021]…

Regret Minimization for LQ Problems


w_t
$x_{t+1} = A_t x_t + B_t u_t + w_t$
$u_t = K_t x_t + K_{t-1} x_{t-1} + \cdots + K_0 x_0$

The *realized* Linear Quadratic cost is written as

$$\mathbf{x}^T\mathbf{x} + \mathbf{u}^T\mathbf{u} = J(\mathbf{K}, \mathbf{w})$$

i.e., a function of chosen policy and *realized* disturbances

- $\mathcal{H}_2$ and $\mathcal{H}_\infty$ costs: minimize expected value or max of $J(\mathbf{K}, \mathbf{w})$ over $\mathbf{w}$
  - Only good if $\mathbf{w}$ is Gaussian ($\mathcal{H}_2$) or worst-case ($\mathcal{H}_\infty$)

Proposal: minimize cost with respect to *the* $\tilde{\mathbf{u}}^\star$ *we would have chosen, had we known* $\mathbf{w}$

causal

non-causal

Regret

$$\min_{\mathbf{K}} \max_{|\mathbf{w}|_2 \le 1} \left[J(\mathbf{K}, \mathbf{w}) - \min_{\tilde{\mathbf{u}}(\mathbf{w})} \tilde{\mathbf{x}}^T\tilde{\mathbf{x}} + \tilde{\mathbf{u}}^T\tilde{\mathbf{u}} \right]$$

Learning from the Optimal Non-causal Policy

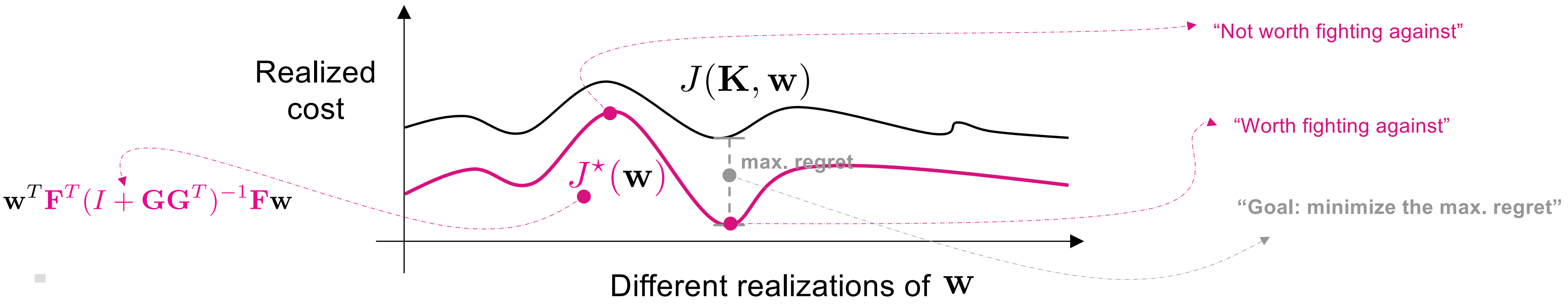
$$\boxed{\min_{\mathbf{K}} \ \max_{|\mathbf{w}|_2\leq 1} \left[J(\mathbf{K},\mathbf{w}) - \min_{\tilde{\mathbf{u}}(\mathbf{w})} \tilde{\mathbf{x}}^T\tilde{\mathbf{x}} + \tilde{\mathbf{u}}^T\tilde{\mathbf{u}}\right]}$$

- Since $\tilde{\mathbf{x}} = \mathbf{G}\tilde{\mathbf{u}} + \mathbf{F}\mathbf{w}$, best *non-causal* policy given by:

$$\boxed{\tilde{\mathbf{u}}^\star(\mathbf{w}) = -(I+\mathbf{G}\mathbf{G}^T)^{-1}\mathbf{G}^T\mathbf{F}\mathbf{w} = \mathbf{\Psi}^\star\mathbf{w}}$$

… Remark: despite being linear, also optimal among *nonlinear* non-causal policies!

- **Interpretation**: optimal non-causal policy teaches what $\mathbf{w}$ is worth fighting against!


"Not worth fighting against"
Realized cost
$J(\mathbf{K},\mathbf{w})$
"Worth fighting against"
max. regret
$J^\star(\mathbf{w})$
$\mathbf{w}^T\mathbf{F}^T(I+\mathbf{G}\mathbf{G}^T)^{-1}\mathbf{F}\mathbf{w}$
"Goal: minimize the max. regret"
Different realizations of $\mathbf{w}$

Main Result: System Level Synthesis for Safe Regret Minimization

$$\mathbf{w}^T\mathbf{F}^T(I+\mathbf{G}\mathbf{G}^T)^{-1}\mathbf{F}\mathbf{w}$$

$$||$$

- The regret-minimization control problem $$\min_{\mathbf{K}}\ \max_{|\mathbf{w}|_2\leq 1}\left[J(\mathbf{K},\mathbf{w}) - \min_{\tilde{\mathbf{u}}(\mathbf{w})}\tilde{\mathbf{x}}^T\tilde{\mathbf{x}}+\tilde{\mathbf{u}}^T\tilde{\mathbf{u}}\right]$$

is equivalent to

$$\begin{aligned}&\min_{\mathbf{\Phi}=[\mathbf{\Phi}_x\ \ \mathbf{\Phi}_u]}\ \lambda_{\max}\left(\mathbf{\Phi}^T\mathbf{\Phi}-\mathbf{\Psi}^{\star T}\mathbf{\Psi}^{\star}\right)\\ &\text{subject to}\ \ \mathbf{\Phi}_x=\mathbf{G}\mathbf{\Phi}_u+\mathbf{F}\\ &\qquad\qquad\ \ \mathbf{\Phi}\ \text{are causal}\end{aligned}$$

$$\tilde{\mathbf{u}}^{\star}(\mathbf{w})=\begin{bmatrix}\tilde{u}_0^{\star}\\ \tilde{u}_1^{\star}\\ \tilde{u}_2^{\star}\end{bmatrix}=\mathbf{\Psi}^{\star}\mathbf{w}=\begin{bmatrix}\blacksquare&\blacksquare&\blacksquare\\ \blacksquare&\blacksquare&\blacksquare\\ \blacksquare&\blacksquare&\blacksquare\end{bmatrix}\begin{bmatrix}w_0\\ w_1\\ w_2\end{bmatrix}$$

$$\mathbf{u}=\mathbf{\Phi}\mathbf{w}=\begin{bmatrix}\blacksquare&&\\ \blacksquare&\blacksquare&\\ \blacksquare&\blacksquare&\blacksquare\end{bmatrix}\begin{bmatrix}w_0\\ w_1\\ w_2\end{bmatrix}$$

- Can easily add safety constraints $x_t\in\mathcal{X},\ u_t\in\mathcal{U},\ \forall t,\ \forall w_t\in\mathcal{W}$
  - … also on the non-causal policy → define a more realistic benchmark!

→ Convex design of safe and regret-optimal control policies

Numerical Examples

$$A_t = \rho \begin{bmatrix} 0.7 & 0.2 & 0 \\ 0.3 & 0.7 & -0.1 \\ 0 & -0.2 & 0.8 \end{bmatrix}, \; B_t = \begin{bmatrix} 1 & 0.2 \\ 2 & 0.3 \\ 1.5 & 0.5 \end{bmatrix}, \; \forall t \in \{0 \dots T-1\},$$

$\mathbf{w}$	$\mathcal{SH}_2$	$\mathcal{SH}_\infty$	$\mathcal{SR}_{nc}$
$\mathcal{N}(0,1)$	**1**	+21.14%	+ 10.89%
$\mathcal{U}_{[0.5,1]}$	+63.42%	>+100%	**1**
$\mathcal{U}_{[0,1]}$	+40.69%	>+100%	**1**
1	+67.74%	>+100%	**1**
sin	+58.12%	>+100%	**1**
sawtooth	+46.27%	>+100%	**1**
step	+66.49%	>+100%	**1**
stairs	+45.27%	>+100%	**1**
worst	+18.45%	**1**	+7.74%

- $\mathcal{H}_2$ wins for Gaussian $\mathbf{w}$, and $\mathcal{H}_\infty$ wins for worst-case $\mathbf{w}$, as expected
  - Regret only slightly worse

- Regret achieves better performance for all non-classical $\mathbf{w}$ realizations!

A New Paradigm in Control?

- Connections with *Imitation Learning*

  [*"Follow the Clairvoyant: An Imitation Learning Approach to Optimal Control",* Andrea Martin, Luca Furieri, Florian Dorfler, John Lygeros, Giancarlo Ferrari-Trecate, IFAC 2023]

$$\min_{\boldsymbol{\pi}} \max_{\|\mathbf{w}\|_2 \leq 1} \left[\boldsymbol{\delta}_{x,\psi}^\top \mathbf{Q} \boldsymbol{\delta}_{x,\psi} + \boldsymbol{\delta}_{u,\psi}^\top \mathbf{R} \boldsymbol{\delta}_{u,\psi}\right]$$

$$\delta = \text{“Difference between causal and optimal non-causal trajectories”}$$

  - **Unconstrained case:** Regret Minimization = Imitation Learning
  - **Constrained case:** Imitation Learning > Regret Minimization!

- Receding-horizon regret minimization (MPC)

  [*"On the Guarantees of Minimizing Regret in a Receding Horizon",* Andrea Martin, Luca Furieri, Florian Dorfler, John Lygeros, Giancarlo Ferrari-Trecate, *under review]*

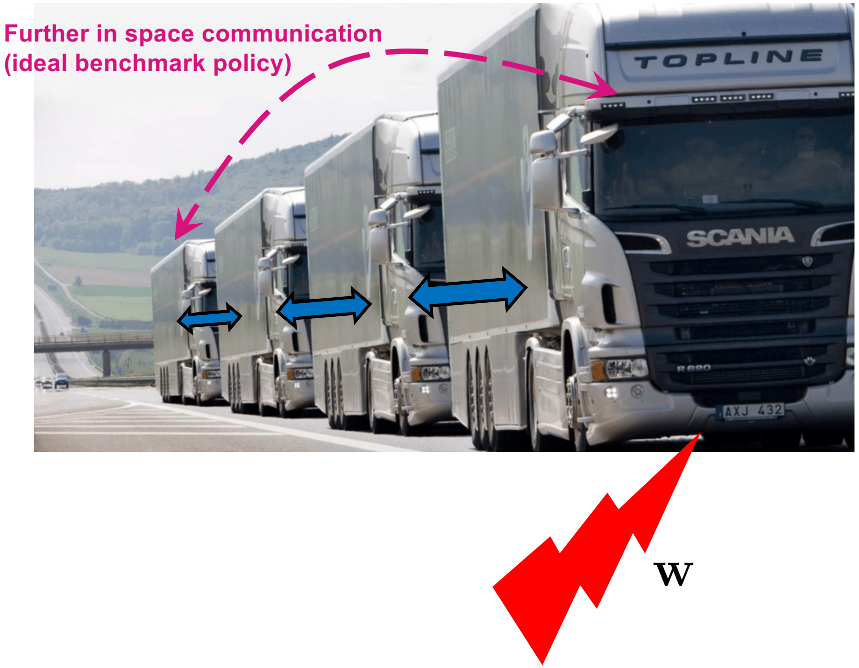
  - **Main result:** stability analysis using regret-based cost
  - **Benefit:** outperforms standard $\mathcal{H}_2/\mathcal{H}_\infty$ receding horizon performance
    - Even when optimizing less frequently (i.e., every 10 time steps…)!

A New Paradigm in Control?

Work in progress

- Optimal distributed control by minimizing *“Spatial Regret”*

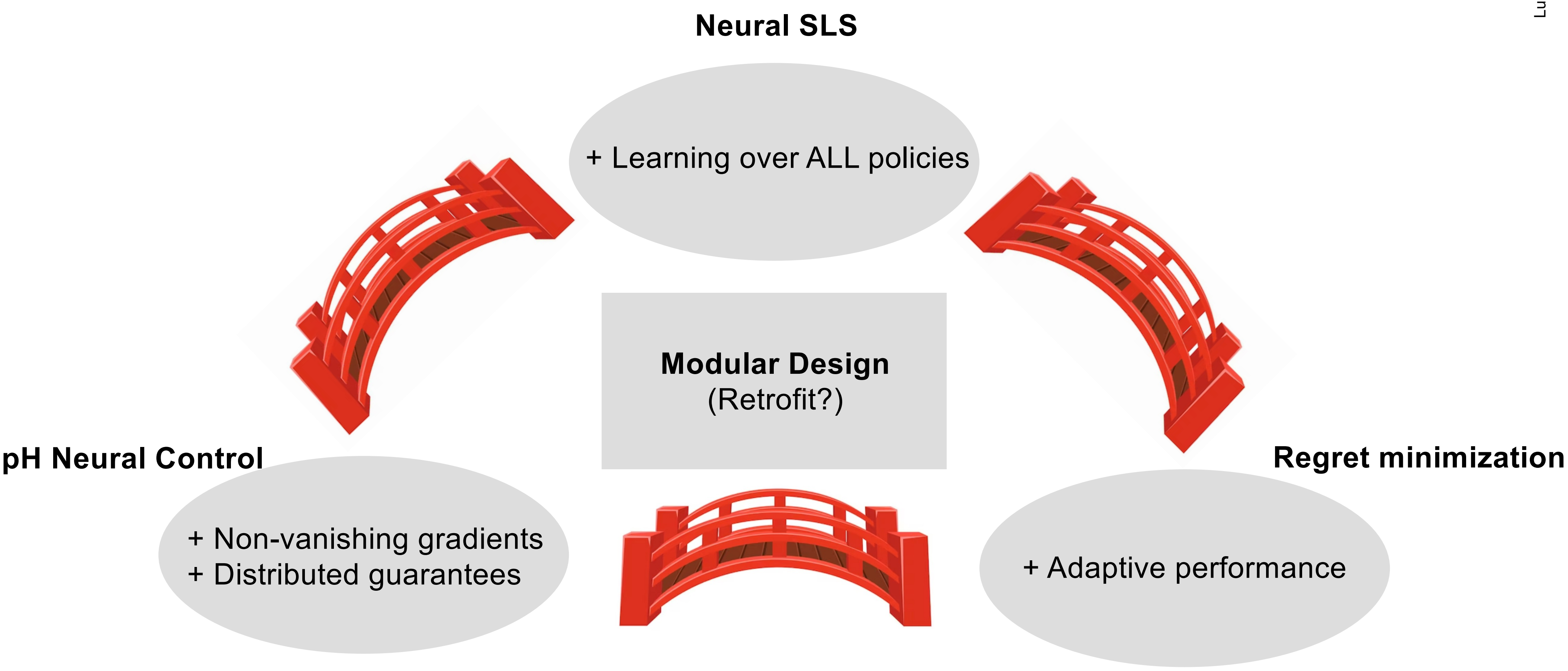
What would have I done, had I seen further in space?”



- Outperform $\mathcal{H}_2/\mathcal{H}_\infty$ against localized disturbances in large-scale control systems
- Combine with “further in time” non-causal benchmarks

Daniele Martinelli (SNSF PhD student)

Outlook: Towards Scalable Nonlinear Design



Neural SLS

+ Learning over ALL policies

Modular Design
(Retrofit?)

pH Neural Control

+ Non-vanishing gradients
+ Distributed guarantees

Regret minimization

+ Adaptive performance

Thank you for your attention!

Swiss National Science Foundation