ISTITUTO ITALIANO
DI TECNOLOGIA
HUMAN-ROBOT INTERFACES
AND INTERACTION

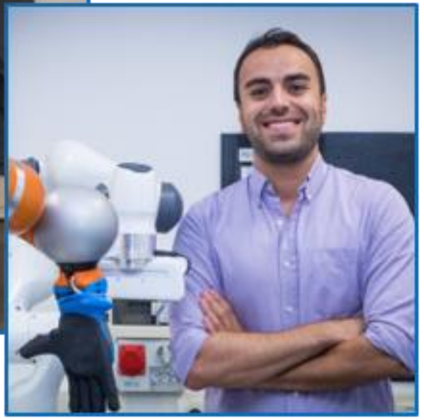
Learning, Control, and Modelling for Deformable Object Manipulation and Safe Interaction

Gokhan Solak

Postdoc @ HRI² lab, IIT, Genoa

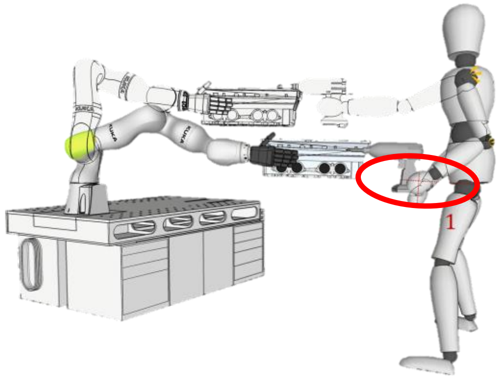
HRI² lab, IIT, Genoa

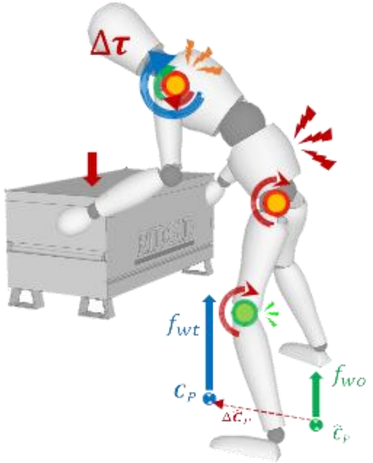
ISTITUTO ITALIANO
DI TECNOLOGIA
HUMAN-ROBOT INTERFACES
AND INTERACTION



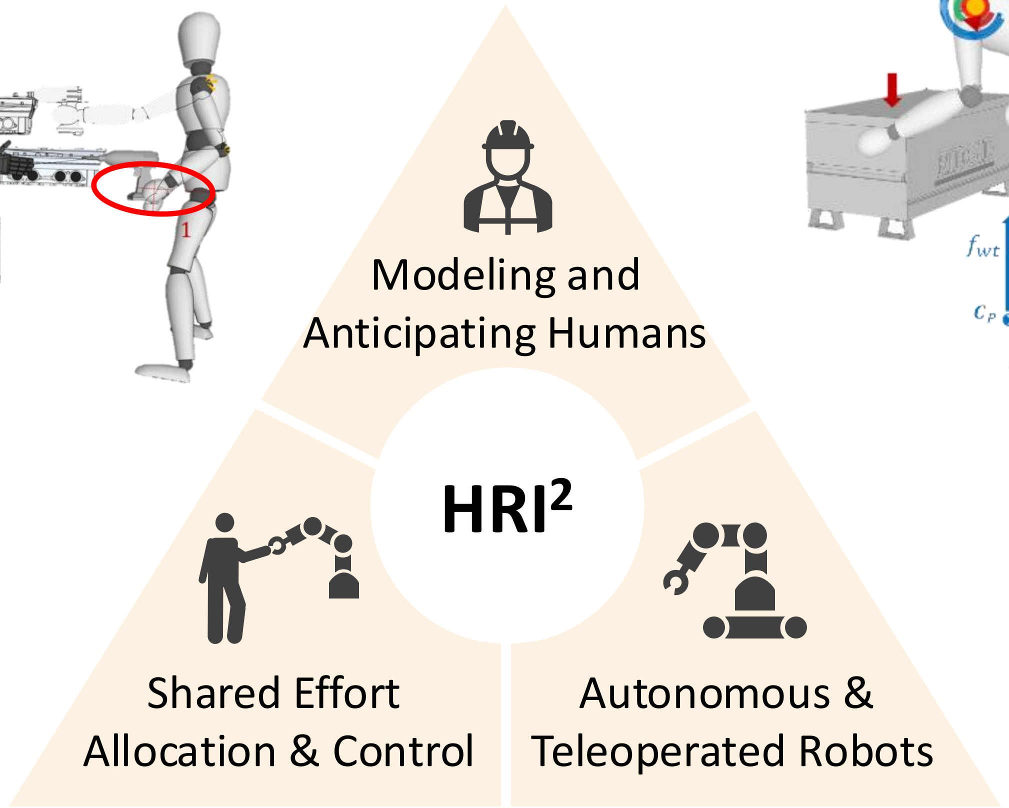
PI: Arash Ajoudani

R&D Pillars

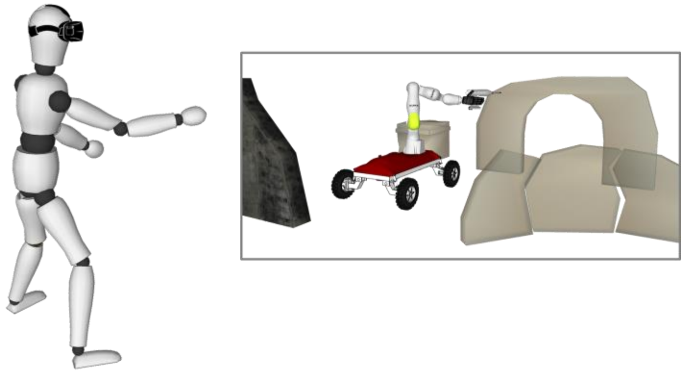


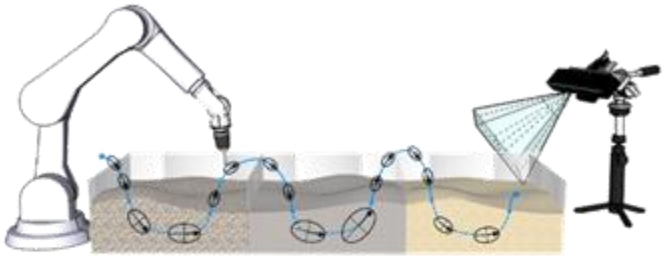


$\Delta\tau$
f_{wt}
f_{wo}

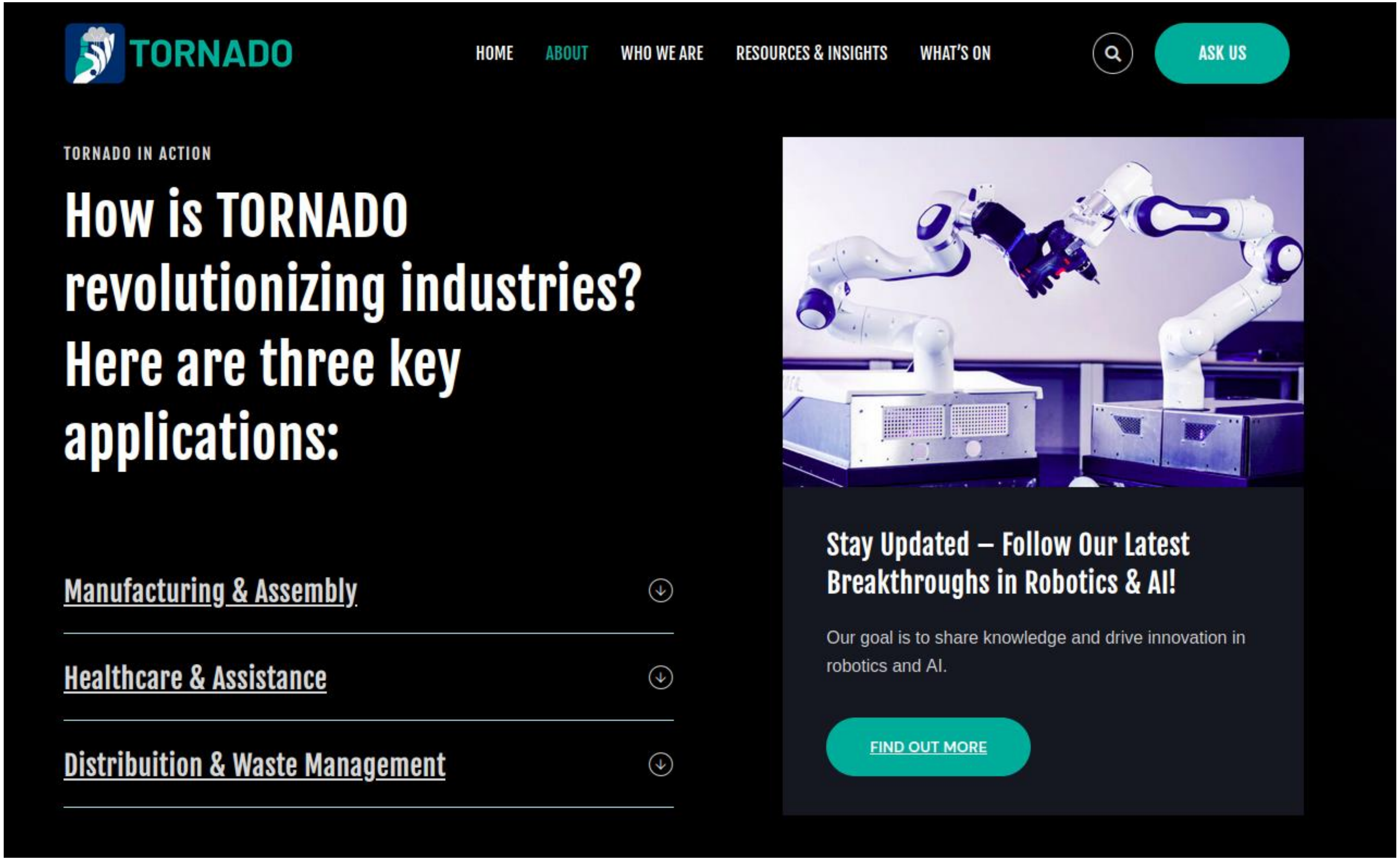


Modeling and Anticipating Humans
HRI²
Shared Effort Allocation & Control
Autonomous & Teleoperated Robots

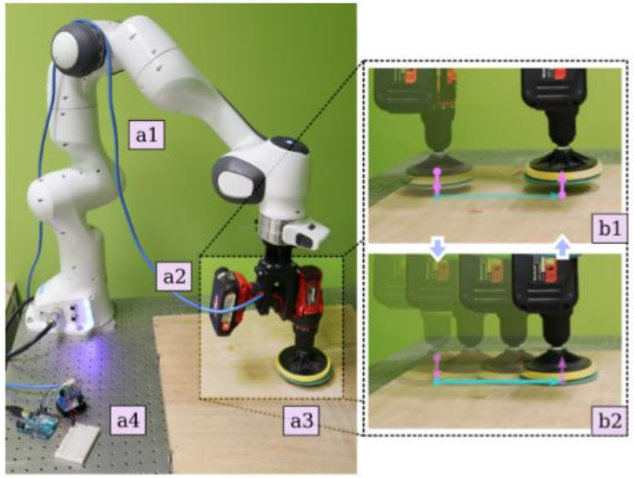




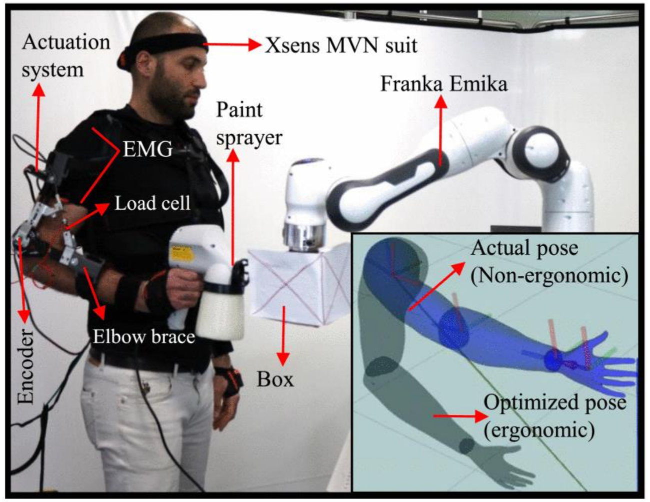
Tornado Project (tornado-horizon.eu)



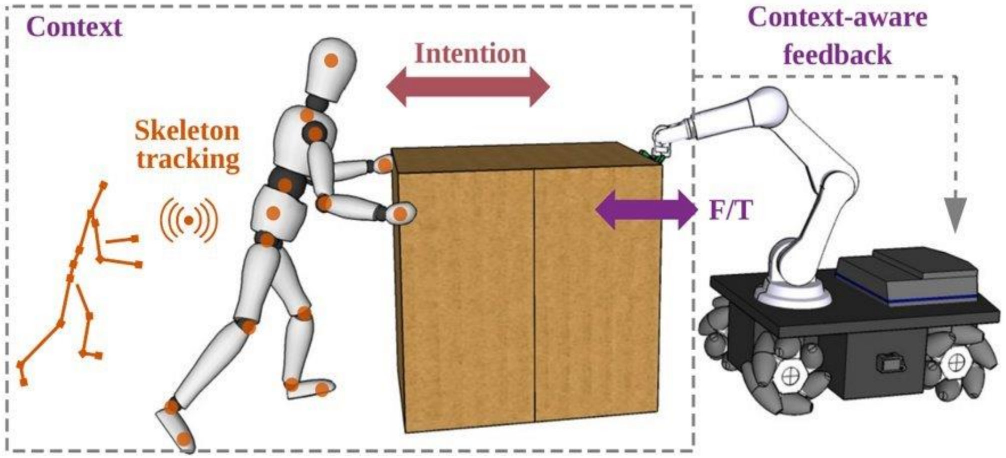
Gokhan Solak, Postdoc @ HRI² lab



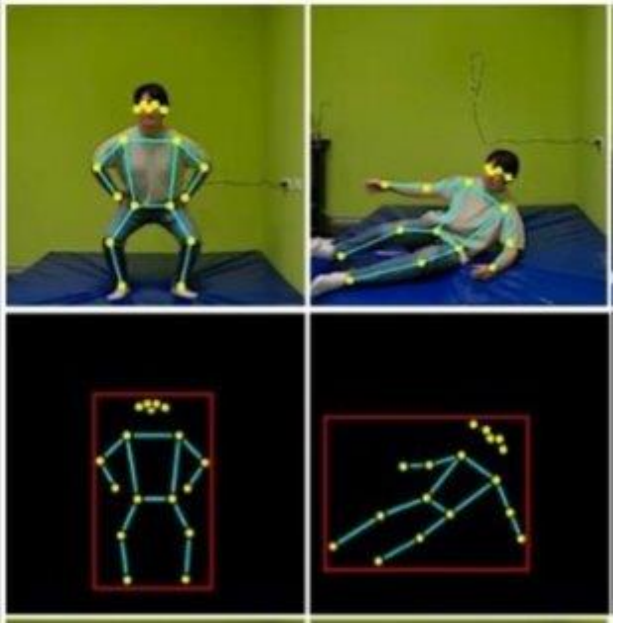
Vibration suppression



Exo-cobot collaboration



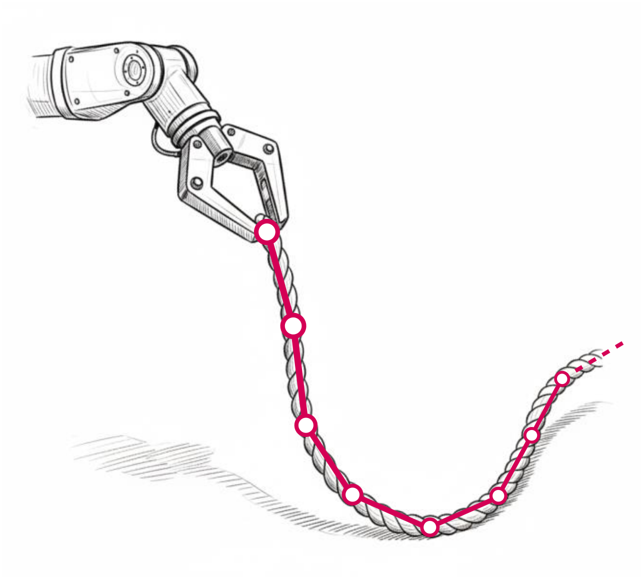
Collaborative pushing

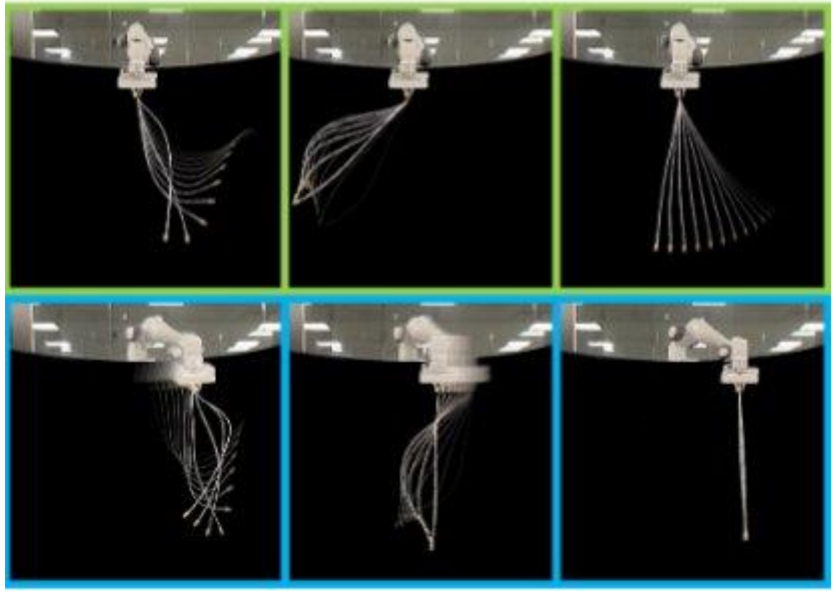


Fall detection

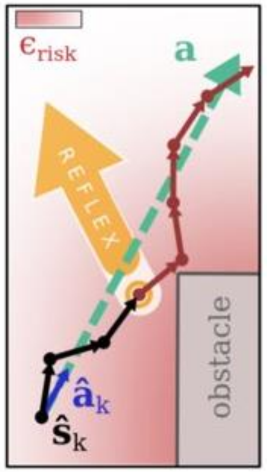
Contents

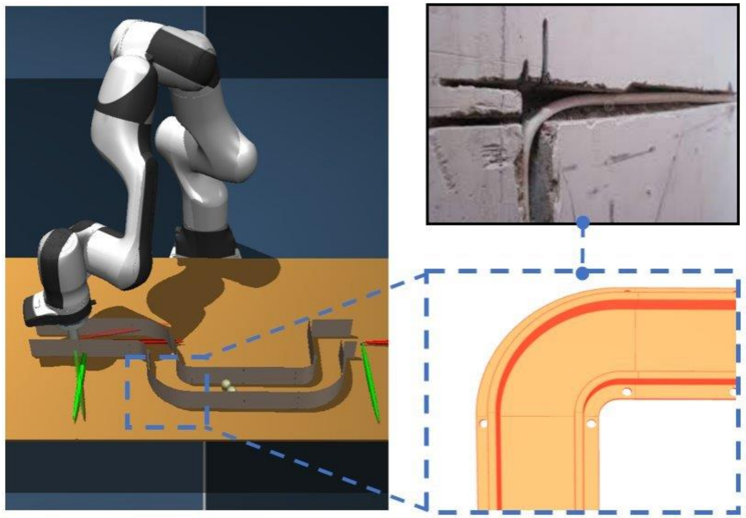
1. Physics-Informed Manipulation
 of Deformable Linear Objects





2. Learning safe interaction from data





3. Key takeaways

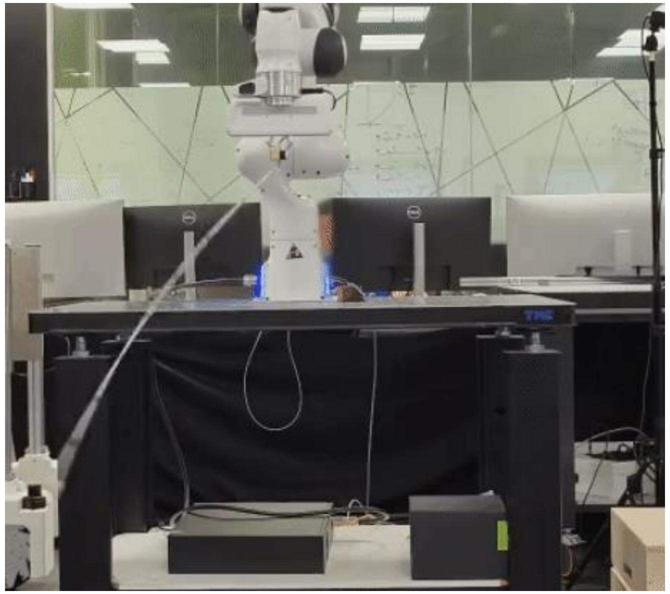
ISTITUTO ITALIANO
DI TECNOLOGIA
HUMAN-ROBOT INTERFACES
AND INTERACTION

Physics-Informed Manipulation of Deformable Linear Objects

Youyuan Long, Gokhan Solak, Sara Zeynalpour, Arash Ajoudani

Dynamic Manipulation of Deformable Linear Objects (DLOs)

- **Deformable objects**
  - Exhibit Infinite degree of freedom
  - Highly non-linear formulation with coupled physical effects (bending, twisting)
- **Dynamic manipulation**
  - Includes high-velocity actions
  - Inertial and time-varying effects play a dominant role
  - Strict control latency requirements



- Traditional optimization is difficult
- **Data-driven can answer these challenges**
- However,
  - Require high amounts of data
  - Suffer limited generalization

Model-based or Model-free?

Model-based

- Physically interpretable ✚
- Theoretical grounding ✚
- Data-efficient ✚

- High-dimensional optimization 🔻
- Slow inference 🔻

Model-free

- Expressive ✚
- Complex behaviors ✚
- Fast inference ✚

- Data-hungry 🔻
  - Expert demonstrations
  - Trial-error
- Overfitting to simulation 🔻

Physics-informed Self-supervised learning

We treat physics model as a differentiable simulator that supervises the neural controller.

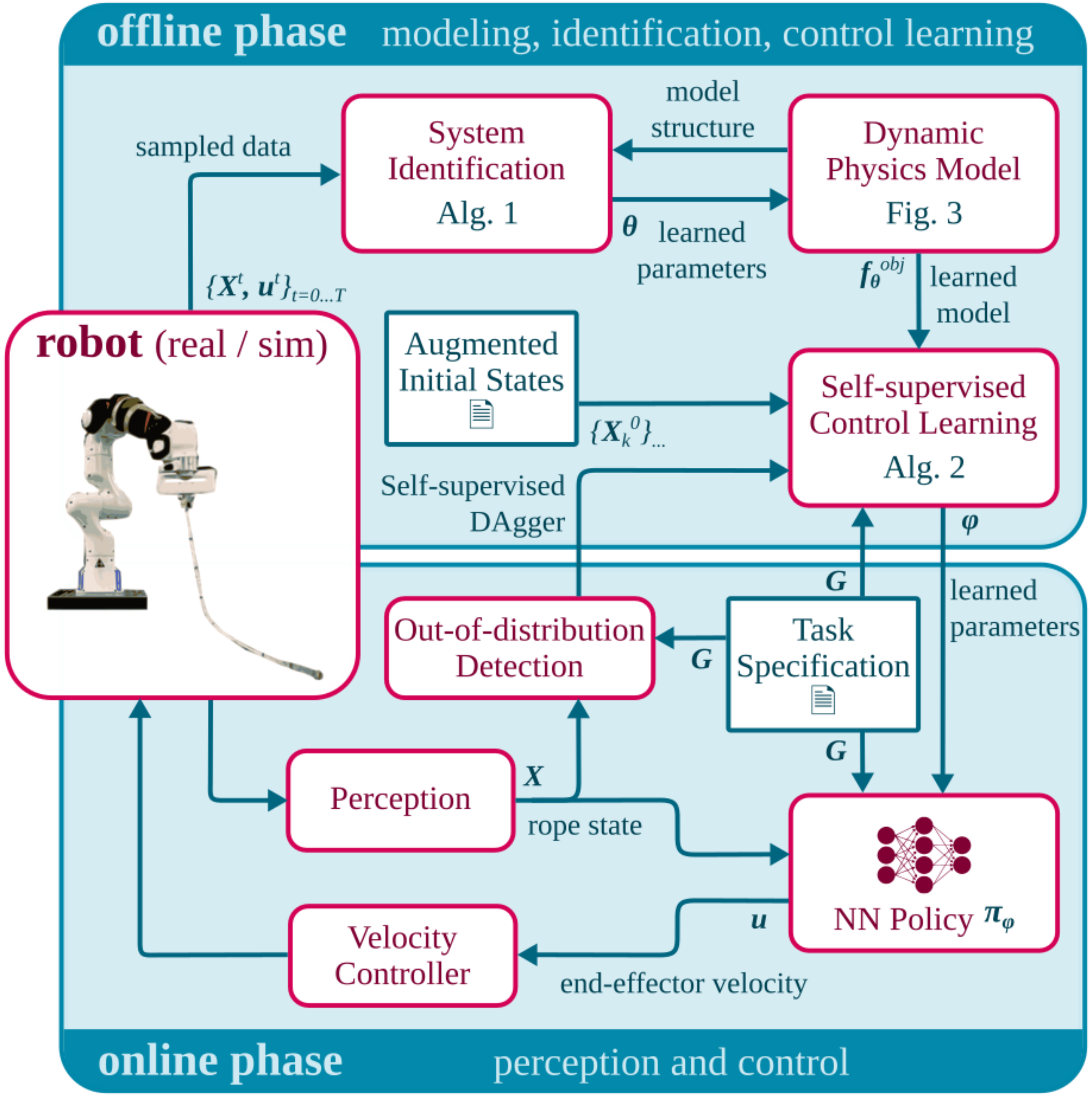
Identify a physics model of the object

- Physically interpretable ✚
- Data-efficient ✚

Learn a task-specific neural controller

- Complex behaviors ✚
- Fast inference ✚

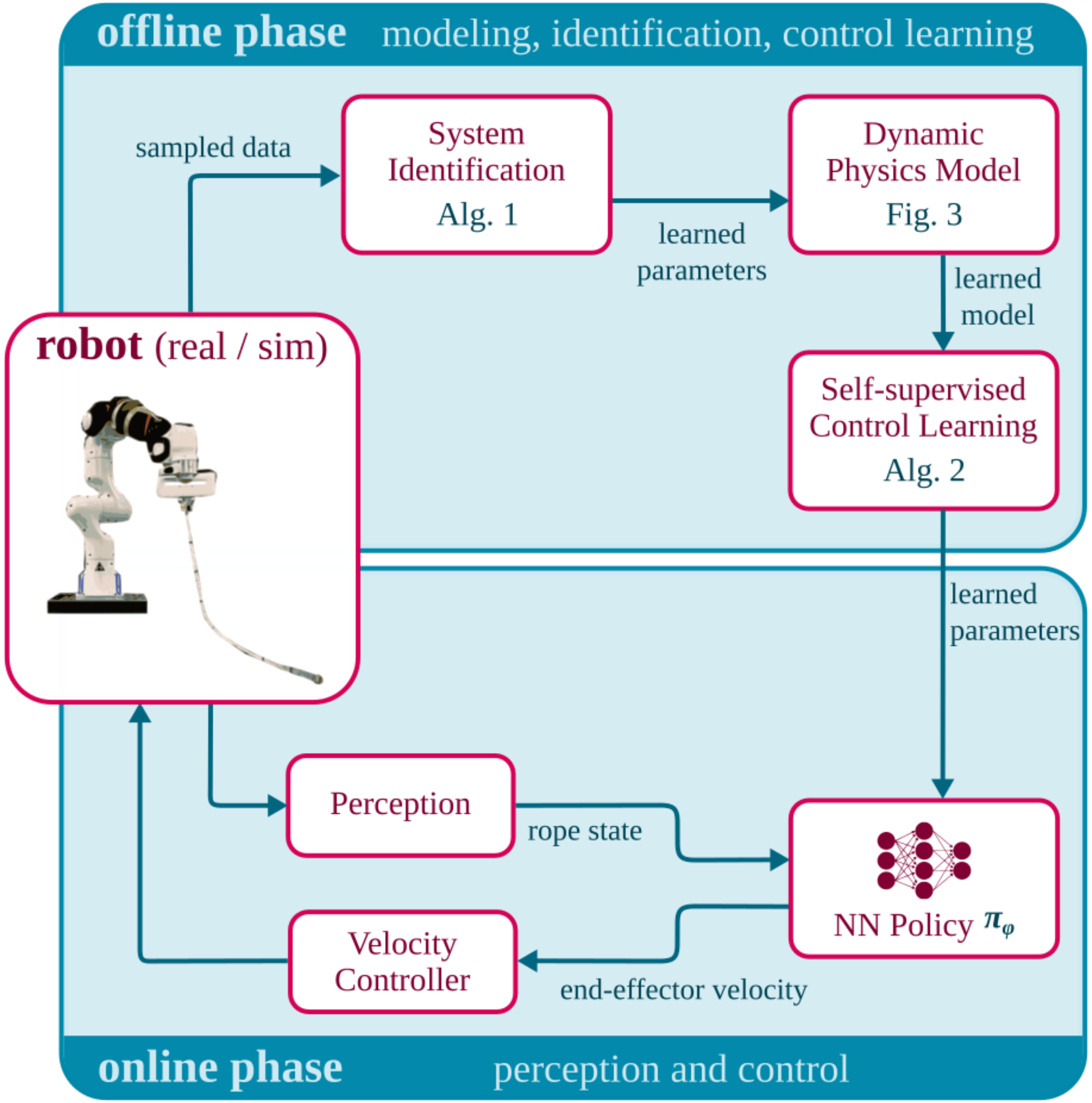
SPiD (Self-supervised Physics-informed Deformable Object Manipulation)



SPiD (Self-supervised Physics-informed Deformable Object Manipulation)

Let's remove

the details:

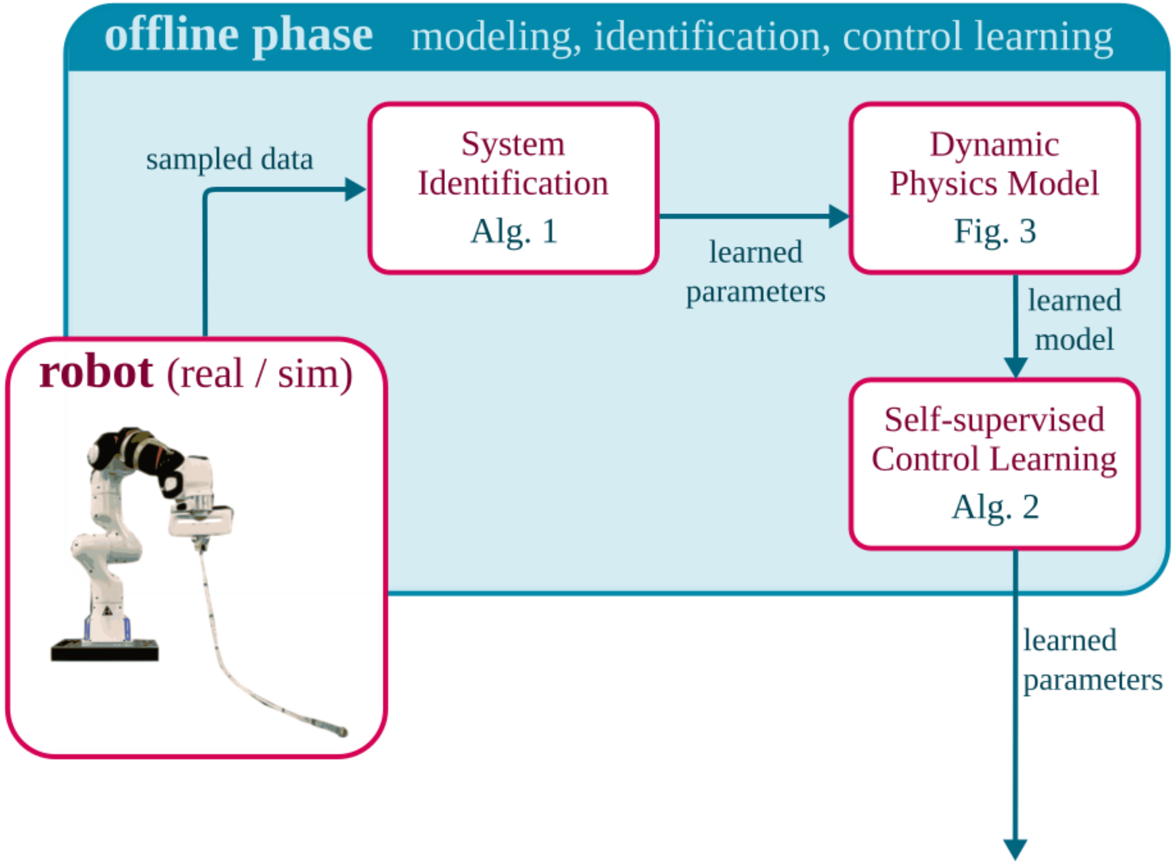


SPiD (Self-supervised Physics-informed Deformable Object Manipulation)

Learning

happens

offline

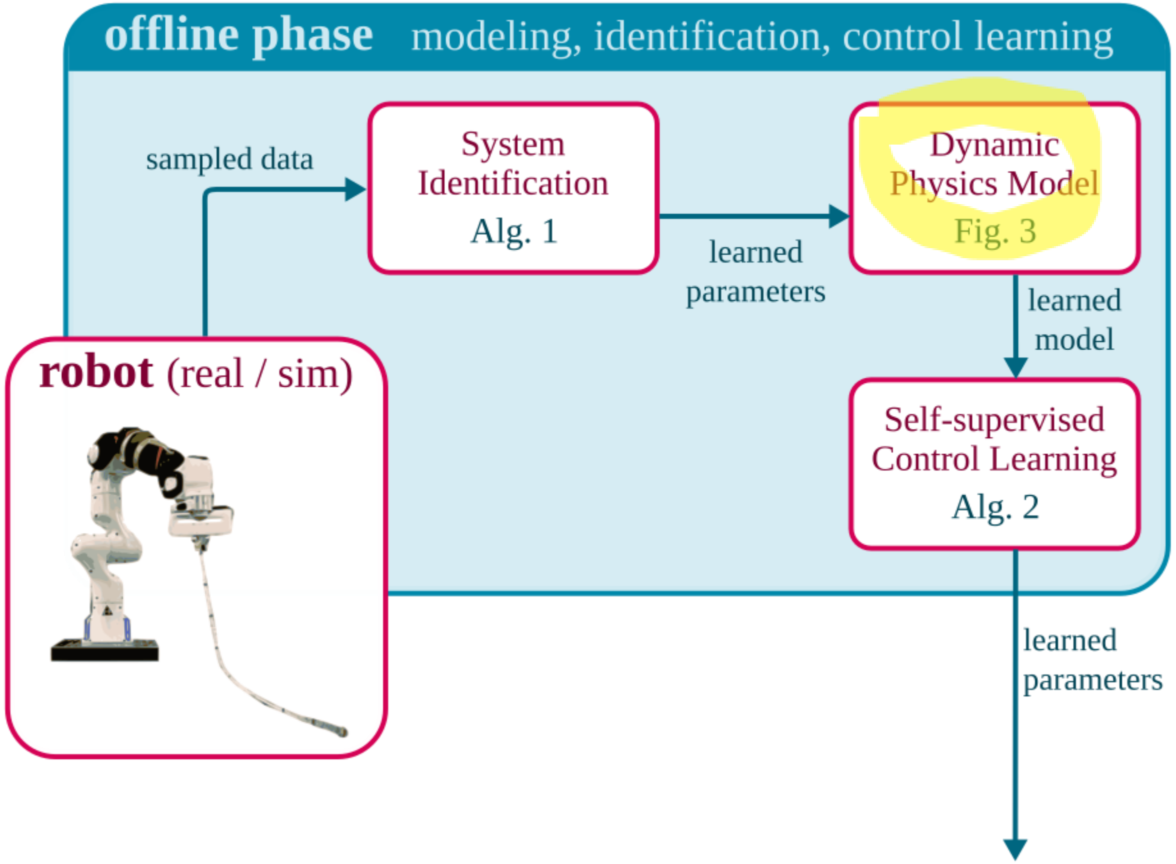


SPiD (Self-supervised Physics-informed Deformable Object Manipulation)

Learning

happens

offline

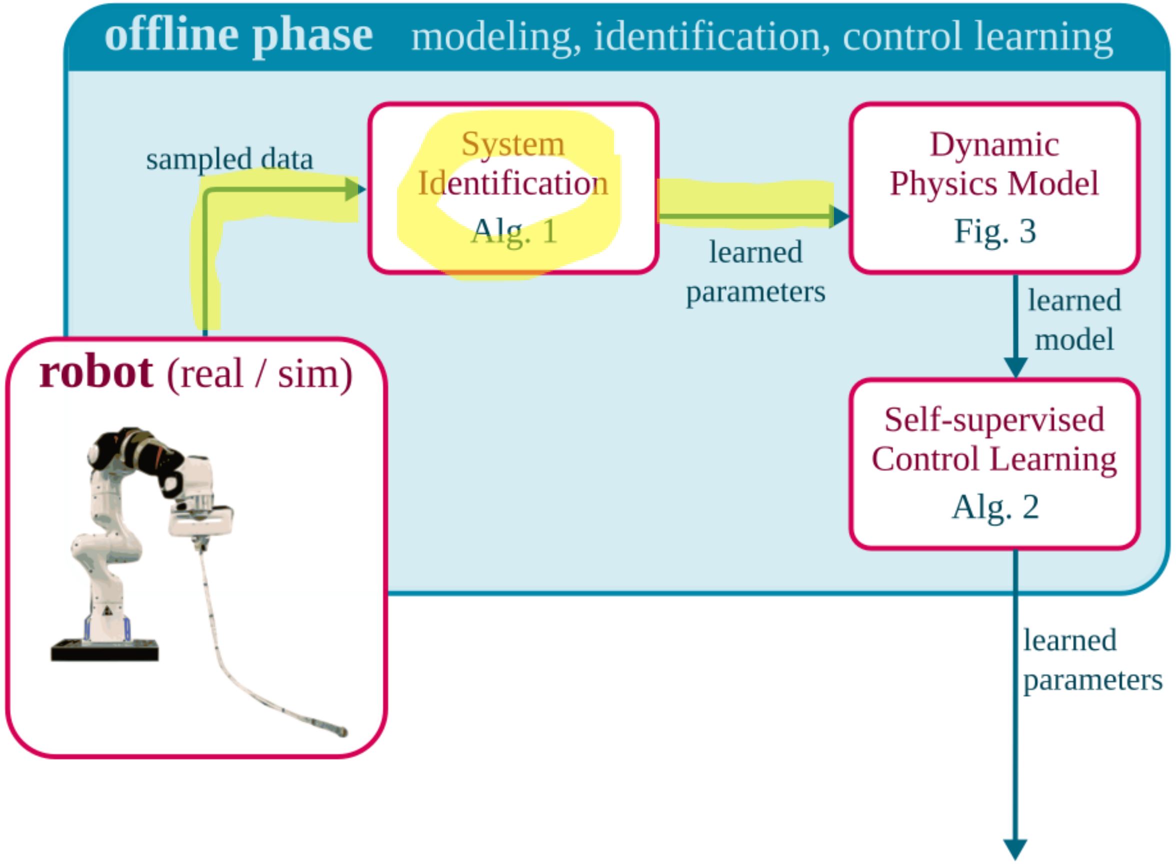


SPiD (Self-supervised Physics-informed Deformable Object Manipulation)

Learning

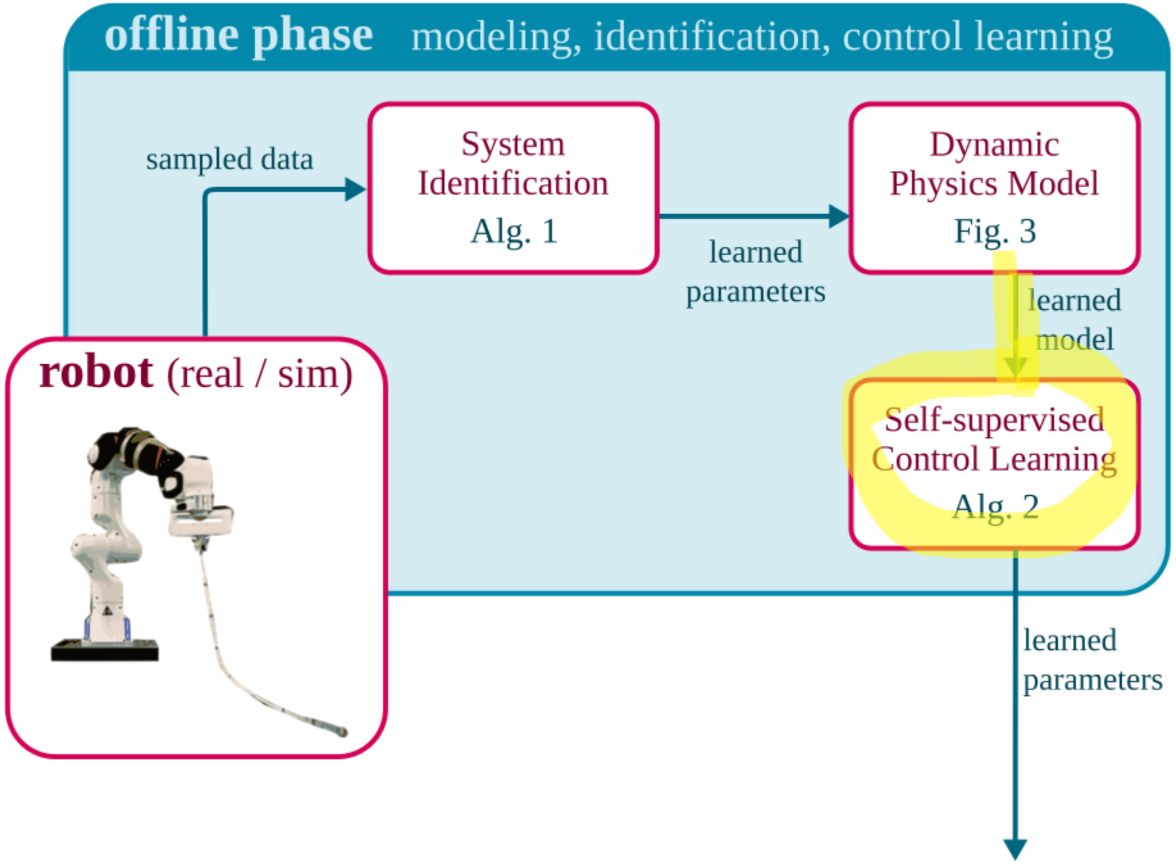
happens

offline



SPiD (Self-supervised Physics-informed Deformable Object Manipulation)

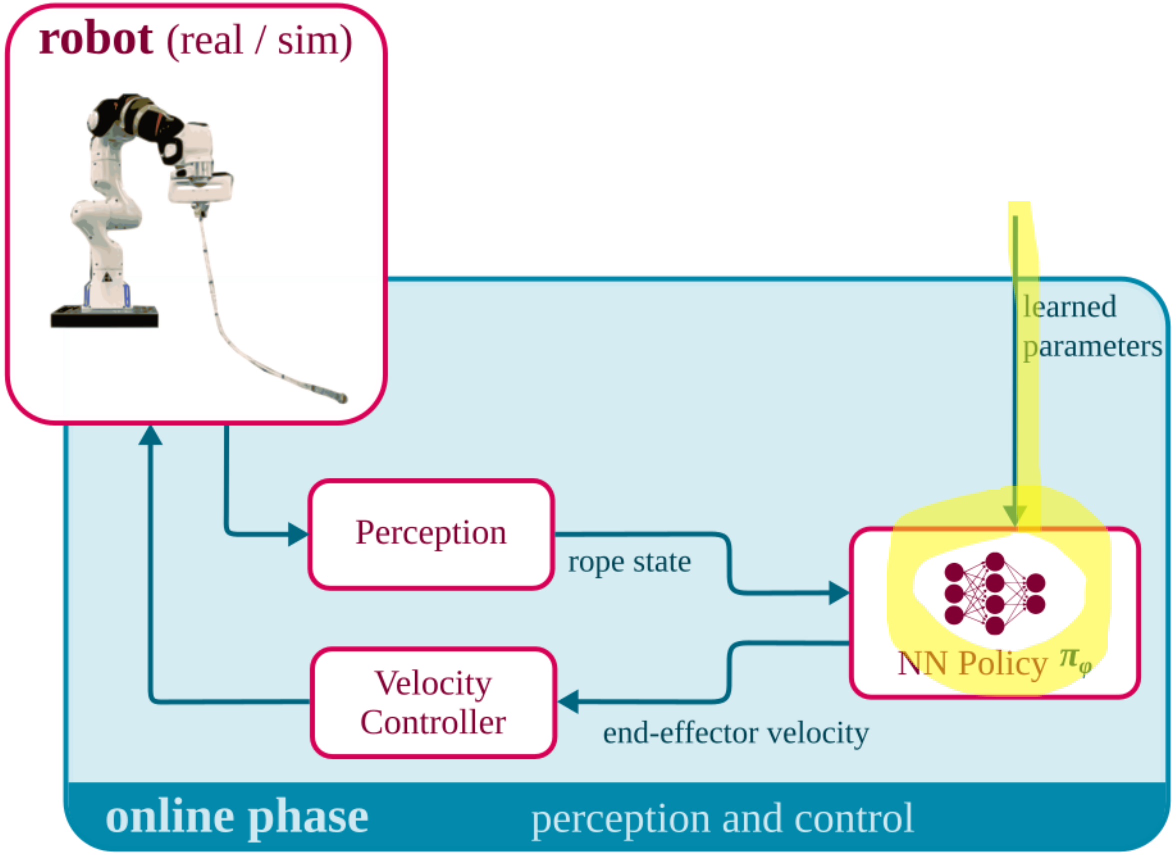
Learning happens offline



SPiD (Self-supervised Physics-informed Deformable Object Manipulation)

Neural controller

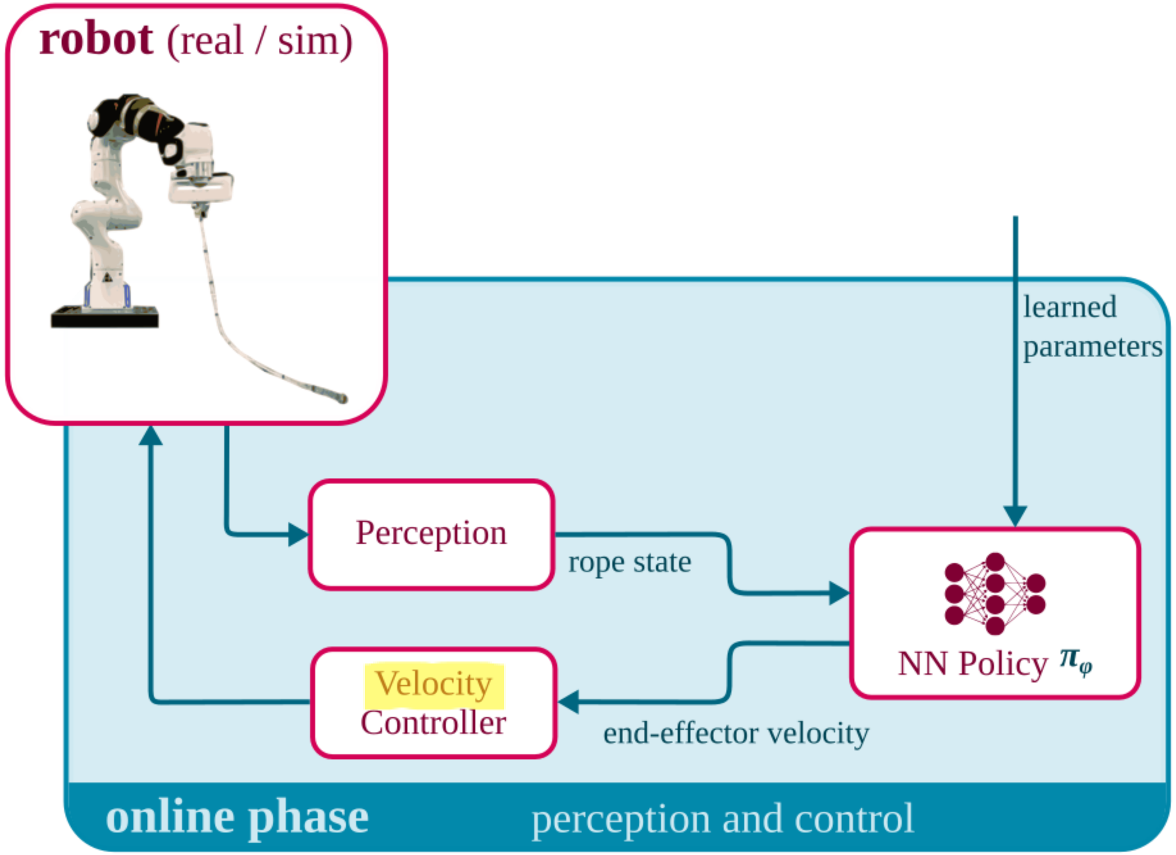
runs at 100 Hz



SPiD (Self-supervised Physics-informed Deformable Object Manipulation)

Neural controller

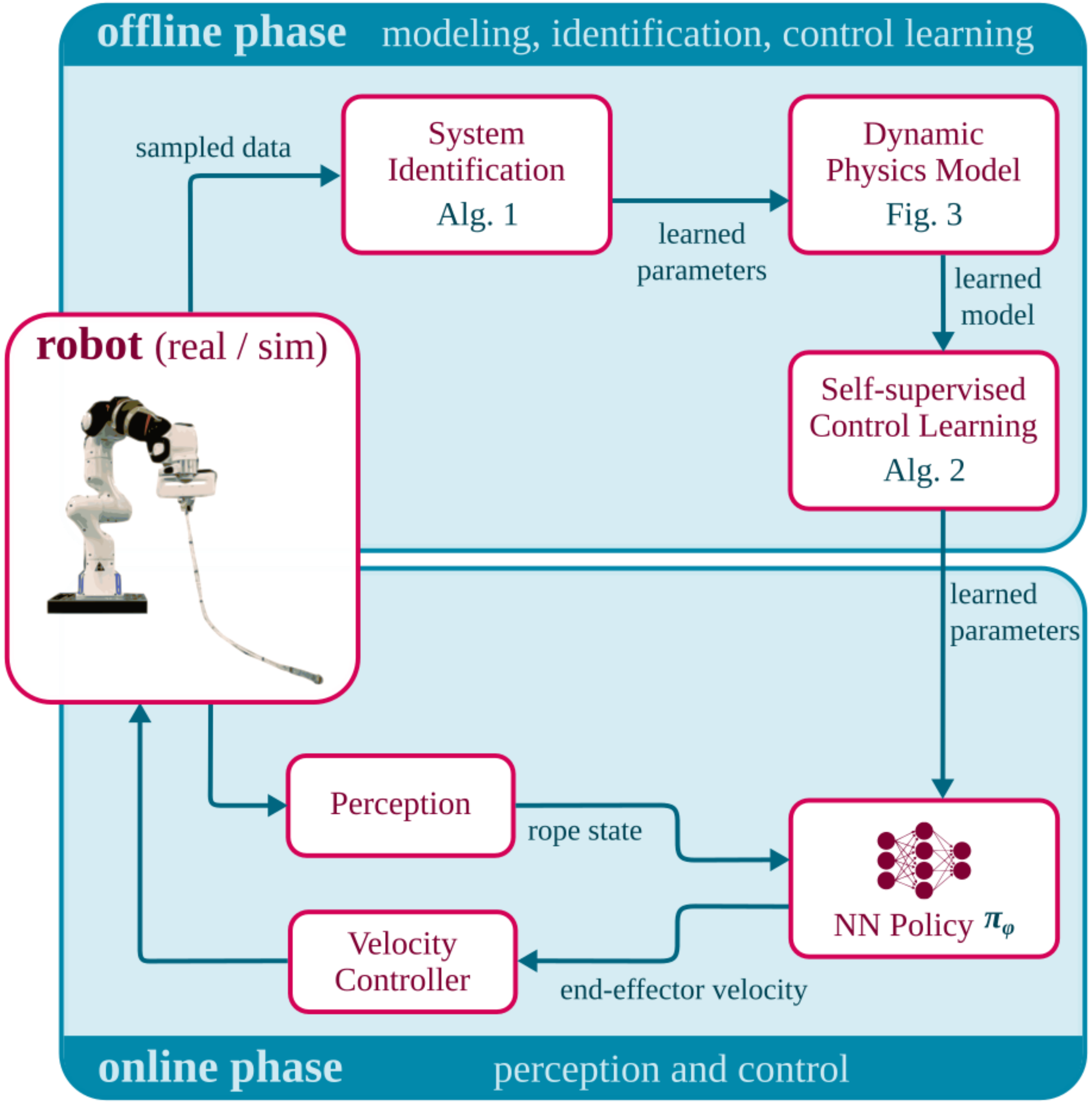
runs at 100 Hz



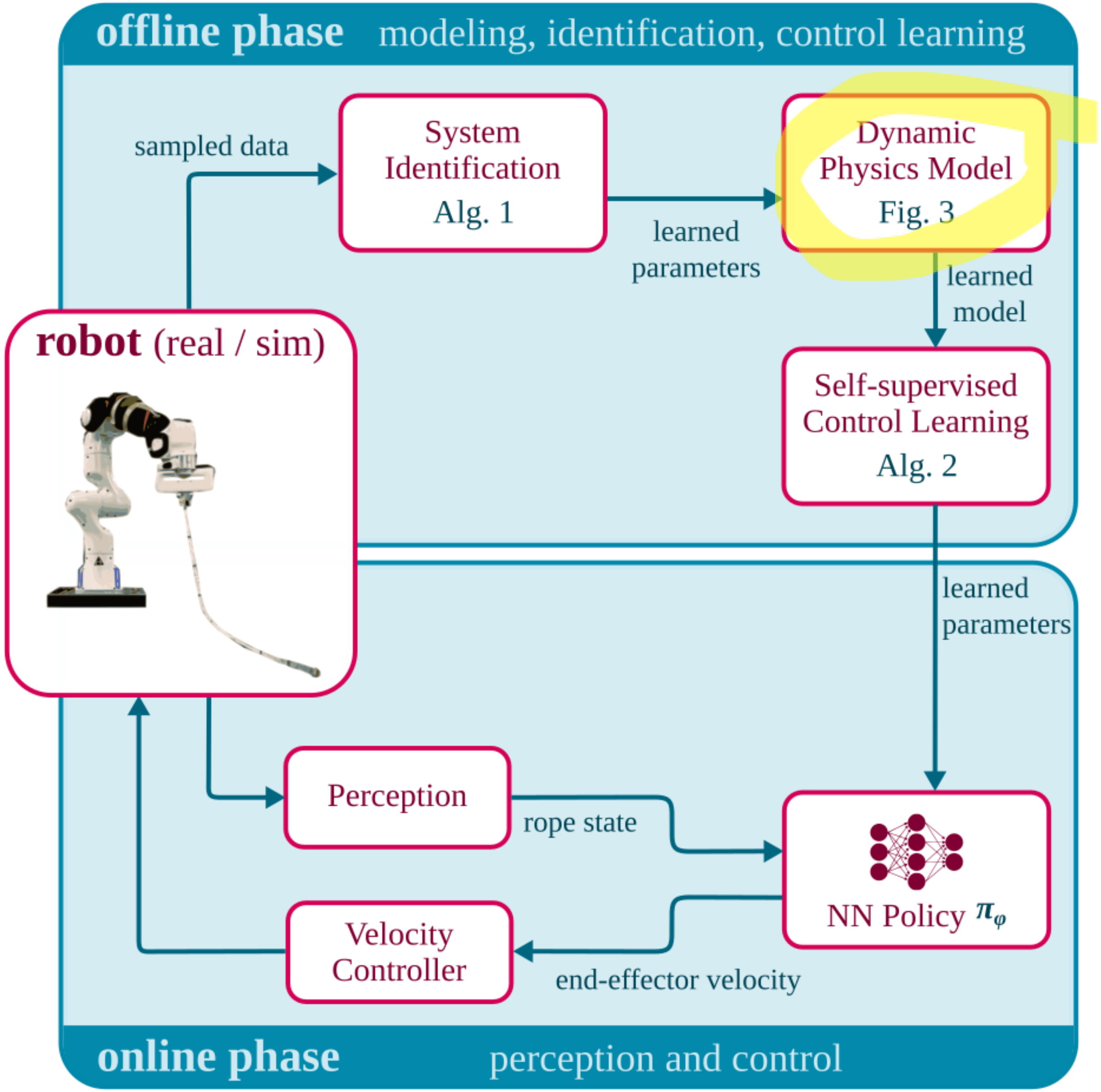
SPiD (Self-supervised Physics-informed Deformable Object Manipulation)

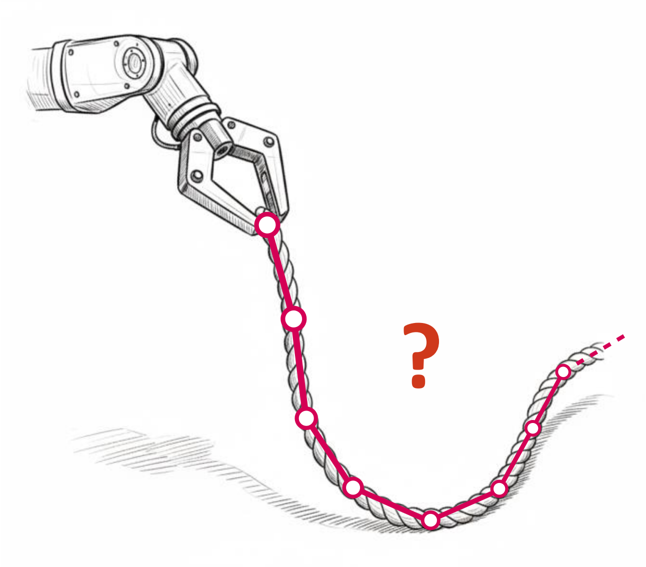
Let's look closer

at the components:



SPiD (Self-supervised Physics-informed Deformable Object Manipulation)





How to model the DLO?

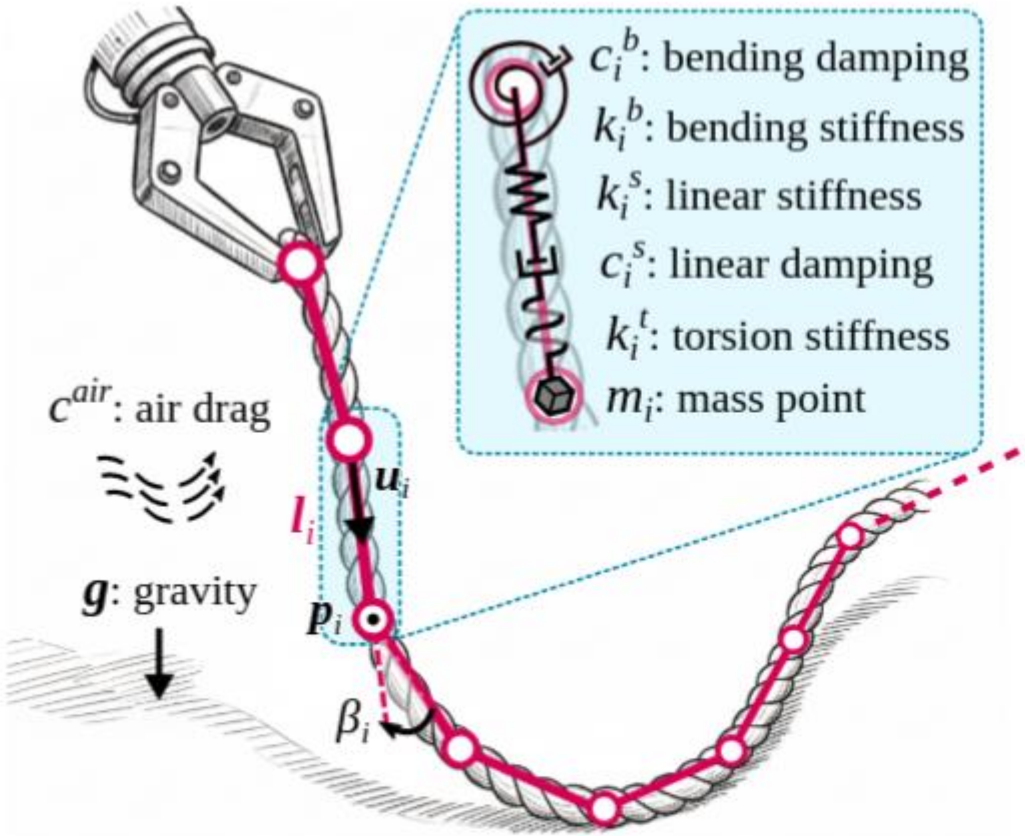
Modeling DLOs

Brief comparison

- **Finite elements method**
  - High precision, expensive computationally → challenging for real-time control
- **Point-based dynamics**
  - Real-time + stable and popular in graphics; tends to prioritize visual plausibility over physical fidelity
- **Elastic rods**
  - Physically accurate for DLOs; still computationally heavy
- **Mass-spring model**
  - Very fast, widely used, but accuracy limited by simplifications

- We chose **mass-spring model** for lightweight modeling for fast computation

Mass-spring-damper (MSD) model (**Ours**)

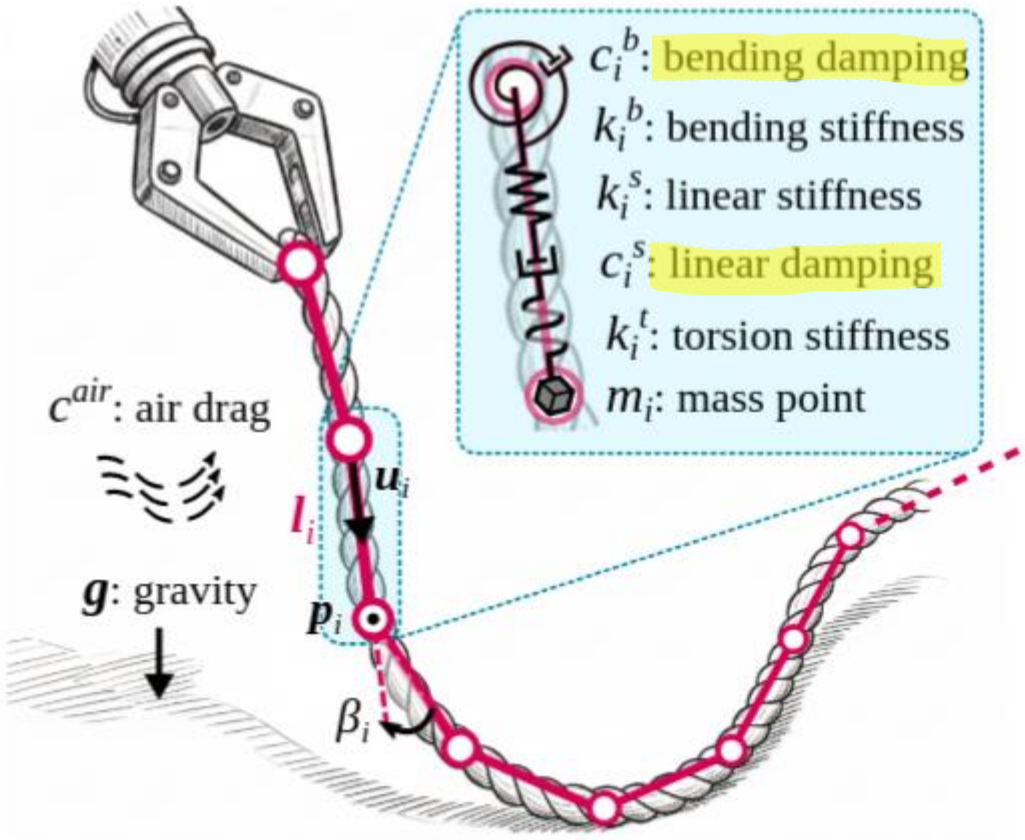


Rope state is a position-velocity pair $\boldsymbol{x}_i = [\boldsymbol{p}_i^\top, \boldsymbol{v}_i^\top]^\top$

- Each segment connects 2 mass points with 6 MSD elements
- 2 global parameters (air drag, gravity)
- We have N+1 mass points (N segments)
- In total N x 6 + 2 + 1 parameters

- **Stiffness**: configuration-dependent restoring forces (quasi-static)
- **Damping:** velocity-dependent dissipative forces (dynamic)

Mass-spring-damper (MSD) model (**Ours**)



Rope state is a position-velocity pair $\boldsymbol{x}_i = [\boldsymbol{p}_i^\top, \boldsymbol{v}_i^\top]^\top$

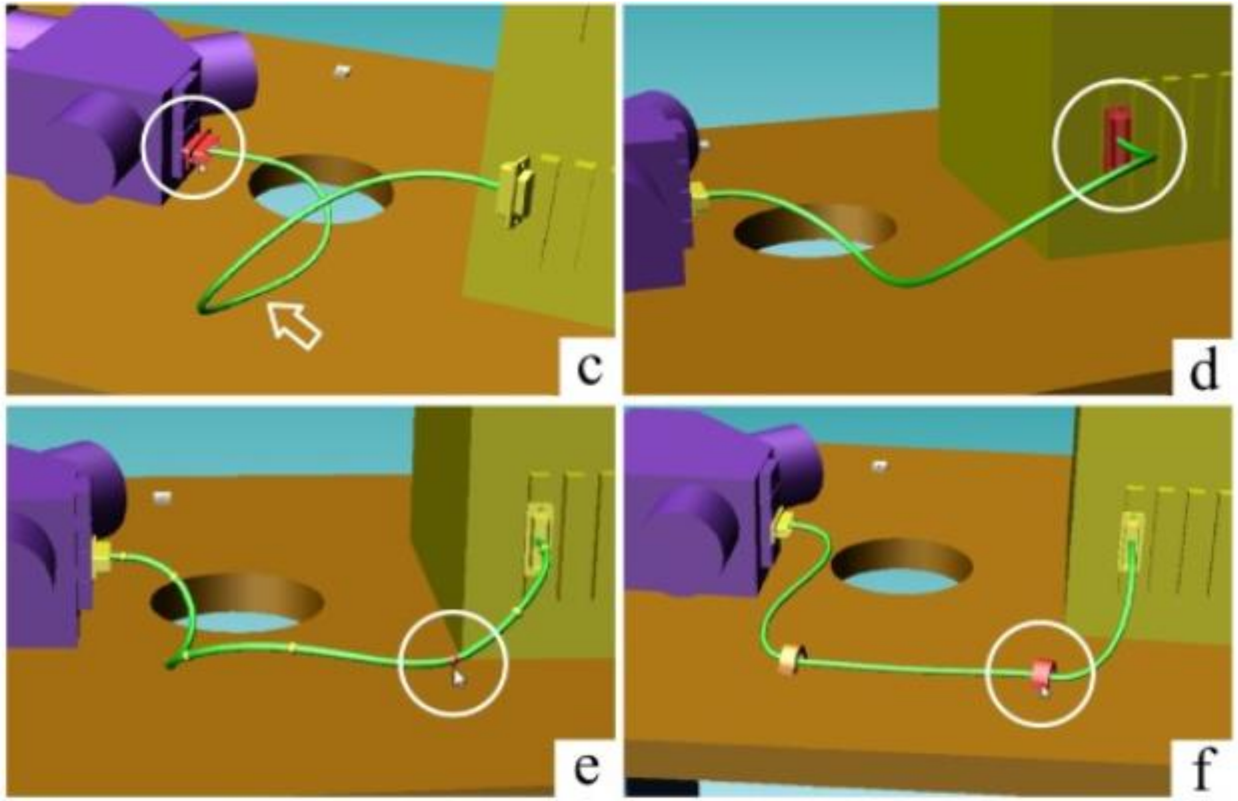
- Each segment connects 2 mass points with 6 MSD elements
- 2 global parameters (air drag, gravity)
- We have N+1 mass points (N segments)
- In total N x 6 + 2 + 1 parameters

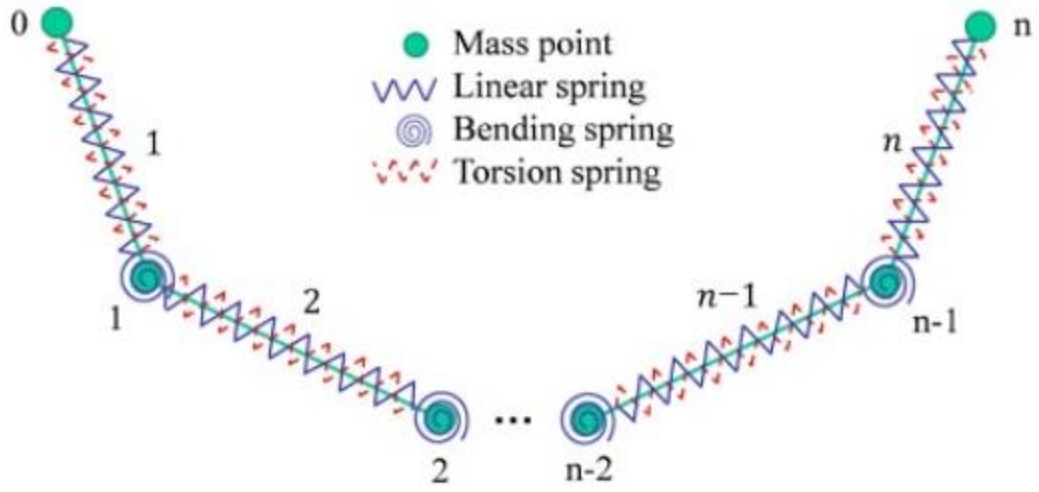
- **Stiffness**: configuration-dependent restoring forces (quasi-static)
- **Damping:** velocity-dependent dissipative forces (dynamic)

Novel

Mass-spring model (**baseline**)

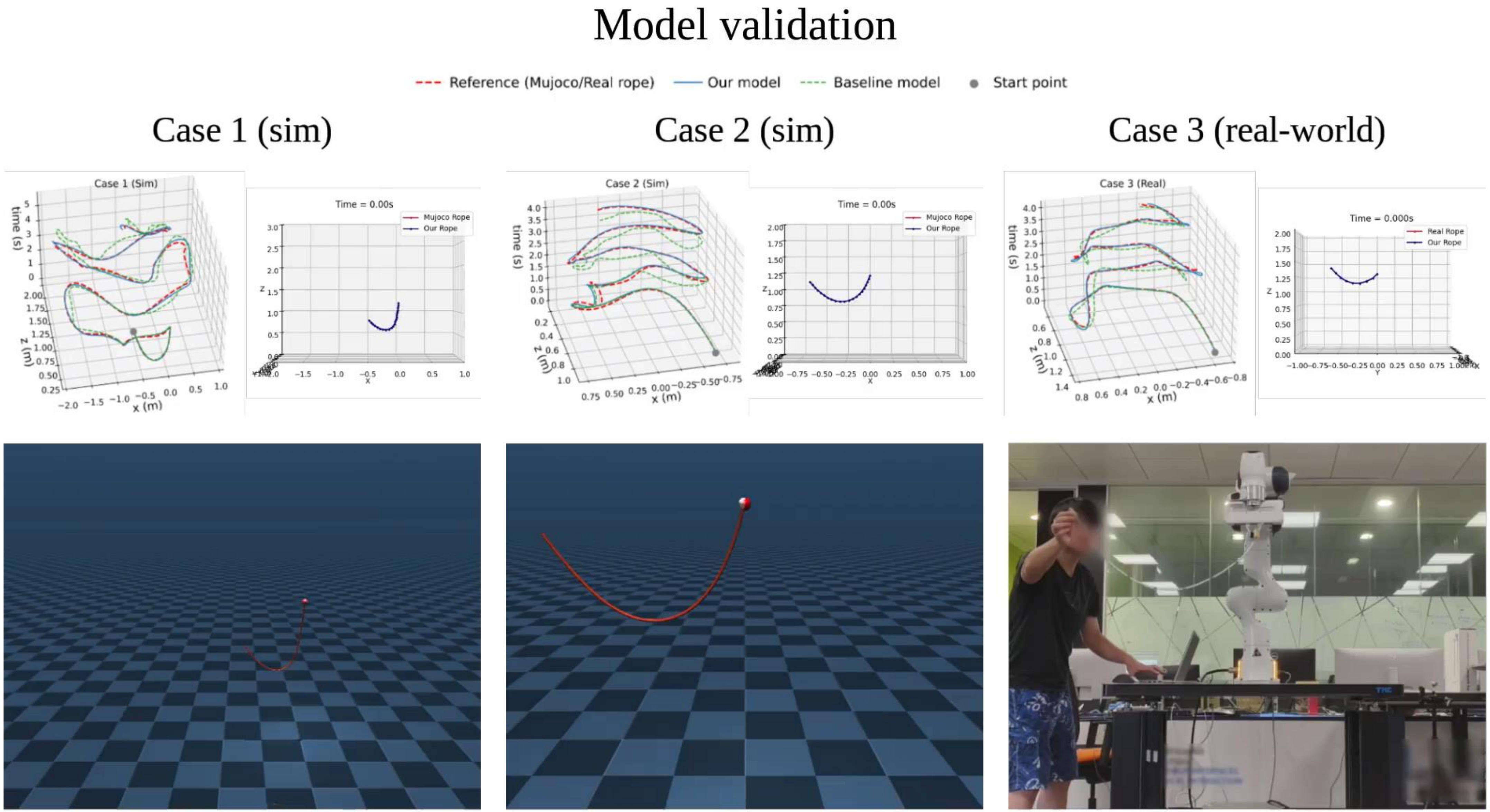
- The state-of-the-art mass-spring modeling method does not consider the damping terms.
- They focused on quasi-static cable manipulation.
- We extend their modeling approach with damping terms.



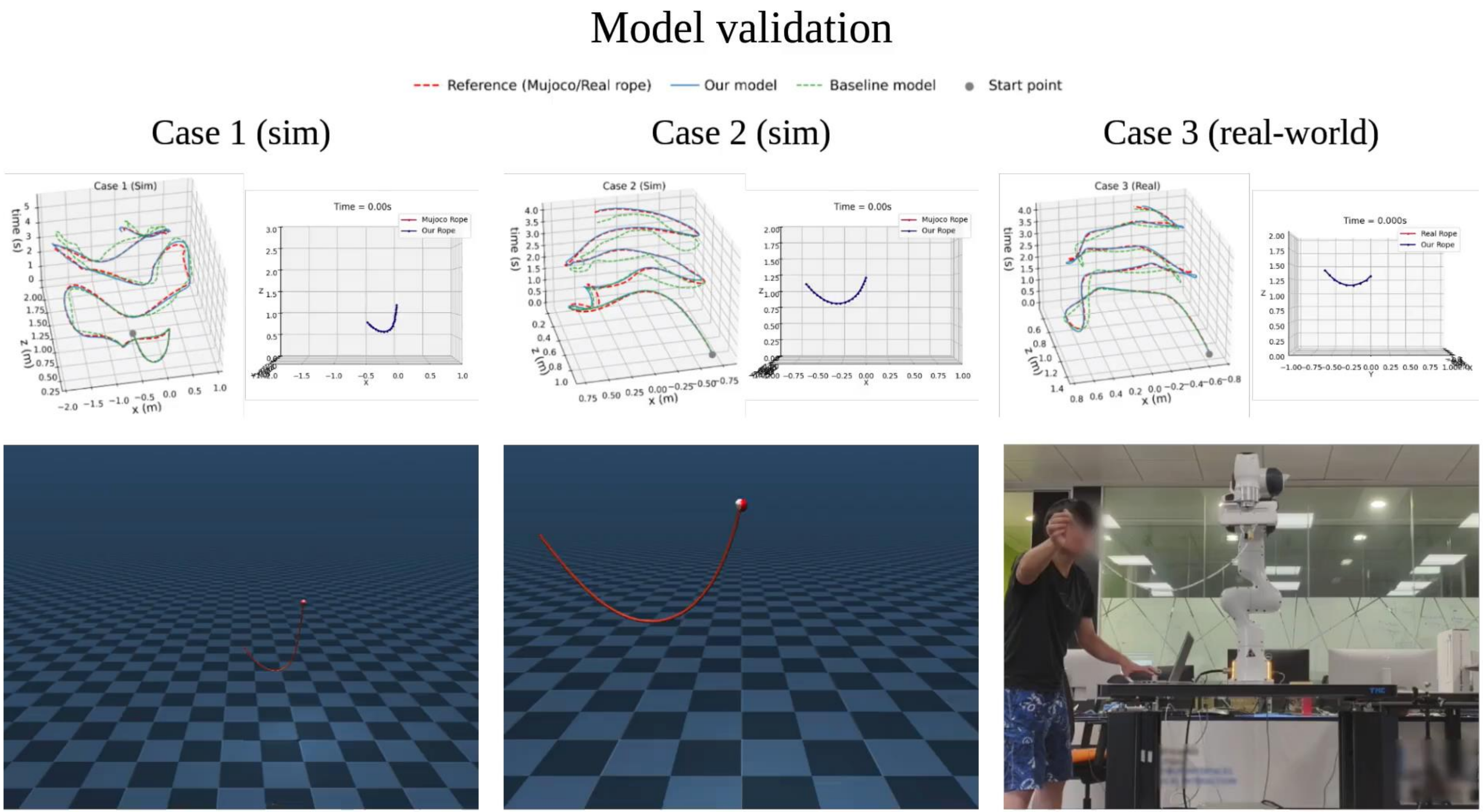


Lv, Naijing, et al. "Physically based real-time interactive assembly simulation of cable harness." *Journal of Manufacturing Systems* 43 (2017): 385-399.

Model validation

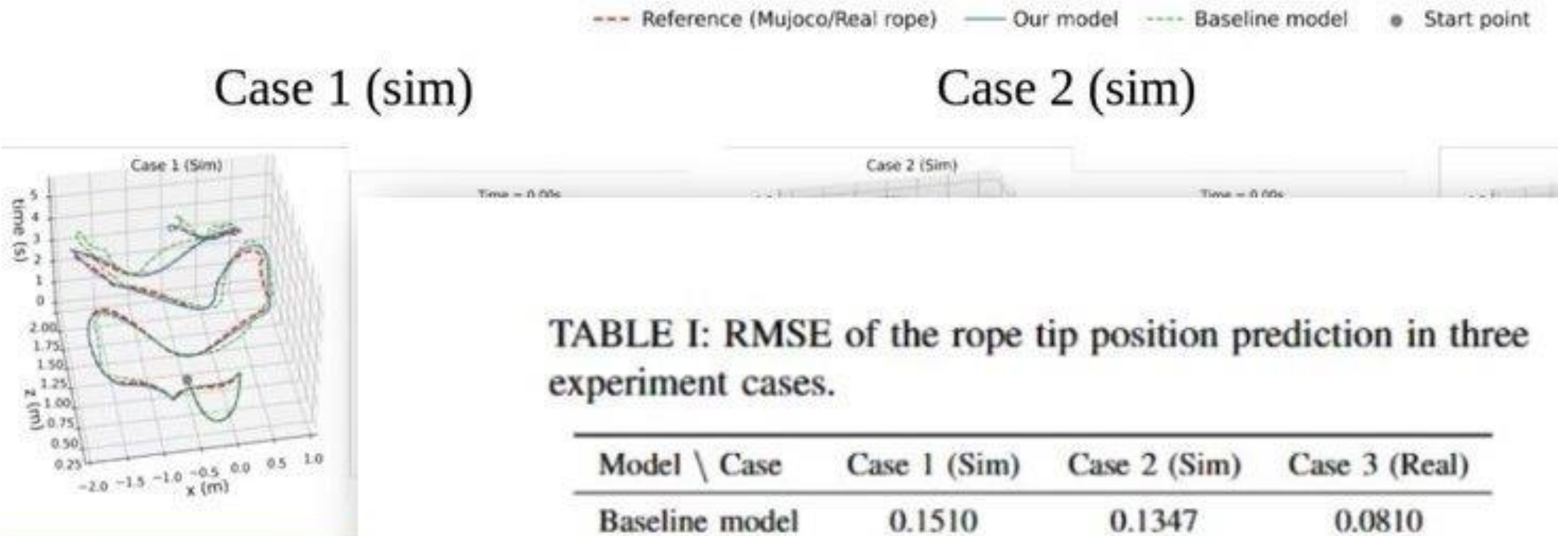


Model validation



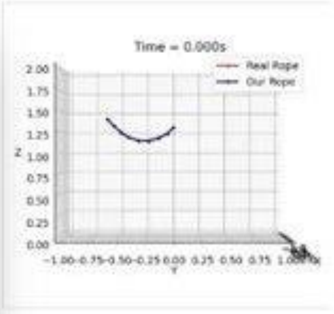
Model validation

Model validation



Case 2 (sim)

Case 3 (real-world)



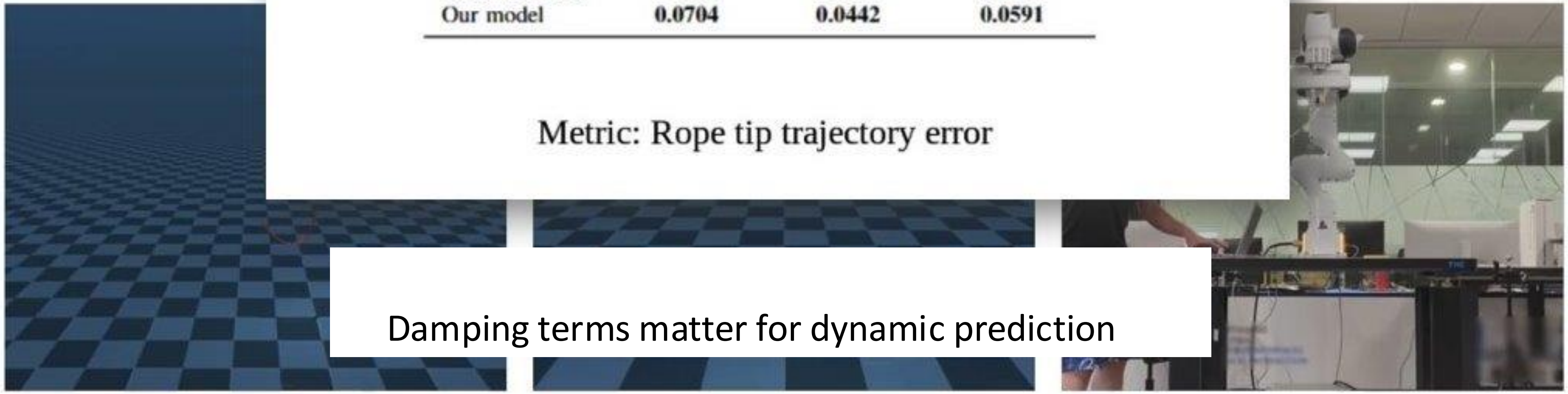


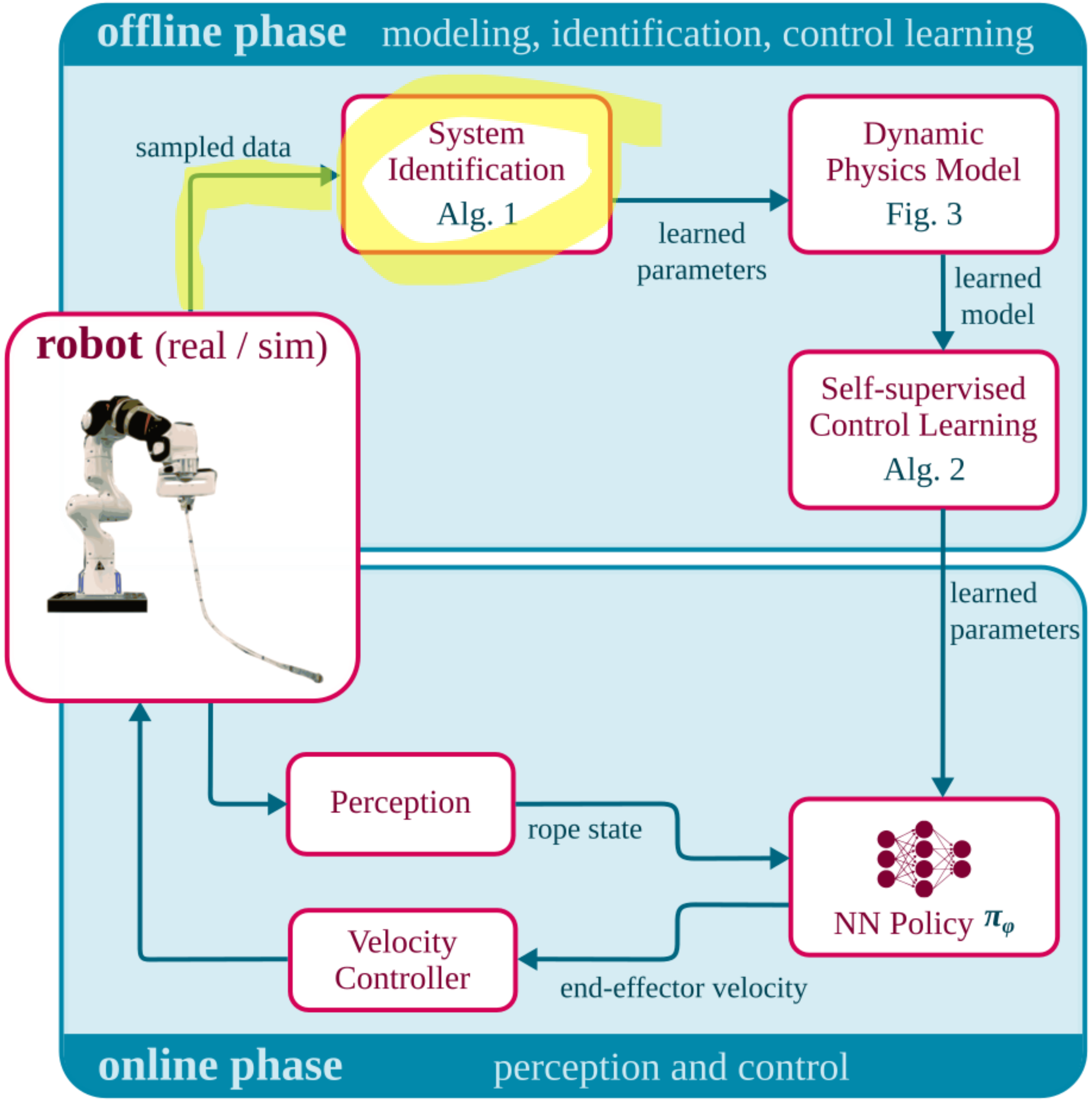
TABLE I: RMSE of the rope tip position prediction in three experiment cases.

Model \ Case	Case 1 (Sim)	Case 2 (Sim)	Case 3 (Real)
Baseline model	0.1510	0.1347	0.0810
Our model	**0.0704**	**0.0442**	**0.0591**

Metric: Rope tip trajectory error

Damping terms matter for dynamic prediction

SPiD (Self-supervised Physics-informed Deformable Object Manipulation)



Algorithm 1 Differentiable Physics System Identification

Input: Dataset $\{(\boldsymbol{P}^t, \boldsymbol{u}^t)\}_{t=0\ldots T}$, initial horizon H, horizon increment ΔH, sampling interval h, integration time step Δt, loss threshold ε

Output: Identified model parameters θ

Initialize parameters θ, $\hat{\boldsymbol{P}}^0 = \boldsymbol{P}^0$, $\hat{\boldsymbol{V}}^0 = \boldsymbol{0}$
Obtain $\hat{\boldsymbol{X}}^0$ by combining $\hat{\boldsymbol{P}}^0$ and $\hat{\boldsymbol{V}}^0$; set $\hat{\boldsymbol{X}}' \leftarrow \hat{\boldsymbol{X}}^0$
while $H \leq T$ **do**
  for $t = 0, \ldots, H$ **do**
    iterate $\hat{\boldsymbol{X}}' \leftarrow f^{\text{dlo}}_{\theta,\Delta t}(\hat{\boldsymbol{X}}', \boldsymbol{u}^t)$ **for** $h/\Delta t$ **steps**
    $\hat{\boldsymbol{X}}^{t+1} \leftarrow \hat{\boldsymbol{X}}'$? ▷ $\hat{\boldsymbol{X}}^{t+1}$ includes $\hat{\boldsymbol{P}}^{t+1}$
  end for
  $\boldsymbol{L_p} \leftarrow \frac{1}{H}\sum_{t=1}^{H} \|\hat{\boldsymbol{P}}^t - \boldsymbol{P}^t\|_2^2$
  Update $\theta \leftarrow \text{Adam}(\theta, \nabla_\theta \boldsymbol{L_p})$ ▷ Autodiff
  if $\boldsymbol{L_p} < \varepsilon$ **then**
    $H \leftarrow H + \Delta H$
  end if
end while

How to identify the model?

Differentiable System Identification

- **Goal**: learn the parameter set

$$\theta = \{\mathbf{m}, \boldsymbol{l}^0, \boldsymbol{g}, c^{air}, \boldsymbol{k}^s, \boldsymbol{c}^s, \boldsymbol{k}^b, \boldsymbol{c}^b, \boldsymbol{k}^t\}.$$

  Of the dynamics model

$$\boldsymbol{X}^{t+1} = \boldsymbol{f}^{\mathrm{dlo}}_{\theta,\Delta t}(\boldsymbol{X}^t, \boldsymbol{u}^t),$$

  That minimizes the loss

$$\boldsymbol{L_p} \leftarrow \frac{1}{H}\sum_{t=1}^{H} \|\hat{\boldsymbol{P}}^t - \boldsymbol{P}^t\|_2^2$$

  With the Dataset $\{(\boldsymbol{P}^t, \boldsymbol{u}^t)\}_{t=0\ldots T}$,
  Sampled from Mujoco Sim or real-robot setup

  Starting from a known initial state

Algorithm 1 Differentiable Physics System Identification

Input: Dataset $\{(\boldsymbol{P}^t, \boldsymbol{u}^t)\}_{t=0\ldots T}$, initial horizon H, horizon increment ΔH, sampling interval h, integration time step Δt, loss threshold ε

Output: Identified model parameters θ

Initialize parameters θ, $\hat{\boldsymbol{P}}^0 = \boldsymbol{P}^0$, $\hat{\boldsymbol{V}}^0 = \mathbf{0}$
Obtain $\hat{\boldsymbol{X}}^0$ by combining $\hat{\boldsymbol{P}}^0$ and $\hat{\boldsymbol{V}}^0$; set $\hat{\boldsymbol{X}}' \leftarrow \hat{\boldsymbol{X}}^0$
while $H \leq T$ **do**
    for $t = 0, \ldots, H$ **do**
        iterate $\hat{\boldsymbol{X}}' \leftarrow \boldsymbol{f}^{\mathrm{dlo}}_{\theta,\Delta t}(\hat{\boldsymbol{X}}', \boldsymbol{u}^t)$ **for** $h/\Delta t$ **steps**
        $\hat{\boldsymbol{X}}^{t+1} \leftarrow \hat{\boldsymbol{X}}'$ ▷ $\hat{\boldsymbol{X}}^{t+1}$ includes $\hat{\boldsymbol{P}}^{t+1}$
    end for
    $\boldsymbol{L_p} \leftarrow \frac{1}{H}\sum_{t=1}^{H} \|\hat{\boldsymbol{P}}^t - \boldsymbol{P}^t\|_2^2$
    Update $\theta \leftarrow \mathrm{Adam}(\theta, \nabla_\theta \boldsymbol{L_p})$ ▷ Autodiff
    if $\boldsymbol{L_p} < \varepsilon$ **then**
        $H \leftarrow H + \Delta H$
    end if
end while

Differentiable System Identification

- **Curriculum learning:**

  Expand the learned horizon gradually

  When the loss target is reached

$$\begin{array}{l}\textbf{if } \boldsymbol{L_p} < \varepsilon \textbf{ then}\\ \quad H \leftarrow H + \Delta H\end{array}$$

- Helps converging faster

Algorithm 1 Differentiable Physics System Identification

Input: Dataset $\{(\boldsymbol{P}^t, \boldsymbol{u}^t)\}_{t=0\dots T}$, initial horizon H, horizon increment ΔH, sampling interval h, integration time step Δt, loss threshold ε

Output: Identified model parameters θ

Initialize parameters θ, $\hat{\boldsymbol{P}}^0 = \boldsymbol{P}^0$, $\hat{\boldsymbol{V}}^0 = \boldsymbol{0}$

Obtain $\hat{\boldsymbol{X}}^0$ by combining $\hat{\boldsymbol{P}}^0$ and $\hat{\boldsymbol{V}}^0$; set $\hat{\boldsymbol{X}}' \leftarrow \hat{\boldsymbol{X}}^0$

while $H \leq T$ **do**

    for $t = 0, \dots, H$ **do**

        iterate $\hat{\boldsymbol{X}}' \leftarrow f^{\text{dlo}}_{\theta,\Delta t}(\hat{\boldsymbol{X}}', \boldsymbol{u}^t)$ **for** $h/\Delta t$ **steps**

        $\hat{\boldsymbol{X}}^{t+1} \leftarrow \hat{\boldsymbol{X}}'$ $\triangleright$ $\hat{\boldsymbol{X}}^{t+1}$ includes $\hat{\boldsymbol{P}}^{t+1}$

    end for

    $\boldsymbol{L_p} \leftarrow \frac{1}{H}\sum_{t=1}^{H} \|\hat{\boldsymbol{P}}^t - \boldsymbol{P}^t\|_2^2$

    Update $\theta \leftarrow \text{Adam}(\theta, \nabla_\theta \boldsymbol{L_p})$ $\triangleright$ Autodiff

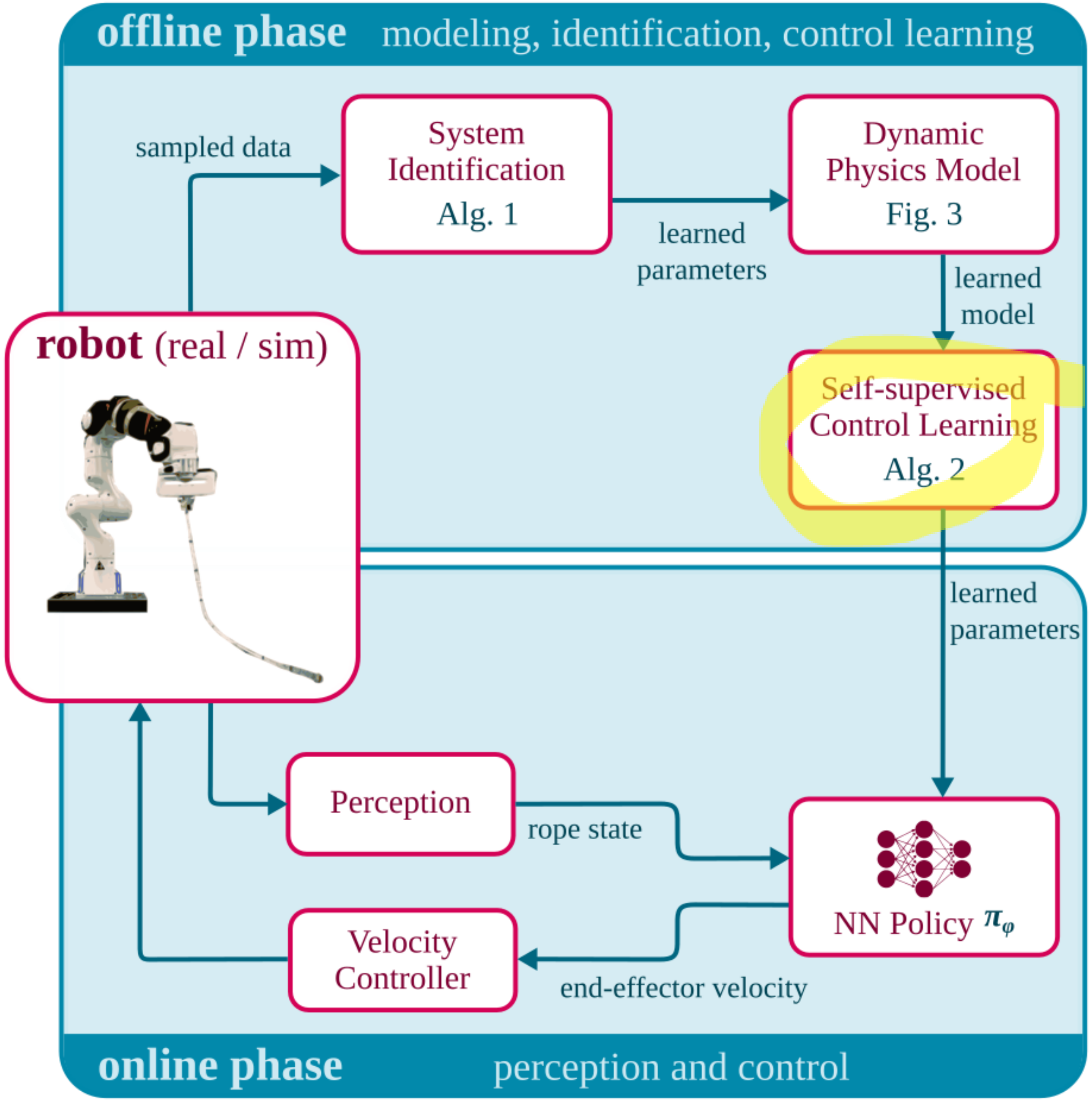
    if $\boldsymbol{L_p} < \varepsilon$ **then**

        $H \leftarrow H + \Delta H$

    end if

end while

SPiD (Self-supervised Physics-informed Deformable Object Manipulation)



Algorithm 2 Physics-informed Self-supervised Training

Input: Initial state set $\{\boldsymbol{X}_k^0\}_{k=1,\dots,K}$, task specification set $\{\boldsymbol{G}_k\}_{k=1,\dots,K}$, model parameters θ, integration time step Δt

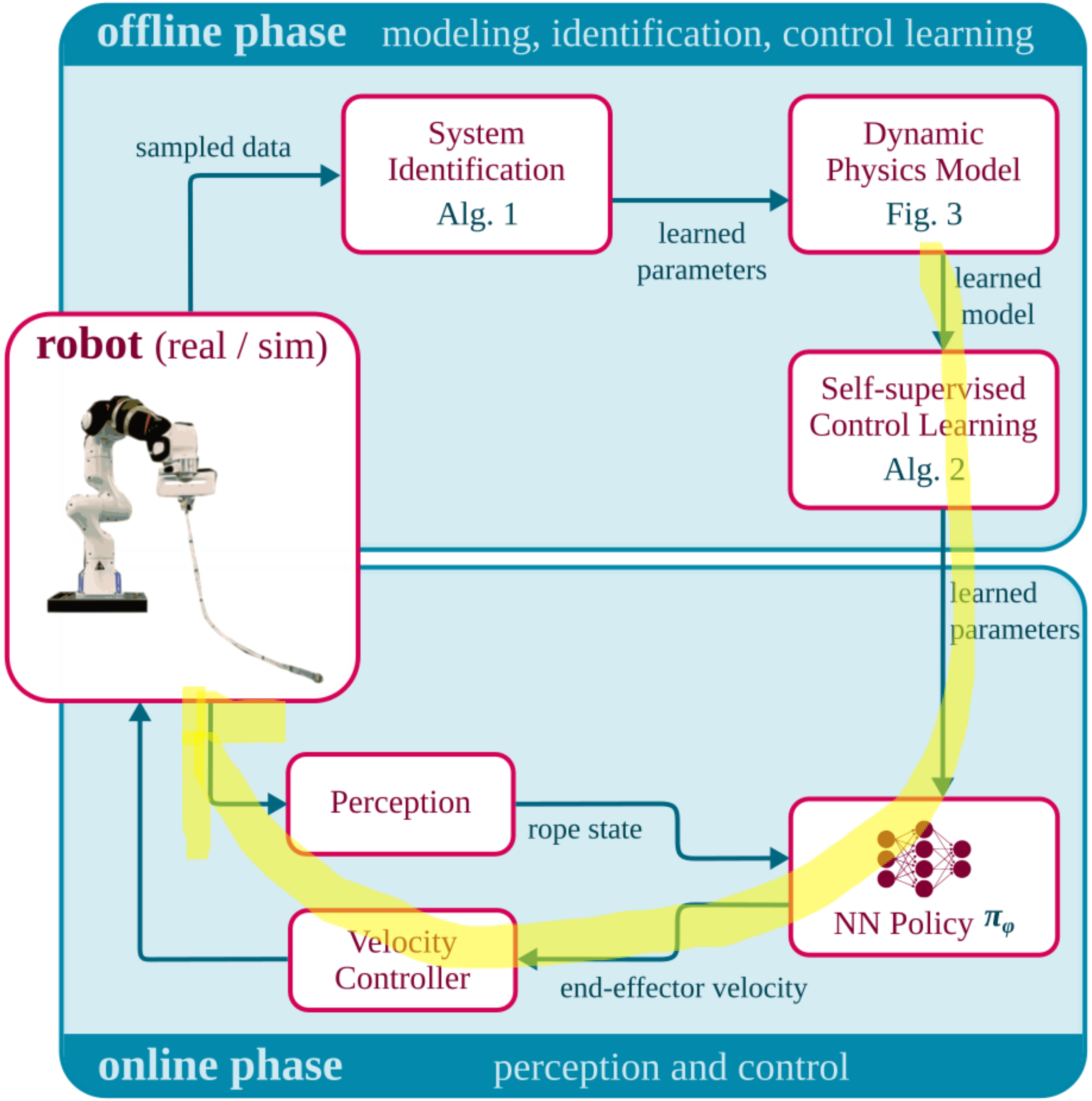
Output: Trained controller parameters ϕ

Initialize controller parameters ϕ
for each training iteration **do**
  $\{\Delta\theta_k\}_{k=1,\dots,K} \leftarrow \text{sample\_noise}(\theta)$
  for $t = 0, \dots, T-1$ **do**
    $\{\boldsymbol{u}_k^t\}_{k=1,\dots,K} \leftarrow \{\pi_\phi(\boldsymbol{X}_k^t, \boldsymbol{G}_k)\}_{k=1,\dots,K}$
    $\{\boldsymbol{X}_k^{t+1}\}_{k=1,\dots,K} \leftarrow \{\boldsymbol{f}_{\theta+\Delta\theta_k,\Delta t}^{\text{dlo}}(\boldsymbol{X}_k^t, \boldsymbol{u}_k^t)\}_{k=1,\dots,K}$
  end for
  $\boldsymbol{L} \leftarrow \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{L}_{\text{task}}(\{\boldsymbol{X}_k^t\}_{t=1}^{T}, \boldsymbol{G}_k)$ ▷ Task-based $\boldsymbol{L}_{\text{task}}(\cdot)$ as in (18) and (19)
  $\phi \leftarrow \text{Adam}(\phi, \nabla_\phi \boldsymbol{L})$ ▷ Autodiff
end for

?

How to learn the controller?

SPiD (Self-supervised Physics-informed Deformable Object Manipulation)



Algorithm 2 Physics-informed Self-supervised Training

Input: Initial state set $\{X_k^0\}_{k=1,\dots,K}$, task specification set $\{G_k\}_{k=1,\dots,K}$, model parameters θ, integration time step Δt
Output: Trained controller parameters ϕ

Initialize controller parameters ϕ
for each training iteration **do**
  $\{\Delta\theta_k\}_{k=1,\dots,K} \leftarrow \text{sample\_noise}(\theta)$
  for $t = 0, \dots, T-1$ **do**
    $\{u_k^t\}_{k=1,\dots,K} \leftarrow \{\pi_\phi(X_k^t, G_k)\}_{k=1,\dots,K}$
    $\{X_k^{t+1}\}_{k=1,\dots,K} \leftarrow \{f^{\text{dlo}}_{\theta+\Delta\theta_k,\Delta t}(X_k^t, u_k^t)\}_{k=1,\dots,K}$
  end for
  $L \leftarrow \frac{1}{K}\sum_{k=1}^{K} L_{\text{task}}(\{X_k^t\}_{t=1}^{T}, G_k)$ ▷ Task-based $L_{\text{task}}(\cdot)$ as in (18) and (19)
  $\phi \leftarrow \text{Adam}(\phi, \nabla_\phi L)$ ▷ Autodiff
end for

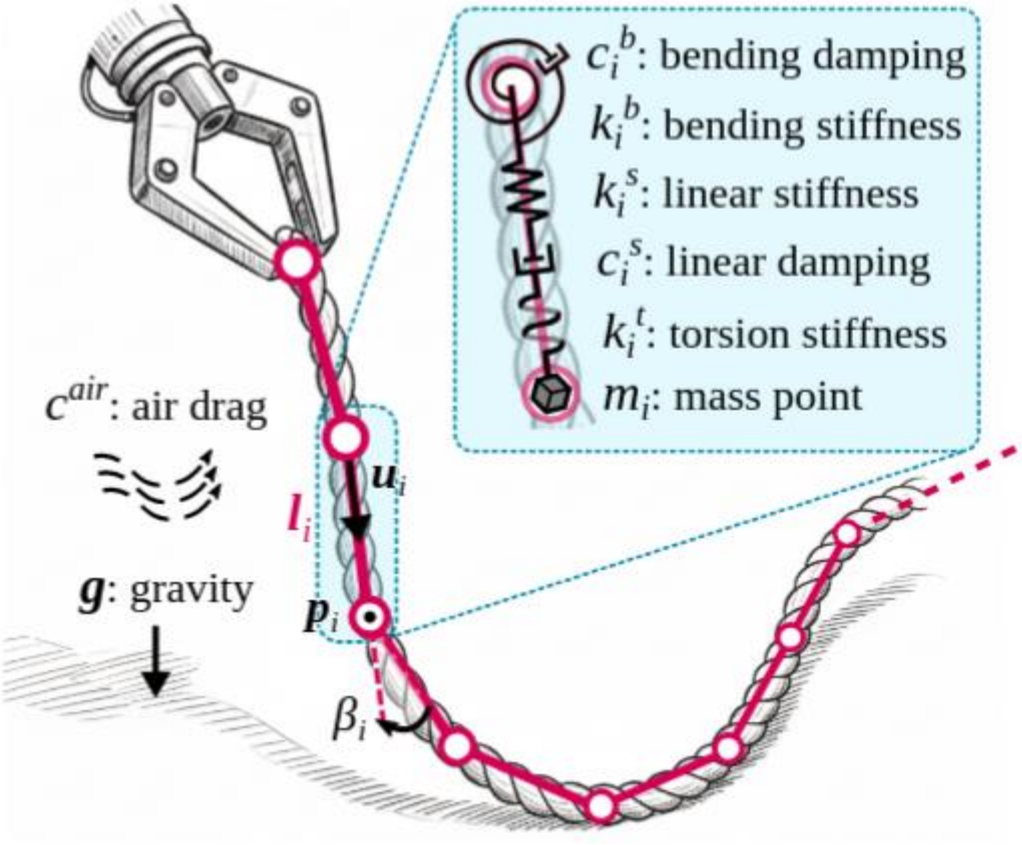
Why learn the controller?

Why not MPC?

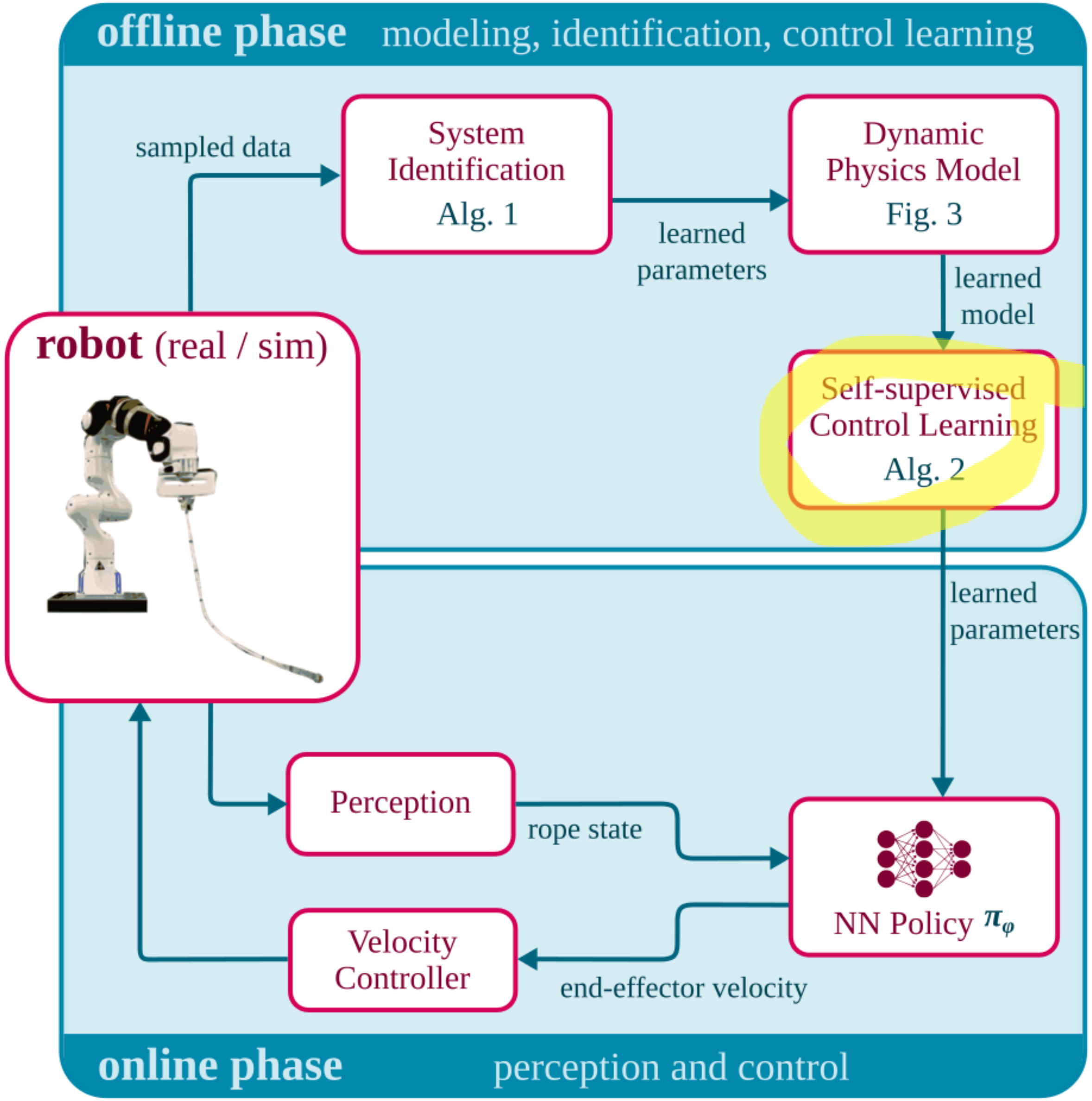
Why not model-predictive control?

- N x 6 + 2 + 1 parameters
- Can become 128-D state space for 21 segments

- Nonconvex dynamics
- Latency constraints



SPiD (Self-supervised Physics-informed Deformable Object Manipulation)



Algorithm 2 Physics-informed Self-supervised Training

Input: Initial state set $\{X_k^0\}_{k=1,\dots,K}$, task specification set $\{G_k\}_{k=1,\dots,K}$, model parameters θ, integration time step Δt
Output: Trained controller parameters ϕ

1: Initialize controller parameters ϕ
2: **for** each training iteration **do**
3: $\{\Delta\theta_k\}_{k=1,\dots,K} \leftarrow$ sample_noise(θ)
4: **for** $t = 0, \dots, T-1$ **do**
5: $\{u_k^t\}_{k=1,\dots,K} \leftarrow \{\pi_\phi(X_k^t, G_k)\}_{k=1,\dots,K}$
6: $\{X_k^{t+1}\}_{k=1,\dots,K} \leftarrow \{f^{\text{dlo}}_{\theta+\Delta\theta_k,\Delta t}(X_k^t, u_k^t)\}_{k=1,\dots,K}$
7: **end for**
8: $L \leftarrow \frac{1}{K}\sum_{k=1}^{K} L_{\text{task}}(\{X_k^t\}_{t=1}^{T}, G_k)$ ▷ Task-based $L_{\text{task}}(\cdot)$ as in (18) and (19)
9: $\phi \leftarrow \text{Adam}(\phi, \nabla_\phi L)$ ▷ Autodiff
10: **end for**

How to learn the controller?

How to learn the controller?

- Why not pure model-free RL?
  - Large data requirements
  - Our problem has high sample complexity (state-space dimensionality)

- Why not imitation learning?
  - Costly expert demonstrations
  - Hard to cover the large task space

- Why self-supervised learning?
  - We have a differentiable model (good gradients)
  - Low-cost data allows more exploration

* Combining self-supervision with RL is still interesting for long horizon tasks

Self-Supervised Neural Control Learning

- The policy interacts with the model

$$
\begin{aligned}
&\textbf{for } t = 0, \ldots, T-1 \textbf{ do} \\
&\quad \{\boldsymbol{u}_k^t\}_{k=1,\ldots,K} \leftarrow \{\pi_\phi(\boldsymbol{X}_k^t, \boldsymbol{G}_k)\}_{k=1,\ldots,K} \\
&\quad \{\boldsymbol{X}_k^{t+1}\}_{k=1,\ldots,K} \leftarrow \{\boldsymbol{f}^{\text{dlo}}_{\theta+\Delta\theta_k,\Delta t}(\boldsymbol{X}_k^t, \boldsymbol{u}_k^t)\}_{k=1,\ldots,K} \\
&\textbf{end for}
\end{aligned}
$$

- Learning to minimize the task-specific cost

$$\boldsymbol{L} \leftarrow \tfrac{1}{K}\textstyle\sum_{k=1}^{K} \boldsymbol{L}_{\text{task}}\left(\{\boldsymbol{X}_k^t\}_{t=1}^{T}, \boldsymbol{G}_k\right) \qquad \triangleright \text{ Task-based } \boldsymbol{L}_{\text{task}}(\cdot)$$

Rope stabilization:

$$\boldsymbol{L}\left(\{\boldsymbol{X}^t\}_{t=1}^{T}, \boldsymbol{G} = \emptyset\right) = E_{\text{rope}}(\boldsymbol{X}^T)$$

Rope trajectory tracking:

$$\boldsymbol{L}(\{\boldsymbol{X}^t\}_{t=1}^{T}, \boldsymbol{G}) = \sum_{t=1}^{T} \gamma^{T-t} \cdot \left\|\boldsymbol{p}_N^t - \boldsymbol{g}_{\text{goal}}^t\right\|_2^2$$

Algorithm 2 Physics-informed Self-supervised Training

Input: Initial state set $\{\boldsymbol{X}_k^0\}_{k=1,\ldots,K}$, task specification set $\{\boldsymbol{G}_k\}_{k=1,\ldots,K}$, model parameters θ, integration time step Δt

Output: Trained controller parameters ϕ

1: Initialize controller parameters ϕ
2: **for** each training iteration **do**
3: $\quad\{\Delta\theta_k\}_{k=1,\ldots,K} \leftarrow \text{sample\_noise}(\theta)$
4: $\quad$**for** $t = 0, \ldots, T-1$ **do**
5: $\quad\quad\{\boldsymbol{u}_k^t\}_{k=1,\ldots,K} \leftarrow \{\pi_\phi(\boldsymbol{X}_k^t, \boldsymbol{G}_k)\}_{k=1,\ldots,K}$
6: $\quad\quad\{\boldsymbol{X}_k^{t+1}\}_{k=1,\ldots,K} \leftarrow \{\boldsymbol{f}^{\text{dlo}}_{\theta+\Delta\theta_k,\Delta t}(\boldsymbol{X}_k^t, \boldsymbol{u}_k^t)\}_{k=1,\ldots,K}$
7: $\quad$**end for**
8: $\quad\boldsymbol{L} \leftarrow \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{L}_{\text{task}}(\{\boldsymbol{X}_k^t\}_{t=1}^{T}, \boldsymbol{G}_k)$ $\triangleright$ Task-based $\boldsymbol{L}_{\text{task}}(\cdot)$ as in (18) and (19)
9: $\quad\phi \leftarrow \text{Adam}(\phi, \nabla_\phi \boldsymbol{L})$ $\triangleright$ Autodiff
10: **end for**

Self-Supervised Neural Control Learning

- The policy interacts with the model

$$
\begin{aligned}
&\textbf{for } t = 0,\dots,T-1 \textbf{ do}\\
&\quad \{\boldsymbol{u}_k^t\}_{k=1,\dots,K} \leftarrow \{\pi_\phi(\boldsymbol{X}_k^t, \boldsymbol{G}_k)\}_{k=1,\dots,K}\\
&\quad \{\boldsymbol{X}_k^{t+1}\}_{k=1,\dots,K} \leftarrow \{\boldsymbol{f}^{\text{dlo}}_{\theta+\Delta\theta_k,\Delta t}(\boldsymbol{X}_k^t, \boldsymbol{u}_k^t)\}_{k=1,\dots,K}\\
&\textbf{end for}
\end{aligned}
$$

- Learning to minimize the task-specific cost

$$\boldsymbol{L} \leftarrow \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{L}_{\text{task}}\left(\{\boldsymbol{X}_k^t\}_{t=1}^{T}, \boldsymbol{G}_k\right)$$
$$\triangleright \text{ Task-based } \boldsymbol{L}_{\text{task}}(\cdot)$$

Rope stabilization:

$$\boldsymbol{L}\left(\{\boldsymbol{X}^t\}_{t=1}^{T}, \boldsymbol{G}=\emptyset\right) = E_{\text{rope}}(\boldsymbol{X}^T)$$

Rope trajectory tracking:

$$\boldsymbol{L}(\{\boldsymbol{X}^t\}_{t=1}^{T}, \boldsymbol{G}) = \sum_{t=1}^{T} \gamma^{T-t} \cdot \left\| \boldsymbol{p}_N^t - \boldsymbol{g}_{\text{goal}}^t \right\|_2^2$$

Algorithm 2 Physics-informed Self-supervised Training

Input: Initial state set $\{\boldsymbol{X}_k^0\}_{k=1,\dots,K}$, task specification set $\{\boldsymbol{G}_k\}_{k=1,\dots,K}$, model parameters θ, integration time step Δt

Output: Trained controller parameters ϕ

1: Initialize controller parameters ϕ
2: **for** each training iteration **do**
3: $\quad\{\Delta\theta_k\}_{k=1,\dots,K} \leftarrow \text{sample\_noise}(\theta)$
4: $\quad$**for** $t = 0,\dots,T-1$ **do**
5: $\quad\quad\{\boldsymbol{u}_k^t\}_{k=1,\dots,K} \leftarrow \{\pi_\phi(\boldsymbol{X}_k^t, \boldsymbol{G}_k)\}_{k=1,\dots,K}$
6: $\quad\quad\{\boldsymbol{X}_k^{t+1}\}_{k=1,\dots,K} \leftarrow \{\boldsymbol{f}^{\text{dlo}}_{\theta+\Delta\theta_k,\Delta t}(\boldsymbol{X}_k^t, \boldsymbol{u}_k^t)\}_{k=1,\dots,K}$
7: $\quad$**end for**
8: $\quad\boldsymbol{L} \leftarrow \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{L}_{\text{task}}\left(\{\boldsymbol{X}_k^t\}_{t=1}^{T}, \boldsymbol{G}_k\right)$
$\triangleright$ Task-based $\boldsymbol{L}_{\text{task}}(\cdot)$ as in (18) and (19)
9: $\quad\phi \leftarrow \text{Adam}(\phi, \nabla_\phi \boldsymbol{L})$ $\triangleright$ Autodiff
10: **end for**

How much should we train?

Self-Supervised Neural Control Learning

- The policy interacts with the model

for $t = 0, \ldots, T-1$ **do**

$\quad \{\boldsymbol{u}_k^t\}_{k=1,\ldots,K} \leftarrow \{\pi_\phi(\boldsymbol{X}_k^t, \boldsymbol{G}_k)\}_{k=1,\ldots,K}$

$\quad \{\boldsymbol{X}_k^{t+1}\}_{k=1,\ldots,K} \leftarrow \{\boldsymbol{f}^{\text{dlo}}_{\theta+\Delta\theta_k,\Delta t}(\boldsymbol{X}_k^t, \boldsymbol{u}_k^t)\}_{k=1,\ldots,K}$

end for

- Learning to minimize the task-specific cost

$$\boldsymbol{L} \leftarrow \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{L}_{\text{task}}\left(\{\boldsymbol{X}_k^t\}_{t=1}^T, \boldsymbol{G}_k\right) \quad \triangleright \text{ Task-based } \boldsymbol{L}_{\text{task}}(\cdot)$$

Rope stabilization:

$$\boldsymbol{L}\left(\{\boldsymbol{X}^t\}_{t=1}^T, \boldsymbol{G} = \emptyset\right) = E_{\text{rope}}(\boldsymbol{X}^T)$$

Rope trajectory tracking:

$$\boldsymbol{L}(\{\boldsymbol{X}^t\}_{t=1}^T, \boldsymbol{G}) = \sum_{t=1}^{T} \gamma^{T-t} \cdot \left\|\boldsymbol{p}_N^t - \boldsymbol{g}_{\text{goal}}^t\right\|_2^2$$

Algorithm 2 Physics-informed Self-supervised Training

Input: Initial state set $\{\boldsymbol{X}_k^0\}_{k=1,\ldots,K}$, task specification set $\{\boldsymbol{G}_k\}_{k=1,\ldots,K}$, model parameters θ, integration time step Δt

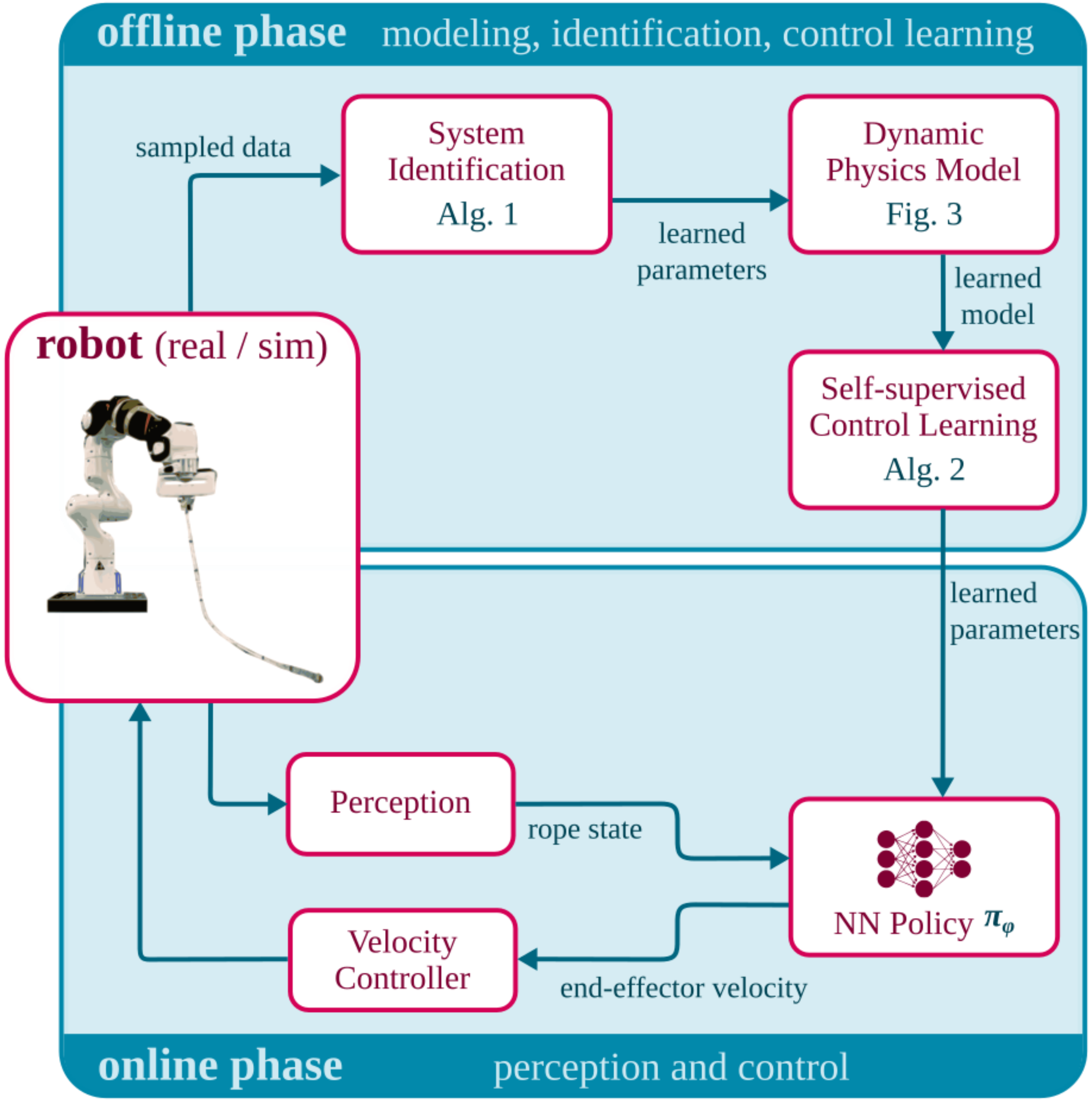
Output: Trained controller parameters ϕ

1: Initialize controller parameters ϕ
2: **for** each training iteration **do**
3: $\quad \{\Delta\theta_k\}_{k=1,\ldots,K} \leftarrow$ sample_noise(θ)
4: $\quad$ **for** $t = 0, \ldots, T-1$ **do**
5: $\quad\quad \{\boldsymbol{u}_k^t\}_{k=1,\ldots,K} \leftarrow \{\pi_\phi(\boldsymbol{X}_k^t, \boldsymbol{G}_k)\}_{k=1,\ldots,K}$
6: $\quad\quad \{\boldsymbol{X}_k^{t+1}\}_{k=1,\ldots,K} \leftarrow \{\boldsymbol{f}^{\text{dlo}}_{\theta+\Delta\theta_k,\Delta t}(\boldsymbol{X}_k^t, \boldsymbol{u}_k^t)\}_{k=1,\ldots,K}$
7: $\quad$ **end for**
8: $\quad \boldsymbol{L} \leftarrow \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{L}_{\text{task}}\left(\{\boldsymbol{X}_k^t\}_{t=1}^T, \boldsymbol{G}_k\right)$ $\triangleright$ Task-based $\boldsymbol{L}_{\text{task}}(\cdot)$ as in (18) and (19)
9: $\quad \phi \leftarrow \text{Adam}(\phi, \nabla_\phi \boldsymbol{L})$ $\triangleright$ Autodiff
10: **end for**

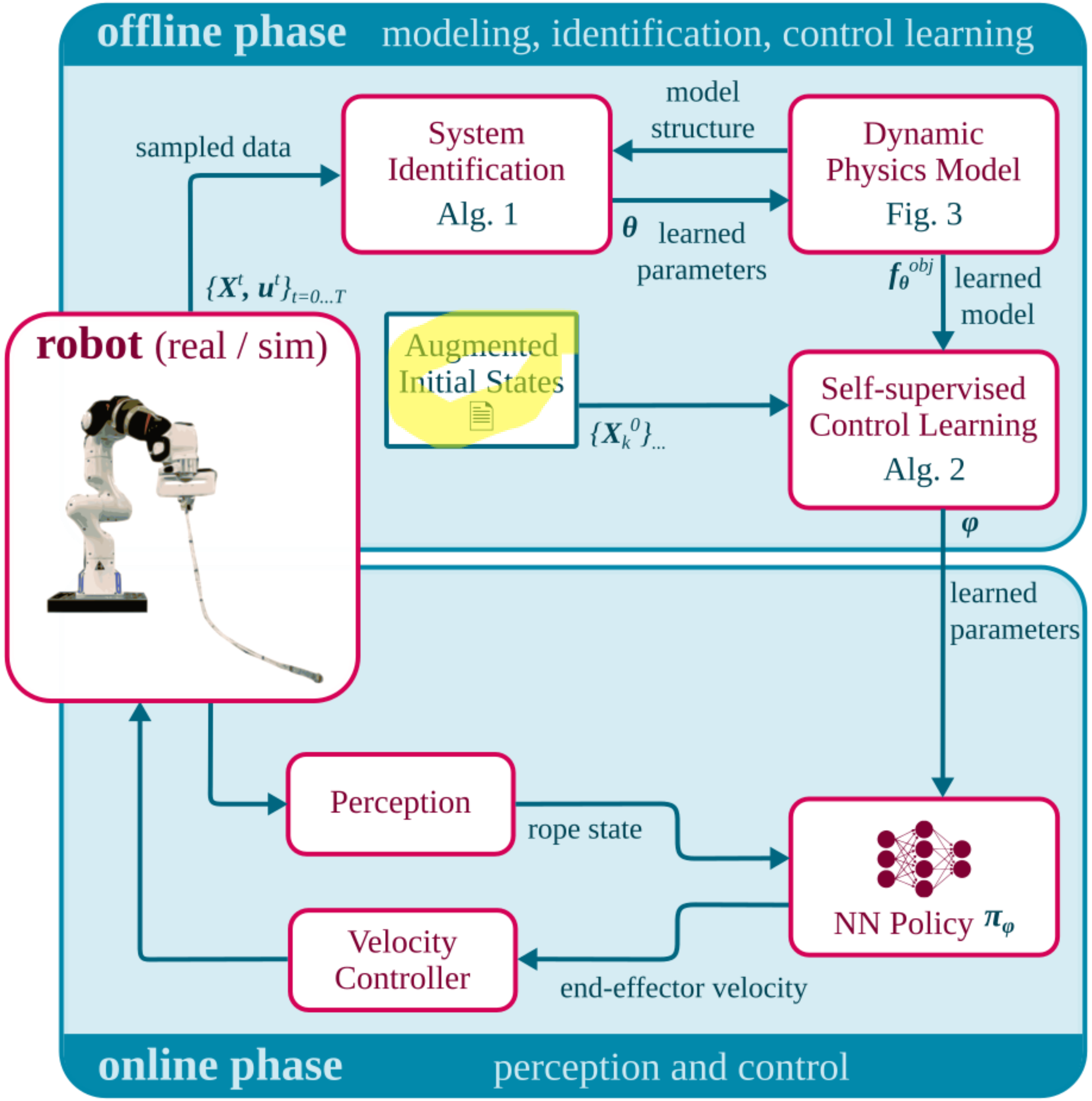
How much should we train?

Very much

SPiD (Self-supervised Physics-informed Deformable Object Manipulation)



SPiD (Self-supervised Physics-informed Deformable Object Manipulation)



- Data augmentation
- Domain randomization

Augmented Self-Supervised Control Learning

- Initial state diversity
  - Apply random motions to the rope and sample states (around 10k states for the main task)
- Data augmentation
  - Rotate in small increments, covering 360^0
  - Translate in x, y, z

  (Around 2 million states)

Algorithm 2 Physics-informed Self-supervised Training

Input: Initial state set $\{\boldsymbol{X}_k^0\}_{k=1,\dots,K}$, task specification set $\{\boldsymbol{G}_k\}_{k=1,\dots,K}$, model parameters θ, integration time step Δt
Output: Trained controller parameters ϕ

Initialize controller parameters ϕ
for each training iteration **do**
  $\{\Delta\theta_k\}_{k=1,\dots,K} \leftarrow \text{sample\_noise}(\theta)$
  for $t = 0, \dots, T-1$ **do**
    $\{\boldsymbol{u}_k^t\}_{k=1,\dots,K} \leftarrow \{\pi_\phi(\boldsymbol{X}_k^t, \boldsymbol{G}_k)\}_{k=1,\dots,K}$
    $\{\boldsymbol{X}_k^{t+1}\}_{k=1,\dots,K} \leftarrow \{\boldsymbol{f}^{\text{dlo}}_{\theta+\Delta\theta_k,\Delta t}(\boldsymbol{X}_k^t, \boldsymbol{u}_k^t)\}_{k=1,\dots,K}$
  end for
  $\boldsymbol{L} \leftarrow \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{L}_{\text{task}}(\{\boldsymbol{X}_k^t\}_{t=1}^{T}, \boldsymbol{G}_k)$
    ▷ Task-based $\boldsymbol{L}_{\text{task}}(\cdot)$ as in (18) and (19)
  $\phi \leftarrow \text{Adam}(\phi, \nabla_\phi \boldsymbol{L})$ ▷ Autodiff
end for

Augmented Self-Supervised Control Learning

- Initial state diversity
  - Apply random motions to the rope and sample states
    (around 10k states for the main task)
- Data augmentation
  - Rotate in small increments, covering 360^0
  - Translate in x, y, z

  (Around 2 million states)

- Domain randomization
  - Rope parameters are perturbed

$$\theta = \{\mathbf{m}, \boldsymbol{l}^0, \boldsymbol{g}, c^{air}, \boldsymbol{k}^s, \boldsymbol{c}^s, \boldsymbol{k}^b, \boldsymbol{c}^b, \boldsymbol{k}^t\}.$$

  - by Gaussian noise
  - To increase robustness to model mismatch
  - The noise amplitude is increased gradually *(curriculum learning)*

Algorithm 2 Physics-informed Self-supervised Training

Input: Initial state set $\{\boldsymbol{X}_k^0\}_{k=1,\dots,K}$, task specification set $\{\boldsymbol{G}_k\}_{k=1,\dots,K}$, model parameters θ, integration time step Δt

Output: Trained controller parameters ϕ

Initialize controller parameters ϕ

for each training iteration **do**

    $\{\Delta\theta_k\}_{k=1,\dots,K} \leftarrow \text{sample\_noise}(\theta)$

    for $t = 0, \dots, T-1$ **do**

        $\{\boldsymbol{u}_k^t\}_{k=1,\dots,K} \leftarrow \{\pi_\phi(\boldsymbol{X}_k^t, \boldsymbol{G}_k)\}_{k=1,\dots,K}$

        $\{\boldsymbol{X}_k^{t+1}\}_{k=1,\dots,K} \leftarrow \{\boldsymbol{f}^{\text{dlo}}_{\theta+\Delta\theta_k,\Delta t}(\boldsymbol{X}_k^t, \boldsymbol{u}_k^t)\}_{k=1,\dots,K}$

    end for

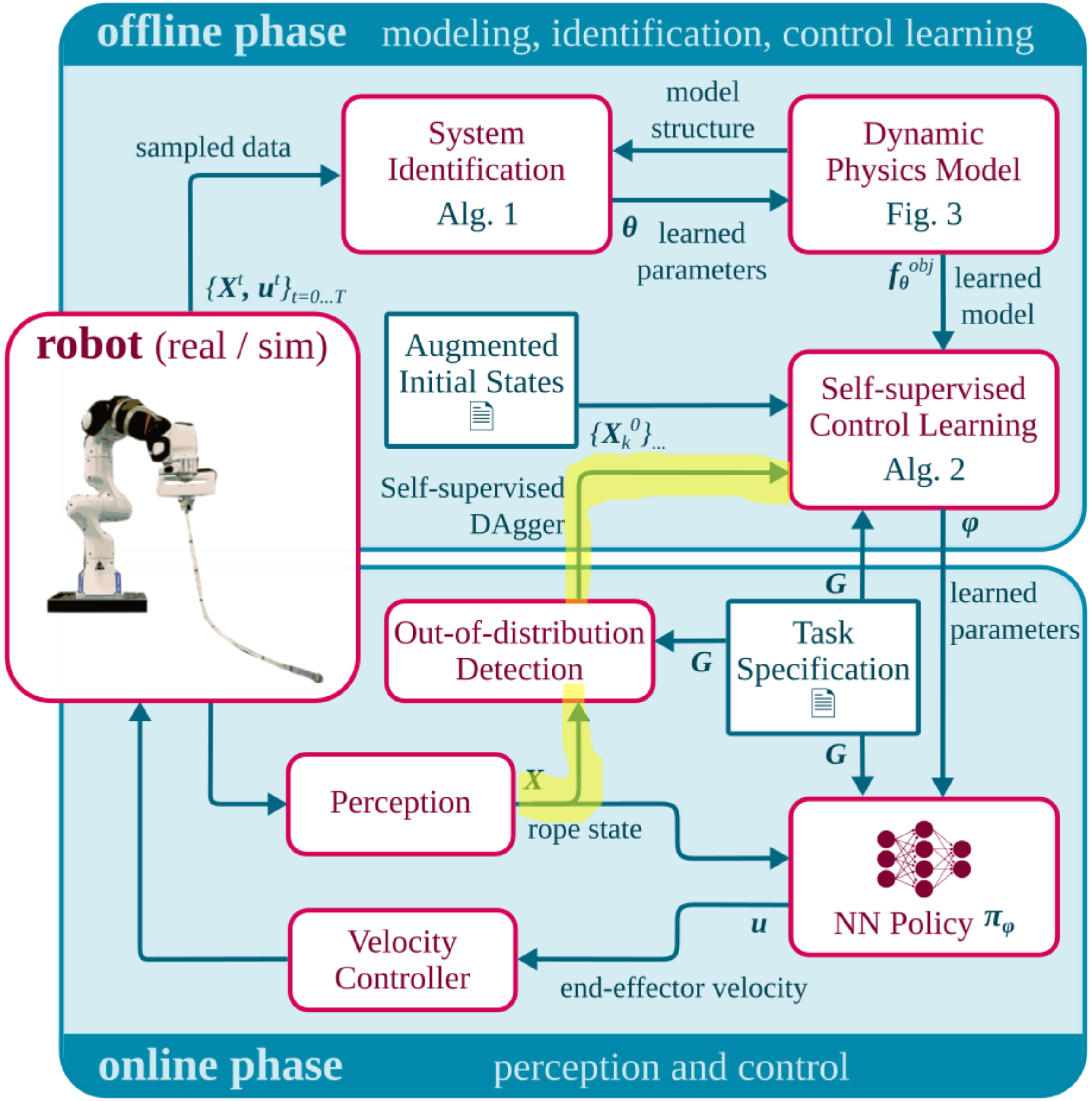
    $\boldsymbol{L} \leftarrow \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{L}_{\text{task}}(\{\boldsymbol{X}_k^t\}_{t=1}^{T}, \boldsymbol{G}_k)$

        ▷ Task-based $\boldsymbol{L}_{\text{task}}(\cdot)$ as in (18) and (19)

    $\phi \leftarrow \text{Adam}(\phi, \nabla_\phi \boldsymbol{L})$ ▷ Autodiff

end for

SPiD (Self-supervised Physics-informed Deformable Object Manipulation)



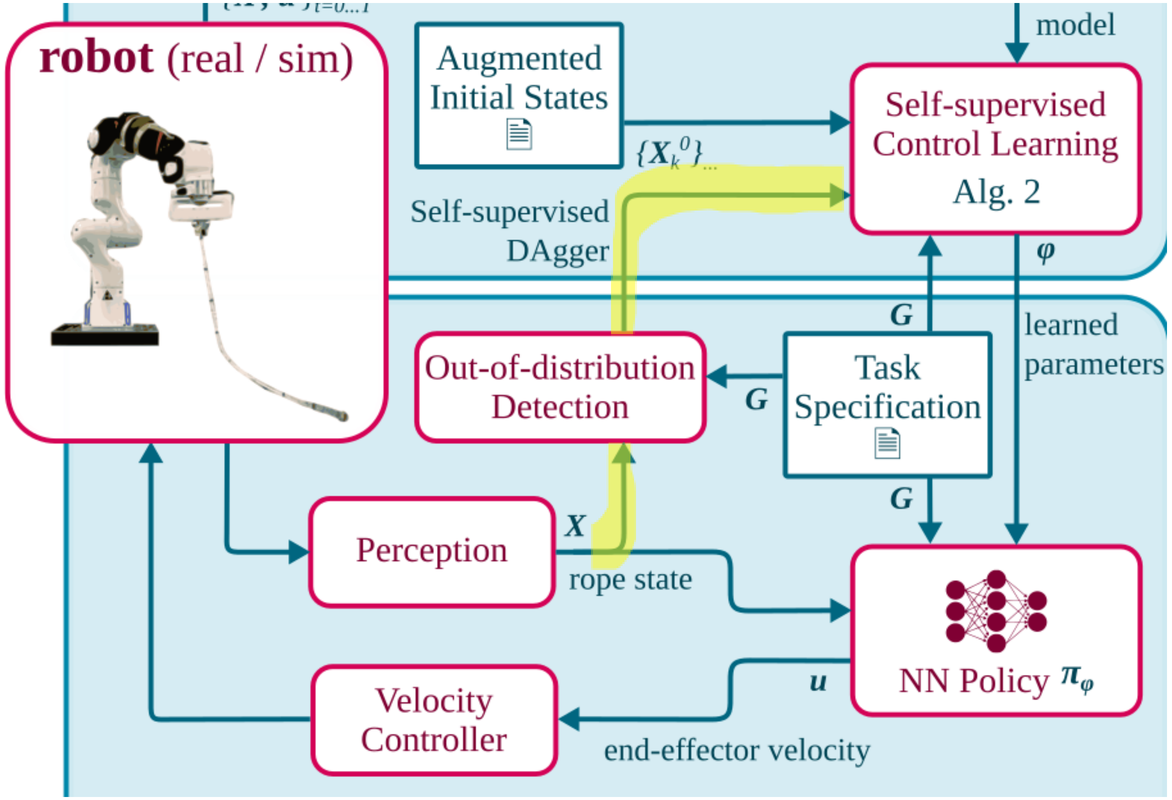
- Data augmentation
- Domain randomization
- Self-supervised DAgger

Self-Supervised DAgger

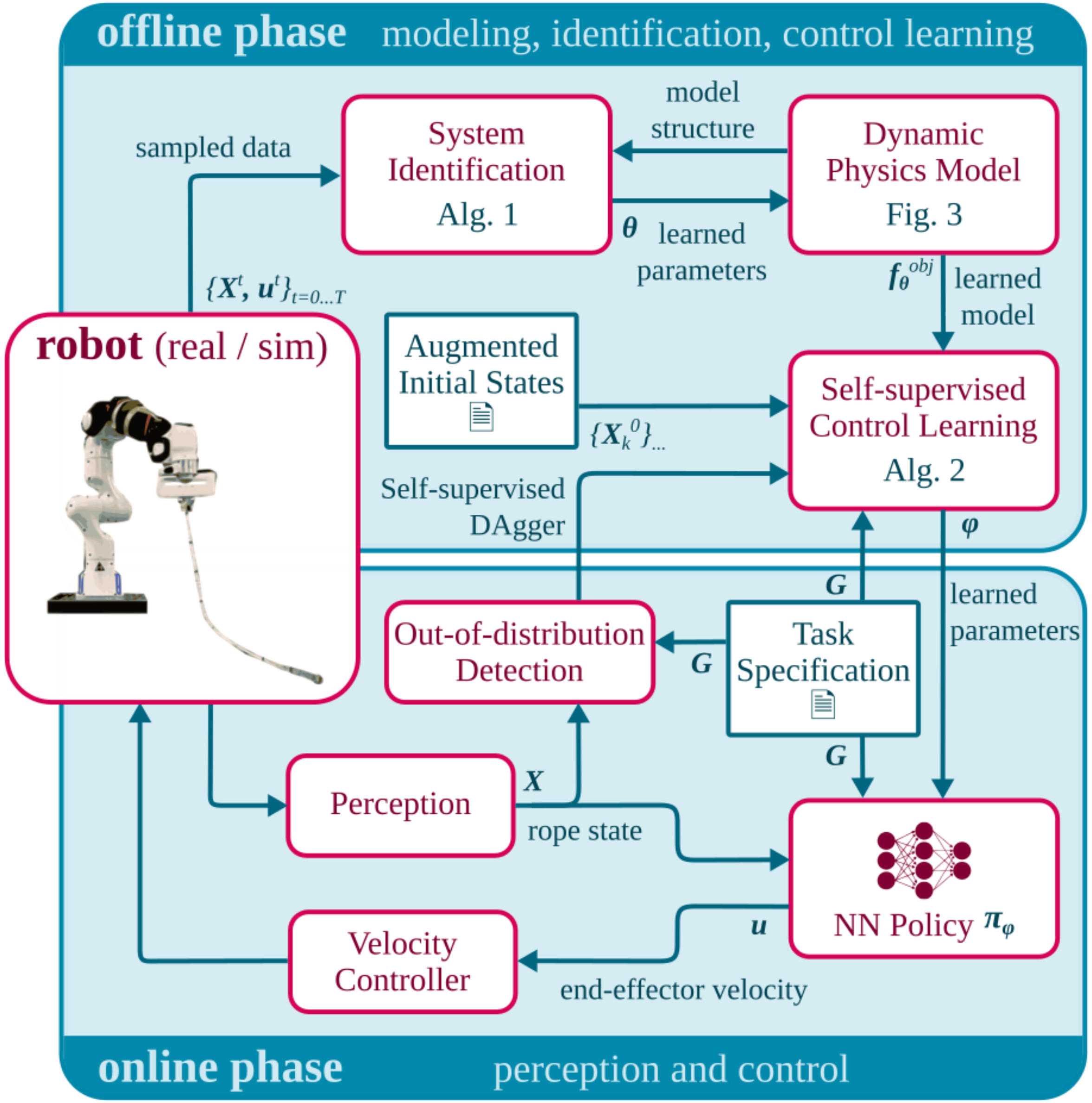
- Automatic detection and correction of distribution shifts
  - Monitoring loss evolution online

- The states are added in the dataset
- Training is still done offline later

- It does *not* require expert data!



SPiD (Self-supervised Physics-informed Deformable Object Manipulation)



Experiments

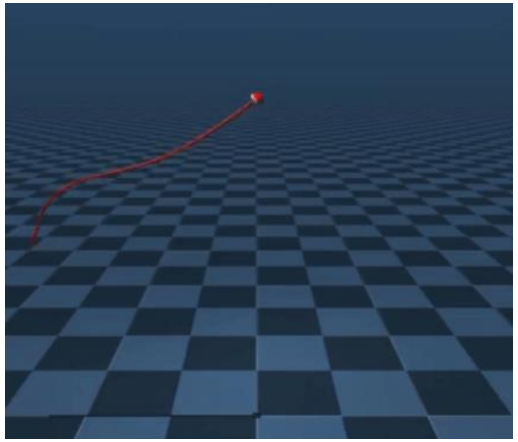
Long, Youyuan, et al. "Self-supervised Physics-informed Deformable Object Manipulation with Non-negligible Dynamics." arXiv preprint (2026).

Experiments

Long, Youyuan, et al. "Self-supervised Physics-informed Deformable Object Manipulation with Non-negligible Dynamics." arXiv preprint (2026).

Rope Stabilization Problem

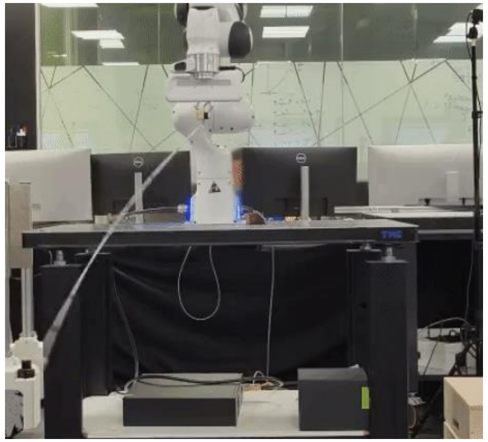
Bring a rope to rest as quickly as possible.



Success is measured by system energy:

$$E_{\text{rope}}(\boldsymbol{X}) = \sum_{i=0}^{N}\left(m_i g z_i + \frac{1}{2} m_i \|\boldsymbol{v}_i\|^2\right) + \frac{1}{2}\sum_{i=1}^{N-1} k_i^b \beta_i^2 + \frac{1}{2}\sum_{i=1}^{N} k_i^s (l_i - l_i^0)^2 + \frac{1}{2}\sum_{i=1}^{N} k_i^t \psi_i^2,$$

including kinetic and potential energy of all rope segments



The rope is considered stabilized when its energy decreases to 1% of the initial value.

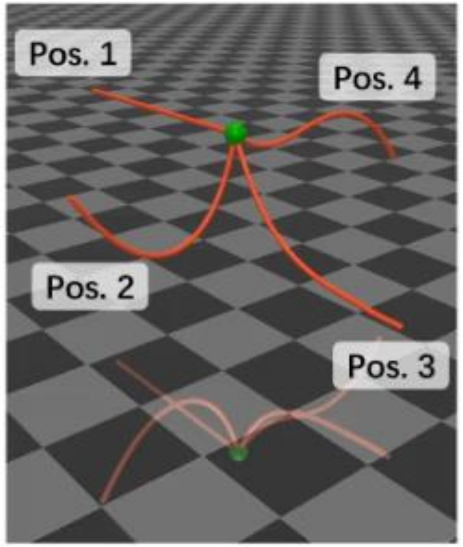
Consequently, control learning loss: $\boldsymbol{L}\left(\{\boldsymbol{X}^t\}_{t=1}^{T}, \boldsymbol{G} = \emptyset\right) = E_{\text{rope}}(\boldsymbol{X}^T)$

Long, Youyuan, et al. "Self-supervised Physics-informed Deformable Object Manipulation with Non-negligible Dynamics." arXiv preprint (2026).

Simulation experiments

The learned controller is compared to the baseline in simulation.

1. Four different initial rope positions are tested for evaluation of SPiD.
2. For each initial position, 125 rope model parameters sets are uniformly sampled.



Metric: Total energy of the rope segments

Self-supervised Physics-informed Manipulation of Deformable Linear Objects with Non-negligible Dynamics

Youyuan Long*, Gokhan Solak, Sara Zeynalpour, Heng Zhang, and Arash Ajoudani

* youyuan.long@iit.it

ISTITUTO ITALIANO DI TECNOLOGIA
HUMAN-ROBOT INTERFACES AND PHYSICAL INTERACTION

hri.iit.it

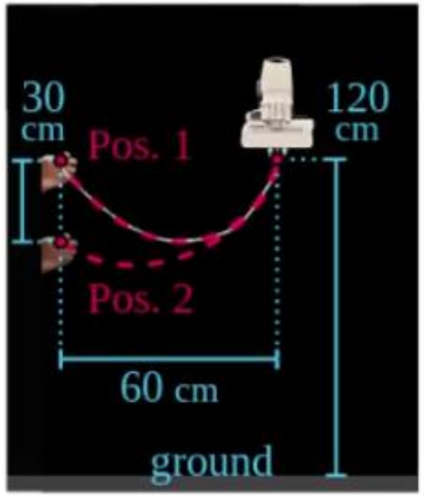
TORNADO

tornado-horizon.eu

https://youtu.be/lgX2J-00TRM

Real-world experiments

Two different initial rope positions are tested for evaluation of SPiD in real-world experiments.



Metric: Total energy of the rope segments

Self-supervised Physics-informed Manipulation of Deformable Linear Objects with Non-negligible Dynamics

Youyuan Long*, Gokhan Solak, Sara Zeynalpour, Heng Zhang, and Arash Ajoudani

* youyuan.long@iit.it

ISTITUTO ITALIANO DI TECNOLOGIA
HUMAN-ROBOT INTERFACES AND PHYSICAL INTERACTION

TORNADO

hri.iit.it

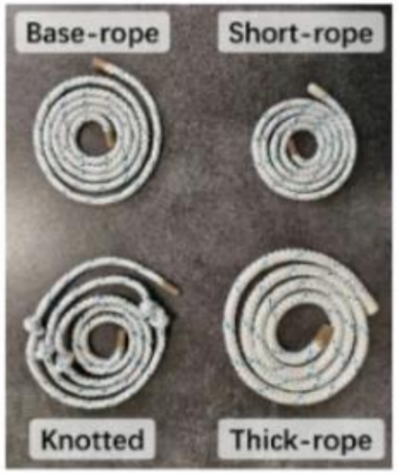
tornado-horizon.eu

https://youtu.be/lgX2J-00TRM

Real-world experiments

Generalization to unseen ropes

Three other ropes with different lengths, masses and non-uniform mass distributions are tested without model re-identification.



Metric: Total energy of the rope segments

Self-supervised Physics-informed Manipulation of Deformable Linear Objects with Non-negligible Dynamics

Youyuan Long*, Gokhan Solak, Sara Zeynalpour, Heng Zhang, and Arash Ajoudani

* youyuan.long@iit.it

ISTITUTO ITALIANO DI TECNOLOGIA
HUMAN-ROBOT INTERFACES AND PHYSICAL INTERACTION

hri.iit.it

TORNADO

tornado-horizon.eu

https://youtu.be/lgX2J-00TRM

Self-supervised Physics-informed Manipulation of Deformable Linear Objects with Non-negligible Dynamics

Youyuan Long*, Gokhan Solak, Sara Zeynalpour, Heng Zhang, and Arash Ajoudani

* youyuan.long@iit.it

ISTITUTO ITALIANO DI TECNOLOGIA
HUMAN-ROBOT INTERFACES AND PHYSICAL INTERACTION

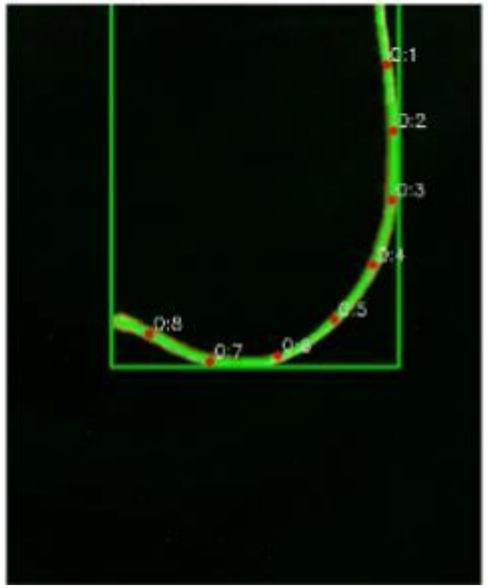
hri.iit.it

TORNADO

tornado-horizon.eu

https://youtu.be/lgX2J-00TRM

Markerless Rope Stabilization



We developed a markerless perception system to validate the application of our method in a more affordable single camera setup.

Despite the physical limitations of the markerless system, our experiments show successful rope stabilization.

Self-supervised Physics-informed Manipulation of Deformable Linear Objects with Non-negligible Dynamics

Youyuan Long*, Gokhan Solak, Sara Zeynalpour, Heng Zhang, and Arash Ajoudani

* youyuan.long@iit.it

ISTITUTO ITALIANO DI TECNOLOGIA
HUMAN-ROBOT INTERFACES AND PHYSICAL INTERACTION

hri.iit.it

TORNADO

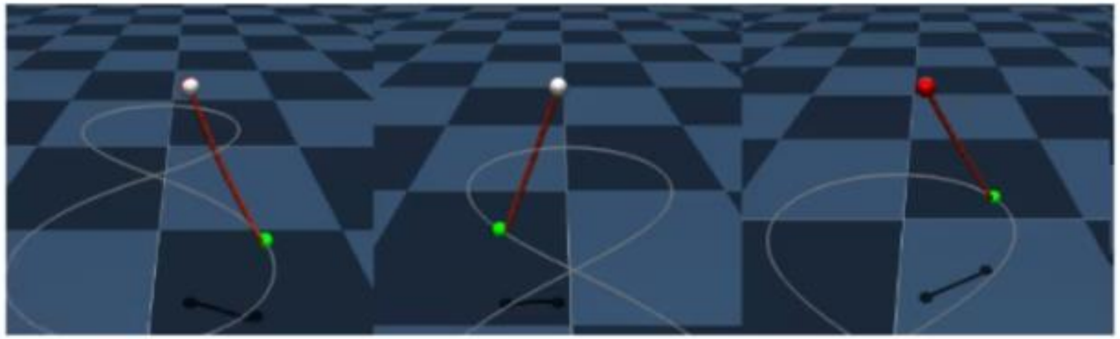
tornado-horizon.eu

https://youtu.be/lgX2J-00TRM

Rope Trajectory Tracking Task

To demonstrate the generalization of SPiD on new tasks,
we demonstrate the rope trajectory tracking task:

Controlling the top end of the rope such that its bottom end follows a desired trajectory.



Metric: Average distance between the executed and target trajectories

$$\boldsymbol{L}(\{\boldsymbol{X}^t\}_{t=1}^T, \boldsymbol{G}) = \sum_{t=1}^{T} \gamma^{T-t} \cdot \left\| \boldsymbol{p}_N^t - \boldsymbol{g}_{\text{goal}}^t \right\|_2^2$$

Self-supervised Physics-informed Manipulation of Deformable Linear Objects with Non-negligible Dynamics

Youyuan Long*, Gokhan Solak, Sara Zeynalpour, Heng Zhang, and Arash Ajoudani

* youyuan.long@iit.it

ISTITUTO ITALIANO DI TECNOLOGIA
HUMAN-ROBOT INTERFACES AND PHYSICAL INTERACTION

hri.iit.it

TORNADO

tornado-horizon.eu

https://youtu.be/lgX2J-00TRM

Questions

ISTITUTO ITALIANO
DI TECNOLOGIA
HUMAN-ROBOT INTERFACES
AND INTERACTION

Learning safe behavior from data

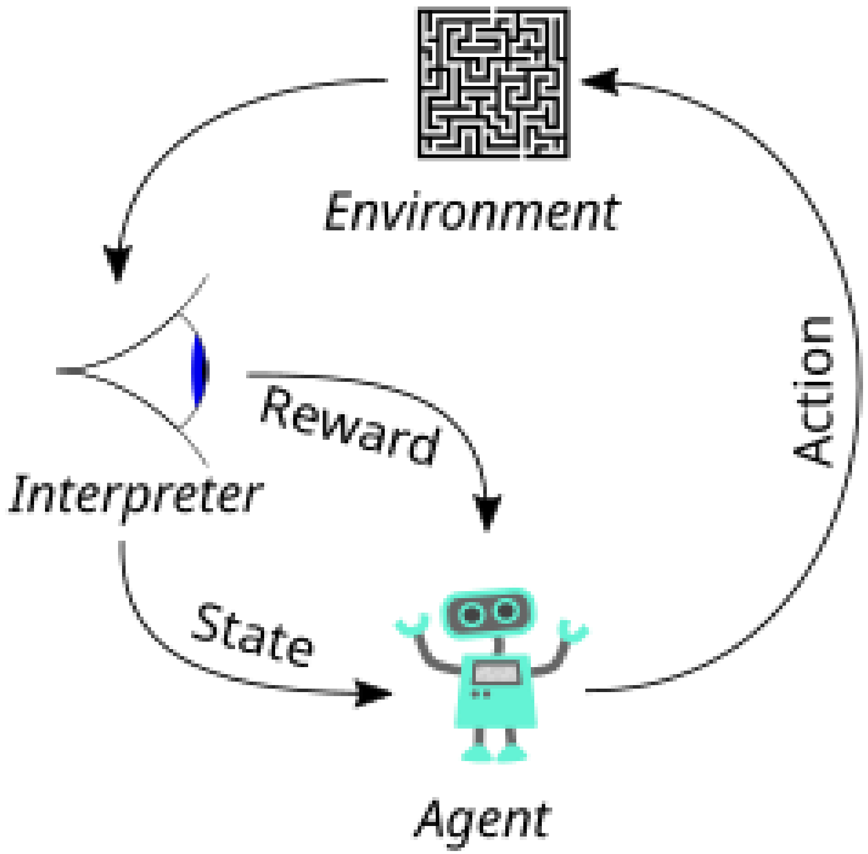
Safe RL for contact-rich interactions

Heng Zhang, Gokhan Solak, Sebastian S. Hjorth,
Gustavo J. G. Lahr, Arash Ajoudani

RL problem:

Markov Decision Process (**MDP**)

$$\mathcal{S}, \mathcal{A}, R, P, \gamma, \mu$$



Goals:

Task accomplishment

$\underbrace{\hspace{6em}}$

Traditional RL

Safe RL problem:

Markov Decision Process (**MDP**)

$$\mathcal{S}, \mathcal{A}, R, P, \gamma, \mu$$

$$\downarrow$$

Constrained MDP (**CMDP**)

$$\mathcal{S}, \mathcal{A}, R, P, \gamma, \mu, \mathcal{C}$$

Added safety constraints

Goals:

$$\underbrace{\underbrace{\text{Task accomplishment}}_{\text{Traditional RL}} + \text{Safety} \quad \text{Stability} \quad \text{Efficiency}}_{\text{Safe Passive RL}}$$

Safe RL problem and our work:

Markov Decision Process (**MDP**)

$$\mathcal{S}, \mathcal{A}, R, P, \gamma, \mu$$

$$\downarrow$$

Constrained MDP (**CMDP**)

$$\mathcal{S}, \mathcal{A}, R, P, \gamma, \mu, \mathcal{C}$$

Added safety constraints

1. Zhang, H., Solak, G., Lahr, G. J., & Ajoudani, A. (2024). **SRL-VIC: ... ment learning for conta...** *IEEE Robotics and Automation Letters*.
2. Zhang, H., Solak, G., Hjorth, S., & Ajoudani, A. (2025). **Towards ... omparative study on conta...** *arXiv preprint arXiv:2503.00287*.
3. Zhang, H., Solak, G., & Ajoudani, A. (2025). **Bresa: Bio-inspired Reflexive Safe Reinforcement Learning for Conta...** *arXiv preprint arXiv:2503.21989*.

SRL + VIC

Passivity-aware learning

Reflex mechanism

Goals:

Task accomplishment + Safety | Stability | Efficiency

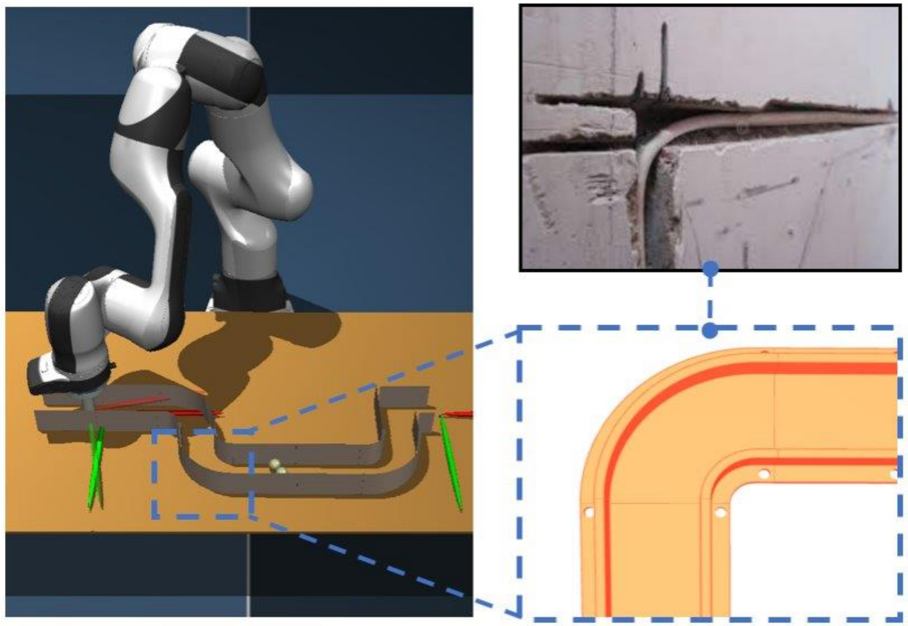
$\underbrace{\text{Task accomplishment}}_{\text{Traditional RL}}$

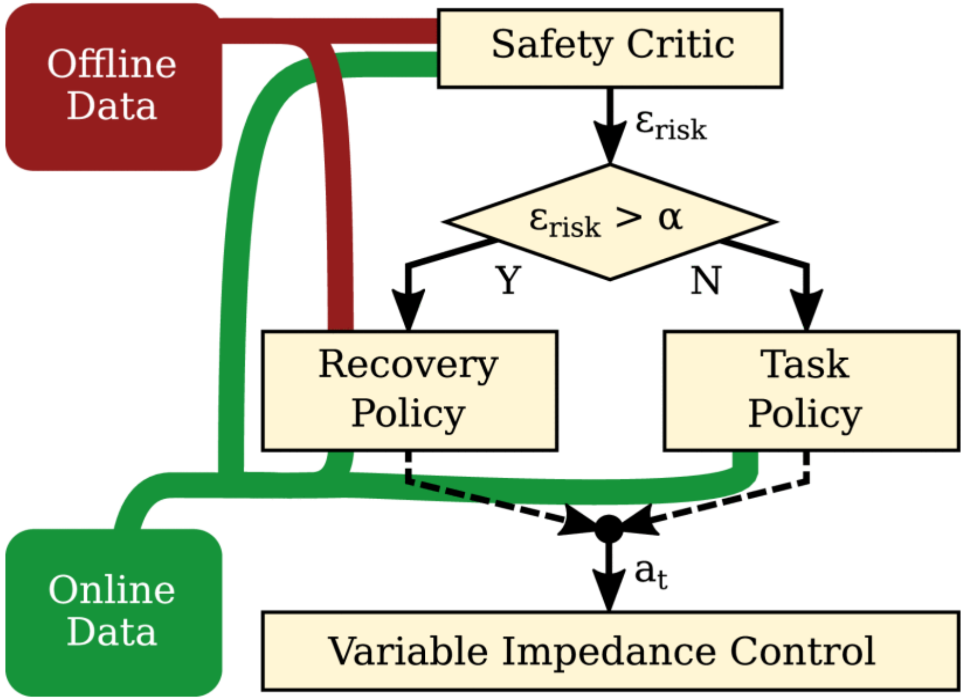
Traditional RL

Safe Passive RL

1. Safe RL – VIC

- Contact-rich maze exploration task
- Force constraint: contact force < threshold
- Combining safe RL with variable impedance control

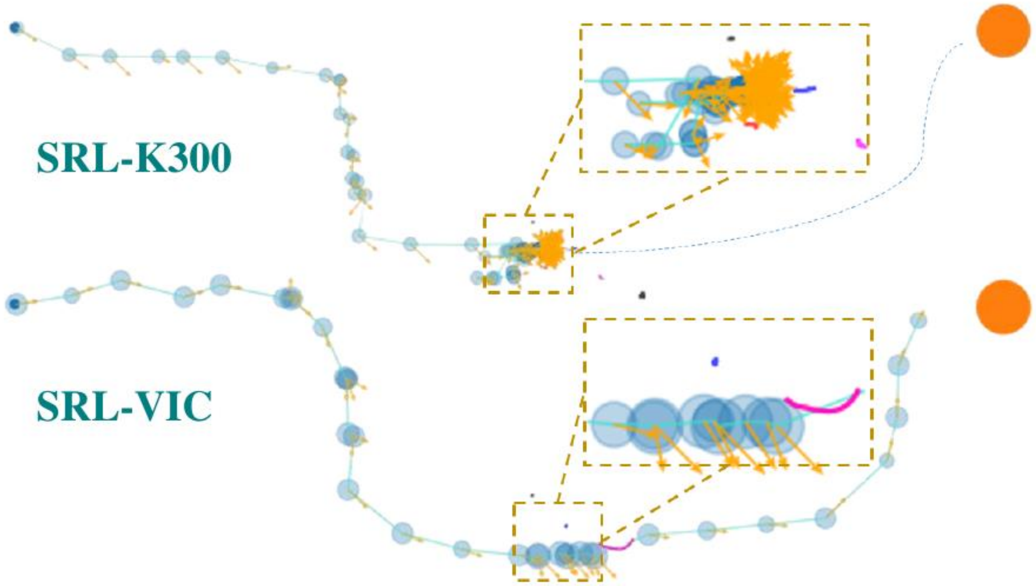


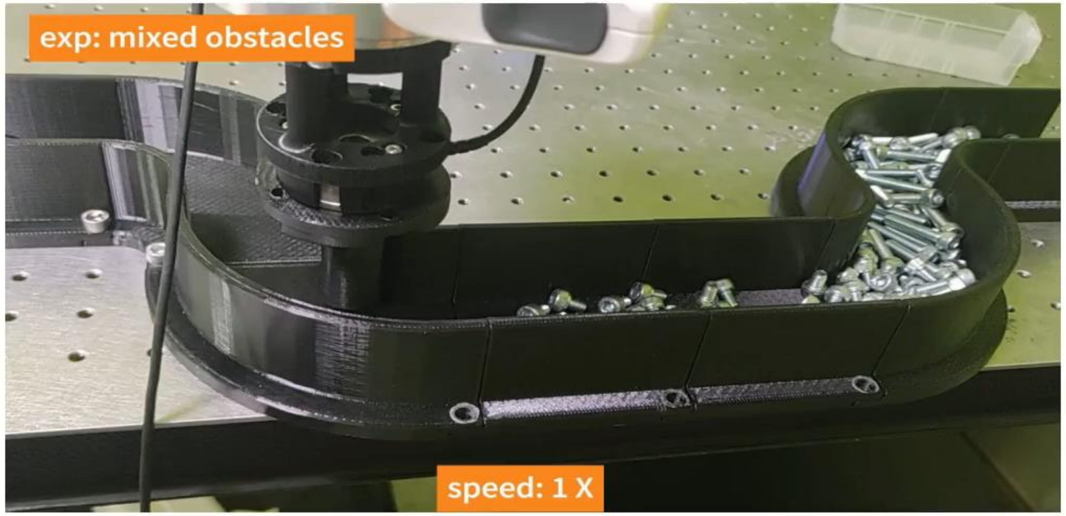


Zhang, H., Solak, G., Lahr, G. J., & Ajoudani, A. (2024). SRL-VIC: A variable stiffness-based safe reinforcement learning for contact-rich robotic tasks. IEEE Robotics and Automation Letters.

1. Safe RL – VIC Results

- The agent learns to modulate the stiffness to
  - Stay compliant to navigate through walls
  - Push through movable obstacles





Zhang, H., Solak, G., Lahr, G. J., & Ajoudani, A. (2024). SRL-VIC: A variable stiffness-based safe reinforcement learning for contact-rich robotic tasks. IEEE Robotics and Automation Letters.

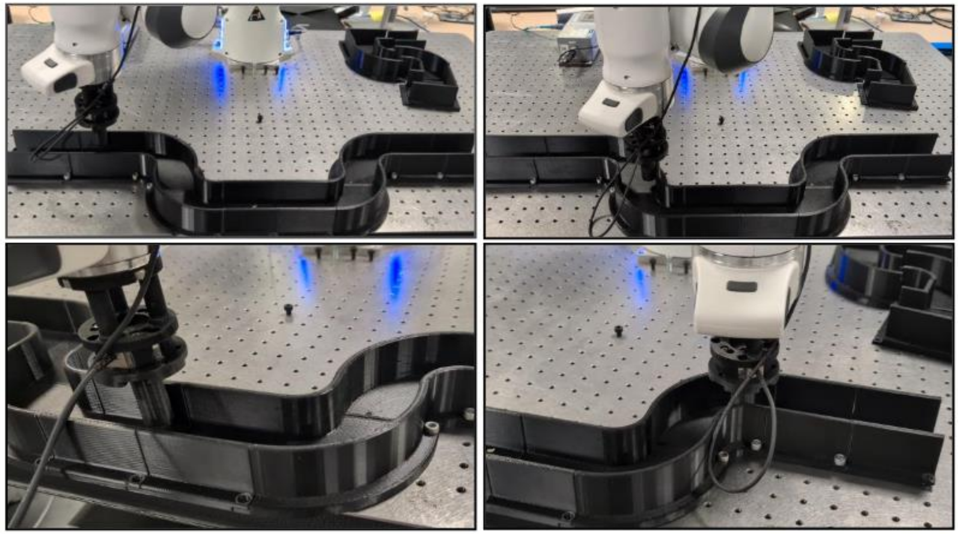
2. Passive Safe RL

Passivity-Centric Safe Reinforcement Learning for Contact-Rich Robotic Tasks

- Passivity-ensured RL system for contact-rich tasks
- Achieve safety and control stability
- Introducing passivity in both phases *(Passivity-ensured)*
  - training *(Passivity-aware)*
  - deployment *(Passivity-filtered)*

	passivity-agnostic	passivity-filtered	passivity-aware	passivity-ensured
passivity guarantee	✗	✓	✗	✓
energy efficiency	✗	✗	✓	✓

- Evaluated on the **blind maze exploration task**



Zhang, H., Solak, G., Hjorth, S., & Ajoudani, A. (2025). Towards passive safe reinforcement learning: A comparative study on contact-rich robotic manipulation. arXiv *preprint arXiv:2503.00287*.

2. Passive Safe RL

Markov Decision Process (**MDP**)

$$\mathcal{S}, \mathcal{A}, R, P, \gamma, \mu$$

$$\downarrow$$

Constrained MDP (**CMDP**)

$$\boxed{\mathcal{S}, \mathcal{A}, R, P, \gamma, \mu, \mathcal{C}}$$

Constraints $\mathcal{C}$

- Safety:
  - Force limit
- Passivity: $\mathcal{C}_e \subset \mathcal{C}$
  - Total energy limit $E_{\mathrm{T}} + \dot{E}_{\mathrm{T}} \leq \underline{E}_{\mathrm{T}}$
  - Instant energy flow limit $\dot{E}_{\mathrm{T}} < \underline{\dot{E}}_{\mathrm{T}}$

E_{T} : Energy tank (budget)

Goals:

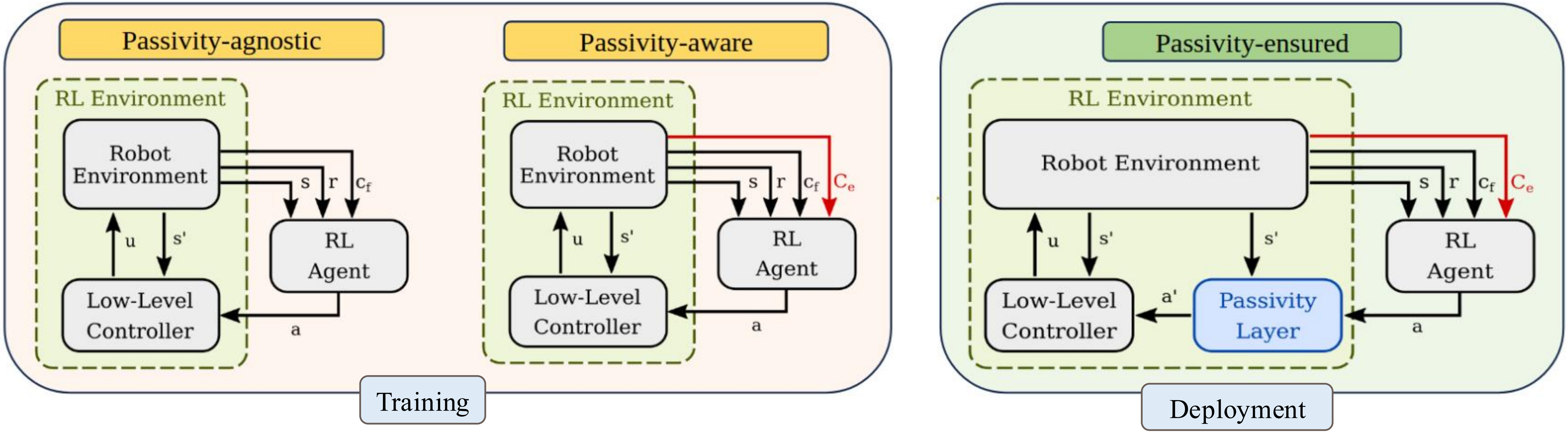
$$\underbrace{\underbrace{\text{Task accomplishment}}_{\text{Traditional RL}} + \text{Safety} \quad \text{Stability} \quad \text{Efficiency}}_{\text{Passivity-centric Safe RL}}$$

Zhang, H., Solak, G., Hjorth, S., & Ajoudani, A. (2025). Towards passive safe reinforcement learning: A comparative study on contact-rich robotic manipulation. arXiv *preprint arXiv:2503.00287*.

2. Passive Safe RL

- Introducing passivity in both in **training** and **deployment**
  - Augmented energy tank
- Training Energy-based constraints $C_e \subset \mathcal{C}$
  - Tank constraint ***Eb*** $E_{\mathrm{T}} + \dot{E}_{\mathrm{T}} \leq \underline{E}_{\mathrm{T}}$
  - Flow constraint ***Ef*** $\dot{E}_{\mathrm{T}} < \underline{\dot{E}}_{\mathrm{T}}$
- Deployment Passivity filtering layer

E_{T} : Energy budget (tank level)

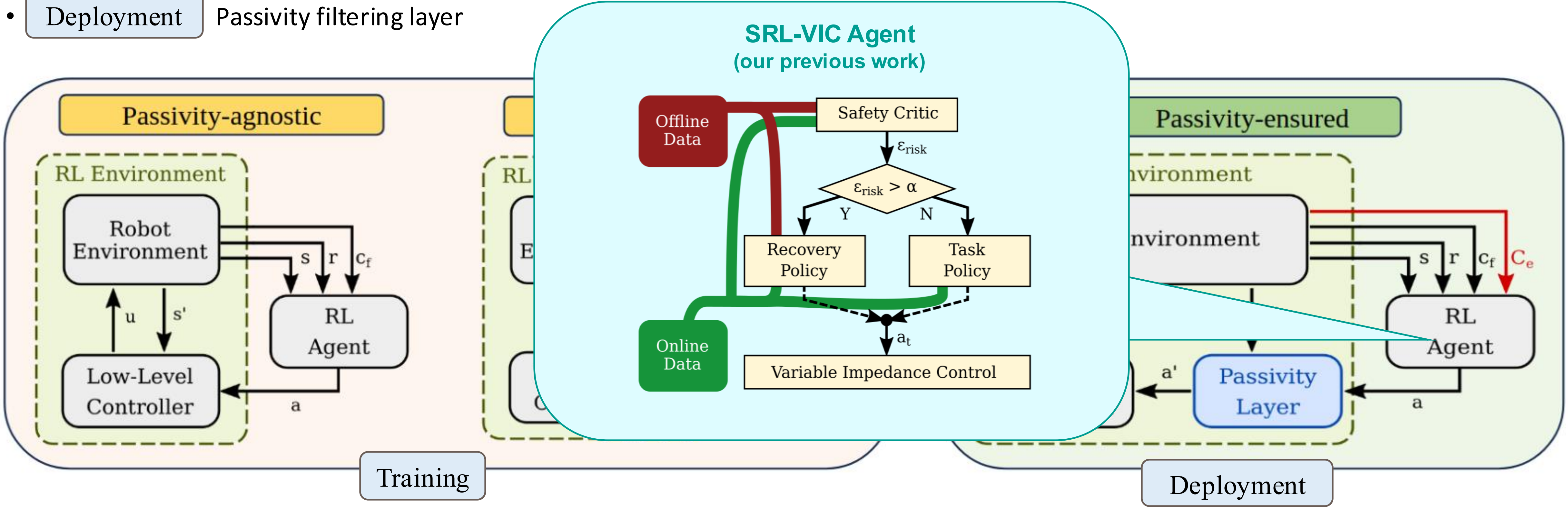


Zhang, H., Solak, G., Hjorth, S., & Ajoudani, A. (2025). Towards passive safe reinforcement learning: A comparative study on contact-rich robotic manipulation. arXiv *preprint arXiv:2503.00287*.

2. Passive Safe RL

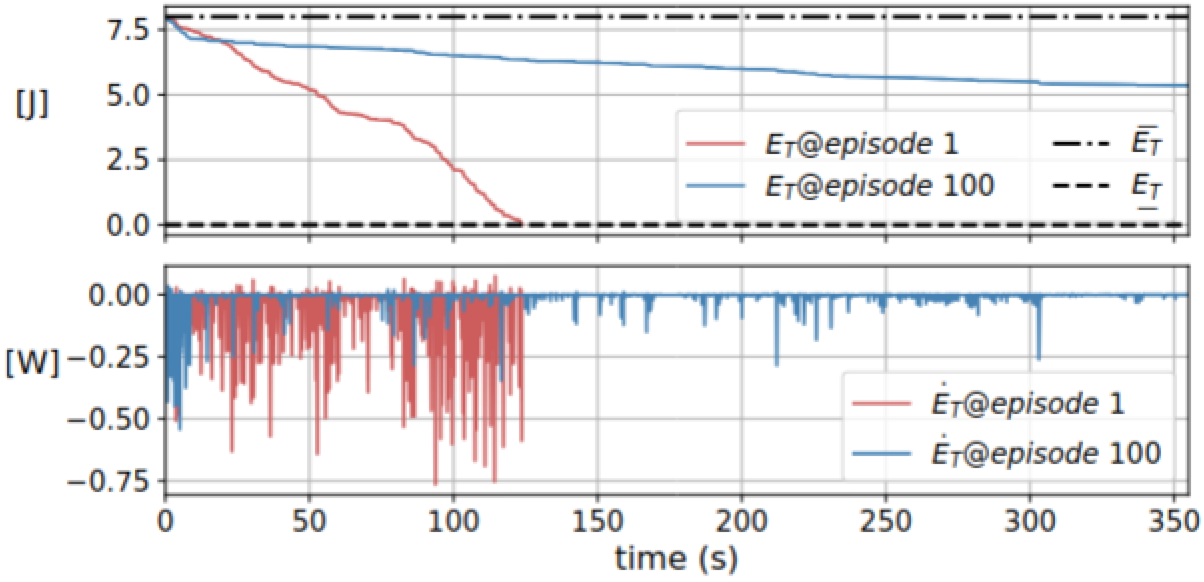
- Introducing passivity in both in **training** and **deployment**
  - Augmented energy tank
- Training Energy-based constraints $C_e \subset \mathcal{C}$
  - Tank constraint ***Eb*** $E_\mathrm{T} + \dot{E}_\mathrm{T} \leq \underline{E}_\mathrm{T}$
  - Flow constraint ***Ef*** $\dot{E}_\mathrm{T} < \underline{\dot{E}}_\mathrm{T}$
- Deployment Passivity filtering layer

E_T : Energy budget (tank level)



Zhang, H., Solak, G., Hjorth, S., & Ajoudani, A. (2025). Towards passive safe reinforcement learning: A comparative study on contact-rich robotic manipulation. arXiv *preprint arXiv:2503.00287*.

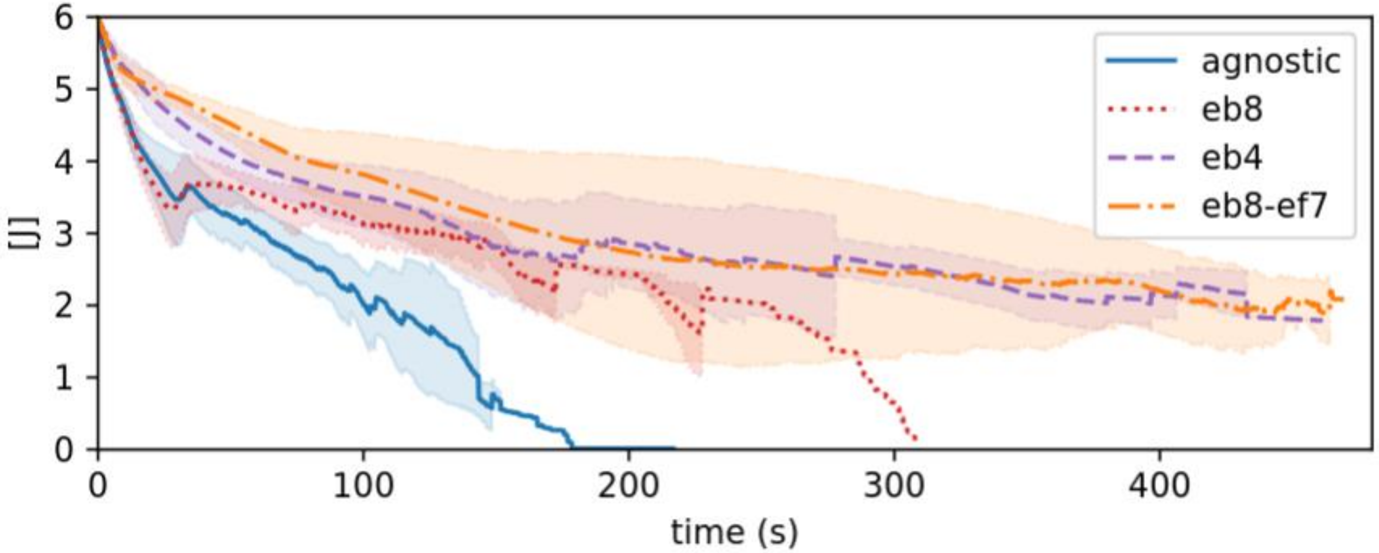
2. Passive Safe RL



E_T: Energy budget (tank level)

E'_T: Energy flow (instant change)

- **Fig. 1**
  - Effect of learning with energy constraints
- **Fig. 2**
  - Comparison of energy economy
  - Training with and without energy constraints
- Results show that
  - the agent learns to regulate stiffness effectively
  - passivity-aware **training** increases the energy efficiency
  - passivity-filtered **deployment** ensures stability



agnostic:
- learning without safety constraints

eb
- energy tank constraint (budget)

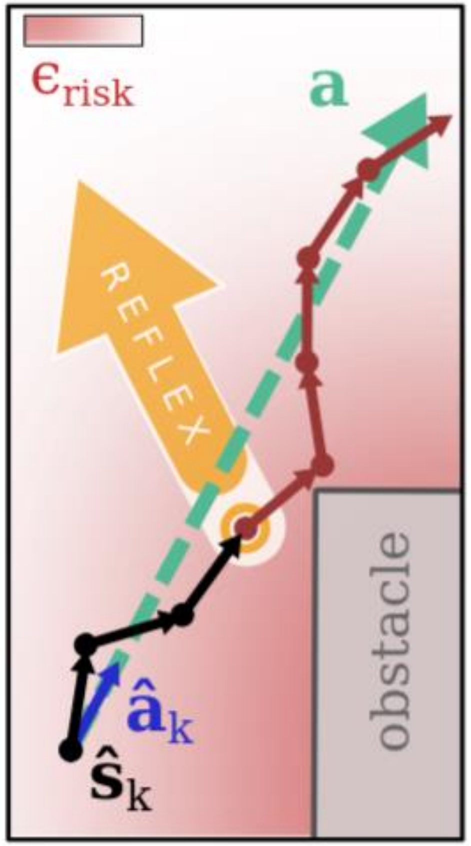
ef
- energy flow constraint

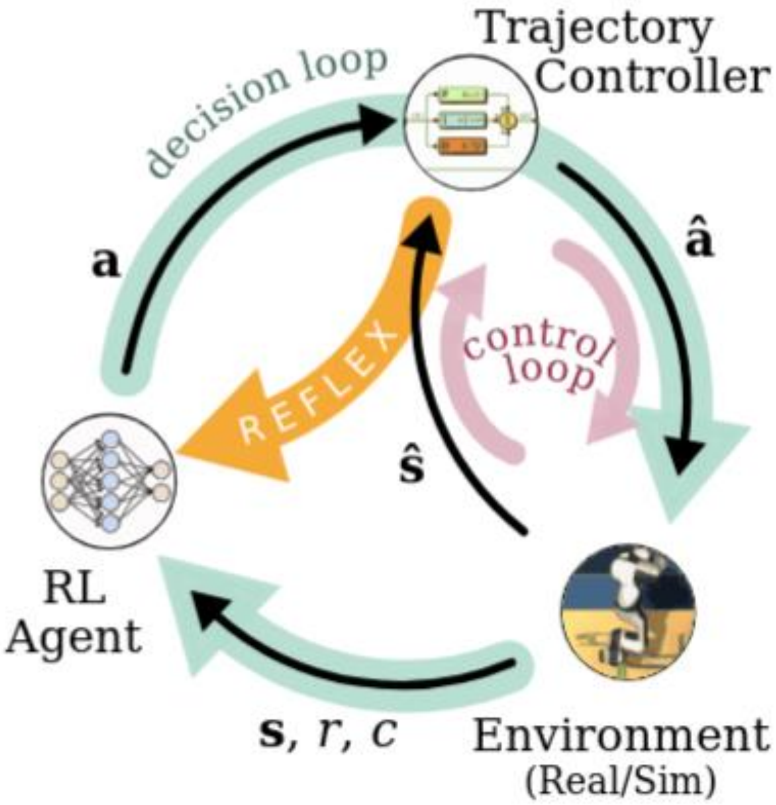
Zhang, H., Solak, G., Hjorth, S., & Ajoudani, A. (2025). Towards passive safe reinforcement learning: A comparative study on contact-rich robotic manipulation. arXiv *preprint arXiv:2503.00287*.

3. Bresa: Reflexive Safe RL

* Bresa: Bio-inspired Reflexive Safe Reinforcement Learning for Contact-Rich Robotic Tasks

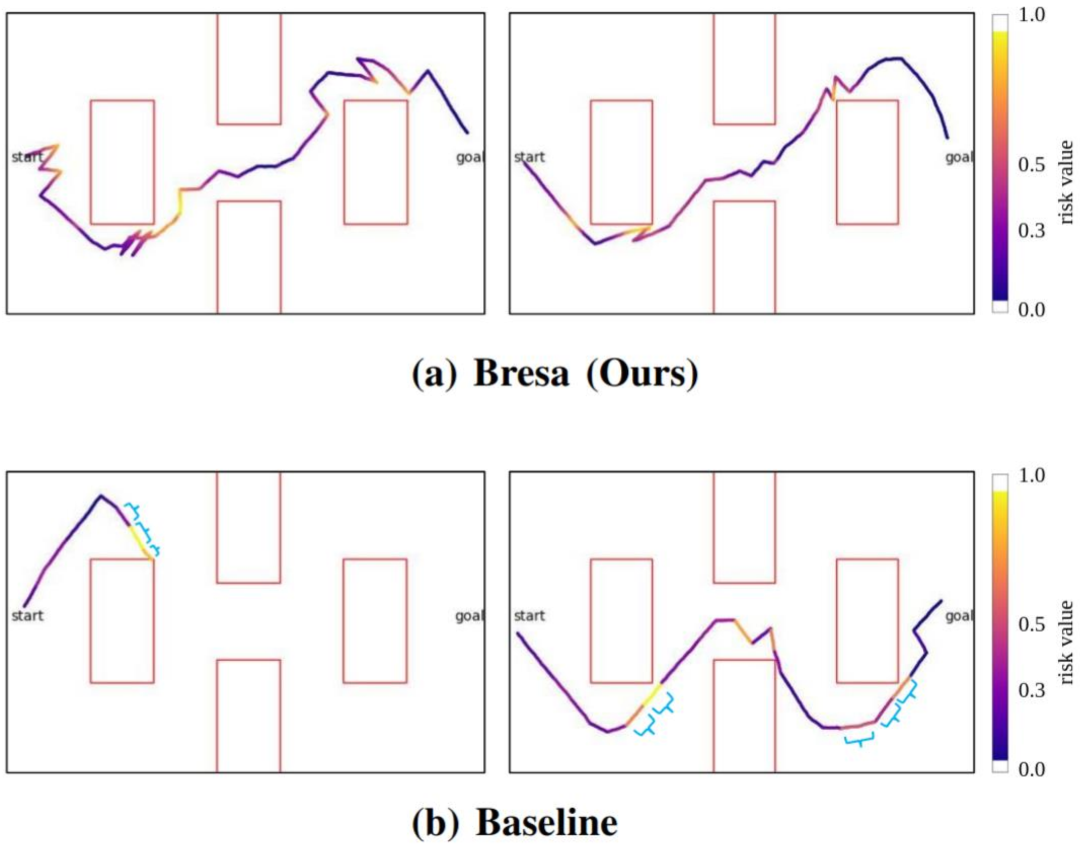
- Bio-inspired approach to **hierarchical** safe RL
  - Task policy -> Planning loop
  - Safety critic and policy -> Control loop
- Safety mechanism runs at a higher frequency
  - More emergent
  - Short horizon
- Task solver runs at lower frequency
  - Sample efficient
  - Long horizon

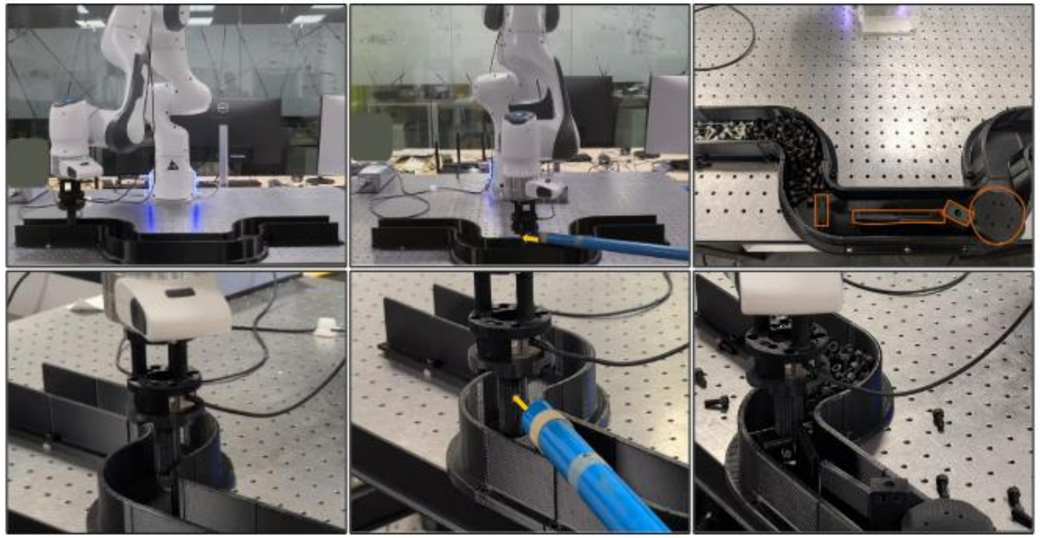




Zhang, H., Solak, G., & Ajoudani, A. (2025). Bresa: Bio-inspired Reflexive Safe Reinforcement Learning for Contact-Rich Robotic Tasks. arXiv preprint arXiv:2503.21989.

3. Bresa: Reflexive Safe RL: Results

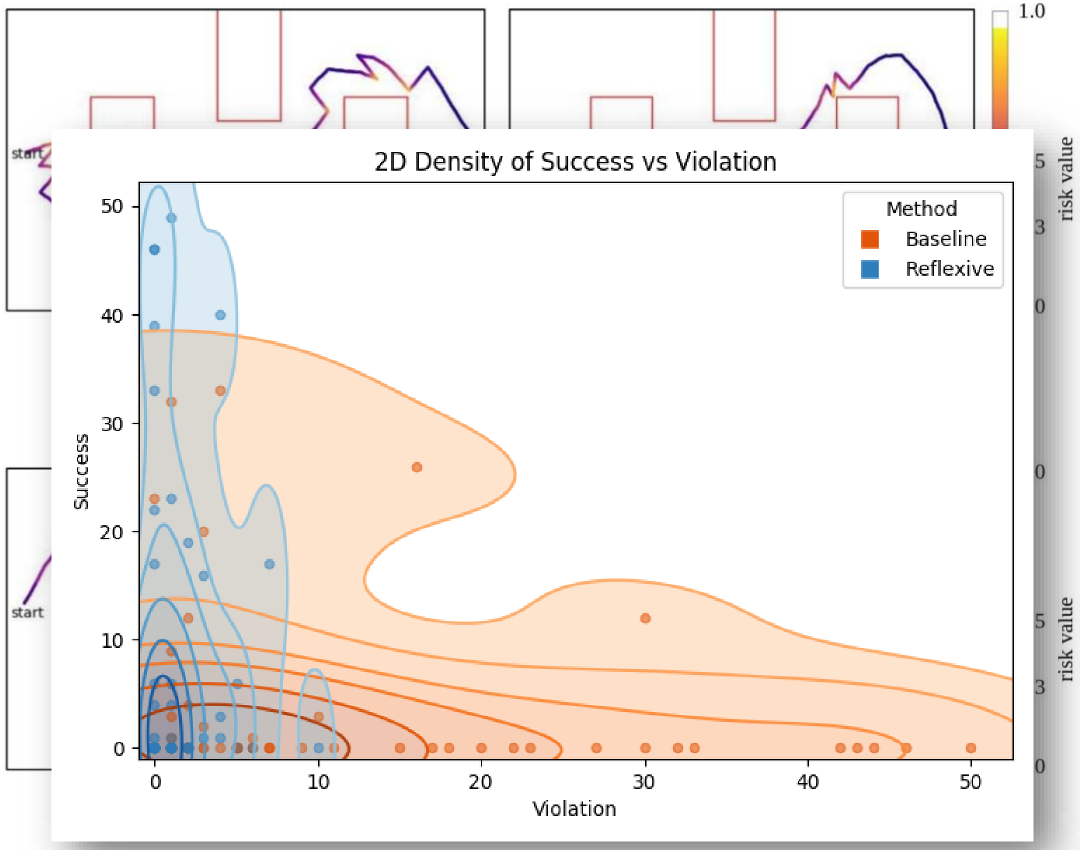


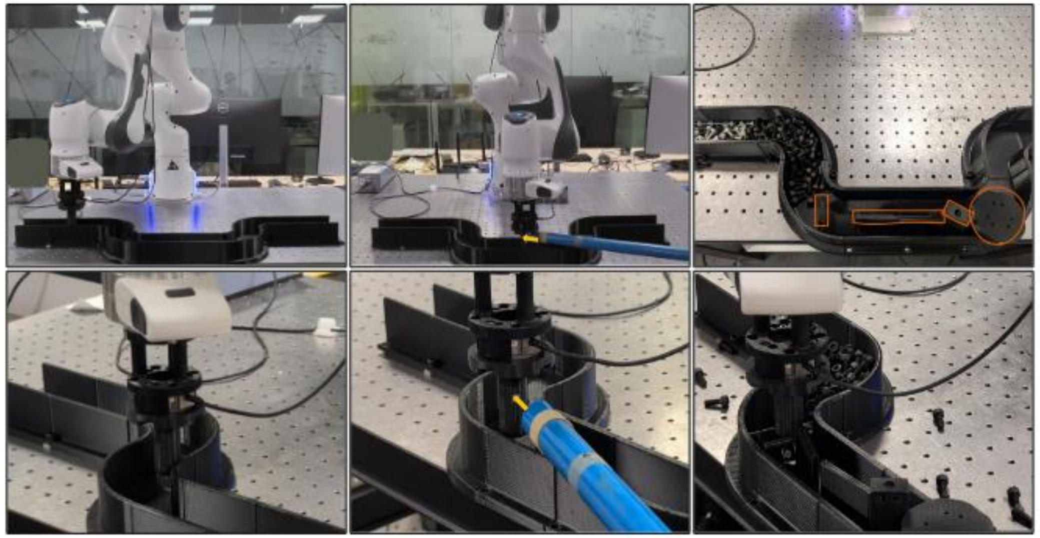


Zhang, H., Solak, G., & Ajoudani, A. (2025). Bresa: Bio-inspired Reflexive Safe Reinforcement Learning for Contact-Rich Robotic Tasks. arXiv preprint arXiv:2503.21989.

3. Bresa: Reflexive Safe RL: Results

The reflex mechanism sharply decrease the constraint violations and lead to better success

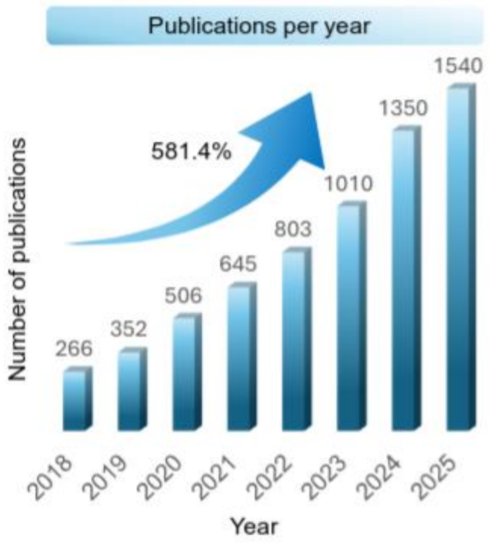


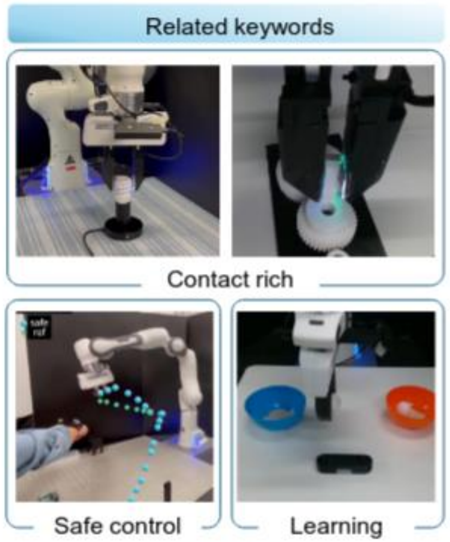


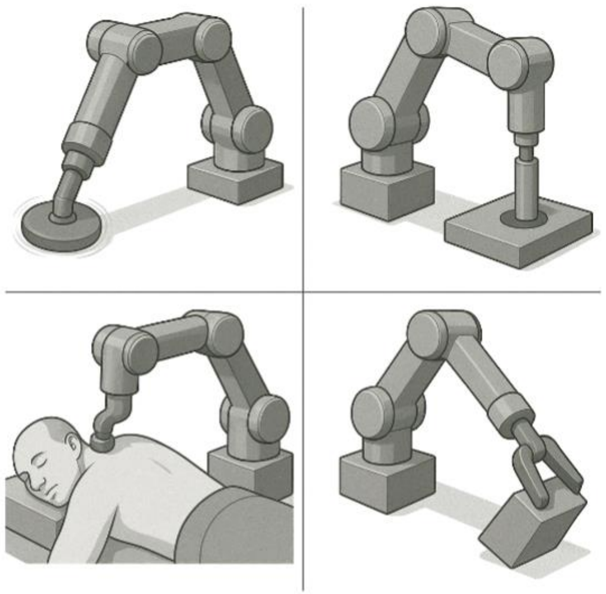
Zhang, H., Solak, G., & Ajoudani, A. (2025). Bresa: Bio-inspired Reflexive Safe Reinforcement Learning for Contact-Rich Robotic Tasks. arXiv preprint arXiv:2503.21989.

Safe Learning for Contact-Rich Tasks

- A **survey** of recent safe learning methods contact-rich manipulation
  - Rising importance of contact-rich robotics
  - Multi-perspective categorization (control, sensing, learning, task)
  - A fresh perspective including
    - Generative methods
    - Foundational models



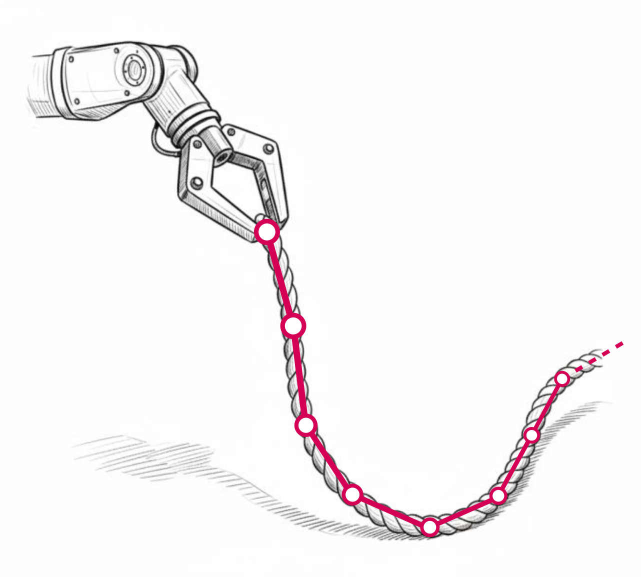


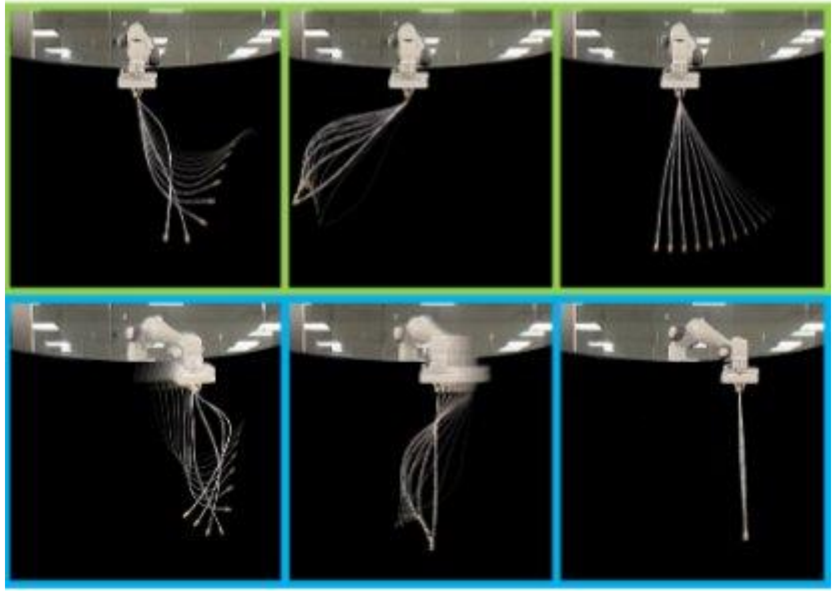


Zhang, H., Dai, R., Solak, G., Zhou, P., She, Y., & Ajoudani, A. (2025). Safe Learning for Contact-Rich Robot Tasks: A Survey from Classical Learning-Based Methods to Safe Foundation Models. arXiv preprint arXiv:2512.11908.

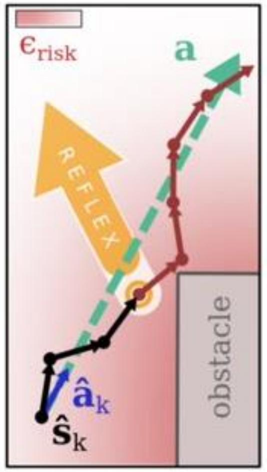
Contents

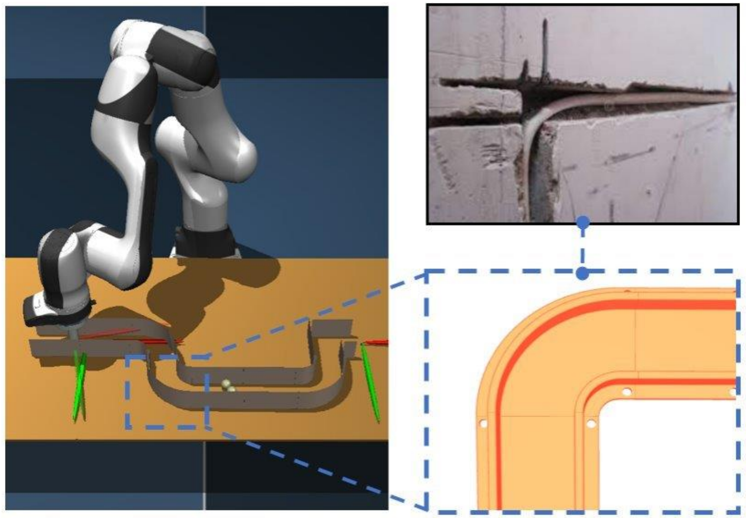
1. Physics-Informed Manipulation
   of Deformable Linear Objects





2. Learning safe interaction from data





3. Key takeaways

Key take-aways

- We can **combine model-based and model-free** to gain advantages of
  - Data efficiency
  - Complex behaviour
  - Fast inference
- Explicitly modeling **damping parameters improves** DLO state prediction in **dynamic tasks**
- Identifying a differentiable model allows efficiently learning a controller through self-supervision
- Curriculum learning helps converging in both model learning and domain randomization

Key take-aways

- We can **combine model-based and model-free** to gain advantages of
  - Data efficiency
  - Complex behaviour
  - Fast inference
- Explicitly modeling **damping parameters improves** DLO state prediction in **dynamic tasks**
- Identifying a differentiable model allows efficiently learning a controller through self-supervision
- Curriculum learning helps converging in both model learning and domain randomization

- Learning contact-rich safety is facilitated by variable impedance control
- Including energy-based passivity constraints leads to better energy economy as an addition to passivity guarantees
- Safety should be handled in short-term, high-frequency loop, while task efficiency increases with higher-level actions

We are hiring!

hrii.iit.it

Openings

Communication and design expert

Mechatronics Technician - Mechanical and Electronic Design for AI Edge-Computing

Software Engineer - Software developer with application on motion capture solutions

Technician position for the hardware design of innovative robotic and mechatronic systems

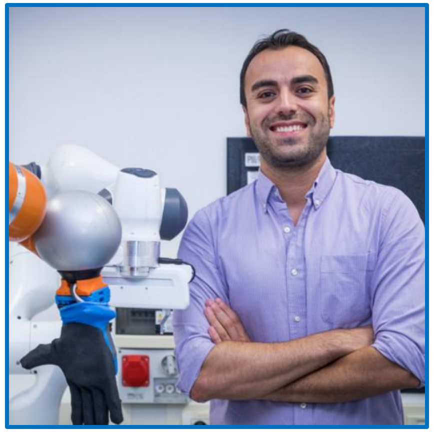
Software Technician - Software developer for Augmented Reality applications

Software Engineer with application on motion capture solutions

ISTITUTO ITALIANO
DI TECNOLOGIA
HUMAN-ROBOT INTERFACES
AND INTERACTION

Thank you!

Gokhan Solak **(Postdoc)**



Arash Ajoudani **(PI)**