Reinforcement Learning: States Representation, Morphological Computation, and Robustness.

Lekan Molu

New York City, NY 10012

Presented by **Lekan Molu** (Lay-con Mo-lu)

August 13, 2025

Talk Outline

- System Identification in Reinforcement Learning (RL);
- Robustness of Deep RL Policies:
  - Iterative Dynamic Game;
  - Convergence analysis in Deep RL: A Mixed H_2/H_∞ perspective.
- Reduced-order modeling and morphological control of emergent robot configurations;
- (Abundant details in Appendices)

Technical Overview

This page is left blank intentionally.

Credits

S. Chen

A. Koul

Y. Efroni

D. Misra

D. Foster

R. Islam

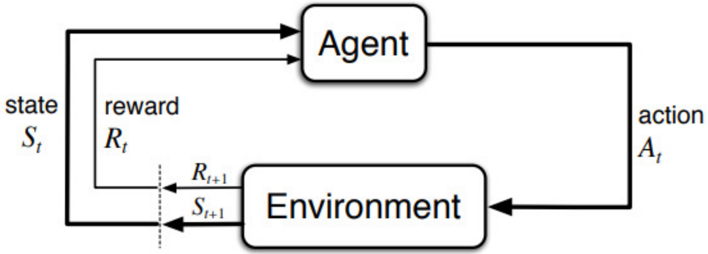
A. Lamb

M. Dudik

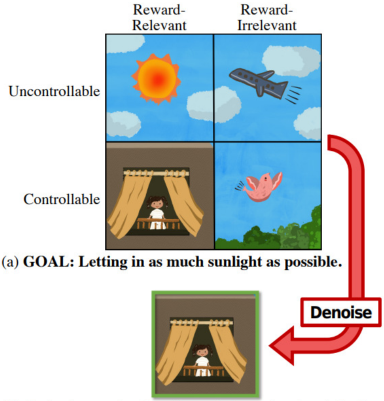
A. Krish.

J. Langford

Standard Reinforcement Learning



Compact States without Exogenous Distractors in RL

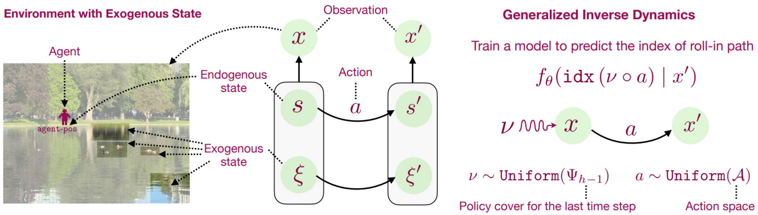


(a) **GOAL: Letting in as much sunlight as possible.**

(b) Optimal control only relies on information that is **both controllable and reward-relevant**. Good world models should ignore other factors as noisy distractors.

Denoised MDPs: Learning World Models Better Than the World Itself [5]

Compact States without Exogenous Distractors in RL



Learning s with [S] whilst ignoring temporally correlated ξ? Source: [3, Fig. 1].

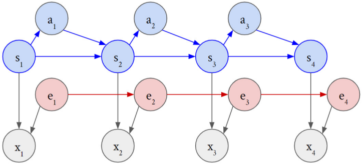
Literature comparison

Algorithms	PPE	OSSR	DBC	CDL	Denoised-MDP	1-Step Inverse	AC-State (Ours)
Exogenous Invariant State	✓	✓	✓	✓	✓	✓	✓
Exogenous Invariant Learning	✓	✓	✗	✗	✗	✓	✓
Flexible Encoder	✓	✗	✓	✗	✓	✓	✓
YOLO (No Resets) Setting	✗	✓	✓	✓	✓	✓	✓
Reward Free	✓	✓	✗	✓	✓	✓	✓
Control-Endogenous Rep.	✓	✓	✗	✓	✓	✗	✓

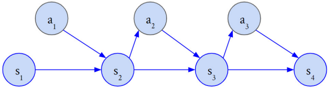
Emphasis on robustness to exogenous information. Comparison with baselines including PPE [3], OSSR [2], DBC [6] , Denoised MDP [5] and One-Step Inverse Models [4].

Rewards-agnostic Exogenous State Invariance in RL

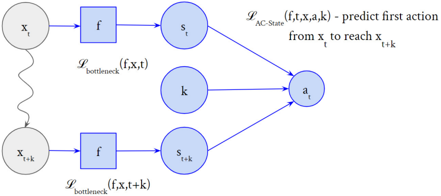
AC-State Discovers the smallest control-endogenous state s assuming factorized dynamics



AC-State collects data with a single random action followed by a high-coverage endogenous policy for k-1 steps



AC-State learns an encoder f for s = f(x) by optimizing a multi-step inverse model with a bottleneck



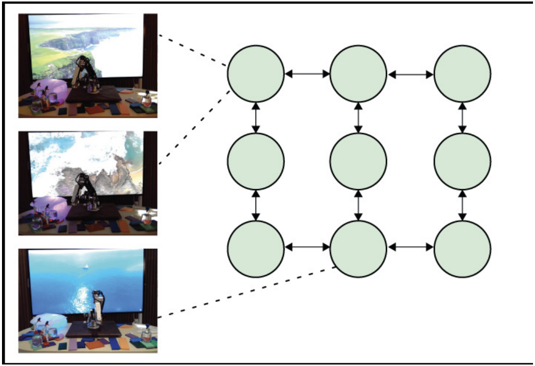
Latent States Discovery – Multi-step Inverse Dynamics

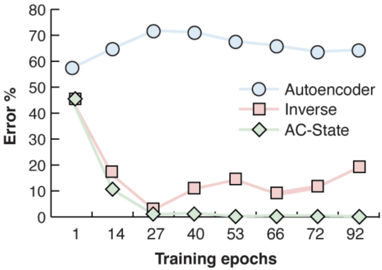
- $\hat{f} \approx \arg\min_{f \in \mathcal{F}} \mathbb{E}_{t,k} \Big[\mathcal{L}_{\mathrm{ACS}}\left(f, x, a, t, k\right) + \mathcal{L}_{\mathrm{B}}(f, x_t) + \mathcal{L}_{\mathrm{B}}(f, x_{t+k}) \Big]$

$$\mathcal{L}_{\mathrm{ACS}}\left(f, x, a, t; k\right) = -\log(\mathbb{P}\left(a_t | f(x_t), f(x_{t+k}); k\right)) \quad (1)$$

Agent Controllable State (AC State)

AC State in Action

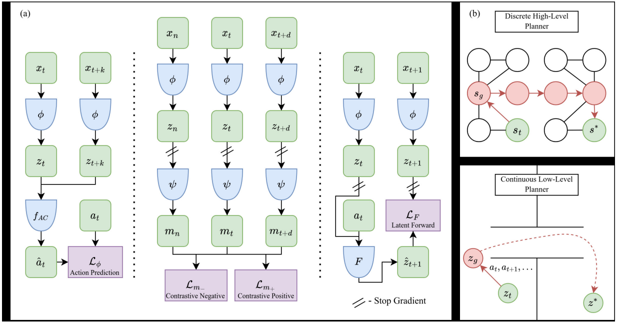




PCLAST: Agent Plannable Continuous Latent States

PcLast: Discovering Plannable Continuous Latent States

Anurag Koul [*1] **Shivakanth Sujit** [*234] **Shaoru Chen** [1] **Ben Evans** [5] **Lili Wu** [1] **Byron Xu** [1] **Rajan Chari** [1]
Riashat Islam [36] **Raihan Seraj** [36] **Yonathan Efroni** [7] **Lekan Molu** [1] **Miro Dudik** [1] **John Langford** [1] **Alex Lamb** [1]



PCLAST Algorithm

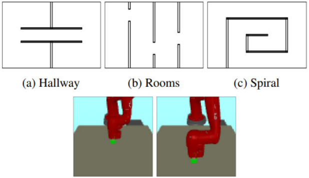
Algorithm 1 n-Level Planner

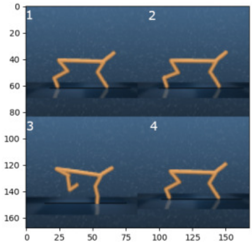
Require:

Current observation x_t
Goal observation x_{goal}
Planning horizon H
Encoder $\phi(\cdot)$
PCLAST map $\psi(\cdot)$
Latent forward dynamics $\delta(\cdot, \cdot)$
Multi-Level discrete transition graphs $\{\mathcal{G}_i\}_{i=2}^{n}$

Ensure: Action sequence $\{a_i\}_{i=0}^{H-1}$

Compute current continuous latent state $\hat{s}_t = \phi(x_t)$ and target latent state $\hat{s}^* = \phi(x_{goal})$.
{See Appendix E for details of high-level planner and low-level planner.}
for $i = n, n-1, \ldots, 2$ **do**
  $\hat{s}^*$ = high-level planner($\hat{s}_t$, $\hat{s}^*$, $\mathcal{G}_i$)
  {Update waypoint using a hierarchy of abstraction.}
end for
$\{a_i\}_{i=0}^{H-1}$ = low-level planner($\hat{s}_t, \hat{s}^*, H, \delta, \psi$)
{Solve the trajectory optimization problem.}


(a) Hallway (b) Rooms (c) Spiral
(d) Sawyer Reach Environment



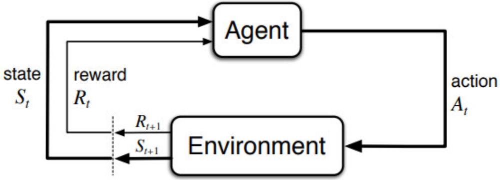
PCLAST Results

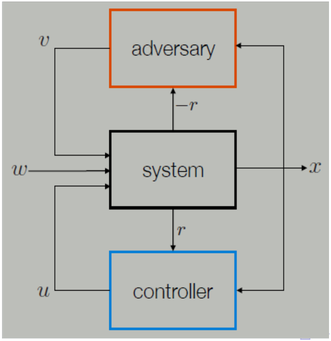
Method	Reward type	Hallway	Rooms	Spiral	Sawyer-Reach
PPO	Dense	6.7 ± 0.6	7.5 ± 7.1	11.2 ± 7.7	$\mathbf{86.00 \pm 5.367}$
PPO + ACRO	Dense	10.0 ± 4.1	23.3 ± 9.4	23.3 ± 11.8	84.00 ± 6.066
PPO + PCLaST	Dense	$\mathbf{66.7 \pm 18.9}$	$\mathbf{43.3 \pm 19.3}$	$\mathbf{61.7 \pm 6.2}$	78.00 ± 3.347
PPO	Sparse	1.7 ± 2.4	0.0 ± 0.0	0.0 ± 0.0	68.00 ± 8.198
PPO + ACRO	Sparse	21.7 ± 8.5	5.0 ± 4.1	11.7 ± 8.5	$\mathbf{92.00 \pm 4.382}$
PPO + PCLaST	Sparse	$\mathbf{50.0 \pm 18.7}$	$\mathbf{6.7 \pm 6.2}$	$\mathbf{46.7 \pm 26.2}$	82.00 ± 5.933
CQL	Sparse	3.3 ± 4.7	0.0 ± 0.0	0.0 ± 0.0	32.00 ± 5.93
CQL + ACRO	Sparse	15.0 ± 7.1	$\mathbf{33.3 \pm 12.5}$	$\mathbf{21.7 \pm 10.3}$	68.00 ± 5.22
CQL + PCLaST	Sparse	$\mathbf{40.0 \pm 0.5}$	23.3 ± 12.5	20.0 ± 8.2	$\mathbf{74.00 \pm 4.56}$
RIG	None	0.0 ± 0.0	0.0 ± 0.0	3.0 ± 0.2	$\mathbf{100.0 \pm 0.0}$
RIG + ACRO	None	$\mathbf{15.0 \pm 3.5}$	$4.0 \pm 1.$	$\mathbf{12.0 \pm 0.2}$	100.0 ± 0.0
RIG + PCLaST	None	10.0 ± 0.5	4.0 ± 1.8	10.0 ± 0.1	90.0 ± 5
Low-Level Planner + PCLaST	None	86.7 ± 3.4	69.3 ± 3.4	50.0 ± 4.3	$\pm$
n-Level Planner + PCLaST	None	$\mathbf{97.78 \pm 4.91}$	$\mathbf{89.52 \pm 10.21}$	$\mathbf{89.11 \pm 10.38}$	95.0 ± 1.54

Iterative Dynamic Game in RL

This page is left blank intentionally.

Inculcating robustness into multistage decision policies





Problem Setup

- To quantify the brittleness, we optimize the stage cost

$$\max_{\boldsymbol{v}_t \sim \psi \in \Psi} \left[\sum_{t=0}^{T} \underbrace{c(\boldsymbol{x}_t, \boldsymbol{u}_t)}_{\text{nominal}} - \gamma \underbrace{g(\boldsymbol{v}_t)}_{\text{adversarial}} \right]$$

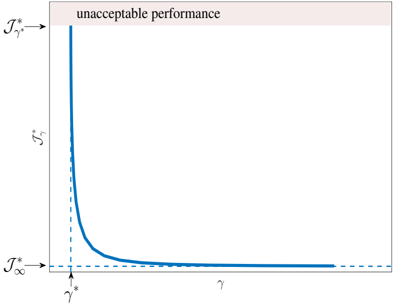
- To mitigate lack of robustness, we optimize the *cost-to-go*

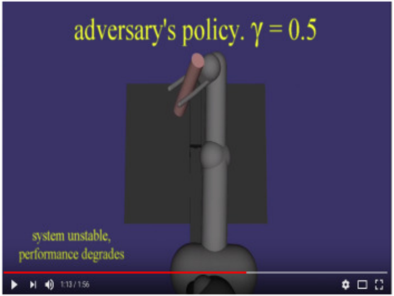
$$c_t(\boldsymbol{x}_t, \pi, \psi) = \min_{\boldsymbol{u}_t \sim \boldsymbol{\pi}} \max_{\boldsymbol{v}_t \sim \psi} \left(\sum_{t=0}^{T-1} \ell_t(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{v}_t) + L_T(\boldsymbol{x}_T) \right),$$

- and seek a saddle point equilibrium policy that satisfies

$$c_t(\boldsymbol{x}_t, \pi^*, \psi) \leq c_t(\boldsymbol{x}_t, \pi^*, \psi^*) \leq c_t(\boldsymbol{x}_t, \pi, \psi^*),$$

Results: Brittleness Quantification

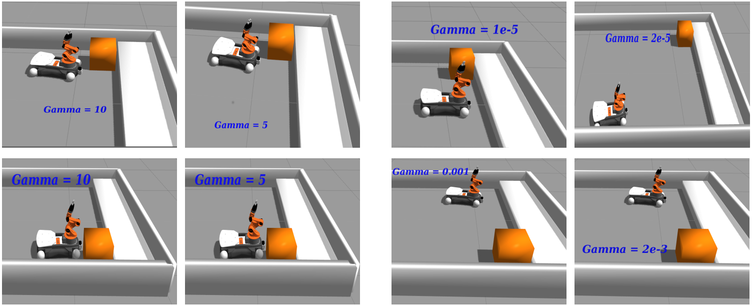




Results: Iterative Dynamic Game

$\boldsymbol{x}_1^\star$

$\boldsymbol{x}_2^\star$



End pose of the KUKA platform with our iDG formulation given different goal states and γ-values.

Mixed H_2/H_∞ Policy Optimization in RL

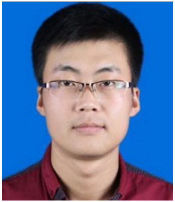
This page is left blank intentionally.

Talk Outline and Overview

- Policy Optimization and Stochastic Linear Control
  - Connections to risk-sensitive control;
  - Mixed $\mathcal{H}_2/\mathcal{H}_\infty$ control theory.
- The case for convergence analysis in stochastic PO.
  - Kleinman's algorithm, *redux.*
  - Kleiman's algorithm in an iterative best response setting;
  - PO Convergence in best response settings.
- Robustness margins in model- and sampling- settings.
  - PO as a discrete-time nonlinear system;
  - Kleiman and input-to-state-stability;
  - Robust policy optimization as a small-input stable state optimization algorithm

Credits

Leilei Cui



Postdoc, MIT

Zhong-Ping Jiang

Professor, NYU

Research Significance

- (Deep) RL and modern AI
  - Robotic manipulation (Levine et al., 2016), text-to-visual processing (DALL-E), Atari games (Mnih et al., 2013), e.t.c.
  - Policy optimization (PO) is fundamental to modern AI algorithms' success.
  - Major success story: functional mapping of observations to policies.
  - But how does it work?

Policy Optimization – General Framework

- PO encapsulates policy gradients (Kakade, 2001) or PG, actor-critic methods (Vrabie and Lewis, 2011), trust region PO Schulman et al. (2015), and proximal PO methods (Schulman et al., 2017).
- PG particularly suitable for complex systems.

$$\min J(K) \quad \text{subject to } K \in \mathcal{K} \tag{1}$$

where $\mathcal{K} = \{K_1, K_2, \cdots, K_n\}$.

- $J(K)$ could be tracking error, safety assurance, goal-reaching measure of performance e.t.c. required to be satisfied.

Policy Optimization – Open questions

- Gradient-based data-driven methods: prone to divergence from true system gradients.
  - Challenge I: Optimization occurs in non-convex objective landscapes.
    - Get performance certificates as a mainstay for control design: Coerciveness property (Hu et al., 2023).
  - Challenge II: Taming PG's characteristic high-variance gradient estimates (REINFORCE, NPG, Zeroth-order approx.).
    - Hello, (linear) robust ($\mathcal{H}_\infty$-synthesis) control!

Policy Optimization – Open questions

- Challenge III: Under what circumstances do we have convergence to a desired equilibrium in RL settings?
- Challenge IV: Stochastic control, not deterministic control settings.
  - models involving round-off error computations in floating point arithmetic calculations; the stock market; protein kinetics.
- Challenge V: Continuous-time RL control.
  - Very little theory. Lots of potential applications encompassing rigid and soft robotics, aerospace or finance engineering, protein kinetics.

(Non-exhaustive) Lit. Landscape on PO Theory

Literature landscape	Cont. time (Kalman '61, Luenberger '63)	Stochastic. LQR (Kalman '60)	Cont. Phase	LEQG or Mixed H_2/H_∞	Finite/Infinite Horizon
Fazel (2018)	No	No	Yes	No	Finite-horizon
Mohammadi (TAC -- 2020)	Yes	No	Yes	No	Finite-Horizon
Zhang (2019)	Yes	Yes (Gaussian)	Yes	Yes	Inf-horizon
Gravell (2021)	No	Multiplicative	Yes	No	Inf-horizon
Zhang (2020)	No	No	Yes	Yes	Rand-horizon
Molu (2022)	Yes	Yes (Brownian)	Yes	Yes	Inf-Horizon
Cui & Molu (2023)	Yes	Yes (Brownian)	Yes	Yes	Inf-Horizon

Model-based Policy Iteration

Algorithm 1: (Model-Based) PO via Policy Iteration

Input: Max. outer iteration $\bar{p}$, $q = 0$, and an $\epsilon > 0$;
Input: Desired risk attenuation level $\gamma > 0$;
Input: Minimizing player's control matrix $R \succ 0$.

```
Compute (K_0, L_0) ∈ 𝒦;                         ▷ From [24, Alg. 1];
Set P_{K,L}^{0,0} = Q_K^0;                         ▷ See equation (9);
for p = 0, ..., p̄ do
    Compute Q_K^p and A_K^p                        ▷ See equation (9);
    Obtain P_K^p by evaluating K_p on (10);
    while ||P_K^p − P_{K,L}^{p,q}||_F ≤ ϵ do
        Compute L_{q+1}(K_p) := γ^{-2} D^⊤ P_{K,L}^{p,q};
        Solve (11) until ||P_K^p − P_{K,L}^{p,q}||_F ≤ ϵ;
        q̄ ← q + 1
    end
    Compute K_{p+1} = R^{-1} B^⊤ P_{K,L}^{p,q̄}     ▷ See (11b) ;
end
```

Convergence of the Inner Loop Iteration

Theorem 3

For a $K \in \check{\mathcal{K}}$, and for any $(p, q) \in \mathbb{N}_+$, there exists $\beta(K) \in \mathbb{R}$ such that

$$Tr(P_K^p - P_{K,L}^{p,q+1}) \le \beta(K) Tr(P_K^p - P_{K,L}^{p,q}). \tag{24}$$

Remark 2

*As seen from Lemma 5, $P_K^p - P_{K,L}^{p,q} \succeq 0$. By the norm on a matrix trace (**?**, Lemma 13) and the result of Theorem 3, we have $\|P_K - P_{K,L}^{p,q}\|_F \le Tr(P_K - P_{K,L}^{p,q}) \le \beta(K) Tr(P_K)$, i.e. $P_{K,L}^{p,q}$ exponentially converges to P_K in the Frobenius norm.*

Robustness Analyses

- Define $\tilde{P} = P_K - \hat{P}_K$ and $\tilde{K} = K - \hat{K}$.
- Keep $|\tilde{K}| < \epsilon$, start with a $K \in \mathcal{K}$: iterates stay in $\mathcal{K}$.

Lemma 7 (Lemma 10, C&M, '23)

For any $K \in \mathcal{K}$, there exists an $e(K) > 0$ such that for a perturbation $\tilde{K}$, $K + \tilde{K} \in \mathcal{K}$, as long as $\|\tilde{K}\| < e(K)$.

Theorem 6

The inexact outer loop is small-disturbance ISS. That is, for any $h > 0$ and $\hat{K}_0 \in \mathcal{K}_h$, if $\|\tilde{K}\| < f(h)$, there exist a $\mathcal{KL}$-function $\beta_1(\cdot,\cdot)$ and a $\mathcal{K}_\infty$-function $\gamma_1(\cdot)$ such that

$$\|P^p_{\hat{K}} - P^\star\| \le \beta_1(\|P^0_{\hat{K}} - P^*\|, p) + \gamma_1(\|\tilde{K}\|). \tag{37}$$

Inner Loop Robustness

Theorem 7

Assume $\|\tilde{L}_q(K_p)\| < e$ *for all* $q \in \mathbb{N}_+$. *There exists* $\hat{\beta}(K) \in [0, 1)$, *and* $\lambda(\cdot) \in \check{\mathcal{K}}_\infty$, *such that*

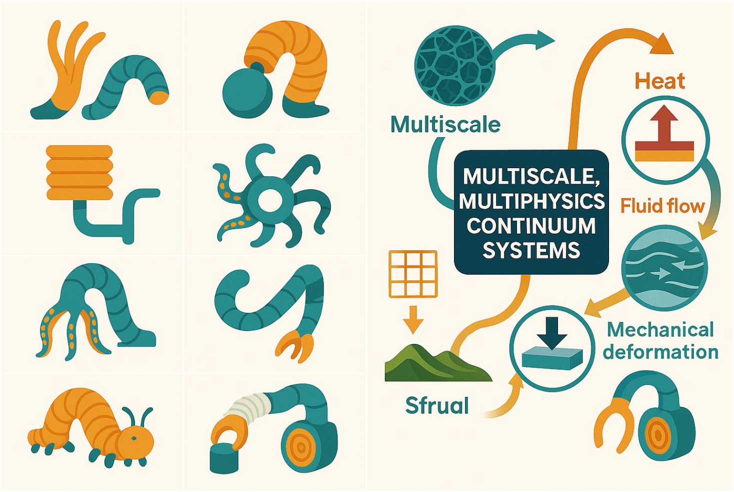
$$\|\hat{P}_{K,L}^{p,q} - P_{K,L}^{p,q}\|_F \leq \hat{\beta}^{q-1}(K)\, Tr(P_{K,L}^{p,q}) + \lambda(\|\tilde{L}\|_\infty). \tag{42}$$

- From Theorem 7, as $q \to \infty$, $\hat{P}_{K,L}^{p,q}$ approaches the solution P_K and enters the ball centered at $P_{K,L}^{p,q}$ with radius proportional to $\|\tilde{L}\|_\infty$.
- The proposed inner-loop iterative algorithm well approximates $P_{K,L}^{p,q}$.

Morphological Computation in Emergent Robotic Systems

This page is left blank intentionally.

Soft Robotic Systems



Credit: Microsoft CoPilot.

The Piecewise Constant Strain (PCS) Cosserat Model

Octopus robot. Courtesy: IEEE Spectrum

Picture generated by Google Gemini

$$\underbrace{\left[\int_0^{L_N} \boldsymbol{J}^\top \boldsymbol{\mathcal{M}}_a \boldsymbol{J} d\boldsymbol{X}\right]}_{\boldsymbol{M}(\boldsymbol{q})} \ddot{\boldsymbol{q}} + \underbrace{\left[\int_0^{L_N} \boldsymbol{J}^\top \mathrm{ad}^\star_{\boldsymbol{J}\dot{\boldsymbol{q}}} \boldsymbol{\mathcal{M}}_a \boldsymbol{J} d\boldsymbol{X}\right]}_{\boldsymbol{C}_1(\boldsymbol{q},\dot{\boldsymbol{q}})} \dot{\boldsymbol{q}} +$$

$$\underbrace{\left[\int_0^{L_N} \boldsymbol{J}^\top \boldsymbol{\mathcal{M}}_a \dot{\boldsymbol{J}} d\boldsymbol{X}\right]}_{\boldsymbol{C}_2(\boldsymbol{q},\dot{\boldsymbol{q}})} \dot{\boldsymbol{q}} + \underbrace{\left[\int_0^{L_N} \boldsymbol{J}^\top \boldsymbol{\mathcal{D}} \boldsymbol{J} \|\boldsymbol{J}\dot{\boldsymbol{q}}\|_p d\boldsymbol{X}\right]}_{\boldsymbol{D}(\boldsymbol{q},\dot{\boldsymbol{q}})} \dot{\boldsymbol{q}} -$$

$$\underbrace{(1-\rho_f/\rho)\left[\int_0^{L_N} \boldsymbol{J}^\top \boldsymbol{\mathcal{M}} \mathrm{Ad}_{\boldsymbol{g}}^{-1} d\boldsymbol{X}\right]}_{\boldsymbol{N}(q)} \mathrm{Ad}_{\boldsymbol{g}_r}^{-1} \boldsymbol{\mathcal{G}} - \underbrace{\boldsymbol{J}^\top(\bar{\boldsymbol{X}})\boldsymbol{\mathcal{F}}_p}_{\boldsymbol{F}(\boldsymbol{q})}$$

$$\underbrace{-\int_0^{L_N} \boldsymbol{J}^\top \left[\nabla_x \boldsymbol{\mathcal{F}}_i - \nabla_x \boldsymbol{\mathcal{F}}_a + \mathrm{ad}^\star_{\eta_n} (\boldsymbol{\mathcal{F}}_i - \boldsymbol{\mathcal{F}}_a)\right] d\boldsymbol{X}}_{\boldsymbol{u}(\boldsymbol{q})} = 0,$$

Renda et al. T-RO 2016

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{z}} + [\boldsymbol{C}_1(\boldsymbol{q},\boldsymbol{z}) + \boldsymbol{C}_2(\boldsymbol{q},\boldsymbol{z}) + \boldsymbol{D}(\boldsymbol{q},\boldsymbol{z})]\,\boldsymbol{z} = \tau(\boldsymbol{q}) + \boldsymbol{F}(\boldsymbol{q}) + \boldsymbol{N}(q)\mathrm{Ad}_{\boldsymbol{g}_r}^{-1}\mathcal{G}.$$

SoRo's control computational complexity is hard!

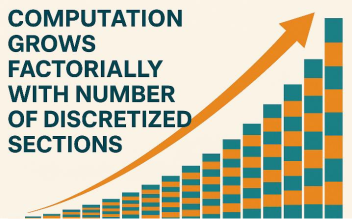
Structural Properties and Control of Soft Robots Modeled as Discrete Cosserat Rods

Lekan Molu and Shaoru Chen

***Abstract*— Soft robots featuring approximate finite-dimensional reduced-order models (undergoing small deformations) are increasingly becoming paramount in literature and applications. In this paper, we consider the piecewise constant strain (PCS) discrete Cosserat model whose dynamics admit the standard Newton-Euler dynamics for a kinetic model. Contrary to popular convention that soft robots under these modeling assumptions admit similar mechanical characteristics to rigid robots, the schemes employed to arrive at the properties for soft robots under finite deformation show a far dissimilarity to those for rigid robots. We set out to first correct the false premise behind this syllogism: from first principles, we established the structural properties of soft slender robots undergoing finite deformation under a discretized PCS assumption; we then utilized these properties to prove the stability of designed proportional-derivative controllers for manipulating the strain states of a prototypical soft robot under finite deformation. Our newly derived results are illustrated by numerical examples on a single arm of the Octopus robot and demonstrate the efficacy of our designed controller based on the derived kinetic properties. This work rectifies previously disseminated kinetic properties of discrete Cosserat-based soft robot models with greater accuracy in proofs and clarity.**

Nonlinear partial differential equations (PDEs) are the standard mathematical machinery for modeling continuum structures with distributed mass. And for soft robots exhibiting infinite degrees-of-freedom (DoF), nonlinear PDEs readily come in handy. However, scanty theory exists for nonlinear PDE analyses. To circumvent the complexity of PDE analyses, researchers have so far exploited approximate finite-dimensional ordinary differential equations (ODEs) [7] for analysis on spatially reduced models.

Tractable reduced-order mathematical models are typically formulated by restricting the range of shapes of the continuum robot to a finite-dimensional functional space over a curve that parameterizes the robot. This is equivalent to taking finite nodal points on the soft robot's body and approximating the dynamics along discretized nodal sections by an ODE. An aggregated ODE of all discretized sections can then be used to model the dynamics of the entire discretized continuum robot. A paramount example is the discrete Cosserat model of Renda et al. [18] whereupon the nonlinear PDE that describes the robot's kinetics in exact form is abstracted to standard Newton-Euler ODEs via



Enter Singularly Perturbed Systems

$$\dot{\boldsymbol{z}}_1 = \boldsymbol{f}(\boldsymbol{z}_1, \boldsymbol{z}_2, \epsilon, \boldsymbol{u}_s, t),\ \ \boldsymbol{z}_1(t_0) = \boldsymbol{z}_1(0),\ \ \boldsymbol{z}_1 \in \mathbb{R}^{6N},$$
$$\epsilon\dot{\boldsymbol{z}}_2 = \boldsymbol{g}(\boldsymbol{z}_1, \boldsymbol{z}_2, \epsilon, \boldsymbol{u}_f, t),\ \ \boldsymbol{z}_2(t_0) = \boldsymbol{z}_2(0),\ \ \boldsymbol{z}_2 \in \mathbb{R}^{6N}$$

General SPT formulation.

$$\dot{\boldsymbol{z}}_1 = \boldsymbol{f}(\boldsymbol{z}_1, \boldsymbol{z}_2, 0, \boldsymbol{u}_s, t),\ \ \boldsymbol{z}_1(t_0) = \boldsymbol{z}_1(0),$$
$$0 = \boldsymbol{g}(\boldsymbol{z}_1, \boldsymbol{z}_2, 0, 0, t).$$

Set ϵ to 0 → Slow subsystem

$$\frac{d\boldsymbol{z}_1}{dT} = \epsilon \boldsymbol{f}(\boldsymbol{z}_1, \tilde{\boldsymbol{z}}_2 + \boldsymbol{\phi}(\boldsymbol{z}_1, t), \epsilon, \boldsymbol{u}_s, t), \tag{8a}$$
$$\frac{d\tilde{\boldsymbol{z}}_2}{dT} = \epsilon\frac{d\boldsymbol{z}_2}{dt} - \epsilon\frac{\partial\boldsymbol{\phi}}{\partial\boldsymbol{z}_1}\dot{\boldsymbol{z}}_1, \tag{8b}$$
$$= \boldsymbol{g}(\boldsymbol{z}_1, \tilde{\boldsymbol{z}}_2 + \boldsymbol{\phi}(\boldsymbol{z}_1, t), \epsilon, \boldsymbol{u}_f, t) - \epsilon\frac{\partial\boldsymbol{\phi}(\boldsymbol{z}_1, t)}{\partial\boldsymbol{z}_1}\dot{\boldsymbol{z}}_1. \tag{8c}$$

Fast subsystem on time scale: $T = t/\epsilon$

Multiphysics, multiscale soft system.

Picture credit: Google Gemini.

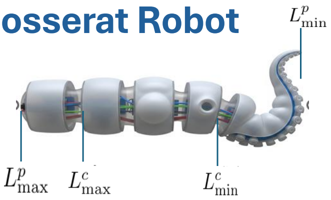
Assumption 1 (Real and distinct root): Equation (5) has the unique and distinct root $\boldsymbol{z}_2 = \boldsymbol{\phi}(\boldsymbol{z}_1, t)$ (for a sufficiently smooth $\boldsymbol{\phi}(\cdot)$) so that

$$0 = \boldsymbol{g}(\boldsymbol{z}_1, \boldsymbol{\phi}(\boldsymbol{z}_1, t), 0, 0, t) \triangleq \bar{\boldsymbol{g}}(\boldsymbol{z}_1, 0, t),\ \ \boldsymbol{z}_1(t_0) = \boldsymbol{z}_1(0). \tag{6}$$

The slow subsystem therefore becomes

$$\dot{\boldsymbol{z}}_1 = \boldsymbol{f}(\boldsymbol{z}_1, \boldsymbol{\phi}(\boldsymbol{z}_1, t), 0, \boldsymbol{u}_s, t) \triangleq \boldsymbol{f}_s(\boldsymbol{z}_1, \boldsymbol{u}_s, t). \tag{7}$$

Singularly Perturbed Soft Cosserat Robot

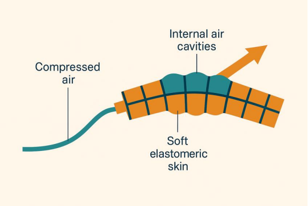


Aggregate the robot's distributed mass, $\mathcal{M}$, inertia into a core active component, $\mathcal{M}_i^{\text{core}}$, and set the passive components as $\mathcal{M}^{\text{pert}} = \mathcal{M} \setminus \mathcal{M}^{\text{core}}$

Then the mass and Coriolis forces adopts the following representation

$$\text{where} \quad \boldsymbol{M}^p = \int_{L^p_{\text{min}}}^{L^p_{\text{max}}} \boldsymbol{J}^\top \mathcal{M}^{pert} \boldsymbol{J} dX$$

$$\boldsymbol{M}(\boldsymbol{q}) = (\boldsymbol{M}^c + \boldsymbol{M}^p)(\boldsymbol{q}), \; \boldsymbol{N} = (\boldsymbol{N}^c + \boldsymbol{N}^p)(\boldsymbol{q}),$$
$$\boldsymbol{F}(\boldsymbol{q}) = (\boldsymbol{F}^c + \boldsymbol{F}^p)(\boldsymbol{q}), \quad \boldsymbol{D}(\boldsymbol{q}) = (\boldsymbol{D}^c + \boldsymbol{D}^p)(\boldsymbol{q})$$
$$\boldsymbol{C}_1(\boldsymbol{q}, \dot{\boldsymbol{q}}) = (\boldsymbol{C}_1^c + \boldsymbol{C}_1^p)(\boldsymbol{q}, \dot{\boldsymbol{q}}),$$
$$\boldsymbol{C}_2(\boldsymbol{q}, \dot{\boldsymbol{q}}) = (\boldsymbol{C}_2^c + \boldsymbol{C}_2^p)(\boldsymbol{q}, \dot{\boldsymbol{q}})$$



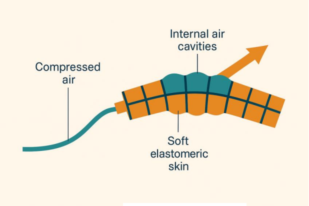
Picture credit: Google Gemini.

Dynamics Separation with Perturbation Parameter

The mass matrix then decomposes as

$$\boldsymbol{M} = \underbrace{\begin{bmatrix} \mathcal{H}_{\text{fast}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}}_{\boldsymbol{M}^c(\boldsymbol{q})} + \underbrace{\begin{bmatrix} \mathbf{0} & \mathcal{H}_{\text{slow}}^{\text{fast}} \\ \mathcal{H}_{\text{slow}}^{\text{fast}\,\top} & \mathcal{H}_{\text{slow}} \end{bmatrix}}_{\boldsymbol{M}^p(\boldsymbol{q})},$$

$\boldsymbol{M}^c(\boldsymbol{q})$ and $\boldsymbol{M}^p(\boldsymbol{q})$ are invertible (Molu & Chen, CDC 2024)



Introducing the perturbation parameter, $\epsilon = \|\boldsymbol{M}^p\| / \|\boldsymbol{M}^c\|$ We may define the matrix, $\bar{\boldsymbol{M}}^p = \boldsymbol{M}^p/\epsilon$

So that we can write,

$$(\boldsymbol{M}^c + \epsilon \bar{\boldsymbol{M}}^p)\dot{\boldsymbol{z}} = \boldsymbol{s} + \boldsymbol{u},$$

where

$$\boldsymbol{s} = \begin{bmatrix} \boldsymbol{s}_{\text{fast}} \\ \boldsymbol{s}_{\text{slow}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{F}^c + \boldsymbol{N}^c \text{Ad}_{\boldsymbol{g}_r}^{-1} \mathcal{G} - [\boldsymbol{C}_1^c + \boldsymbol{C}_2^c + \boldsymbol{D}^c]\boldsymbol{z}_{\text{fast}} \\ \boldsymbol{F}^p + \boldsymbol{N}^p \text{Ad}_{\boldsymbol{g}_r}^{-1} \mathcal{G} - [\boldsymbol{C}_1^p + \boldsymbol{C}_2^p + \boldsymbol{D}^p]\boldsymbol{z}_{\text{slow}} \end{bmatrix}. \quad (13)$$

Singularly perturbed soft robot form

Suppose that

$$\bar{M}^p = \begin{bmatrix} \bar{M}^p_{11} & \bar{M}^p_{12} \\ \bar{M}^p_{21} & \bar{M}^p_{22} \end{bmatrix} \text{ and } \boldsymbol{\Delta} = \begin{bmatrix} 0 & 0 \\ \bar{M}^p_{21}\mathcal{H}^{-1}_{\text{fast}} & 0 \end{bmatrix},$$

Then, we may write

$$\begin{bmatrix} \mathcal{H}_{\text{fast}} & \bar{M}^p_{12} \\ 0 & \bar{M}^p_{22} \end{bmatrix} \begin{bmatrix} \dot{z}_{\text{fast}} \\ \epsilon \dot{z}_{\text{slow}} \end{bmatrix} = \begin{bmatrix} s_{\text{fast}} \\ s_{\text{slow}} - \epsilon \bar{M}^p_{21}\mathcal{H}^{-1}_{\text{fast}} s_{\text{fast}} \end{bmatrix} + \begin{bmatrix} u_{\text{fast}} \\ u_{\text{slow}} - \epsilon \bar{M}^p_{21}\mathcal{H}^{-1}_{\text{fast}} u_{\text{fast}} \end{bmatrix} \quad (16)$$

Fast subdynamics extraction

Set $T = t/\epsilon$, with $dT/dt = 1/\epsilon$

Then, $\dot{z}_{\text{fast}} = \dfrac{dz_{\text{fast}}}{dt} \equiv \dfrac{1}{\epsilon}\dfrac{dz_{\text{fast}}}{dT} \triangleq \dfrac{1}{\epsilon} z'_{\text{fast}}$

and $\epsilon \dot{z}_{\text{slow}} = z'_{\text{slow}}$.

So that,

$$\begin{aligned} z'_{\text{fast}} &= \epsilon \mathcal{H}^{-1}_{\text{fast}}(s_{\text{fast}} + u_{\text{fast}}) - \mathcal{H}^{-1}_{\text{fast}}\mathcal{H}^{\text{fast}}_{\text{slow}} z'_{\text{slow}} \\ z'_{\text{slow}} &= \mathcal{H}^{-1}_{\text{slow}}(s_{\text{slow}} - u_{\text{slow}}) - \mathcal{H}^{-1}_{\text{fast}}(s_{\text{fast}} - u_{\text{fast}}) \end{aligned}$$

A backstepping nonlinear multi-scale controller

Theorem 1: The control law

$$\boldsymbol{q}^d_{\text{fast}}(t_f) - \boldsymbol{q}_{\text{fast}}(t_f) + \boldsymbol{q}'^d_{\text{fast}}(t_f)$$

is sufficient to guarantee an exponential stability of the origin of $\boldsymbol{\theta}' = \boldsymbol{\nu}$ such that for all $t_f \geq 0$, $\boldsymbol{q}_{\text{fast}}(t_f) \in S$ for a compact set $S \subset \mathbb{R}^{6N}$. That is, $\boldsymbol{q}_{\text{fast}}(t_f)$ remains bounded as $t_f \to \infty$.

Where,

$$[\boldsymbol{\theta}^\top, \boldsymbol{\phi}^\top]^\top = [\boldsymbol{q}^\top_{\text{fast}}, \boldsymbol{z}^\top_{\text{fast}}]^\top \text{ where } \boldsymbol{\theta}' = \epsilon \boldsymbol{z}_{\text{fast}}.$$

Theorem 2: Under the tracking error $\boldsymbol{e}_2 = \boldsymbol{\phi} - \boldsymbol{\nu}$ and matrices $(\boldsymbol{K}_p, \boldsymbol{K}_q) = (\boldsymbol{K}_p^\top, \boldsymbol{K}_q^\top) > 0$, the control input

$$\boldsymbol{u}_{\text{fast}} = \frac{1}{\epsilon}\boldsymbol{\mathcal{H}}_{\text{fast}}[\boldsymbol{q}''^d_{\text{fast}} + \boldsymbol{e}_1 - 2\boldsymbol{e}_2 - \boldsymbol{K}_q^\top(\boldsymbol{K}_q\boldsymbol{K}_q^\top)^{-1}\boldsymbol{K}_p\boldsymbol{e}_1] + \frac{1}{\epsilon}\boldsymbol{\mathcal{H}}^{\text{fast}}_{\text{slow}}\boldsymbol{z}'_{\text{slow}} - \boldsymbol{s}_{\text{fast}} \quad (24)$$

exponentially stabilizes the fast subdynamics (18).

Theorem 3: The control law

$$\boldsymbol{u}_{\text{slow}} = \boldsymbol{\mathcal{H}}_{\text{slow}}(\boldsymbol{e}_1 - \boldsymbol{e}_2 - \boldsymbol{e}_3 + \ddot{\boldsymbol{q}}^d_{\text{fast}}) - \boldsymbol{s}_{\text{slow}}$$

exponentially stabilizes the slow subdynamics.

A backstepping nonlinear multi-scale controller

4) Stability of the singularly perturbed interconnected system: Let $\varepsilon = (0, 1)$ and consider the composite Lyapunov function candidate $\Sigma(\boldsymbol{z}_{\text{fast}}, \boldsymbol{z}_{\text{slow}})$ as a weighted combination of $\boldsymbol{V}_2$ and $\boldsymbol{V}_3$ i.e. ,

$$\boldsymbol{\Sigma}(\boldsymbol{z}_{\text{fast}}, \boldsymbol{z}_{\text{slow}}) = (1-\varepsilon)\boldsymbol{V}_2(\boldsymbol{z}_{\text{fast}}) + \varepsilon \boldsymbol{V}_3(\boldsymbol{z}_{\text{slow}}),\ 0 < \varepsilon < 1. \tag{35}$$

It follows that,

$$\begin{aligned}\dot{\boldsymbol{\Sigma}}(\boldsymbol{z}_{\text{fast}}, \boldsymbol{z}_{\text{slow}}) &= (1-\varepsilon)[\boldsymbol{e}_1^\top \boldsymbol{K}_p \dot{\boldsymbol{e}}_1 + \boldsymbol{e}_2^\top \boldsymbol{K}_q \dot{\boldsymbol{e}}_2] + \varepsilon \boldsymbol{e}_3^\top \boldsymbol{K}_r \dot{\boldsymbol{e}}_3, \\ &= -2(\boldsymbol{V}_2 + \boldsymbol{V}_3) + 2\varepsilon \boldsymbol{V}_2 \leq 0 \end{aligned} \tag{36}$$

which is clearly negative definite for any $\varepsilon \in (0, 1)$. Therefore, we conclude that the origin of the singularly perturbed system is asymptotically stable under the control laws.

$$\boldsymbol{u}(\boldsymbol{z}_{\text{fast}}, \boldsymbol{z}_{\text{slow}}) = (1-\varepsilon)\boldsymbol{u}_{\text{fast}} + \varepsilon \boldsymbol{u}_{\text{slow}}. \tag{37}$$

Numerical Results

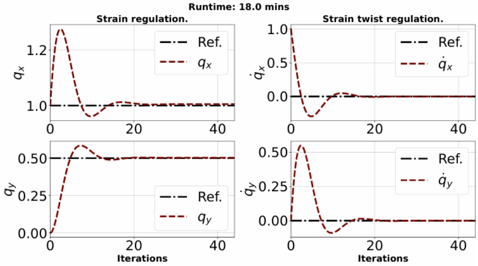


Fig. 2. Backstepping control on the singularly perturbed soft robot system with 10 discretized pieces, divided into 6 fast and 4 slow pieces. For a tip load of $\mathcal{F}_p^y = 10\,N$, the backstepping gains were set as $\boldsymbol{K}_p = 10$, $\boldsymbol{K}_d = 2.0$ for a desired joint configuration $\xi^d = [0, 0, 0, 1, 0.5, 0]^\top$ and $\eta^d = \mathbf{0}_{6\times1}$ that is uniform throughout the robot sections.

Pieces			Runtime (mins)	
Total	Fast	Slow	Hierarchical SPT (mins)	Single-layer PD control (hours)
6	4	2	18.01	51.46
8	5	3	30.87	68.29
10	7	3	32.39	107.43

TABLE I

TIME TO REACH STEADY STATE.

Numerical Results – System Setup

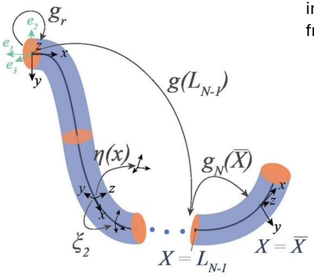


Fig. 1. Simplified configuration of an Octopus arm, reprinted from Molu and Chen [9].

The robot's z-axis is offset in orientation from the inertial frame by -90 deg so that a transformation from the base to inertial frames is

$$\boldsymbol{g}_r = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Tip wrench at $\bar{X} = L$ is ,

$$\boldsymbol{\mathcal{F}}_p = \text{diag}\left(\boldsymbol{R}^\top(L), \boldsymbol{R}^\top(L)\right)\begin{pmatrix} \boldsymbol{0}_{3\times1} & 0 & 10 & 0 \end{pmatrix}^\top$$

Param	Symbol	Value
Reynold's #		0.82
Young's Mod.	**E**	110*kPa*
Shear visc.	**J**	3 *kPa*

Numerical Results – System Setup

Param	Symbol	Value
Reynold's #		0.82
Young's Mod.	**E**	110*kPa*
Shear visc.	**J**	3 *kPa*
Bending 2nd Inertia	$I_y = I_z = \pi r^4/4$	
Torsion 2nd Inert	$I_x = \pi r^4/2$	
Material abscissa	$L = 2m$	
Poisson ratio	ρ	0.45
Mass density	$\mathcal{M} = \rho \cdot \mathrm{diag}([I_x, I_y, I_z, A, A, A])$	
Drag stiffness matrix	$\boldsymbol{D} = -\rho_w \nu^T \nu \check{\boldsymbol{D}} \nu / \lvert\nu\rvert$	

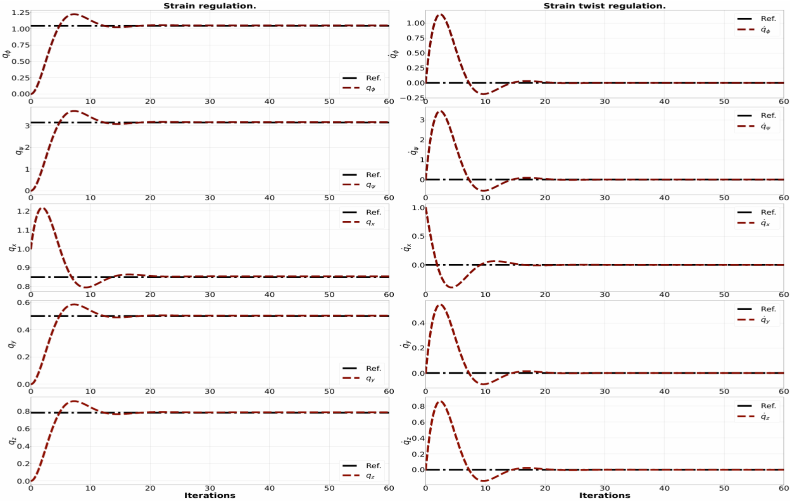


Fig. 3. Backstepping control on the singularly perturbed soft robot system with 10 pieces 4 slow and 6 fast sections.