

Wiggle and Go! System Identification for Zero-Shot Dynamic Rope Manipulation

Arthur Jakobsson¹, Abhinav Mahajan¹, Karthik Pullalarevu¹, Krishna Suresh¹, Yunchao Yao²,
Yuemin Mao¹, Bardienus Duisterhof¹, Shahram Najam Syed¹, Jeffrey Ichnowski¹

¹Robotics Institute, Carnegie Mellon University

²University of North Carolina at Chapel Hill

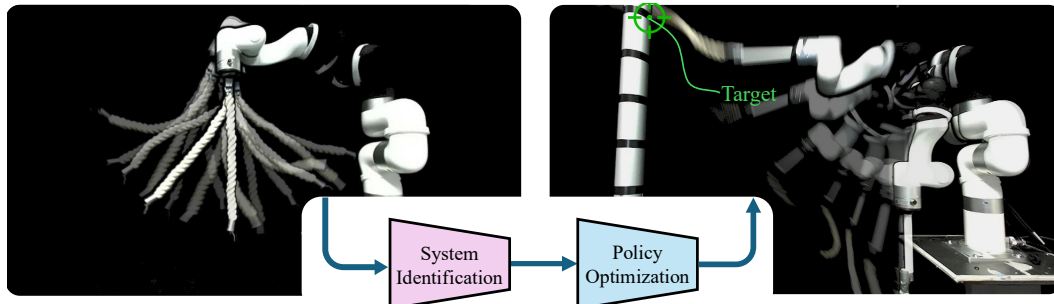


Figure 1: Wiggle and Go! uses a brief *wiggle* motion (left) to identify the dynamic behavior of a rope using a neural network model. This is fed into a goal-conditioned policy optimizer, which then *goes!*, performing a dynamic fling (right), hitting the target in one shot. Our system identification model and policy optimizer work entirely in simulation enabling the wiggle and task rollout to be the only actions we perform in real.

Abstract:

Many robotic tasks are unforgiving; a single mistake in a dynamic throw can lead to unacceptable delays or unrecoverable failure. We introduce Wiggle and Go!, a two-stage framework for zero-shot rope manipulation: a brief, safe *wiggle* action is observed to predict descriptive rope parameters, which then conditions a trajectory optimizer for zero-shot goal-conditioned execution. Unlike prior dynamic rope manipulation methods that require large real-world datasets or iterative real-world refinement, our identification module is task-agnostic, supporting diverse manipulation policies without retraining. We achieve a 3.55 cm average accuracy on 3D target striking in real using rope system parameters in comparison to 15.29 cm for uninformed baselines, and over 50% success on multi-objective lobbing and draping tasks. Predicted parameters transfer to unseen motions with 0.95 Pearson correlation between simulated and real rope dynamics, indicating that the identification module generalizes across the task corpus. Project website: <https://wiggleandgo.github.io/>

Keywords: System-Identification, Deformable Object Manipulation, Model Learning, Sim-to-Real Transfer, State Estimation, Perception for Manipulation

1 Introduction

Before a sailor throws a mooring line or a climber flings a rope, they might instinctively give it a brief shake. This ‘wiggle’ is not just a habit; it is a sophisticated probe: the human nervous system reads the rope’s characteristics to calibrate the subsequent throw. Robots performing similar dynamic tasks are blind to these hidden dynamics, and when failures are unrecoverable (tangled, broken, or caught ropes) or costly, iterating over attempts is too risky.

Dynamic rope manipulation is hard precisely because rope behavior emerges from an interplay of stiffness, damping, inertia, and geometry that varies across ropes and cannot be read from a label. Iterative refinement methods Chi et al. [1] and Zhang et al. [2] achieve strong results but require 5-10 real-world trials *per goal*, a cost that is unacceptable when a single failed throw risks equipment

damage or an unrecoverable tangle. Large-dataset approaches [3] require tens of thousands of autonomous interactions and do not adapt to a new rope without re-collection. Parameter-conditioned methods, Kuroki et al. [4, 5], estimate material properties such as Young’s modulus from point-cloud observations, but these do not map one-to-one onto the behavioral parameters needed for physics-based trajectory optimization of dynamic 3D tasks.

We bridge this gap by replicating the sailor’s strategy: a single short, safe probing action decodes a rope’s physical identity before committing to a high-stakes dynamic maneuver. We propose Wiggle and Go! (WaG), a two-stage framework in which (a) a brief *wiggle* excites rope dynamics for task-agnostic system identification, and (b) the predicted parameters condition a zero-shot trajectory optimizer. Because identification is decoupled from execution, one wiggle supports diverse downstream tasks without retraining. Our primary contributions are: a task-agnostic system identification module, trained entirely in simulation, that infers nine behavioral rope parameters from a single planar wiggle and transfers to unseen motions at 0.95 Pearson correlation between simulated and real rope dynamics; a decoupled two-stage pipeline in which one wiggle observation supports zero-shot deployment across qualitatively different tasks (striking, lobbing, draping) without task-specific retraining or additional real-world trials; and a comprehensive real-world evaluation (with over 600 trials) across diverse ropes that achieves a four-fold accuracy improvement on 3D target striking over our baseline (3.55 cm vs. 15.29 cm) and over 50% success on lobbing and draping, with a parameter-importance ablation establishing that non-directly-measurable properties (stiffness, damping, mass distribution) drive ~ 10 cm of the improvement.

2 Related Works

Rope Dynamics Through Data and Simulation. Learning rope dynamics from large-scale data is constrained by the difficulty and safety of collecting diverse real-world demonstrations. Early work [3] required approximately 60,000 autonomous interactions; self-supervised methods [6] reduce this cost but still depend on extensive rollouts. Benchmark and imitation-learning approaches [7, 8, 9] scale via demonstrations rather than environment interaction, but neither is well-suited to dynamic settings where individual failures can damage equipment or cause safety hazards. Sim-to-real transfer offers an alternative by training primarily in simulation. Domain randomization methods [10, 11] train policies across varied simulation parameters to transfer policies to real deformable objects. Differentiable physics simulators [12, 13, 14, 15] enable gradient-based parameter identification and trajectory optimization. Trajectory-optimization approaches for deformable linear objects [16, 17, 18, 19] synthesize motions with explicit physics constraints, spanning quasi-static Cosserat-rod shape control to analytical high-speed models for specific dynamic tasks. Real-time cable-shaping MPC [20] similarly operates in quasi-static regimes due to re-planning cost. We show that identification from a wiggle complements these approaches by supplying better parameter estimates for sub-second dynamic motions where pre-execution planning suffices.

System Identification for Deformable Objects. System identification approaches explicitly estimate physical parameters to bridge simulation and reality. Real2Sim2Real (Lim et al. [21]) optimizes simulator parameters to match observed trajectories for dynamic casting, though requiring extensive physical experiments for each new cable. GenORM (Kuroki et al. [4]) and GenDOM (Kuroki et al. [5]) condition policies on Young’s modulus and Poisson’s ratio estimated from point cloud observations during predefined lifting motions, demonstrating one-shot manipulation on single-stage planar tasks. Wiggle and Go! differs in three key ways: (1) we use higher-dimensional, rope-specific parameters rather than material properties alone, Lim et al. [21] argues that Young’s modulus-based simulation can create discrepancies for ropes with leads, (2) we demonstrate transferability across multiple complex 3D manipulation tasks rather than single-stage goals, and (3) we perform an analysis of how observation strategy (wiggle design) affects parameter prediction quality.

Learning-based dynamics models [22, 23, 24] capture rope behavior through learned recurrent, graph, or local-linear representations; these predict implicit state-space rollouts rather than interpretable physical parameters, so they cannot be instantiated inside an analytic physics simulator for parameter-based trajectory optimization. Unlike privileged-learning frameworks such as Rapid Motor Adaptation [25, 26, 27] that infer hidden factors implicitly during execution, we estimate parameters explicitly and offline through a safe observation action, enabling task-agnostic identi-

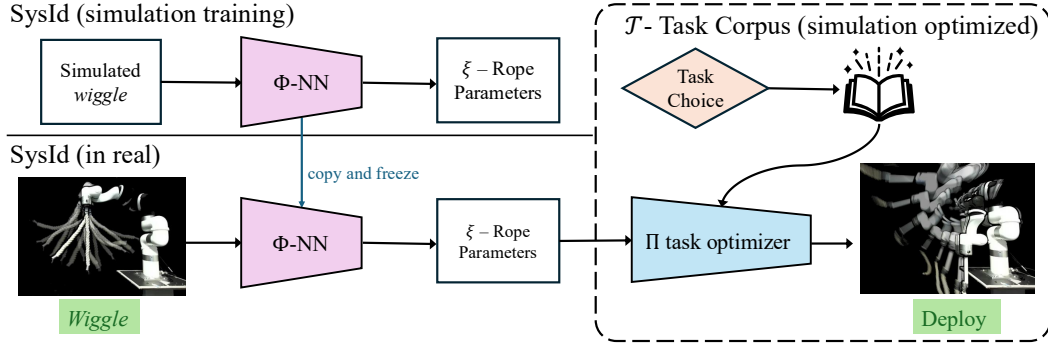


Figure 2: The WaG pipeline. Φ -NN is trained entirely in simulation; at deployment, a real-world wiggle feeds Φ -NN, whose predicted parameters inform a task-specific trajectory optimizer that is then executed on the robot. Items in green are the only real-world steps, executed once each.

cation. Concurrent work, Sobanbabu et al. [28], also uses CMA-ES-based sim-ID, but in legged locomotion and via iterative active exploration; we instead identify a rope from a single wiggle for one-shot, zero-shot deployment.

Iterative Learning Methods. Iterative learning control refines policies through repeated trials Bristow et al. [29]. For dynamic rope manipulation, Chi et al. [1], Zhang et al. [2] achieve strong results across diverse ropes and tasks through progressive refinement, typically requiring 5-10 trials per goal to achieve precise manipulation. Suresh and Atkeson [30] extend dynamic rope manipulation task-level ILC, learning a flying-knot skill with a critical-point objective on the rope-state trajectory. Our method builds towards addressing scenarios where repeated attempts pose risks of rope damage, equipment failure, or environmental hazards. Critically different, our system identification module is task-agnostic so the same wiggle observation can support multiple downstream manipulation tasks, enabling seamless switching between different goals without additional real-world trials.

3 Method

Wiggle and Go! has two stages: a safe *wiggle* excites rope dynamics for parameter estimation, and the predicted parameters condition a task-specific trajectory optimizer. Decoupling observation from execution lets one wiggle support multiple downstream tasks without retraining. Formally, WaG observes rope motion during the wiggle to obtain visual features $\mathcal{O}$. The System Identification (SysId) model $\Phi : \mathcal{O} \rightarrow \Xi$ predicts rope system parameters $\xi \in \Xi$ (e.g., stiffness, damping, mass distribution, link count). Given a task goal $g \in \mathcal{G}$ (e.g., 3D target position) and rope parameters ξ , an action policy $\Pi : \Xi \times \mathcal{G} \rightarrow \mathcal{A}$ generates a robot trajectory $\vec{a} \in \mathcal{A}$.

A task corpus $\mathcal{T}$ collects per-task policies (target striking, lobbing, draping). Because Φ is task-agnostic, every policy in $\mathcal{T}$ takes the same input form. Figure 2 illustrates the complete system architecture.

3.1 Action Policy Optimization Π

Each task in the corpus requires a policy $\Pi_{\text{task}} : \Xi \times \mathcal{G} \rightarrow \mathcal{A}$ that maps rope parameters and goals to executable trajectories. To avoid real-world trial and error, we optimize trajectories in simulation using predicted rope parameters $\hat{\xi}$ from the adaptation model (3.2).

We use Drake [31] for high-fidelity rope simulation of dynamic motions. We employ data-efficient, derivative-free optimization that can discover effective trajectories within a few hundred simulations per rope-task pair.

Task Setup. We choose three toy tasks to represent our task corpus: 3D point striking, lobbing, and draping shown in Figure 3. The lobbing task is motivated by controlled placement strategies in deformable object manipulation [32]. Lobbing and draping use multi-objective reward formulations defined in Appendix K. Our task setup (Figure 3) uses a UFactory xArm 7 robot manipulator [33].

For all tasks, we attach ropes to a 20 cm pole extender to increase the moment arm; the xArm’s joint velocity limits require this leverage for dynamic motions like whipping. The optimization objective minimizes 3D Euclidean distance between the rope tip and target.

CMA-ES Trajectory Optimization. We use Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [34], a stochastic optimization algorithm that adapts its search distribution

based on the success of sampled solutions. At each iteration, CMA-ES-traj (CMA-ES for trajectory optimization) samples a population of candidate trajectories from a multivariate Gaussian distribution, evaluates their fitness (task error in simulation), and updates the distribution’s mean and covariance matrix to concentrate search around promising regions. Each task uses a task-specific fitness combining a distance term with task-shaped auxiliaries: a proximity reward and motion regularizer for pole striking, an additional dwell term for lobbing that activates once the tip enters the target neighborhood, and an intermediate waypoint plus height penalty for draping. Joint-velocity and collision penalties are folded into the fitness directly. Formal definitions are given in Appendix K.

We parameterize trajectories using three sequential waypoints in the robot’s joint space and interpolate with a cubic spline to generate smooth, continuous joint trajectories within velocity limits. This low-dimensional representation (21D for a 7-DOF arm) is substantially smaller than the full per-step joint trajectory. For the more complex tasks, we limit exploration to planar actions by optimizing only the three joints that move in the plane of motion (9D parameters). Joint-space waypoints also avoid inverse kinematics singularities and naturally respect joint limits and velocity constraints. For each rope parameter set ξ and goal g , CMA-ES-traj runs for up to 25 iterations with 60 samples per iteration. The initial sampling distribution is unbiased for 3D pole-striking; for lobbing and draping, where the optimization landscape is more multi-modal, we bias the initial distribution toward a lifting motion as a warm start.

Simulation Setup and Implementation. We represent ropes in simulation as link/ball-joint chains, following Lim et al. [21], where the rope is modeled as a series of rigid spheres connected by ball joints with stiffness k and damping parameters c . Each link has a mass and radius for accurate inertial calculations. We train on parameters spanning the ranges in Table 2, chosen to match real ropes in household and industrial settings. We also, following [21], include lead weights at the rope tip to model common real-world scenarios where ropes are used to cast or manipulate attached objects. The lead weight also increases the effective reach by providing momentum to the rope tip during dynamic motions. The nine behavioral descriptors in Table 2 characterize the rope’s dynamic response inside our simulator and do not correspond one-to-one to measurable physical properties.

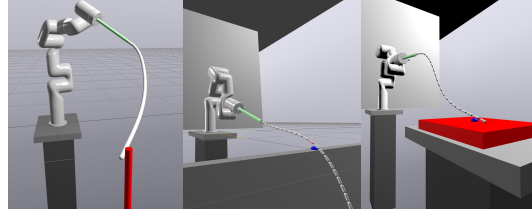


Figure 3: Drake tasks left to right: 3D striking, draping, lobbing. Green: extender; Red/Blue: targets

3.2 Adaptation Model Φ

Trajectory optimization (Section 3.1) requires the rope’s parameters. Even when physical properties are available, they do not map one-to-one onto the behavioral parameters of our ball-joint simulator, so we infer them from observed simulation behavior. The mapping from motion to parameters is non-linear; we frame parameter inference as supervised learning and train Φ -NN (Φ -neural network) on simulated ropes that span the ranges in Table 2, each with known ground-truth labels that are unavailable for physical ropes (e.g. stiffness, damping, and link count cannot be directly measured). At deployment, Φ -NN consumes a wiggle observation $\mathcal{O}$ and returns $\hat{\xi}$, which is then passed to the trajectory optimizer.

3.2.1 Observation Collection with Safe Action

We collect observations by executing a planar wiggle trajectory ω_{wiggle} that oscillates the rope with the robot end-effector. The wiggle is designed to excite key rope properties (stiffness, damping, inertia, etc.) through controlled acceleration while maintaining visibility for tracking. An ablation

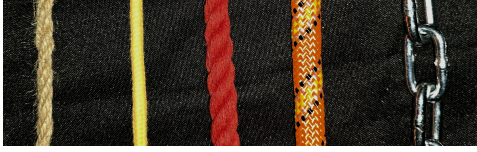


Figure 4: Test ropes (order: Brown, Yellow, Red, Orange, Chain).

Table 1: Material and physical characterization of test ropes.

Rope	Mat.	Stiff.	Damp.	Wt. (kg/m)	Rad. (mm)
Brown	Twine	Slack	Low	0.013	6
Yellow	Cotton	Stiff	Med	0.011	5
Red	Cotton	Slack	Low	0.033	9.5
Orange	Polyester	Stiff	High	0.082	12.7
Chain	Steel	Slack	Low	0.52	4.8×18

in the appendix (Sec. F) shows that the specific wiggle trajectory is not critical; any motion that sufficiently excites rope dynamics performs comparably.¹

In simulation, for a rope with N links, we extract 3D positions of the link centers $\{p_t^{(i)} \in \mathbb{R}^3\}_{i=1}^N$ at each timestep t . We project these to 2D image coordinates using a calibrated pinhole camera model. Since our wiggle is planar by design, depth information is largely redundant, the rope motion occurs primarily in the image plane and all links remain visible from a single fixed viewpoint. While stereo setups with non-planar motion could potentially improve inference of damping and stiffness, the single-camera approach balances simplicity with empirically validated performance (Section 4.3).

Feature Engineering. We normalize link positions at deployment relative to the first link and use only position and angle features, $\mathcal{O} = \{\tilde{p}_t, \theta_t\}$, omitting velocity and acceleration of the points to ensure minimal feature drift during real-world deployment. Details are in Appendix E.

3.2.2 Network Architecture and Sim-to-Real Training

Φ -NN is a temporal convolutional encoder followed by an MLP that maps observations to normalized parameters $\hat{\xi} \in [0, 1]^9$ (Table 2). We close the sim-to-real gap through domain randomization (calibration noise, anisotropic temporally-correlated tracking noise, and randomized trajectory padding) and curriculum masking that progressively occludes random and beginning-biased frame blocks to prevent overfitting to specific temporal windows. Full architecture, randomization, and training specifications are in Appendix J.

3.3 Real-World Deployment

Real-world deployment uses a ZED Mini 2i camera calibrated to the simulation’s pinhole model; Grounding SAM [35] segments the rope in the first frame, and CoTracker [36] tracks centerline keypoints across the wiggle. The tracked 2D keypoints are converted to the same normalized position-and-angle features used during training and passed to Φ -NN to obtain $\hat{\xi}$; CMA-ES-traj is then run offline in Drake with $\hat{\xi}$, and the resulting joint trajectory is executed open-loop on the xArm 7. No real-world feedback is used during task execution, so residual real-world error reflects sim-to-real gap and identification quality rather than online correction. Task error is measured by HSV-based endpoint tracking, with depth from the ZED for 3D distance. Full deployment details are in Appendix D.

4 Experiments and Results

We perform a series of experiments to evaluate the efficacy of our adaptation model, the transferability of rope system parameters from one dynamic context to another, and the full-pipeline accuracy of our system-identification-enabled zero-shot manipulation method. We perform our tests on a diverse range of ropes and a chain. The chain represents an out-of-distribution example of a *rope-behaving* object, though we note that ropes are more accurately represented as ball joints than chains which would be closer to an alternating hinge joint.

The five ropes are shown in Fig. 4 and characterized in Tab. 1. They span the qualitative axes of the training distribution: slack/stiff stiffness, low/medium/high damping, 0.011–0.082 kg/m, and 5–

¹We do not provide any theoretical guarantees on the safety of the wiggle action, but we use this terminology to suggest that a known, more controlled action is less likely to cause physical damage to surroundings and itself.

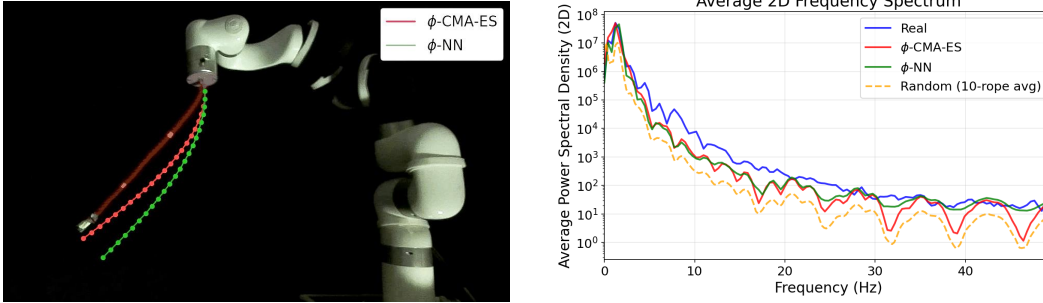


Figure 5: Real vs. simulation comparison for Φ -NN and Φ -CMA-ES on an unseen dynamic motion (Red rope, 45 cm, 10 g lead): point-wise trajectory overlay (left) and Fourier frequency distribution (right). See Appendix H for similar trends across ropes.

12.7 mm radius (from thin twine to stiff polyester). We evaluate the information conveyed by the wiggle, the simulation precision, the cross-motion transferability of predicted parameters, and the importance of identification for real-world rollout.

4.1 Adaptation Model Evaluation

We evaluate Φ -NN on three criteria: parameter estimation accuracy, robustness to distribution shift, and transferability to downstream tasks. All experiments use the model trained on 9000 simulated ropes with domain randomization enabled.

4.1.1 Parameter Estimation Accuracy and Robustness

On a test set of 1000 held-out simulated ropes, Φ -NN reaches sub-millimeter MAE on geometric parameters (rope length 6 mm, rope radius 2 mm, number of links 0.098). Inertial and damping parameters have larger absolute MAE (ball damping 0.010 N·s/m, ball stiffness 0.111 N/m, mass per unit length 0.007 kg/m), reflecting the difficulty of inferring these from a planar projected motion. Parameter Importance (Section 4.3) demonstrates that these absolute errors still leave the predictions useful for downstream trajectory optimization, and multicollinearity in the ball-joint model allows neighboring parameters to compensate for one another. We additionally find that the choice of wiggle is not critical: alternative planar wiggles of varied amplitude and frequency yield comparable parameter estimates (Appendix F), and the model gracefully clamps to its training bounds rather than extrapolating on out-of-distribution parameters (Appendix G).

Table 2: In-distribution parameter ranges and mean absolute error on 1000 test samples.

Parameter	Parameter Range			MAE
	Min	Max	Units	
Number of links	20	26	—	0.098
Rope length	0.45	0.65	m	0.006
Ball damping	0.001	0.05	N·s/m	0.010
Ball stiffness	0.05	1.0	N/m	0.111
Rope radius	0.003	0.015	m	0.002
Mass per unit len	0.02	0.12	kg/m	0.007
Lead mass	0.02	0.12	kg	0.005
Lead radius	0.015	0.045	m	0.008
Link extra scale	50	120	%	0.152
Overall				0.045

4.1.2 Transferability to Downstream Tasks

Parameter-wise accuracy alone is insufficient and predicted parameters must work together to enable task execution. We therefore validate transferability through motion fidelity (whether the rope with predicted parameters reproduces real rope dynamics under different motions) and task-level performance (whether they enable effective trajectory optimization).

Baseline: Φ -CMA-ES. To benchmark Φ -NN, we implement an optimization-based baseline that directly fits simulation parameters to a real wiggle trajectory: 60 samples per iteration $\times$ 50 CMA-ES iterations minimize pixel-wise distance between the simulated and real 2D rope trajectories. Without learned priors, Φ -CMA-ES intentionally overfits each rope to its wiggle, costing $\sim$ 3000 simulations per inference but offering robustness to out-of-distribution objects like chains where Φ -NN’s priors no longer apply.

Motion Fidelity. To test fidelity beyond static accuracy, for 10 real ropes (5 types $\times$ 10/30 g leads) we predict parameters from one wiggle, then compare a held-out motion in sim and real via Fourier and point-wise trajectory analysis (Fig. 5 and details in Appendix H). Both Φ -NN and Φ -CMA-ES achieve 0.95 average frequency correlation. Φ -NN achieves 5.4 cm/frame average error versus 5.8 cm/frame for Φ -CMA-ES, both substantially outperforming random parameters (13.6 cm/frame). The $\sim 2.5\times$ gap over random parameters indicates the wiggle-inferred parameters reproduce rope dynamics throughout an unseen trajectory. Φ -CMA-ES does better on the out-of-distribution chain (0.99 vs. 0.78 correlation), where per-rope fitting beats learned priors.

Task-Level Validation. We first validate in simulation that predicted parameters enable downstream trajectory optimization. For 100 random simulated ropes, we predict parameters with Φ -NN, optimize trajectories using those predictions (CMA-ES-traj, Section 3.1), and measure task error (Table 4). Trajectories optimized with predicted parameters achieve 2.1 cm median error versus 1.2 cm with ground truth and 12.8 cm with random in-distribution parameters. Predicted parameters provide substantial improvement over uninformed baselines while remaining within 0.9 cm of ground truth performance. This demonstrates that parameter estimation errors do not compound catastrophically, suggesting Φ -NN provides sufficient information for control, validating the two-stage pipeline design.

4.2 Action Policy Optimization Evaluation

To attribute residual real-world error to SysId quality rather than the optimizer, we need an optimizer that reliably finds near-optimal trajectories given ground-truth parameters. CMA-ES-traj satisfies this: across 100 ground-truth ropes (Appendix Tab. 4) it reaches 1.2 cm median and 0.25 cm minimum tip-to-target error. We pre-filter samples for joint-velocity feasibility so reported errors come from hardware-feasible trajectories. The 25–120 minute CPU cost per rope-task is acceptable when each rope is identified once and keeps *system identification quality* as the primary varying factor across our experiments. An RL policy conditioned on $\hat{\xi}$ is a viable alternative that could amortize this cost at deployment; it would, however, require additional reward, exploration, and architecture tuning and may converge less uniformly across the rope range than the per-task optimization used here. The higher convergence times were largely due to simulations with more rope-object contacts. Across tasks CMA-ES-traj converges consistently within budget (25 iterations for pole striking, 50 for lobbing, 15 for draping). On pole striking it reaches 0.8 cm average tip-to-target distance; with a simulation timestep of 0.01 s and the high speeds of these motions, the ground-truth instant of minimum distance is often missed by a small margin, so the true achieved accuracy is likely lower than reported.

4.3 Full Pipeline Results in Real

We test four in-domain ropes (Fig. 4) at 45/55/65 cm with 5 g lead and the 45 cm rope with 10/20/30 g leads; a metal chain serves to demonstrate capabilities of out-of-distribution generalization. The benchmark Φ -Random executes trajectories optimized for 20 unrelated in-distribution ropes, simulating an educated but unmatched action (omitted on the chain for safety). Φ -NN outperforms Φ -CMA-ES on in-distribution ropes (3.55 cm vs. 4.16 cm average), and Φ -CMA-ES wins on the chain by overfitting to its dynamics.

Parameter Importance: Do Other Parameters Actually Move the Outcome? A reasonable concern is that the two directly-measurable parameters (length, lead mass) dominate the benefit and the other seven are cosmetic. Replacing subsets of Φ -NN’s predictions with random in-distribution values (Appendix Tab. 5) shows otherwise: full identification reaches 0.9 cm mean error versus 14.2 cm for fully random, but retaining only length and lead-mass closes just 3.3 cm of that 13.3 cm gap: the remaining seven parameters (stiffness, damping, mass distribution) drive ~ 10 cm. The real-world data in Table 3 agrees: Φ -NN reduces mean error over Φ -Random by $3.3\text{--}13.1\times$, and ropes sharing length and lead but differing in stiffness/damping span a 7.5 cm range in Φ -Random error.

Secondary Tasks: Lobbing and Draping. Each multi-objective task uses two success metrics (Appendix Tab. 6). **Lobbing:** a full success requires the rope tip to knock down a domino *and* remain on the board (a gentle, controlled placement); the domino is positioned closer to the robot than the strike target to require overshoot. **Draping:** a success requires the tip to clear the wall and

Table 3: Full pipeline results after rope wiggle, parameter prediction and CMA-ES-traj execution. Average distance (cm) from target from 600 task-policy rollouts (5 per rope-config-method). Each (length, lead) configuration shows three methods side-by-side: **NN** = Φ -NN, **CMA** = Φ -CMA-ES, **Rnd** = Φ -Random. Best in-domain method per (rope, config) in **bold**.

Rope	45cm, 5g			45cm, 10g			45cm, 20g			45cm, 30g			55cm, 5g			65cm, 5g			Average		
	NN	CMA	Rnd	NN	CMA	Rnd	NN	CMA	Rnd	NN	CMA	Rnd	NN	CMA	Rnd	NN	CMA	Rnd	NN	CMA	Rnd
Brown	0.1	0.0	15.9	2.0	0.6	17.3	5.3	8.9	18.3	0.8	5.2	17.7	0.0	0.2	15.5	2.1	4.4	22.1	1.7	3.2	17.8
Yellow	8.1	1.5	14.7	6.3	3.8	15.8	0.8	0.8	16.5	4.7	3.2	17.0	3.5	7.7	15.0	0.0	8.1	19.2	3.9	4.2	16.4
Red	0.0	0.0	17.6	16.0	3.2	15.2	0.9	2.3	16.9	1.4	7.2	16.6	0.3	1.4	14.1	4.3	0.0	18.2	3.8	2.4	16.4
Orange	10.1	0.0	10.9	2.2	8.7	10.0	3.7	1.5	11.8	0.0	0.0	13.5	11.4	12.0	8.0	1.3	19.1	9.1	4.8	6.9	10.6
Chain*	51.4	3.9	—	29.6	0.6	—	12.3	2.6	—	3.3	1.6	—	51.9	4.5	—	0.7	6.7	—	24.9	3.3	—
Avg In-Domain	4.6	0.4	14.8	6.6	4.0	14.6	2.7	3.4	15.9	1.7	3.9	16.2	3.8	5.3	13.1	1.9	7.9	17.2	3.5	4.2	15.3

* out-of-domain; Φ -Random not run on chain for safety.

hang on the far side; secondary success uses the rope midpoint distance to the wall top, matching the CMA-ES-traj reward. Because Φ is shared across the task corpus, adding a new task requires only a representative simulation environment and a reward function, no retraining. Across the four in-domain ropes, Φ -NN achieves a 54% lobbing target-hit rate (21% full success including the stay condition) and a 63% draping success rate at 4.6 cm mean midpoint distance; Φ -CMA-ES is comparable (63%/38% lobbing, 67%/4.1 cm draping). The shared identification module thus supports qualitatively different downstream tasks (fast strikes, controlled placement, over-the-wall drapes) without any task-specific retraining.

5 Limitations

CMA-ES-traj under Drake is CPU-bound, capping per-task throughput at 25 to 120 minutes depending on task and environment complexity (Sec. 4.2); as discussed in Sec. 3.1, Φ -NN’s task-agnostic output is theoretically compatible with any future GPU-accelerated or gradient-based optimizer. An RL policy conditioned on ξ could amortize this per-task cost into a single training phase and run in milliseconds at deployment, and is an exciting future direction. Φ -NN’s predictions saturate at the bounds of its training distribution (Appendix G), which limits generalization to objects far outside the rope regime (like the chain).

Segmentation and tracking introduce noticeable noise in the real-to-sim transfer, and our current simulation has limited fidelity for behaviors like rope kinks.

6 Conclusion

This paper introduces a novel, task-agnostic system identification method for rope manipulation. Through the use of a single safe action and the leveraging of simulation priors with a neural network model we can fit descriptive system parameters to a rope in real. On 3D pole-striking this yields roughly a four-fold improvement in real-world accuracy over non-system-identified baselines, and the same identification module extends without modification to lobbing and draping. Targeted wiggles that excite individual parameters in isolation is a promising direction for future work.

Acknowledgments

This work used computational resources at Pittsburgh Supercomputing Center through allocation CIS251423 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, which is supported by U.S. National Science Foundation grants #2138259, #2138286, #2138307, #2137603, and #2138296. [37]

We also thank Professor Chris Atkeson for his advice on the paper and Alan Wang for his early work on this project.

References

- [1] C. Chi, B. Burchfiel, E. Cousineau, S. Feng, and S. Song. Iterative Residual Policy: for Goal-Conditioned Dynamic Manipulation of Deformable Objects, Apr. 2022. arXiv:2203.00663.
- [2] H. Zhang, J. Ichnowski, D. Seita, J. Wang, H. Huang, and K. Goldberg. Robots of the Lost Arc: Self-Supervised Learning to Dynamically Manipulate Fixed-Endpoint Cables, May 2020. arXiv:2011.04840.
- [3] A. Nair, D. Chen, P. Agrawal, P. Isola, P. Abbeel, J. Malik, and S. Levine. Combining self-supervised learning and imitation for vision-based rope manipulation, 2017. URL <https://arxiv.org/abs/1703.02018>.
- [4] S. Kuroki, J. Guo, T. Matsushima, T. Okubo, M. Kobayashi, Y. Ikeda, R. Takanami, P. Yoo, Y. Matsuo, and Y. Iwasawa. GenORM: Generalizable One-shot Rope Manipulation with Parameter-Aware Policy, Jan. 2023. arXiv:2306.09872.
- [5] S. Kuroki, J. Guo, T. Matsushima, T. Okubo, M. Kobayashi, Y. Ikeda, R. Takanami, P. Yoo, Y. Matsuo, and Y. Iwasawa. GenDOM: Generalizable One-shot Deformable Object Manipulation with Parameter-Aware Policy, Jan. 2024. arXiv:2309.09051.
- [6] J. Wang, H. Huang, V. Lim, H. Zhang, J. Ichnowski, D. Seita, Y. Chen, and K. Goldberg. Self-supervised learning of dynamic planar manipulation of free-end cables, 2024. URL <https://arxiv.org/abs/2405.09581>.
- [7] X. Lin, Y. Wang, J. Olkin, and D. Held. Softgym: Benchmarking deep reinforcement learning for deformable object manipulation. In *Conference on Robot Learning*, pages 432–448. PMLR, 2021.
- [8] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.
- [9] J. Luo, C. Xu, X. Geng, G. Feng, K. Fang, L. Tan, S. Schaal, and S. Levine. Multistage cable routing through hierarchical imitation learning. *IEEE Transactions on Robotics*, 40: 1476–1491, 2024.
- [10] J. Matas, S. James, and A. J. Davison. Sim-to-real reinforcement learning for deformable object manipulation, 2018. URL <https://arxiv.org/abs/1806.07851>.
- [11] G. Tiboni, K. Arndt, and V. Kyrki. DROPO: Sim-to-real transfer with offline domain randomization. *arXiv preprint arXiv:2203.00352*, 2022.
- [12] Y. Hu, J. Liu, A. Spielberg, J. B. Tenenbaum, W. T. Freeman, J. Wu, D. Rus, and W. Matusik. Chainqueen: A real-time differentiable physical simulator for soft robotics. In *2019 International conference on robotics and automation (ICRA)*, pages 6265–6271. IEEE, 2019.
- [13] J. K. Murthy, M. Macklin, F. Golemo, V. Voleti, L. Petrini, M. Weiss, B. Considine, J. Parent-Lévesque, K. Xie, K. Erleben, et al. gradsim: Differentiable simulation for system identification and visuomotor control. In *International conference on learning representations*, 2021.

- [14] Y. Chen, Y. Zhang, Z. Brei, T. Zhang, Y. Chen, J. Wu, and R. Vasudevan. Differentiable discrete elastic rods for real-time modeling of deformable linear objects. *arXiv preprint arXiv:2406.05931*, 2024.
- [15] Z. Si, G. Zhang, Q. Ben, B. Romero, Z. Xian, C. Liu, and C. Gan. Diff tactile: A physics-based differentiable tactile simulator for contact-rich robotic manipulation. *arXiv preprint arXiv:2403.08716*, 2024.
- [16] A. Jnadi, H. Salloum, Y. Kholodov, A. Gasnikov, and K. Almaghout. Scope: Smooth convex optimization for planned evolution of deformable linear objects. *arXiv preprint arXiv:2601.19742*, 2026.
- [17] A. Artinian, Q. Huet, F. Benamar, and V. Perdereau. Optimal cosserat-based deformation control for robotic manipulation of linear objects. In *2023 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 381–388. IEEE, 2023.
- [18] Y. Yamakawa, A. Namiki, and M. Ishikawa. Simple model and deformation control of a flexible rope using constant, high-speed motion of a robot arm. In *2012 IEEE International Conference on Robotics and Automation*, pages 2249–2254, 2012. doi:10.1109/ICRA.2012.6224763.
- [19] Y. Yamakawa, A. Namiki, and M. Ishikawa. Dynamic high-speed knotting of a rope by a manipulator. *International Journal of Advanced Robotic Systems*, 10(10):361, 2013. doi:10.5772/56783. URL <https://doi.org/10.5772/56783>.
- [20] R. Viljoen, S. Mamedov, J. Ubbink, W. Decré, E. Aertbeliën, J. Swevers, and J. De Schutter. Cable shaping with mpc: A reliable, fast, and real-time feasible formulation. *IFAC-PapersOnLine*, 58(25):156–161, 2024.
- [21] V. Lim, H. Huang, L. Y. Chen, J. Wang, J. Ichnowski, D. Seita, M. Laskey, and K. Goldberg. Planar Robot Casting with Real2Sim2Real Self-Supervised Learning, June 2022. arXiv:2111.04814.
- [22] Y. Yang, J. A. Stork, and T. Stoyanov. Learning differentiable dynamics models for shape control of deformable linear objects. *Robotics and Autonomous Systems*, 158:104258, 2022.
- [23] C. Wang, Y. Zhang, X. Zhang, Z. Wu, X. Zhu, S. Jin, T. Tang, and M. Tomizuka. Offline-online learning of deformation model for cable manipulation with graph neural networks. *IEEE Robotics and Automation Letters*, 7(2):5544–5551, 2022.
- [24] M. Yu, H. Zhong, and X. Li. Shape control of deformable linear objects with offline and online learning of local linear deformation models, 2022. URL <https://arxiv.org/abs/2109.11091>.
- [25] A. Kumar, Z. Fu, D. Pathak, and J. Malik. RMA: Rapid Motor Adaptation for Legged Robots, July 2021. arXiv:2107.04034.
- [26] H. Qi, A. Kumar, R. Calandra, Y. Ma, and J. Malik. In-Hand Object Rotation via Rapid Motor Adaptation, Oct. 2022. arXiv:2210.04887.
- [27] Y. Liang, K. Ellis, and J. Henriques. Rapid Motor Adaptation for Robotic Manipulator Arms, Mar. 2024. arXiv:2312.04670.
- [28] N. Sobanbabu, G. He, T. He, Y. Yang, and G. Shi. Sampling-based system identification with active exploration for legged robot sim2real learning. In *Proceedings of the 9th Conference on Robot Learning (CoRL)*, volume 305 of *Proceedings of Machine Learning Research*, pages 578–598. PMLR, 2025. arXiv:2505.14266.
- [29] D. Bristow, M. Tharayil, and A. Alleyne. A survey of iterative learning control. *IEEE Control Systems Magazine*, 26(3):96–114, 2006. doi:10.1109/MCS.2006.1636313.

- [30] K. Suresh and C. G. Atkeson. Learning dynamic rope manipulation using task-level iterative learning control. In *Robotics: Science and Systems (RSS)*, 2026. arXiv:2602.21302.
- [31] R. Tedrake and the Drake Development Team. Drake: Model-based design and verification for robotics, 2019. URL <https://drake.mit.edu>.
- [32] L. Y. Chen, H. Huang, E. Novoseller, D. Seita, J. Ichnowski, M. Laskey, R. Cheng, T. Kollar, and K. Goldberg. Efficiently learning single-arm fling motions to smooth garments. In *The International Symposium of Robotics Research*, pages 36–51. Springer, 2022.
- [33] UFACTORY. Ufactory xarm 7 - 7 dof collaborative robot, 2022. URL <https://www.ufactory.cc/products/xarm-7-2020>. Accessed: 2024-01-26.
- [34] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001. doi:10.1162/106365601750190398.
- [35] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, et al. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024.
- [36] N. Karaev, I. Makarov, J. Wang, N. Neverova, A. Vedaldi, and C. Rupprecht. Co-tracker3: Simpler and better point tracking by pseudo-labelling real videos. *arXiv preprint arXiv:2410.11831*, 2024.
- [37] T. J. Boerner, S. Deems, T. R. Furlani, S. L. Knuth, and J. Towns. Access: Advancing innovation: Nsf’s advanced cyberinfrastructure coordination ecosystem: Services & support. In *Practice and Experience in Advanced Research Computing 2023: Computing for the Common Good*, PEARC ’23, pages 173–176, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9781450399852. doi:10.1145/3569951.3597559. URL <https://doi.org/10.1145/3569951.3597559>.

Appendix

A Action Policy Optimization: Simulation Pipeline Results

To benchmark CMA-ES-traj’s performance with ground-truth, predicted, and random parameters, we evaluate the full pipeline on 100 simulated ropes (Table 4). Trajectories optimized with ground-truth parameters reach 1.4 cm mean error (1.2 cm median); predicted parameters reach 5.7 cm mean (2.1 cm median); random in-distribution parameters reach 16.3 cm mean (12.8 cm median). This confirms that CMA-ES-traj reliably finds near-optimal trajectories given accurate parameters, isolating SysId quality as the dominant factor in real-world residual error.

Table 4: Average of 100 adaptation-predicted ropes on full pipeline in simulation. Comparison of ground truth, predicted and 100 random in-distribution ropes. Here, CMA-ES-traj was optimized until 25 iterations *or* min-distance of <1.5 cm.

	Median	Min	Max	Mean
Ground Truth	1.2	0.25	21.6	1.4 $\pm$ 2.2
Predicted	2.1	0.27	70.2	5.7 $\pm$ 11.5
100 random ropes	12.8	0.14	104.3	16.3 $\pm$ 12.1

B Parameter Importance Ablation Details

This appendix expands the parameter-importance analysis referenced in Section 4.3. Table 5 reports mean 3D tip-to-target error in simulation for five identification conditions: full Φ -NN; three partial-identification settings where only a subset of Φ -NN’s predictions are kept and the rest are sampled at random from the training distribution (length and lead mass together, length only, lead mass only); and fully random parameters. Each row averages 10 random parameter draws per configuration over the 30 rope configurations used in our real-world evaluation. The full pipeline reaches 0.9 cm mean error; supplying only the easily-measured length and lead-mass predictions closes only 3.3 cm of the 13.3 cm gap to fully random parameters, leaving 10.9 cm of residual error driven by the remaining seven (stiffness, damping, mass-distribution, link-count) parameters that the wiggle identifies.

Table 5: Parameter importance ablation. Each row replaces a subset of Φ -NN’s nine predictions with random in-distribution values, keeping Φ -NN’s predictions only for the parameters listed. Evaluated in simulation over 10 random parameter sets per each of 30 rope configurations from our real-world evaluation.

Condition (simulation)	Mean (cm)	Std (cm)
Φ -NN (full pipeline)	0.9	0.4
Random + Φ -NN length & lead mass	10.9	10.1
Random + Φ -NN length only	13.3	13.5
Random + Φ -NN lead mass only	12.3	10.5
Fully random in-distribution	14.2	12.5

C Secondary Task Results: Lobbing and Draping

This appendix expands the secondary-task results referenced in Section 4.3. Table 6 reports per-rope success rates and distances on the lobbing and draping tasks, averaged over 6 configurations per rope (45/55/65 cm lengths with 5 g lead, plus 45 cm with 10/20/30 g leads). Across the four in-domain ropes, the in-domain averages are: lobbing target-hit 54% (Φ -NN) vs. 63% (Φ -CMA-ES); lobbing stay 21% vs. 38%; draping success 63% vs. 67%; and draping midpoint distance 4.6 cm vs. 4.1 cm. Both methods exceed 50% success on draping for all four ropes and on lobbing target-hit for three of four—the Orange rope (heavy, stiff polyester) is the hardest, with both methods at 17% lobbing target-hit and 50% draping success. The complete per-configuration results, including which configurations achieved both lobbing sub-goals and the per-configuration draping midpoint distances, are in Table 7.

Table 6: Performance across lobbing and draping tasks. Each metric shows both methods side-by-side: **NN** = Φ -NN, **CMA** = Φ -CMA-ES. Best per (rope, metric) in **bold**. Arrows: $\uparrow$ = higher better, $\downarrow$ = lower better. Averaged over 6 configurations per rope; full per-configuration results in Tab. 7.

Rope	Lobbing				Draping			
	Target $\uparrow$		Stay $\uparrow$		Success $\uparrow$		Distance (cm) $\downarrow$	
	NN	CMA	NN	CMA	NN	CMA	NN	CMA
Brown	67%	83%	50%	33%	67%	83%	4.1	3.8
Yellow	67%	67%	17%	50%	67%	67%	7.4	4.1
Red	67%	83%	17%	50%	67%	67%	3.6	4.8
Orange	17%	17%	0%	17%	50%	50%	3.4	3.5
Avg In-Domain	54%	63%	21%	38%	63%	67%	4.6	4.1

D Real-World Deployment Pipeline

Our complete system is designed for real-world deployment on physical robots for ropes with unknown quantitative parameters. The deployment pipeline is as follows: (1) execute wiggle and extract visual observations, (2) have Φ -NN estimate rope parameters $\hat{\xi}$, (3) optimize trajectory using $\hat{\xi}$ in simulation, (4) execute trajectory on the robot, and (5) measure task performance. The key challenges are obtaining visual observations compatible with Φ ’s training data and accurately measuring task error in metric space.

Wiggle Observation. A ZED Mini 2i camera is positioned to match the simulation’s pinhole calibration. We segment the first frame with Grounding SAM [35] and color refinement, fit a spline to the rope mask, and resample keypoints along the centerline at uniform arc-length. CoTracker [36] tracks these keypoints across the wiggle, after which we convert to the same normalized angular features used in training. For real-world deployment, we use only position and angle features ($\tilde{p}_t, \theta_t$) to avoid noise amplification from temporal derivatives. Domain randomization during Φ ’s training (calibration noise, tracking noise) ensures robustness to real-world imperfections.

Trajectory Execution and Evaluation. After Φ -NN predicts parameters ($\hat{\xi}$), we optimize a trajectory in Drake with CMA-ES as explained in Sec. 3.1 and execute it on the xArm 7. To measure task error, we track the rope endpoint and target with HSV-based color filtering. The ZED camera’s depth estimates convert 2D detections to 3D metric coordinates, yielding the minimum distance between rope endpoint and target.

E Feature Engineering Details

We normalize positions relative to the first link: $\tilde{p}_t^{(i)} = p_t^{(i)} - p_t^{(1)}$. This provides two benefits: (1) the first link is attached to the robot end-effector (effectively a fixed hinge), conveying minimal rope-specific information, and (2) normalization provides robustness to calibration shifts between simulation and deployment. We then compute angular features from normalized positions, including angles relative to the first link and their temporal derivatives (angular velocity and acceleration) using centered finite differences with Gaussian smoothing. While derivative features improve performance in simulation, they amplify noise in real-world due to tracking errors. Therefore, for simulation training we use only position and angle features: $\mathcal{O} = \{\tilde{p}_t, \theta_t\}$ to ensure minimal feature drift during real world deployment.

F Wiggle Ablation Study

We validate that the specific wiggle trajectory used for system identification is not critical by training 8 additional models with different planar wiggles: 6 systematically varying amplitude ($20^\circ, 30^\circ$) and frequency (0.5–1.0 Hz), and 2 random predefined trajectories (Table 8). The main wiggle achieves best overall performance, with Ablation 1 ($20^\circ, 0.5$ Hz) close behind. Performance remains consistent across planar wiggles that sufficiently excite rope dynamics, and no single wiggle dominates all

Table 7: Complete experimental results across all rope configurations and parameter prediction methods. Results were repeatable, so one trial was performed for each task-rope combination.

Rope	Method	Lobbing								Draping																	
		45cm 5g		45cm 10g		45cm 20g		45cm 30g		55cm 5g		65cm 5g		45cm 5g		45cm 10g		45cm 20		45cm 30g		55cm 5g		65cm 5g			
		T	S	T	S	T	S	T	S	T	S	T	S	S	D	S	D	S	D	S	D	S	D	S	D	S	D
Brown	Φ-NN	✓	—	✓	✓	✓	✓	—	—	—	—	✓	✓	✓	3.5	✓	5.0	✓	5.0	✓	3.0	—	—	—	—	—	—
	Φ-CMA-ES	✓	—	✓	—	—	—	✓	—	✓	✓	✓	✓	✓	4.5	✓	4.5	✓	4.5	✓	5.5	✓	0.0	—	—	—	—
Yellow	Φ-NN	—	—	✓	—	✓	—	✓	—	—	—	✓	✓	✓	10.0	✓	8.5	✓	11.0	—	11.0	✓	0.0	—	—	—	—
	Φ-CMA-ES	✓	✓	—	—	—	—	✓	—	✓	✓	✓	✓	✓	3.5	✓	4.0	✓	3.5	—	6.0	✓	5.5	—	—	—	—
Red	Φ-NN	—	—	✓	—	✓	—	✓	✓	—	—	✓	4.5	✓	4.5	✓	3.5	✓	1.8	—	—	—	—	—	—	—	—
	Φ-CMA-ES	✓	✓	✓	—	—	—	✓	—	✓	✓	✓	✓	✓	9.5	✓	4.5	✓	3.0	✓	2.3	—	—	—	—	—	—
Orange	Φ-NN	—	—	—	—	—	—	—	—	—	—	✓	—	✓	3.9	✓	4.8	—	—	—	✓	1.5	—	—	—	—	—
	Φ-CMA-ES	—	—	—	—	—	—	—	—	—	—	✓	✓	✓	4.5	✓	5.4	—	—	—	✓	0.5	—	—	—	—	—

T = Target hit, S = Stayed, D = Distance (cm). ✓ = success, — = failure or not measured.

parameters. Random trajectories (Ablations 7–8) show degraded performance on complex parameters. This validates our design principle: planar wiggles with adequate excitation enable accurate parameter inference regardless of specific trajectory. We hypothesize that stereo setups with depth cameras and non-planar motion could further improve inference of certain parameters like damping and stiffness. Per-parameter gradient sensitivity maps (Appendix Tab. 10) further suggest that different parameters are informed by different timesteps of the wiggle: inertial and damping parameters (ball stiffness, ball damping, mass per unit length) peak at later trajectory inflection points where dynamically similar ropes diverge, while geometric parameters are identifiable earlier. Sensitivity diminishes toward the sequence end, indicating that wiggle duration is sufficient but not over-utilized.

Table 8: Wiggle Study: Mean Absolute Error across wiggle variants (1000 test samples). Ablations systematically vary wiggle parameters: Abl 1-3 use 20° amplitude at 0.5, 0.75, 1.0 Hz; Abl 4-6 use 30° amplitude at 0.5, 0.75, 1.0 Hz; Abl 7-8 use random predefined trajectories. Main is the primary wiggle (joint 6 oscillation of the xArm) used throughout all experiments. Color gradient indicates relative performance per parameter (green = best, red = worst); **bold** marks the best wiggle for each parameter (ties bolded).

Parameter	Abl 1 20°, 0.5Hz	Abl 2 20°, 0.75Hz	Abl 3 20°, 1.0Hz	Abl 4 30°, 0.5Hz	Abl 5 30°, 0.75Hz	Abl 6 30°, 1.0Hz	Abl 7 Random 1	Abl 8 Random 2	Main Joint 6
Number of links	0.122	0.127	0.225	0.205	0.206	0.360	2.098	1.453	0.098
Rope length (m)	0.007	0.013	0.017	0.015	0.018	0.022	0.057	0.028	0.006
Ball damping (N-s/m)	0.010	0.033	0.038	0.018	0.031	0.096	0.036	0.040	0.010
Rope radius (m)	0.002	0.003	0.003	0.003	0.004	0.006	0.003	0.004	0.002
Mass per unit length (kg/m)	0.006	0.010	0.010	0.013	0.012	0.014	0.019	0.014	0.007
Link extra scale	0.153	0.160	0.160	0.179	0.169	0.241	0.205	0.208	0.152
Lead mass (kg)	0.005	0.011	0.014	0.008	0.011	0.018	0.021	0.016	0.005
Lead radius (m)	0.008	0.008	0.008	0.008	0.008	0.008	0.007	0.008	0.008
Ball stiffness (N/m)	0.118	0.257	0.258	0.224	0.244	0.336	0.388	0.322	0.111

G Out-of-Distribution Robustness

We test Φ -NN’s generalization by extending four parameters (ball damping, mass per unit length, lead mass, ball stiffness) beyond training bounds. The model clamps predictions to training-distribution bounds rather than extrapolating: ball damping predictions saturate at [0.016, 0.089] when targets span [0.100, 1.499]. This is a fundamental limit of supervised models trained on bounded distributions.

H Transferability

Appendix Table 9 provides the per-rope motion-fidelity results referenced in Section 4.3 of the main paper. We execute a higher-jerk motion in real and compare the rope’s behavior to simulations parameterized by Φ -NN, Φ -CMA-ES, and a random in-distribution baseline. Across 10 in-distribution

rope/lead pairs, Φ -NN and Φ -CMA-ES both achieve 0.95 average Fourier frequency correlation, versus 0.81 for random parameters. Per-frame point distance follows the same pattern: 5.4 cm/frame (Φ -NN), 5.8 cm/frame (Φ -CMA-ES), and 13.6 cm/frame (random). The chain (out-of-distribution) is an exception: Φ -CMA-ES’s per-rope overfitting outperforms Φ -NN’s learned priors there.

Table 9: Estimation of parameters, material, and physical description of test ropes on different Φ methods.

Rope	Lead (g)	Frequency correlation			Average Point Distance**		
		NN	CMAES	Random*	NN	CMAES	Random*
Brown	10	0.98	0.94	0.76	4.8	6.9	13.4
	30	0.97	0.91	0.85	4.7	8.4	14.7
Yellow	10	0.98	0.96	0.80	4.0	5.6	13.7
	30	0.97	0.94	0.77	4.9	7.4	15.2
Red	10	0.97	0.95	0.81	7.7	5.8	14.2
	30	0.97	0.93	0.78	6.3	8.3	15.9
Orange	10	0.97	0.89	0.91	4.5	4.3	12.4
	30	0.99	0.99	0.90	3.2	2.8	10.7
Chain***	10	0.77	0.99	0.76	8.4	3.2	14.2
	30	0.98	0.99	0.78	6.0	5.2	12.8
Average		0.95	0.95	0.81	5.4	5.8	13.6
Std. Dev.		0.06	0.03	0.05	1.7	2.0	1.6

* For Φ -Random we randomly sample 10 (simulation stable) ropes as a control. Distance is mean trajectory error in the image plane (ZED Mini camera model, depth 0.65 m), in centimeters.

** cm per frame

*** out-of-distribution

I Φ -NN Activations Analysis

We perform a case study of Φ -NN activations by computing gradient-based sensitivity $\frac{\partial p_i}{\partial \mathbf{x}_t}$ for each predicted parameter p_i with respect to input joint positions $\mathbf{x}_t$ and angles θ_t at each timestep, revealing which spatiotemporal regions of the wiggle most influence each parameter’s prediction (Appendix Table 10). Ball stiffness, ball damping, and mass parameters exhibit their highest sensitivity later in the wiggle, with peaks at apparent trajectory inflection points where physical differences emerge most clearly. Sensitivity diminishes toward the sequence end, suggesting that while extended duration is needed to capture temporal dynamics, yet additional wiggle time may provide limited new information. We hypothesize that multi-parameter relationships, such as stiff, light-lead ropes initially wiggling similarly to non-stiff, heavy-lead ropes before the latter gain momentum, contribute to these later activations.

J Implementation Details

J.1 Network Architecture

Φ -NN uses a temporal convolutional encoder followed by a multi-layer perceptron. The encoder consists of three 1D convolutional blocks applied along the temporal dimension. Each block uses kernel size 8, stride 1, followed by layer normalization, GELU activation, average pooling (stride 2), and dropout (rate 0.3). After the convolutional blocks, adaptive pooling produces a fixed-length representation, which is passed through a linear layer to produce a 256-dimensional embedding. The MLP head uses hidden dimensions [128, 64] to map the embedding to 9 normalized rope parameters $\hat{\xi} \in [0, 1]^9$. The complete model has 1.2M parameters.

J.2 Feature Engineering

Angular velocity is computed using unwrapped finite differences with Gaussian smoothing ($\sigma = 1.0$, kernel size 5). Angular acceleration uses the same approach with $\sigma = 1.5$ and kernel size 7. We record rope motion for 400 frames at 60 FPS (~ 6.7 seconds).

J.3 Domain Randomization

Calibration noise: Gaussian noise ($\sigma = 2$ cm) applied to camera position and lookat point. Tracking noise: Anisotropic noise spanning 0-3 pixels with temporal correlation coefficient $\alpha = 0.8$, with higher variance in the longitudinal direction than lateral to mimic realistic tracking errors. Trajectory padding: Randomly add 0-20 frames at the start of each trajectory to simulate recording delays.

J.4 Curriculum Masking Schedule

Masking begins at epoch 50 using 50-frame contiguous blocks. From epochs 50-200, we mask 1-2 random blocks per trajectory. From epochs 200-400, we apply beginning-biased masking where blocks near the trajectory start are preferentially masked. From epochs 400-500, we increase to 7 masked blocks per trajectory. All masked frames are set to zero.

J.5 Training Configuration

Dataset: 9000 training ropes and 1000 validation ropes with parameters sampled using Latin Hypercube Sampling (LHS) from the ranges in Table II (Main Paper). Loss: Mean squared error on normalized parameters. Optimizer: Adam with initial learning rate 10^{-3} , cosine annealing schedule, and 5-epoch warmup. Batch size: 32. Training duration: 500 epochs.

J.6 Trajectory Parameterization and Velocity Filtering

Trajectories are parameterized by three sequential joint-space waypoints, interpolated by a cubic spline to produce smooth continuous joint trajectories. For the full 7-DOF arm this gives 21 parameters; for the planar lobbing and draping tasks we restrict to the three in-plane joints (9 parameters). At sample time, each candidate’s spline-implied joint velocity profile is evaluated against the xArm 7 joint-velocity limits, and samples that violate any joint’s limit are rejected before fitness evaluation, so all evaluated trajectories are hardware-feasible. CMA-ES-traj runs up to 25 iterations of 60 samples each for pole striking, 50 iterations for lobbing, and 15 for draping. For pole striking the initial sampling distribution is unbiased; for lobbing and draping it is biased toward a lifting motion as a warm start.

K CMA-ES Fitness Functions

We describe the CMA-ES fitness $f(\tau)$ minimized for each task. $p_{\text{tip}}(t)$ is the rope tip position at simulation time t along trajectory τ , and p_{tgt} the task target. Numeric scale factors are fixed hyperparameters tuned once and omitted from the expressions below for clarity. Candidate trajectories are pre-filtered at sample time to satisfy joint-velocity limits. Hardware feasibility is also driven into the reward through soft penalties on joint-velocity-limit violations and on collisions, both added to the fitness as positive terms.

K.1 Shared Quantities

$$d_{\min} = \min_t \|p_{\text{tip}}(t) - p_{\text{tgt}}\|_2 \quad (1)$$

$$R_{\text{prox}} = \frac{1}{1+d_{\min}} \quad (2)$$

$M(\tau)$ regularizes total joint motion. $P_{\text{vel}}(\tau) \geq 0$ accumulates the magnitude of joint-velocity-limit violations along τ and vanishes when all joints stay within limits. $P_{\text{coll}}(\tau) \geq 0$ sums contact-count penalties for rope-robot contacts, extender-pole-robot contacts, and any contact entering the robot’s weighted base region, with the base-region term weighted more heavily than the others.

K.2 Pole Striking

$$f(\tau) = d_{\min} - R_{\text{prox}} + M(\tau) + P_{\text{vel}}(\tau) + P_{\text{coll}}(\tau) \quad (3)$$

K.3 Lobbing

The lobbing task augments the striking fitness with a dwell term that rewards the tip for remaining near the target after arrival. Define

$$T_{\text{near}}(\tau) = |\{t : \|p_{\text{tip}}(t) - p_{\text{tgt}}\|_2 < \epsilon\}|, \quad (4)$$

the number of simulation timesteps for which the tip is within distance ϵ of the target. By construction $T_{\text{near}}(\tau) > 0$ if and only if $d_{\text{min}} < \epsilon$, i.e., only when the tip actually enters the target neighborhood at some point along the trajectory. The dwell reward applies a piecewise scale to T_{near} :

$$R_{\text{dur}}(\tau) = s(d_{\text{min}}) \cdot T_{\text{near}}(\tau), \quad s(d) = \begin{cases} \alpha_{\text{hi}} & d < \epsilon \\ \alpha_{\text{lo}} & d \geq \epsilon, \end{cases} \quad (5)$$

with $\alpha_{\text{hi}} \gg \alpha_{\text{lo}}$. The fitness is

$$f(\tau) = d_{\text{min}} - R_{\text{prox}} - R_{\text{dur}} + M(\tau) + P_{\text{vel}}(\tau) + P_{\text{coll}}(\tau). \quad (6)$$

This produces a staggered objective. Before any candidate trajectory achieves arrival, $T_{\text{near}} = 0$, so R_{dur} vanishes and the fitness reduces to the pole-striking fitness. Once a trajectory reaches $d_{\text{min}} < \epsilon$, the scale switches to α_{hi} and every additional near-target timestep contributes a large reduction in fitness. CMA-ES therefore first finds trajectories that arrive at the target, and then refines them to maximize dwell time, producing the slow controlled lobbing motion rather than a fast strike-and-leave.

K.4 Draping

Target marked over the wall with an intermediate waypoint over the wall edge $p_{\text{wall}} = p_{\text{tgt}} + (\Delta x, 0, -L/2)$. Let $d_{\text{tgt}}, d_{\text{wall}}$ be the minimum tip distances to each, and $P_{\text{height}} = \max(0, \max_t p_{\text{tip},z}(t) - z_{\text{max}})$ discourage high arcs:

$$f(\tau) = d_{\text{tgt}} + d_{\text{wall}} + P_{\text{height}} + P_{\text{vel}}(\tau) + P_{\text{coll}}(\tau). \quad (7)$$

d_{wall} encourages the tip to pass over the wall edge before descending to the target.

Table 10: Neural network activation heatmaps showing gradient flow for each rope parameter during wiggle observation. These are for one demonstration of our 45cm brown rope with five gram lead.

