CORRECT THE ACTION, NOT THE POLICY: ONLINE RECOVERY UNDER DYNAMICS SHIFTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Changes in object dynamics can cause a pretrained manipulation policy to fail even when object geometry and task goals remain unchanged. We study whether such failures can be recovered during execution without updating the policy itself. We introduce a model-based action correction approach that estimates effective dynamics from interaction records, trains a lightweight predictive model under the estimated dynamics, and modifies only a low-dimensional subset of the frozen policy's actions. During execution, candidate corrections are evaluated from the current observation and selected to keep predicted motion close to a successful trajectory produced by the same policy under reference dynamics. Across shifts in object mass and contact friction, the method raises success from 0–4% to 52–78% in rope routing and T-pushing, and from 65% to 78% in toy packing. A separate physical rope experiment further shows that a simulation-selected correction can recover an execution that fails after a change in object properties. These results show that some failures caused by dynamics shifts can be recovered through execution-time action correction rather than policy retraining.

1 INTRODUCTION

Learning-based policies can successfully solve a range of robotic manipulation tasks under the dynamics encountered during training (Chi et al., 2025). However, changes in physical properties such as mass, friction, or stiffness can alter the motion produced by the same action and cause a previously successful policy to fail even when object geometry and the task goal remain unchanged (Peng et al., 2018; Zhang et al., 2024). Retraining or replacing the policy after each such dynamics shift would require additional demonstrations or interactions in the target environment.

Prior work addresses physical variation by broadening the policy's training distribution through domain randomization (Peng et al., 2018), adapting simulation before retraining the policy (Chebotar et al., 2019), or learning an additional residual policy (Johannink et al., 2019). These approaches modify what is learned before or after the shift.

Other approaches modify executed actions while retaining a pretrained policy. Iterative Residual Policy (Chi et al., 2024) revises actions using feedback from previous executions and a learned model of trajectory changes, whereas DiPac (Chen et al., 2024) optimizes and replans action sequences through a differentiable simulator calibrated to observations. These approaches leave open a different question: when a pretrained policy fails specifically because the physical dynamics have changed, can task success be recovered by applying only limited corrections to its actions while keeping the policy weights fixed?

We use an adjustable-physics simulator to infer an effective dynamics setting from interaction records collected after the shift. PhysTwin (Jiang et al., 2025) provides such a simulation model, and prior real-to-sim evaluation has reported strong agreement between simulated and physical policy performance in related manipulation tasks (Zhang et al., 2026). These approaches leave open a different question: when a pretrained policy fails specifically because the physical dynamics have changed, can task success be recovered during the same execution by applying only limited corrections to the policy's actions?

Our approach first fits effective simulator parameters to interaction records from the shifted environment and trains a lightweight dynamics surrogate under the estimated setting. The pretrained

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

policy remains the task-level controller: at execution time, we repeatedly search only over a low-dimensional residual to its actions, using the surrogate to predict which correction best preserves a successful reference behavior. The correction is recomputed from new observations without simulator calls during candidate evaluation.

Across shifts in object mass and contact friction, online correction raises success from 0–4% to 52–78% in rope routing and T-pushing, while toy-packing success increases from 65% to 78%. We compare against direct simulator search, residual reinforcement learning, and dynamics-randomized behavior cloning. A separate physical rope experiment further shows that a simulation-selected correction can recover an execution that fails after a change in object properties. Together, these results show that failures induced by changed dynamics can, in selected settings, be recovered by correcting the actions of a frozen policy rather than retraining the policy itself.

2 Related Work

2.1 Policy Learning and Action Correction under Physics Shifts

Demonstration-based policies (Chi et al., 2025) can accommodate physical variation through training. Domain randomization (Peng et al., 2018) trains across dynamics; UP-OSI (Yu et al., 2017) and GenDOM (Kuroki et al., 2024) condition policies on estimated physical properties. SimOpt (Chebotar et al., 2019) and AdaptSim (Ren et al., 2023) adapt the training distribution using execution records or task performance. These approaches motivate our DR+BC baseline.

Residual Reinforcement Learning (Johannink et al., 2019) and Residual Policy Learning (Silver et al., 2018) add learned actions to an existing controller; Ankile et al. (2025) learns a closed-loop residual for an imitation policy. Our residual-RL baseline retains the original policy while learning an additional model.

Actions can also be revised without policy updates. Iterative learning control (Tharayil & Alleyne, 2006) uses errors from previous attempts, while Iterative Residual Policy (IRP) (Chi et al., 2024) predicts the trajectory response to action changes and revises actions over repeated executions on a new object. Both use feedback across attempts.

DINO-WM (Zhou et al., 2024) plans before execution using predicted visual features. DreamTrajectory (Yang et al., 2026) compares candidate predictions with the trajectory intended by a pretrained policy; its policy and predictor are trained beforehand. Adjustable-physics simulation offers another way to predict candidate outcomes.

2.2 Observation-Based Physics Simulation and Action Selection

Simulation-based action selection requires the shifted dynamics to be reproduced. DiffCloud (Sundaresan et al., 2022) estimates deformable-object mass and stiffness from point clouds; Antonova et al. (2022) infer property distributions for rope and cloth. PhysTwin (Jiang et al., 2025) reconstructs geometry and spring–mass properties from interaction videos. We fit its simulator parameters to shifted-environment records.

Zhang et al. (2026) reports highly correlated real and simulated success rates across policy checkpoints in PhysTwin-reconstructed scenes. Under shifted properties, we evaluate both motion reproduction and success after correction.

PlasticineLab (Huang et al., 2021) optimizes actions in a differentiable soft-body simulator, while CORD-SLS (Li et al., 2026) replans with a contact-aware simulator. DiPac (Chen et al., 2024) fits particle-simulator stiffness and friction to observations, then optimizes and replans action sequences from a learned policy. We instead train a surrogate under estimated physics and update only a low-dimensional correction to the frozen policy's output as observations arrive; the methods differ in optimization scope and predictive model.

3 PROBLEM FORMULATION

3.1 SETTING

For task q, we apply a policy π_0 trained under reference physical parameters θ_0 to a scene with the same object geometry, robot, and task goal, in which only physical parameters such as mass, stiffness, and friction have changed to θ_*. Because θ_* is generally not directly observable in the physical environment, we infer corresponding simulator parameters $\hat{\theta}$ from object motion. In simulated evaluations, θ_* is likewise withheld from parameter estimation and action correction. Let s_t denote the object and robot state at time t and g_q the task goal. At every time step, the policy with fixed weights computes its base action from the observation o_t:

$$o_t = \phi_q(s_t, g_q, t), \qquad a_t^0 = \pi_0(o_t). \tag{1}$$

3.2 PHYSICAL PARAMETER ESTIMATION

We replay the action sequences A_j of M executions recorded in the shifted environment in a simulator with matching initial conditions, and seek parameters for which the simulated trajectory $F_\theta(A_j)$ matches the observed object trajectory X_j^{obs}:

$$\hat{\theta} \in \arg\min_{\theta \in \Theta_q} \frac{1}{M} \sum_{j=1}^{M} \ell_q\big(F_\theta(A_j), X_j^{\mathrm{obs}}\big)\,. \tag{2}$$

Here, Θ_q is the admissible parameter range, and ℓ_q compares object motion at corresponding time steps.

3.3 ACTION CORRECTION

To address the physics shift, we add an action correction Δa_t to the base action at each time step:

$$\tilde{a}_t = a_t^0 + \Delta a_t. \tag{3}$$

The applied correction sequence $\Delta A = (\Delta a_0, \ldots, \Delta a_{T-1})$ belongs to a set $\mathcal{C}_q$ of sequences that satisfy robot feasibility constraints and affect only the permitted action components. Let $Q_q(\Delta A; \hat{\theta})$ denote the score obtained by executing the frozen policy together with ΔA under the estimated parameters. The idealized sequence-level objective is

$$\Delta A^* \in \arg\max_{\Delta A \in \mathcal{C}_q} Q_q(\Delta A; \hat{\theta}). \tag{4}$$

The method generates ΔA through successive observation-dependent selections of a single correction vector, rather than fixing the entire sequence before execution. In this work, Q_q reflects agreement between the predicted trajectory and a reference trajectory $Y^{\mathrm{ref}} = (y_0^{\mathrm{ref}}, \ldots, y_T^{\mathrm{ref}})$, where each y_t selects from s_t the object and robot features needed for tracking.

4 METHOD

4.1 OVERVIEW

We identify $\hat{\theta}$, train a dynamics surrogate under these parameters, and use it for online correction of the frozen policy π_0 (Figure 1).

4.2 PHYSICAL PARAMETER IDENTIFICATION

The trajectory error ℓ_q compares the positions of all corresponding object particles at each observation time, unlike the policy observation o_t, which uses selected features. We search over parameter candidates with CMA-ES Hansen (2006) and validate the selected $\hat{\theta}$ on records withheld from identification. Thus, $\hat{\theta}$ is a simulator setting that reproduces observed motion, rather than a direct measurement of physical parameters.

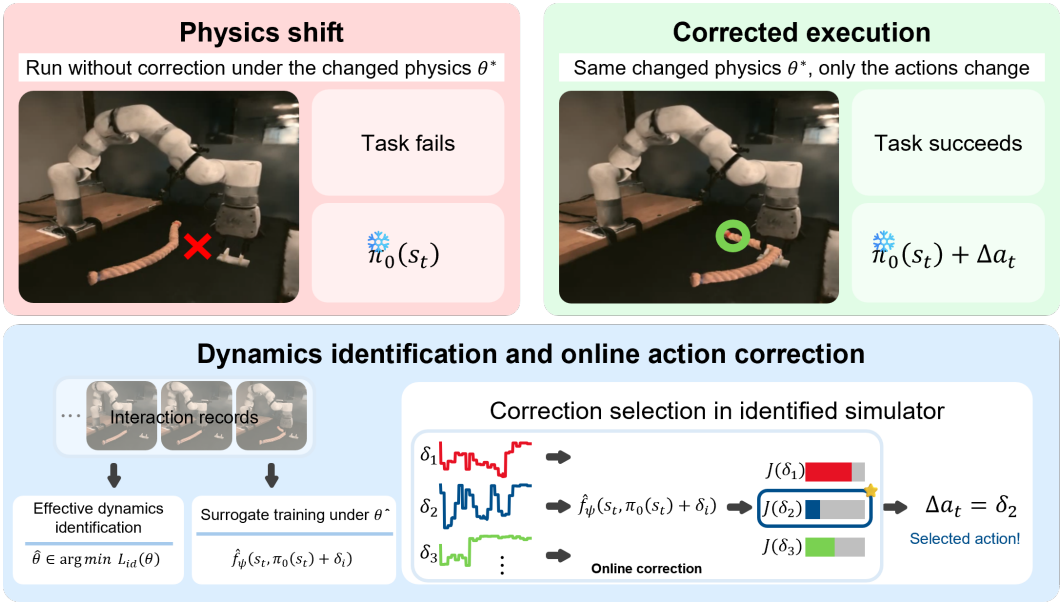


Figure 1: Overview of dynamics identification and online action correction. After a physics shift causes a pretrained policy to fail, we identify effective dynamics from interaction records and train a lightweight surrogate. During execution, the frozen policy provides the nominal action and the surrogate selects a low-dimensional residual, giving $\tilde{a}_t = \pi_0(s_t) + \Delta a_t$. The residual is updated online without modifying the policy weights.

4.3 Low-Dimensional State Dynamics Surrogate

Because simulating each candidate at every correction step would be computationally costly, we train a dynamics surrogate $\widehat{f}_\psi$ that predicts motion from a low-dimensional state z_t combining representative object positions and velocities with robot motion and contact information. Training data are collected by executing the frozen policy in the simulator under $\hat{\theta}$ while perturbing the permitted action components by ϵ_t:

$$\tilde{a}_t^{\text{data}} = a_t^0 + \epsilon_t, \qquad \widehat{z}_{t+1} = \widehat{f}_\psi(z_{t-L+1:t}, a_t^{\text{data}}). \tag{5}$$

Here, a_t^{data} is the executed action after feasibility constraints are applied to the requested action $\tilde{a}_t^{\text{data}}$, and L is the length of the input state history. The perturbations expose the model to outcomes near the policy's usual actions, and the model does not take $\hat{\theta}$ as input.

4.4 Reference-Trajectory Action Correction

The reference trajectory Y^{ref} is a successful execution of the pretrained policy under the reference parameters. At each correction step, we use a short prediction horizon to select one correction vector from the current observation. A candidate correction δ is held constant over a prediction horizon of $H = 30$ frames, during which the frozen policy recomputes its base action from the predicted state at every step. The discrepancy between the predicted tracking state $\widehat{y}_{t+h}(\delta)$ and the reference trajectory is

$$C_t^{\text{track}}(\delta) = \frac{1}{H} \sum_{h=1}^{H} d_q\left(\widehat{y}_{t+h}(\delta), y^{\text{ref}}_{\min(t+h,T)}\right), \tag{6}$$

where d_q is a task-specific distance and predictions beyond the reference horizon use the final reference state. The candidate cost $\widehat{C}_t$ also accounts for disagreement among model predictions, and a penalty R_t discourages large corrections and changes from the previous correction δ_{prev}:

$$J_t(\delta) = \widehat{C}_t(\delta) + R_t(\delta, \delta_{\text{prev}}), \qquad \delta_t^* \in \arg\min_{\delta \in \mathcal{D}_q} J_t(\delta). \tag{7}$$

The feasible range $\mathcal{D}_q$ covers gripper target positions and, where needed, gripper opening. We search candidates with the cross-entropy method and select a nonzero correction only when its predicted cost improves sufficiently over no correction. The chosen δ_t^* is applied as Δa_t for the next 10 frames and then recomputed from a new observation. Executed actions also satisfy task-specific feasibility constraints (Appendix A.2).

5 EXPERIMENTS

5.1 TASKS AND PHYSICS SHIFTS

We evaluate whether a pretrained policy that fails under a physics shift can be recovered without retraining the policy. All experiments use a PhysTwin-based simulator in which the scene and object geometry are kept fixed while physical parameters are changed. In rope routing, we separately increase the rope mass multiplier from 1 to 5 and the floor-friction coefficient from approximately 0.50 to 2.0. In T-pushing, we increase the pusher-friction coefficient from 1.0 to 1.6, and in toy packing, we reduce the gripper-friction coefficient from 1.0 to 0.5. Each task uses a policy π_0 trained under the reference physical parameters θ_0, and its weights are fixed for both our method and the uncorrected baseline. The complete parameter vectors are listed in Table 1.

5.2 EVALUATED METHOD AND BASELINES

We apply the same reference-tracking correction to all four shifts, updating one correction vector from the latest observation every 10 frames without simulator calls during candidate evaluation. A variant with an added terminal task cost is evaluated separately.

We compare against four baselines. The *uncorrected* baseline executes the policy unchanged under the shifted physics. *Direct simulator search* selects time-varying correction values before evaluation through complete rollouts in the estimated-physics simulator, parameterizing the correction sequence by temporal knots and fixing it before evaluation; details are given in Appendix A.3. *Residual RL* learns a new correction model in the shifted-physics environment, and *DR+BC* learns a new policy from demonstrations that cover a range of physical parameters. Our residual RL baseline belongs to the same category as prior work that adds a learned residual to an existing controller Johannink et al. (2019). It keeps the pretrained policy fixed and trains over 160 executions with success rewards in the shifted-physics simulation; the learned model determines the correction at the start of each execution. Thus, our method and direct search keep the policy frozen, whereas residual RL trains an additional correction model and DR+BC trains a new policy.

DR+BC combines dynamics randomization Peng et al. (2018) with behavior cloning from demonstrations Pomerleau (1991). It is told in advance which physical parameter changes and over what range, and one human demonstration is collected at each of nine levels along that single axis. A new policy is trained on state histories without a time input, with the history length selected per task under the reference parameters. This baseline is therefore not an update of the policy checkpoint used by our method. In toy packing, we did not obtain a DR+BC policy suitable for final evaluation, and the corresponding entry is marked `n/a` in Figure 4.

5.3 SUCCESS CRITERIA AND REPEATED EVALUATION

The success rate is the fraction of single policy executions under the shifted parameters θ_* that satisfy the task-specific termination condition. In rope routing, at least 100 particles must cross each of the two decision planes of one clip, and the gripper must remain released from the rope throughout the final 30 frames. In T-pushing, the RMS error of the object particle configuration at the final frame must be at most 15 mm. In toy packing, at least 3050 object particles must lie inside the box in at least 30 of the final 100 frames of a 750-frame execution, and the execution safety conditions must be satisfied.

Online correction and its paired uncorrected baseline are evaluated in 23 executions per condition, and direct search, residual RL, and DR+BC are each evaluated in 20 executions per condition.

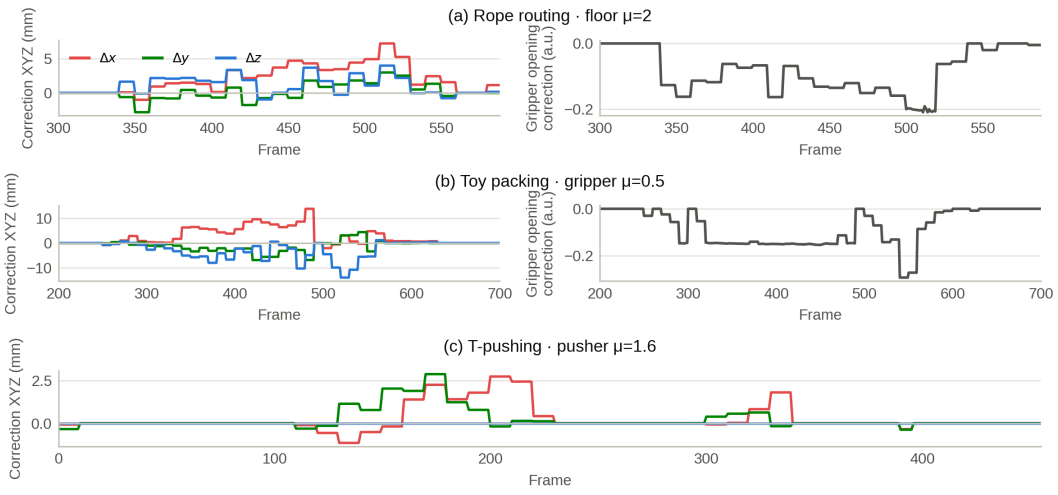


Figure 2: Online correction signals from successful executions under (a) a floor-friction shift in rope routing, (b) a gripper-friction shift in toy packing, and (c) a pusher-friction shift in T-pushing. Panels (a) and (b) show corrections to the gripper target position in x, y, and z and to gripper opening; panel (c) shows position corrections in x and y. Negative opening corrections close the gripper further. The rotational command in T-pushing is not corrected.

5.4 Physical Parameter Identification and Additional Cost

Physical parameters are identified as described in Section 4.2, with the identification loss computed over intervals in which robot–object interaction is visible. Identification accuracy is compared with the ground-truth shifted parameters in Section 6.3. The additional training and simulator use that each method requires for adaptation are summarized in Appendix Table A1, and per-condition record and execution counts are given in Appendix Table A2.

6 Results

6.1 Online Action Correction Recovers Successful Executions

Online action correction improves the success rate of the frozen policy under all four physics shifts (Figure 4). We examine the corrections applied in successful executions (Figure 2) and the rope-routing and toy-packing executions (Figure 3) to show how the executed actions change.

After floor friction increases from approximately 0.5 to 2.0 in rope routing, the uncorrected execution loses the rope at frame 494 and fails to pass it through the clip (Figure 3, left). The correction in Figure 2(a) closes the gripper further, and the corrected execution retains the rope and passes it through the clip.

In toy packing, the reduced gripper friction likewise causes the uncorrected execution to drop the toy from the closed gripper at frame 460 (Figure 3, right). The correction in Figure 2(b) also closes the gripper further, and the corrected execution carries the toy into the box.

6.2 Comparison with Adaptation Baselines

Figure 4 compares online correction with the four baselines across repeated executions.

Relative to the paired uncorrected baseline, online correction raises success from 0% to 78% for the rope mass shift, from 4% to 52% for the rope floor-friction shift, and from 0% to 74% for T-pushing. Under the same conditions, DR+BC and direct simulator search reach 100%, and residual RL reaches 100% for the rope mass shift and T-pushing. The contribution of online correction is therefore not the highest success rate, but recovery from failure through action correction during

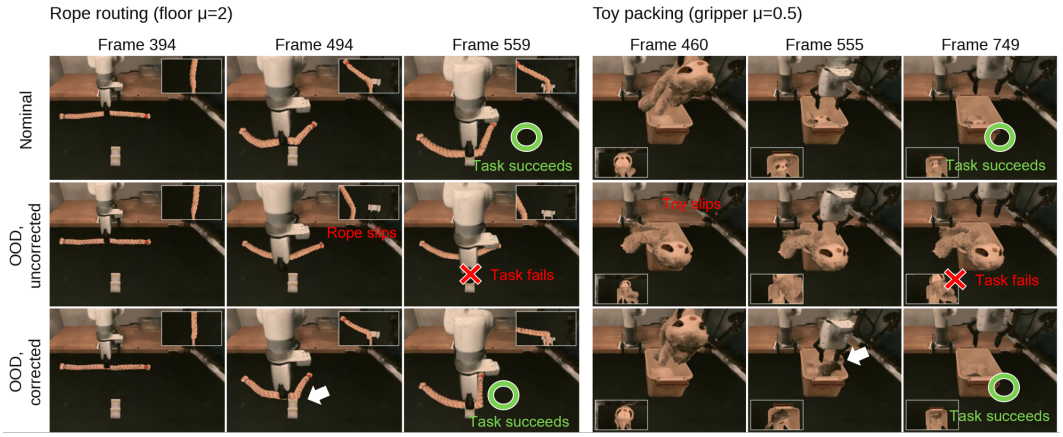


Figure 3: Representative rope-routing (left) and toy-packing (right) executions. Rows show success under reference physics, uncorrected failure under shifted physics, and success with online correction under shifted physics. Insets show additional camera views.

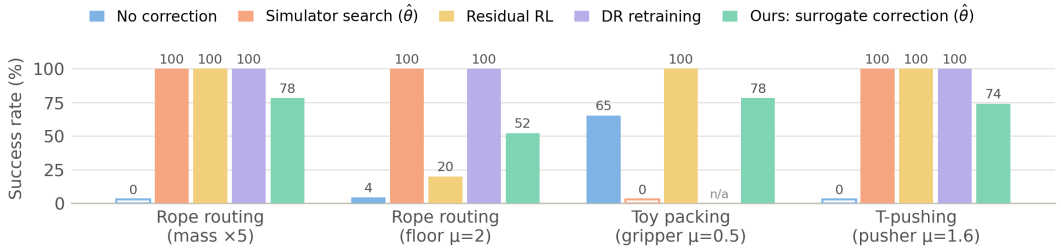


Figure 4: Task success under shifted physics. Online correction and its uncorrected baseline use 23 executions per condition; direct simulator search, residual RL, and DR+BC each use 20. The toy-packing DR+BC entry is marked `n/a` because no final comparison policy was obtained.

execution while the policy weights remain unchanged. In toy packing, the uncorrected policy already succeeds in 65% of executions, so the additional gain is smaller.

Baseline performance also depends on the condition. Under the rope floor-friction shift, residual RL succeeds in 20% of executions, the same as its paired uncorrected baseline. In toy packing, direct search passes its selection check in the estimated-physics simulator but succeeds in 0% of shifted-physics evaluations, whereas online correction succeeds in 78%. This case shows that a correction that succeeds under estimated physics need not transfer to the shifted physics. Because direct search and online correction also differ in their objectives and correction variables, this difference cannot be attributed to online updating alone.

These results involve different adaptation requirements: DR+BC trains a new policy from nine demonstrations with the changed parameter axis and range supplied; residual RL trains a correction model over 160 shifted-physics executions; and direct search uses 30–144 additional simulator rollouts after identification. Our method requires identification and surrogate training but no simulator calls when selecting corrections during execution (Appendix Table A1).

6.3 Validation of Motion Prediction under Identified Physics

Identification compares replayed particle motion over informative interaction intervals: frames 361–406 after grasping for rope routing, 270–420 for toy packing, and 110–455 for T-pushing, sampled every five frames. Later clip passage and release in rope routing are excluded. Table 1 lists reference θ_0, estimated $\hat{\theta}$, and shifted θ_* parameters.

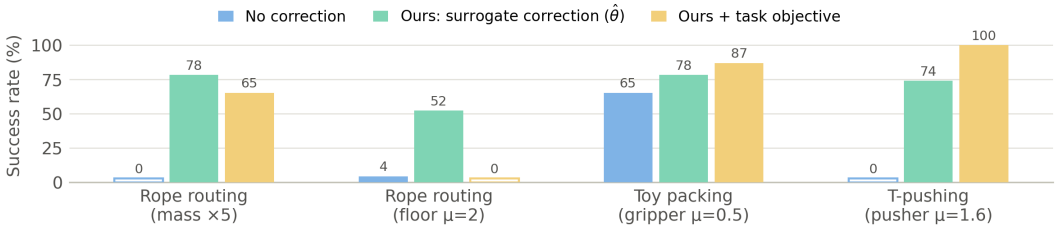


Figure 5: Success rates with a terminal task cost. Reference-trajectory tracking and the variant are compared over 23 executions per condition.

When the same actions are replayed under the three parameter settings, the object shapes under estimated parameters are closer to those under shifted parameters than to those under reference parameters for both rope-routing shifts and for T-pushing (Appendix Figure A2).

Measured on held-out records as the distance between corresponding positions of all particles, the replay error decreases from 5.683 to 0.249 mm for the rope mass shift, from 8.671 to 3.497 mm for the floor-friction shift, and from 5.395 to 0.684 mm for T-pushing. For toy packing, it decreases from 15.14 to 12.92 mm but remains above the 7.54 mm obtained by replaying under the ground-truth parameters. These errors measure how accurately object motion is reproduced under recorded actions, not task success.

For the rope mass shift, estimated mass and stiffness $(3.73, 0.75)$ differ from ground truth $(5, 1)$ yet reproduce the recorded motion more accurately than the ground-truth setting. Ground-truth replay error is also nonzero for T-pushing. Thus, motion fit does not establish parameter accuracy or guarantee success under newly corrected actions, though effective simulator parameters may still support correction.

6.4 Benefits and Limitations of a Terminal Task Cost

Reference-trajectory tracking follows the order of the motion but does not directly reduce error in the final goal state. We therefore compare it with a variant that adds the task goal as a terminal cost (Figure 5). Both use the surrogate trained under estimated parameters and correct the frozen policy every 10 frames.

For T-pushing and toy packing, the variant increases success from 74% to 100% and from 78% to 87%, respectively. A representative T-pushing execution with the terminal task cost is shown in Appendix Figure A5. For rope routing, success decreases from 78% to 65% under the mass shift and from 52% to 0% under the floor-friction shift. Under the floor-friction shift, the variant releases the rope before clip passage, and gripper contact force vanishes early (Appendix Figure A4). These results suggest that a final-state cost helps in some tasks, but rope routing requires the objective to represent the order of grasp maintenance, clip passage, and release.

Table 1: Physical parameters by task. Mass and stiffness are multipliers relative to their reference values; friction entries are simulator coefficients, rounded to two decimal places. Parameters are ordered as (mass, stiffness, floor friction) for rope routing, (mass, stiffness, gripper friction) for toy packing, and (mass, floor friction, pusher friction) for T-pushing.

Task	Reference θ_0	Estimated $\hat{\theta}$	Shifted θ_*
Rope routing, mass shift	$(1.00, 1.00, 0.50)$	$(3.73, 0.75, 0.53)$	$(5.00, 1.00, 0.50)$
Rope routing, floor-friction shift	$(1.00, 1.00, 0.50)$	$(2.23, 1.17, 1.97)$	$(1.00, 1.00, 2.00)$
Toy packing	$(1.00, 1.00, 1.00)$	$(1.02, 1.34, 0.52)$	$(1.00, 1.00, 0.50)$
T-pushing	$(1.00, 0.50, 1.00)$	$(1.71, 0.45, 1.86)$	$(1.00, 0.50, 1.60)$

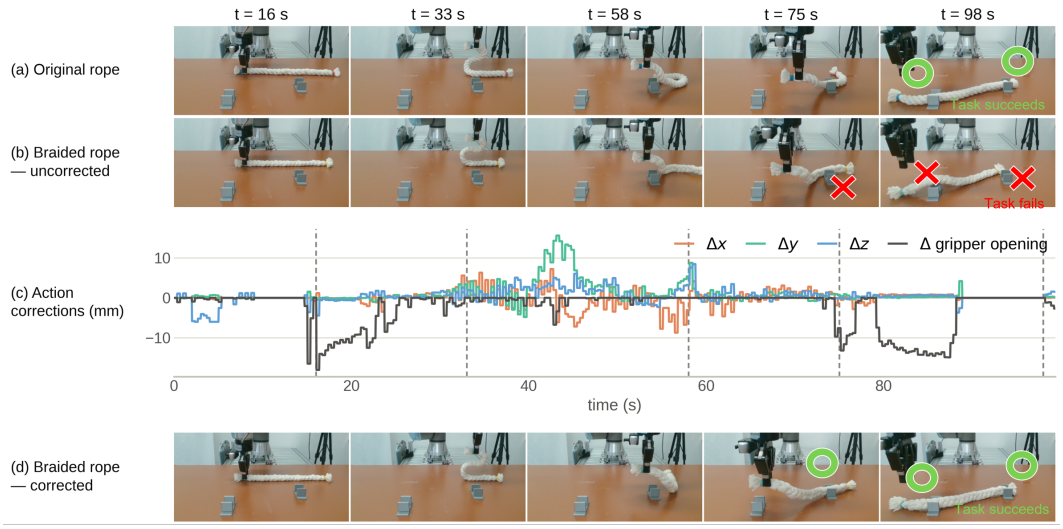


Figure 6: Physical rope-routing experiment under a dynamics shift. (a) The frozen policy succeeds with the original rope. (b) The same policy fails with the braided rope without correction. (c) Time-varying position and gripper-opening corrections applied to the frozen policy. Negative opening corrections close the gripper further. (d) With the corrected actions, the braided rope passes both clips and the task succeeds. The policy weights remain unchanged.

6.5 Action Correction for a Physical Rope with Shifted Properties

We perform a separate physical experiment in which a rope must pass through two clips, using a braided rope that keeps its appearance close to the original while increasing its resistance to bending (Figure 6). The action trajectory of the policy that succeeds with the original rope fails at the second clip with the braided rope, whereas the corrected actions pass both clips and release the rope.

Unlike the reference parameters θ_0 of the original rope, the parameters $\hat{\theta}$ identified from video of the braided-rope failure reproduce the failure in the simulator. The mean distance between reconstructed rope points and simulated particles also decreases from 29.9 to 26.5 mm. Writing the parameter vector as (mass multiplier, tensile-spring stiffness multiplier, floor-friction coefficient), we obtain $\theta_0 = (1, 1, 0.544)$ and $\hat{\theta} = (1.111, 0.586, 0.524)$. The estimate is an effective simulator setting fitted to observed motion, not a measurement of the actual bending stiffness.

The correction selected by a surrogate trained under these parameters also succeeds in the simulator. On the physical robot, the correction closes the gripper further while transporting the rope and shifts the gripper by up to 15.6 mm in the robot-base y direction (Figure 6(c)).

7 Conclusion and Limitations

Low-dimensional online correction recovered successful executions of a frozen policy under physics shifts, raising success from 0% to 78% for rope mass, 4% to 52% for rope floor friction, and 0% to 74% for T-pushing. A simulation-selected correction also succeeded in a separate physical rope case.

Our method is best suited to settings in which the policy can be recovered by adjusting its existing actions around a successful reference trajectory, and task objectives that reflect the order of task phases may further improve correction. Extending the evaluation to simultaneous shifts in multiple physical parameters and to repeated physical experiments with observation-dependent correction updates remains future work.

AI USE STATEMENT

We used generative AI tools to draft, translate, and edit manuscript text, provide feedback on experimental presentation and result interpretation, and assist with code development and debugging. The authors reviewed AI-assisted text and code and checked the reported claims against experimental records. We take responsibility for the final content of this work, including all AI-assisted text, claims, and artifacts.

REPRODUCIBILITY STATEMENT

Sections 3–5 specify the setting, method, physics shifts, baselines, success criteria, and evaluation protocol. Section 6 reports the results. The appendix provides the surrogate and correction settings, adaptation counts, and diagnostic figures.

REFERENCES

Lars Ankile, Anthony Simeonov, Idan Shenfeld, Marcel Torne, and Pulkit Agrawal. From imitation to refinement-residual rl for precise assembly. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 01–08. IEEE, 2025.

Rika Antonova, Jingyun Yang, Priya Sundaresan, Dieter Fox, Fabio Ramos, and Jeannette Bohg. A bayesian treatment of real-to-sim for deformable object manipulation. *IEEE Robotics and Automation Letters*, 7(3):5819–5826, 2022.

Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan Ratliff, and Dieter Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *2019 international conference on robotics and automation (ICRA)*, pp. 8973–8979. IEEE, 2019.

Siwei Chen, Yiqing Xu, Cunjun Yu, Linfeng Li, and David Hsu. Differentiable particles for general-purpose deformable object manipulation. *arXiv preprint arXiv:2405.01044*, 2024.

Cheng Chi, Benjamin Burchfiel, Eric Cousineau, Siyuan Feng, and Shuran Song. Iterative residual policy: for goal-conditioned dynamic manipulation of deformable objects. *The International Journal of Robotics Research*, 43(4):389–404, 2024.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

Nikolaus Hansen. The cma evolution strategy: a comparing review. *Towards a new evolutionary computation: Advances in the estimation of distribution algorithms*, pp. 75–102, 2006.

Zhiao Huang, Yuanming Hu, Tao Du, Siyuan Zhou, Hao Su, Joshua B. Tenenbaum, and Chuang Gan. Plasticinelab: A soft-body manipulation benchmark with differentiable physics. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=xCcdBRQEDW.

Hanxiao Jiang, Hao-Yu Hsu, Kaifeng Zhang, Hsin-Ni Yu, Shenlong Wang, and Yunzhu Li. Phystwin: Physics-informed reconstruction and simulation of deformable objects from videos. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7219–7230. IEEE, 2025.

Tobias Johannink, Shikhar Bahl, Ashvin Nair, Jianlan Luo, Avinash Kumar, Matthias Loskyll, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. Residual reinforcement learning for robot control. In *2019 international conference on robotics and automation (ICRA)*, pp. 6023–6029. IEEE, 2019.

So Kuroki, Jiaxian Guo, Tatsuya Matsushima, Takuya Okubo, Masato Kobayashi, Yuya Ikeda, Ryosuke Takanami, Paul Yoo, Yutaka Matsuo, and Yusuke Iwasawa. Gendom: Generalizable one-shot deformable object manipulation with parameter-aware policy. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 14792–14799. IEEE, 2024.

Wei-Chen Li, Jeffrey Fang, Sasanka Polisetti, Yuexi Song, and Glen Chou. Robustness without wrinkles: Parallel simulation and robust mpc for certified deformable manipulation. *arXiv preprint arXiv:2606.14188*, 2026.

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 3803–3810. IEEE, 2018.

Dean A. Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3(1):88–97, 1991. doi: 10.1162/neco.1991.3.1.88.

Allen Z. Ren, Hongkai Dai, Benjamin Burchfiel, and Anirudha Majumdar. Adaptsim: Task-driven simulation adaptation for sim-to-real transfer. In Jie Tan, Marc Toussaint, and Kourosh Darvish (eds.), *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 3434–3452. PMLR, 06–09 Nov 2023.

Tom Silver, Kelsey Allen, Josh Tenenbaum, and Leslie Kaelbling. Residual policy learning. *arXiv preprint arXiv:1812.06298*, 2018.

Priya Sundaresan, Rika Antonova, and Jeannette Bohgl. Diffcloud: Real-to-sim from point clouds with differentiable simulation and rendering of deformable objects. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10828–10835. IEEE, 2022.

DABM Tharayil and Andrew G Alleyne. A survey of iterative learning control: A learning-based method for highperformance tracking control. *IEEE Control systems magazine*, 26(3):96–114, 2006.

Zheng Yang, Wenjie Zhang, Xiangyu Chen, Wenxuan Song, Xianpeng Wang, Yihang Kang, Wen Chen, Lujia Wang, Renjing Xu, and Xiaowen Chu. Dreamtrajectory: Trajectory-guided action generation with world model alignment for mobile manipulation. *arXiv preprint arXiv:2608.01381*, 2026.

Wenhao Yu, Jie Tan, C. Karen Liu, and Greg Turk. Preparing for the unknown: Learning a universal policy with online system identification. In *Proceedings of Robotics: Science and Systems*, Cambridge, Massachusetts, July 2017. doi: 10.15607/RSS.2017.XIII.048.

Kaifeng Zhang, Baoyu Li, Kris Hauser, and Yunzhu Li. Adaptigraph: Material-adaptive graph-based neural dynamics for robotic manipulation. *URL https://arxiv. org/abs/2407*, 7889, 2024.

Kaifeng Zhang, Shuo Sha, Hanxiao Jiang, Matthew Loper, Hyunjong Song, Guangyan Cai, Zhuo Xu, Xiaochen Hu, Changxi Zheng, and Yunzhu Li. Real-to-Sim Robot Policy Evaluation with Gaussian Splatting Simulation of Soft-Body Interactions. In *2026 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2026.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. Dino-wm: World models on pre-trained visual features enable zero-shot planning. *arXiv preprint arXiv:2411.04983*, 2024.

A Method Details and Correction Diagnostics

A.1 Dynamics Surrogate

The reduced state consists of dynamic variables from which the policy input can be reconstructed. For the deformable-object tasks, it contains the positions and velocities of selected particles, the gripper position, orientation, and opening, and available grasp information. For T-pushing, it contains the object center and orientation, their velocities, and the robot position. A history of the most recent $L = 8$ states is encoded by a one-dimensional convolution along the time axis and combined with the current action to predict the next state. Training proceeds sequentially with one-step, five-step, and twenty-step rollout losses. Three models trained from different initializations are used in candidate search. The models are trained on state–action trajectories collected under the estimated parameters and do not take the physical parameters as input.

A.2 Scoring and Selecting a Correction

Let $C_{k,t}(\delta)$ denote the reference-tracking cost predicted by model k for a candidate correction δ. Each candidate is a single correction vector held fixed over a 30-frame surrogate rollout; the frozen policy recomputes the base action at every predicted frame. The implemented score adds the mean and standard deviation of this cost across models and penalizes both the magnitude of the correction and its change from the previous correction. With a per-component scale vector σ_q and previous correction δ_{prev}, the score is

$$J_t(\delta) = \text{mean}_k\, C_{k,t}(\delta) + \text{std}_k\, C_{k,t}(\delta) + 0.05\, \|\delta/\sigma_q\|_2^2 + 0.05\, \|(\delta - \delta_{\text{prev}})/\sigma_q\|_2^2 . \tag{8}$$

Division is element-wise, so components with different units, such as position and gripper-opening corrections, are compared after scaling by their own scales. $C_{k,t}$ is the mean tracking error over the $H = 30$ frames following each correction time. In the full-state tracking configuration for rope routing, the error includes representative particle positions, gripper position and opening, grasp state, and finger force below the reference; no force penalty is applied when finger force exceeds the reference. In the main T-pushing comparison, the reference is a successful execution of the same policy under reference physics. Its tracking cost uses the full nine-dimensional state: object center and yaw, robot end-effector position, and object linear and angular velocities.

A single correction is selected from the latest observation every 10 frames and held until the next update. The cross-entropy method evaluates 256 candidates per generation for the deformable-object tasks and 512 for T-pushing, with three generations and 32 elites in both implementations. The online correction is set to zero unless the predicted cost of the best candidate is at least 0.02 lower than that of the zero correction. For the deformable-object tasks, σ_q is 5 mm for position components and 0.15 for gripper opening; T-pushing uses 5 mm for the x and y components. These values normalize the penalties and are distinct from the admissible search ranges.

The rope-routing executor checks the minimum distance between sampled points on the robot geometry and the floor but does not perform exact collision checking against the clips. Online prediction for T-pushing does not reproduce every candidate-rejection rule of the actual executor. The feasibility constraints in the main text should therefore not be interpreted as guaranteeing collision avoidance with all obstacles.

A.3 Direct Simulator Search

The direct-search baseline parameterizes the correction sequence ΔA by a small number of temporal knot values v. Corrections to the gripper target position are linearly interpolated between knots; tasks that require it also include a correction to gripper opening. Gripper orientation is not corrected. At each time step, the resulting $\Delta a_t(v)$ is added to the policy's base action a_t^0, while the policy continues to recompute that base action from the state observed during the corrected execution. Each candidate v is evaluated through a rollout in the simulator under the estimated parameters and compared by a task-performance score; the rule combining success and continuous task scores varies by task and parameter condition. We search with the cross-entropy method and validate the leading candidates in separate rollouts before selection. The selected v^* is fixed before final evaluation and is not searched again during evaluation. This approach does not train a dynamics surrogate.

A.4 Adaptation Cost and Implementation Counts

Table A1 summarizes the additional training and simulator use that each method requires for adaptation, and Table A2 gives the per-condition counts. Identification candidate replays are the simulator replays used in the CMA-ES search; for rope routing and toy packing, additional validation replays of the selected parameters are performed separately. Surrogate data count all collected executions, including those used for validation and testing, and each condition uses one additional execution under the reference parameters to obtain the reference trajectory.

A.5 Physical Scene Reconstruction and Object Shapes

Figure A1 shows the physical braided-rope setup and its PhysTwin reconstruction from the same side-camera viewpoint. The reconstruction includes the robot, tabletop, rope, and two clips used in

Table A1: Additional training and simulator use required for adaptation. The candidate replays for physical-parameter identification (80–240 per task) are shared by our method and direct simulator search.

Method	Policy weights	Additional training and executions	Correction updates during evaluation
Uncorrected	Frozen	None	None
Direct simulator search	Frozen	Identification; 30–144 full simulator executions for correction selection	None (fixed before evaluation)
Residual RL	Frozen; correction model trained	160 training executions with rewards in shifted-physics simulation	None (set at the start of execution)
DR+BC	New policy trained	9 human demonstrations along a known parameter axis and range	Not applicable
Ours	Frozen	Identification; 31–48 simulator executions for the surrogate (20 for training); 1 reference trajectory	Every 10 frames, without simulator calls

Table A2: Records and simulator executions per condition. Surrogate data are listed as collected executions, with training executions in parentheses.

Condition	Identification / validation records	Identification replays	Direct-search executions	Surrogate data (training)
Rope routing, mass shift	3 / 3	240	30	31 (20)
Rope routing, floor-friction shift	3 / 3	240	30	31 (20)
T-pushing	2 / 1	80	49	48 (20)
Toy packing	3 / 3	192	144	31 (20)

the physical correction experiment. This image documents the scene used for simulation; motion agreement is evaluated separately below.

The reference model for the physical experiment was obtained by fitting a PhysTwin spring–mass rope model to interaction videos of the original rope. We keep that model, including its other fitted properties, fixed and identify three simulator parameters from a failed execution with the braided rope. CMA-ES evaluates 40 parameter candidates by replaying the recorded robot actions and comparing simulated particles with rope points reconstructed from three cameras. The fitting loss is the symmetric Chamfer distance over 31 observations from 10 to 70 s at 2 s intervals, with each nearest-neighbor distance capped at 50 mm.

Table A3: Simulator parameters for the physical rope experiment. Mass and tensile-spring stiffness are multipliers relative to the PhysTwin model of the original rope; floor friction is a simulator coefficient. Chamfer errors are measured against the braided-rope failure recording used for identification.

Setting	m/m_0	k/k_0	μ_{floor}	Error (mm)
Original rope θ_0	1.000	1.000	0.544	29.9
Braided rope $\hat{\theta}$	1.111	0.586	0.524	26.5

The fitted stiffness multiplier scales resistance to stretching, not bending. Thus, the estimate is an effective simulator setting that reproduces the recorded braided-rope motion; it is not a direct measurement of the rope's bending stiffness or its ground-truth physical parameters. Replaying the uncorrected action trajectory with $\hat{\theta}$ also reproduces the physical failure, whereas the reference setting succeeds in simulation.

Figure A2 compares object configurations when recorded actions are replayed under reference, estimated, and shifted simulator parameters.

Figure A3 instead compares the planned simulation under the estimated braided-rope parameters with camera observations from the corrected physical execution. The simulator was not refitted to that execution. The rope configurations broadly agree across the selected times, although local differences remain.

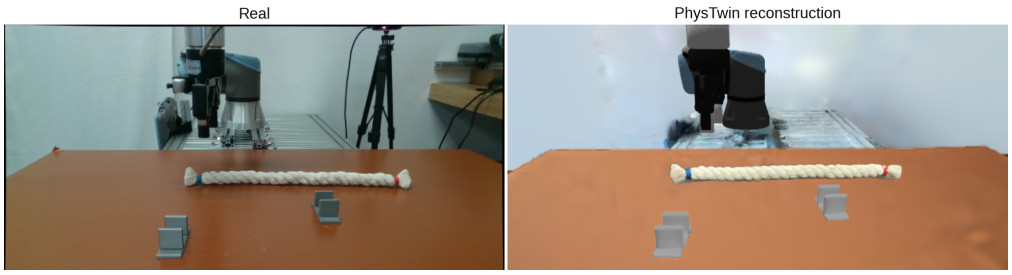


Figure A1: Physical braided-rope scene (left) and its PhysTwin reconstruction (right), viewed from the same side camera at the same time.

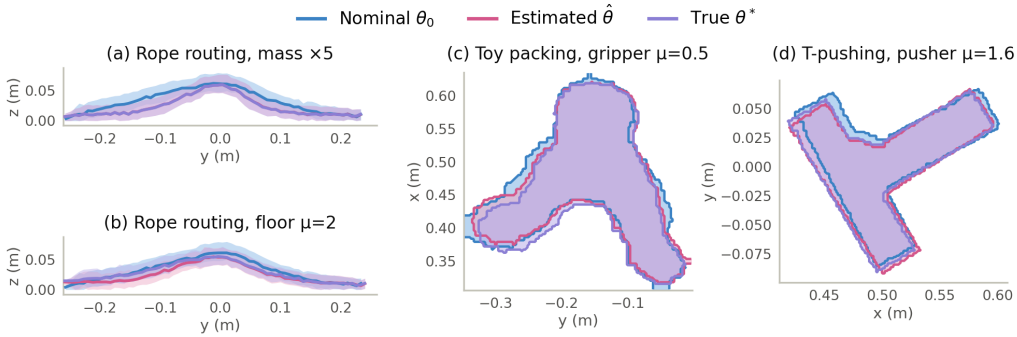


Figure A2: Object shapes under the same actions replayed with reference, estimated, and shifted parameters. (a, b) Side-view centerlines and particle extents for the rope mass and floor-friction shifts. (c, d) Top-view silhouettes for toy packing and T-pushing. Each panel shows one time point.

A.6 Gripper Signals in Correction Failures

Figure A4 shows time-varying signals from one selected execution per correction condition. The base correction with the estimated-physics surrogate keeps closing the gripper and maintains the grasp for 191 frames, compared with 188 frames in the reference-physics execution. The terminal-cost variant maintains the grasp for only 63 frames, the ground-truth-physics surrogate correction for 85 frames, and the uncorrected execution for 47 frames. In the terminal-cost variant, the gripper-opening correction approaches zero early and the contact force vanishes, consistent with a failure in which the gripper opens before the rope passes through the clip.

This pattern is not limited to the selected execution. In a separate analysis of 10 executions per condition under the same floor-friction shift, the grasp is maintained for 177–195 frames with the base correction and 41–78 frames with the terminal-cost variant. These diagnostic runs succeed in 7 and 0 of 10 executions, respectively, and are not combined with the 23-execution evaluations in the main text. Success in rope routing requires both clip passage and subsequent release. A terminal cost that sums these two terms does not specify their order and can therefore assign a low cost to candidates that release early. An additional variant that applies the release cost only when clip passage is predicted also fails in all 10 executions, so a simple ordering constraint does not resolve the problem. Because the terminal-cost variant also changes the predicted states and horizon, the figure alone does not attribute early release to a single cost term.

A.7 T-pushing Execution with a Terminal Task Cost

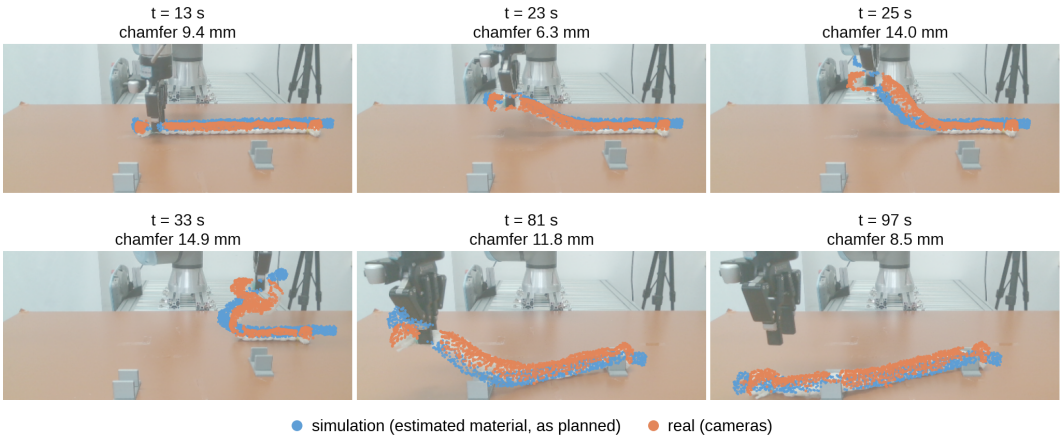


Figure A3: Rope shape during the corrected physical execution. Blue points show simulated particles under the estimated braided-rope parameters and planned correction; orange points show rope positions reconstructed from three cameras and projected onto the side-view image. Each panel reports the symmetric Chamfer distance between the two point sets at that time, with individual nearest-neighbor distances capped at 50 mm.

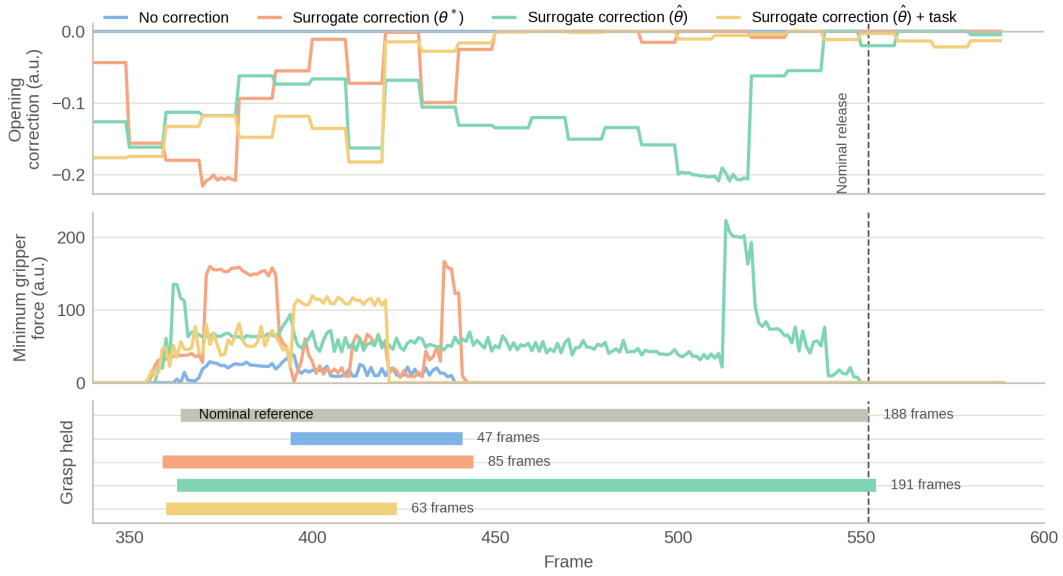


Figure A4: Early grasp release in rope routing under the floor-friction shift. Selected executions are compared for no correction, correction with a ground-truth-physics surrogate, correction with an estimated-physics surrogate, and the estimated-physics correction with an added terminal task cost. Top: gripper-opening correction (negative values close the gripper further). Middle: the smaller contact force of the two fingers. Bottom: intervals during which the grasp is maintained. Gray bars show the grasp interval of the reference-physics execution, and the dashed line marks its gripper release. Position corrections are not shown.

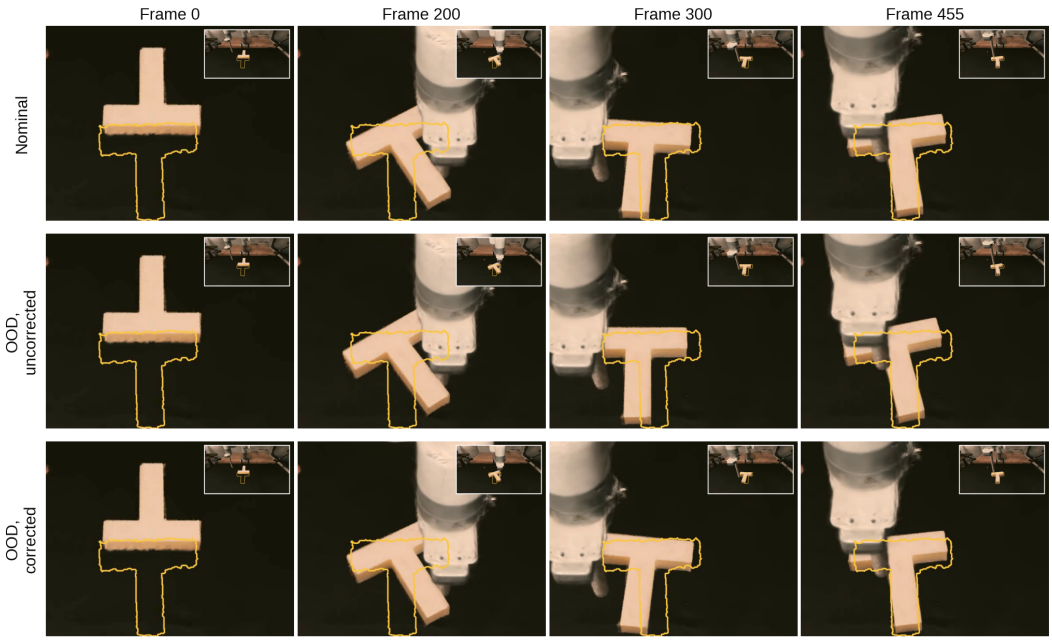


Figure A5: Selected T-pushing frames under reference physics (top), shifted pusher friction without correction (middle), and shifted friction with the terminal-cost correction (bottom). The yellow outline marks the target object configuration.