CLOSING THE LOOP ON TASK STATE: ROBUST MULTI-STAGE ROBOT MANIPULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

In multi-stage robot manipulation, local deviations can propagate to later stages. Policies learned from successful demonstrations can be data-efficient, but deployment disturbances can separate the controller's assumed task state from the physical task state. Anomaly monitors detect unreliable execution, but detection signals alone do not specify how to revise task state. Supervised failure handling requires additional data from failed rollouts or corrective supervision, with effectiveness depending on coverage of deployment-time failures. The challenge is to use runtime evidence to guide recovery without collecting additional failure data. We introduce **TSF** (**T**ask-**S**tate **F**eedback), which models valid task evolution from successful demonstrations alone. The TSF representation combines object-relative motion with interaction relations. At deployment, the inferred relations and progress regulate advancement under benign timing variations and guide recovery to a model-supported state after invalid transitions. On held-out failure types in RLBench, TSF achieves $2.44\times$ the average detection F1 of the failure-supervised GRU. With TSF monitoring fixed, full recovery achieves $2.46\times$ the intention-to-treat success rate of generic skill retry. TSF improves perturbed-task success by 65% over the same action policy without task-state feedback while satisfying aggregate nominal non-inferiority.

1 INTRODUCTION

Learning from demonstrations allows robots to acquire complex manipulation skills without online exploration. Recent behavioral-cloning methods can capture multimodal and temporally extended action distributions (Florence et al., 2022; Shafiullah et al., 2022; Chi et al., 2023; Zhao et al., 2023). Learning from demonstrations is especially attractive when robot data are scarce, but collecting demonstrations is expensive and policy performance depends heavily on demonstration coverage (Mandlekar et al., 2022; 2023). Learning efficiently from successful behavior does not ensure reliable execution. Actuation errors, imperfect contacts, or external disturbances can move the system outside the demonstrated state distribution. In a multi-stage task, a local deviation can disrupt later stages because each interaction establishes conditions for the next (Lee et al., 2022; Jain et al., 2025).

A grasp followed by object transport illustrates the need for task-state feedback. If the gripper closes without acquiring the object, the controller may still advance to transport. The resulting action can be geometrically plausible but fail to advance the task. Conversely, slower-than-expected execution may remain valid at an earlier task state. Related mismatches arise when an established interaction breaks or collaborating arms reach a shared transition asynchronously. In each case, the controller's assumed task state can diverge from the physical task state. The controller needs task-state feedback to distinguish benign timing variation from invalid transitions and guide realignment after a deviation.

Approaches to deployment robustness include training-time data augmentation and runtime monitoring. Training-time methods use expert interventions, synthesized corrections, or generated failure trajectories (Ross et al., 2011; Hoque et al., 2022; 2024; Ke et al., 2024; Duan et al., 2025). Additional supervision can improve correction or failure reasoning when the added data cover deployment-time deviations. Runtime monitors detect departures using action consistency, uncertainty, or visual evidence of task progress (Agia et al., 2025; Xu et al., 2025). A detection signal

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.

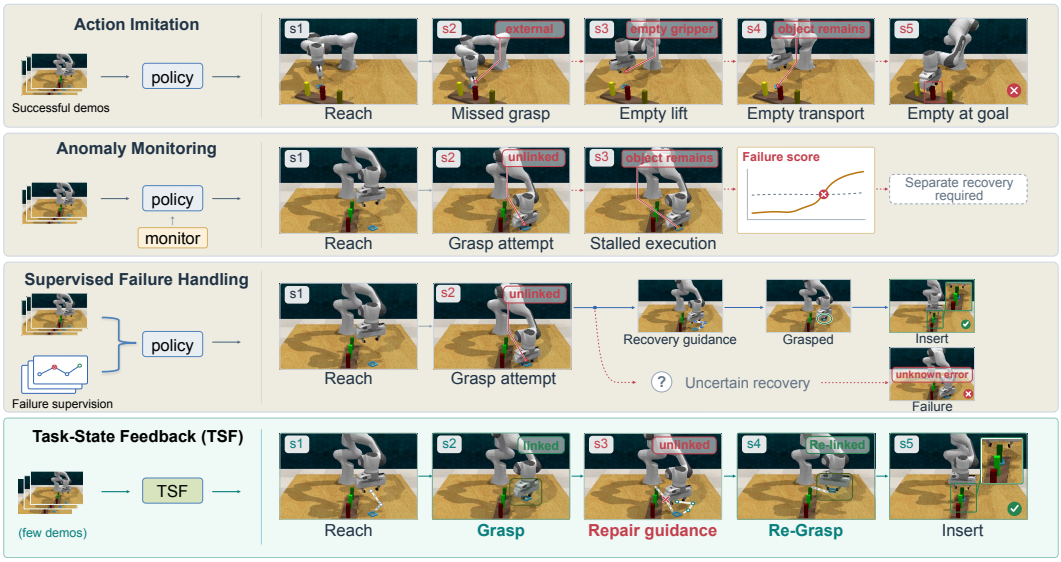


Figure 1. Comparison of action imitation, anomaly monitoring, supervised failure handling, and TSF. TSF models valid task evolution from successful demonstrations. The inferred relations and progress guide task state realignment and recovery without failure data.

alone does not specify how to revise the controller's task state or where the policy can resume. To guide progression and recovery, runtime evidence must therefore be interpreted in terms of task state.

Object-centric representations can ground task state in physical interaction. By expressing motion relative to task entities, object-centric policies adapt behavior across object configurations (Calinon, 2016; von Hartz et al., 2024; 2026a). DynaMAC extends this principle to moving objects and collaborating arms (von Hartz et al., 2026b). Object-centric representations primarily adapt actions to the current geometry without establishing whether the interaction history and current scene support the controller's assumed progress. Successful demonstrations record actions together with the evolution of object-relative motion, interaction relations, and scene configurations across valid task transitions. A model of how motion and relations evolve with the scene can therefore support task-state inference and the choice of where to resume after a deviation.

We introduce Task-State Feedback (TSF), a framework for closed-loop object-centric manipulation. From successful demonstrations, TSF models how interaction relations and task progress evolve with the physical scene. At deployment, TSF jointly infers the current relations and progress to assess whether execution remains consistent with the model. The inferred relations and progress regulate task progression under timing variations and guide repair and re-entry after a deviation. The same principle closes the execution loop for both single-arm and bimanual manipulation without requiring failure trajectories for training. Overall, our contributions are as follows.

- **Robust task-state alignment.** We formulate execution as task-state alignment, using successful demonstrations to distinguish valid timing variation from invalid transitions.
- **Structured state inference.** We jointly infer interaction relations and task progress by combining action responses with scene evidence, identifying physically supported task states.
- **Unified closed-loop execution.** We integrate relation repair and verified re-entry into TSF for one or two arms, improving perturbed success by 65% over the frozen action policy without failure trajectories.

2 Related Work

Structured representations for manipulation. Task-parameterized models adapt demonstrated motion through object-relative coordinates (Calinon, 2016; Rozo et al., 2020). Visual and entity-

centric representations extend transfer across object poses, categories, and compositions (Florence et al., 2018; Zeng et al., 2021; Shridhar et al., 2023; Ze et al., 2024; Zhu et al., 2023; Haramati et al., 2024; Wang et al., 2023; Zhu et al., 2025; Qi et al., 2025). Bimanual policies encode coupled motion, asymmetric arm roles, or joint action sequences (Silvério et al., 2015; Grannen et al., 2023; Zhao et al., 2023; Liu et al., 2025). DynaMAC treats objects and the peer end effector as moving task parameters, filtering frames through demonstrated kinematic links (von Hartz et al., 2026b). TSF extends this motion representation with progress-dependent relations and transition conditions to assess whether the observed execution supports the assumed task state.

Robust imitation under distribution shift. Behavior cloning can compound errors outside the demonstrated state distribution (Ross et al., 2011; Mandlekar et al., 2022). Interactive imitation labels learner-visited states while limiting human intervention (Celemin et al., 2022; Hoque et al., 2022). Data-generation methods expand demonstrations or synthesize corrections (Mandlekar et al., 2023; Ke et al., 2024; Zhang et al., 2024; Hoque et al., 2024). Failure-focused methods add induced failures, descriptions, or recovery trajectories (Duan et al., 2025; Dai et al., 2025). Contractive policies promote convergence between trajectories (Soleimani Abyaneh et al., 2025). We keep the action policy and the five training demonstrations fixed to isolate the effect of task-state feedback.

Task progress, monitoring, and recovery. Long-horizon policies organize behavior through segments, task machines, or skill hierarchies (Kipf et al., 2019; Liang & Boularias, 2021; Gupta et al., 2020; Di Palo & Johns, 2022; Lee et al., 2022; Mishra et al., 2023; Belkhale et al., 2023). An unsupported transition can compromise the remaining rollout (Pirk et al., 2021; Jain et al., 2025). Runtime monitors track action consistency, uncertainty, or progress (Agia et al., 2025; Xu et al., 2025). VICtoR learns progress rewards for reinforcement learning (Hung et al., 2025). Recovery methods learn convergent dynamics (Soleimani Abyaneh et al., 2025) or add recovery branches from demonstrations (Willibald & Lee, 2026). Other methods retract toward the initial pose (Dai et al., 2026) or return to verified checkpoints (Zheng et al., 2026). In TSF, detection, relation repair, and re-entry state selection use the same model of valid task evolution. The inferred relations and progress govern advancement and constrain re-entry.

3 PRELIMINARY

3.1 DEMONSTRATIONS AND TASK STATE

We consider a multi-stage manipulation task with N successful demonstrations $\mathcal{D} = \{\mathcal{T}^{(n)}\}_{n=1}^{N}$. Each demonstration records observations o_t, actions a_t, and task-relevant entity configurations. The dataset contains no failed executions, failure labels, or recovery actions. An object-centric multi-stream policy predicts actions in entity-relative frames and fuses the frame-conditioned predictions in a common action space (Calinon, 2016; von Hartz et al., 2026a;b). Segmentation partitions the demonstrations into K skills, and temporal alignment indexes progress within each skill. The aligned demonstrations define the ordered state space $\mathcal{S} = \{(k, \tau) \mid k \in \{1, \dots, K\}, \tau \in \{1, \dots, T_k\}\}$, where k identifies a skill, τ indexes progress within the skill, and T_k is the number of aligned states in skill k.

The ordered state space specifies available policy references without establishing whether the physical conditions for a queried state have been reached. The controller queries state $s_t^{\text{ref}} \in \mathcal{S}$, while $s_t \in \mathcal{S}$ denotes the latent physical progress when a normal-state explanation exists. Execution delays can cause s_t to lag behind s_t^{ref}, while early completion can create the opposite mismatch. A skill may also be queried before the required interaction or scene conditions are established. TSF must therefore assess whether execution remains valid and infer the task state supported by current physical evidence.

We assess compatibility using the task entities underlying the action policy. The frame set $\mathcal{F}$ contains the selected object, fixture, and goal frames, together with peer-end-effector frames for bimanual tasks. The world-frame poses $T_{e,t}^{W}$ and $T_{f,t}^{W}$ correspond to end effector e and frame $f \in \mathcal{F}$, respectively. The end-effector pose relative to frame f is therefore $\xi_{f,t} = (T_{f,t}^{W})^{-1} T_{e,t}^{W}$. The variable $z_{f,t} \in \{0, 1\}$ marks frame f as external to (0) or kinematically linked with (1) the acting robot. To infer $z_{f,t}$, TSF uses the motion of frame f in response to robot actions, since static contact alone does not establish kinematic linkage. For the task-state comparison, the physical configuration x_t collects the relative poses $\{\xi_{f,t}\}$ and relations $\{z_{f,t}\}$, together with scene configurations derived

from entity poses or articulation states. The relative poses and scene configurations are observed, while $z_{f,t}$ and s_t are inferred from the history $h_t = (o_{1:t}, a_{1:t-1})$.

3.2 Task-State Alignment Objective

The successful demonstrations associate each state s with compatible motion, interaction relations, and scene configurations. The candidate set $\mathcal{C}_t \subseteq \mathcal{S}$ contains states permitted by the task ordering and committed skill boundaries. For each candidate, the score $\mathcal{E}_{\mathcal{D}}(x_t, s) \in [0, 1]$ measures how well the physical configuration x_t matches state s. Maximizing the score over the permitted candidates gives the strongest available explanation

$$E_t^\star = \max_{s \in \mathcal{C}_t} \mathcal{E}_{\mathcal{D}}(x_t, s). \tag{1}$$

Execution is supported when $E_t^\star$ exceeds the threshold ϵ_{exp}. The absolute compatibility test distinguishes timing variation, which may align with a neighboring state, from a broken relation or incompatible scene that leaves every candidate unsupported.

When a supported state exists, the controller can hold, advance, or realign s_t^{ref}. Otherwise, normal progression is suspended until demonstrated support is re-established. Section 4 describes how TSF infers the required state and uses the estimate to control execution.

4 Task-State Feedback

TSF augments an object-centric action policy with a model of valid task evolution learned from successful demonstrations. Figure 2 connects offline model construction to online task-state feedback. Algorithm 1 gives the full TSF control loop. The feedback state is $B_t = (q_t, \beta_t, \mathcal{E}_t, E_t^\star)$, where q_t and β_t are relation and progress beliefs and $\mathcal{E}_t$ measures absolute compatibility. The mode m_t selects normal execution or intervention, and e_t identifies a demonstrated relation event.

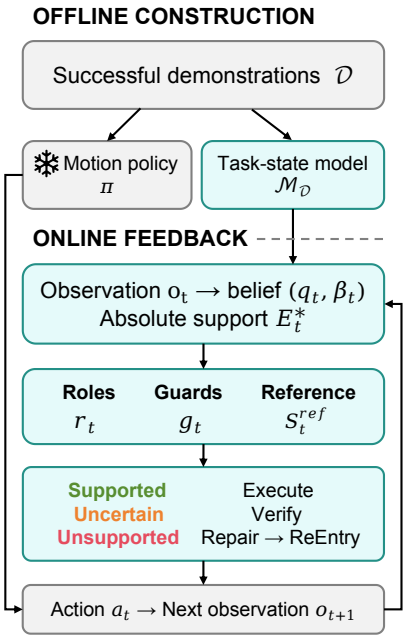


Figure 2. TSF offline construction and online task-state feedback.

Algorithm 1 Task-State Feedback

Input Successful demonstrations $\mathcal{D}$, frozen policy π, cycle limit $T_{\max}$

Construct $\mathcal{M}_{\mathcal{D}}$, initialize feedback state
for $t = 1, \ldots, T_{\max}$ **do**
Joint Task-State Inference (Section 4.1)
  if $m_{t-1} =$ Task **then** update $\bar{\beta}_t \to q_t \to \beta_t$
  else update q_t, keep $\beta_t = \beta_{t-1}$
  Evaluate $\mathcal{E}_t$, $E_t^\star \leftarrow \max_{s \in \mathcal{C}_t} \mathcal{E}_t(s)$
  $r_t \leftarrow \textsc{Roles}(\beta_t, q_t)$
  $(m_t, e_t) \leftarrow \textsc{RouteMode}(B_t, r_t)$
  $s_t^{\text{ref}} \leftarrow s_{t-1}^{\text{ref}}$, $a_t \leftarrow$ Hold
Feedback-Guided Execution (Section 4.2)
  if $m_t =$ Task **then**
    Hold, advance, or realign s_t^{ref}
    Commit guard-ready boundaries (Eq. 11)
    $a_t \leftarrow \textsc{Fuse}(\pi, r_t, s_t^{\text{ref}})$
Support-Constrained Recovery (Section 4.3)
  else if $m_t =$ Verify **then** $a_t \leftarrow \textsc{ProbeReturn}$
  else if $m_t =$ Repair **then** $a_t \leftarrow \textsc{EventStep}(e_t)$
  else if $m_t =$ Reentry **then**
    $(\mathcal{C}_t^{\text{legal}}, \mathcal{C}_t^{\text{align}}) \leftarrow \textsc{Screen}(e_t, B_t)$
    if $\mathcal{C}_t^{\text{legal}} \neq \varnothing$ **then** atomically admit $s_t^\dagger$
    else if $\mathcal{C}_t^{\text{align}} \neq \varnothing$ **then** $a_t \leftarrow \textsc{Align}(\tilde{s}_t)$
  Execute a_t and observe o_{t+1}

Output executed actions $\{a_t\}$ and feedback states $\{B_t\}$

4.1 JOINT TASK-STATE INFERENCE

Given the observation and action history $h_t = (o_{1:t}, a_{1:t-1})$, TSF maintains a relation belief $q_t^f(z)$ over $z \in \{0, 1\}$ and a progress belief $\beta_t(s)$ over $s \in \mathcal{C}_t$ (Thrun et al., 2005). The action reference s_t^{ref} is maintained separately from β_t, so a change in the progress belief does not by itself advance the controller. The same formulation extends to bimanual execution by maintaining $q_t^{i,f}$, β_t^i, and $s_t^{i,\text{ref}}$ for each arm i. Coordinated transitions additionally enforce shared relation and scene constraints. We suppress the arm index when the acting arm is clear from context.

TSF first predicts progress from the previous posterior β_{t-1} and the executed command. Before incorporating current evidence, the predicted progress prior $\bar{\beta}_t(s)$ is

$$\bar{\beta}_t(s) = \sum_{s' \in \mathcal{C}_{t-1}} p_{\text{prog}}(s \mid s', s_{t-1}^{\text{ref}}, a_{t-1}) \beta_{t-1}(s'). \tag{2}$$

The fixed kernel p_{prog} concentrates mass at the executed reference, with limited support at the immediate successor and near the previous belief for incomplete execution. Probability enters the next skill only after a verified skill transition.

The demonstration prior $\pi_{f,s}^{\text{demo}}(z)$ specifies the expected linkage of frame f at state s. Averaging with $\bar{\beta}_t(s)$ gives the weak relation prior $\pi_{f,t}^{\text{prior}}(z) = \sum_{s \in \mathcal{C}_t} \bar{\beta}_t(s) \pi_{f,s}^{\text{demo}}(z)$. TSF also propagates the previous relation posterior as $\bar{q}_t^f(z) = \sum_{z'} p_{\text{rel}}(z \mid z') q_{t-1}^f(z')$, where p_{rel} favors persistence. The evidence vector $y_{f,t}$ records the observed robot and frame motions following action a_{t-1}. The likelihood $\ell_{f,t}(z) = p(y_{f,t} \mid z, a_{t-1})$ scores the observed response under relation z. Combining the predicted relation belief $\bar{q}_t^f$ with the demonstration prior and observed response yields the relation posterior $q_t^f(z)$ as

$$q_t^f(z) = \frac{1}{Z_{f,t}^q} \bar{q}_t^f(z) \left[\left(\pi_{f,t}^{\text{prior}}(z) \right)^{\lambda_{\text{demo}}} \ell_{f,t}(z) \right]^{\omega_{f,t}}, \tag{3}$$

where $Z_{f,t}^q$ normalizes the binary belief. The parameter λ_{demo} controls the weak demonstration prior, while $\omega_{f,t} \in [0, 1]$ measures how informative the observed response is. As the current response becomes uninformative and $\omega_{f,t}$ approaches zero, the update approaches the persistent prediction $\bar{q}_t^f$. TSF accepts a new discrete relation decision only when evidence is sufficient, retaining the binary belief otherwise.

TSF then corrects $\bar{\beta}_t(s)$ using current motion and scene evidence together with the inferred relations. The scores $L_t^{\text{robot}}(s)$ and $L_t^{\text{scene}}(s)$ measure robot-motion and sparse-scene compatibility, respectively. The weight $\nu_{f,t} \in [0, 1]$ controls the usable relation evidence from frame f. We set $\nu_{f,t} = 0$ when no reliable relation decision is available. The resulting progress posterior $\beta_t(s)$ combines the prior with current compatibility evidence as

$$\begin{aligned} \beta_t(s) = \frac{1}{Z_t^\beta} \bar{\beta}_t(s) L_t^{\text{robot}}(s) \left[L_t^{\text{scene}}(s) \right]^{\lambda_{\text{scene}}} \\ \times \left[\prod_{f \in \mathcal{F}_s} \left(\sum_{z \in \{0,1\}} q_t^f(z) \pi_{f,s}^{\text{demo}}(z) \right)^{\nu_{f,t}} \right]^{\lambda_{\text{rel}}}, \qquad s \in \mathcal{C}_t, \end{aligned} \tag{4}$$

where $\mathcal{F}_s \subseteq \mathcal{F}$ contains the relation-relevant frames. The weights λ_{scene} and λ_{rel} control the two evidence families. The factor Z_t^β normalizes over $\mathcal{C}_t$.

Normalization in Equation 4 makes $\beta_t(s)$ a relative ranking, so a sharp posterior can still assign high probability to a poorly supported state. TSF therefore checks absolute compatibility with a separate score $\mathcal{E}_t(s)$, computed as

$$\mathcal{E}_t(s) = \widetilde{L}_t^{\text{robot}}(s) \left[\widetilde{L}_t^{\text{scene}}(s) \right]^{\lambda_{\text{scene}}} \left[\widetilde{L}_t^{\text{rel}}(s) \right]^{\lambda_{\text{rel}}}, \tag{5}$$

where each tilded term is scaled to an attainable maximum of one and accounts for the number and reliability of contributing factors. The score instantiates $\mathcal{E}_{\mathcal{D}}(x_t, s)$ using current observations and the inferred relation belief, without access to latent relation labels. Maximizing $\mathcal{E}_t(s)$ over $\mathcal{C}_t$ gives Equation 1. Appendix A.2 gives the exact factor construction. The controller uses β_t to select a reference and $E_t^\star$ to decide whether normal execution remains supported.

4.2 FEEDBACK-GUIDED EXECUTION

The inferred task state changes both which state is commanded and how each object-centric stream is used. For frame f, the progress belief defines the expected relation distribution $\pi^{\mathrm{ctrl}}_{f,t}(z) = \sum_{s\in\mathcal{C}_t} \beta_t(s)\pi^{\mathrm{demo}}_{f,s}(z)$. Agreement between $\pi^{\mathrm{ctrl}}_{f,t}$ and q^f_t assigns an externally anchored stream to action generation or a linked stream to relation monitoring. Reliable disagreement indicates that a required link has not formed or a required separation has not occurred. A mismatched stream becomes a recovery target when the corresponding demonstrated relation event provides an executable anchor. If evidence is insufficient, the controller can defer the relation-dependent decision. A linked stream therefore monitors the interaction without imposing a redundant action constraint on the robot that drives the stream's reference frame. Within a skill, the controller may hold s^{ref}_t, advance by one legal successor, or realign the reference with a supported earlier state.

A discrete skill transition additionally requires a sufficient absolute normal-task score and posterior mass in the demonstrated terminal window. Every required relation and boundary-scene condition must also be confirmed. The boundary conditions must persist for the duration estimated from successful demonstrations. Independent arm boundaries may commit separately. A coordinated boundary uses one shared observation to evaluate all participating arms, then commits all participating arm transitions atomically. Appendix A.3 details the boundary checks and stream roles, including how equivalent control states are grouped and coordinated transitions are committed.

4.3 SUPPORT-CONSTRAINED RECOVERY

TSF separates relation ambiguity from physical incompatibility. When motion during ordinary execution cannot resolve a required link, TSF holds task progress and applies a bounded probe followed by a return motion. The observed response updates the relation filter.

A persistently low absolute normal-task score or a reliable directional relation mismatch instead suspends normal execution. TSF repairs a persistent relation mismatch through a demonstrated LINK or UNLINK event when an executable anchor is available. A low absolute score without a relation mismatch instead triggers re-entry search. No fault-specific recovery policy is required. After any required relation repair is verified, re-entry screening checks task topology and permitted boundaries, then compatibility with current physical evidence. The set $\mathcal{C}^{\mathrm{legal}}_t$ contains candidates that pass every admission condition. Candidates in $\mathcal{C}^{\mathrm{align}}_t$ pass the non-robot conditions and have observable robot evidence, but still require geometric alignment. From $\mathcal{C}^{\mathrm{legal}}_t$ and $\mathcal{C}^{\mathrm{align}}_t$, TSF selects the re-entry state $s^\dagger_t$ and alignment target $\tilde{s}_t$, respectively,

$$s^\dagger_t = \arg\max_{s\in\mathcal{C}^{\mathrm{legal}}_t} \mathcal{E}_t(s), \qquad \tilde{s}_t = \arg\max_{s\in\mathcal{C}^{\mathrm{align}}_t} \widetilde{L}^{\mathrm{robot}}_t(s). \tag{6}$$

The state $s^\dagger_t$ is committed only after full admission. The target $\tilde{s}_t$ guides alignment without updating β_t or s^{ref}_t. After physical alignment makes the target legal, the controller jointly updates β_t and s^{ref}_t while preserving validated relation evidence. Execution thus resumes from a state supported by current physical evidence, rather than from a reset of the pre-failure execution clock.

In Algorithm 1, lines 3 to 5 update beliefs and absolute compatibility. Lines 9 to 12 regulate the reference and commit verified boundaries. During recovery, the controller holds progress until verification or re-entry permits normal execution. The loop runs for at most $T_{\max}$ cycles, although the evaluator may terminate an episode earlier. Success labels remain unavailable to TSF. Appendix A.3 specifies the operators and routing priorities.

5 EXPERIMENTS

We evaluate TSF on six single-arm RLBench tasks (James et al., 2020) and four bimanual tasks from the RLBench2 extension (Grotz et al., 2024). Controlled comparisons share a frozen DynaMAC policy trained on five successful demonstrations per task, with access to the same low-dimensional simulator states. The main comparison uses 200 paired episodes per task in each nominal or perturbed condition. Appendix B details the protocol and information boundaries.

5.1 Effect of Closing the Task-State Loop

We compare the action policy without task-state feedback, restart, three monitors followed by generic skill retry, and full TSF. All methods use paired initializations and scheduled interventions.

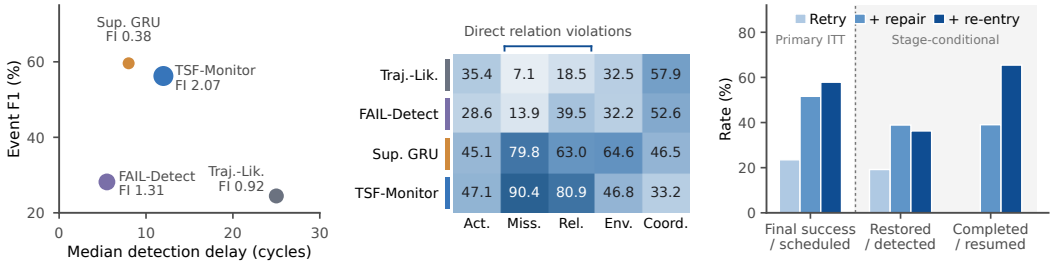
Table 1. End-to-end task success (%) over 200 paired episodes per task and condition. All methods share a frozen action policy. Sup. GRU is a three-seed mean. Row-wise best results are bold.

	DynaMAC		Restart		Traj.-Lik.		FAIL-Detect		Sup. GRU		TSF (Ours)	
Task	Nom.	Pert.	Nom.	Pert.	Nom.	Pert.	Nom.	Pert.	Nom.	Pert.	Nom.	Pert.
Single-arm												
Close Jar	86	55	**87**	54	**87**	55.5	86	56.5	85.7	54.2	82	**66.5**
Open Drawer	3.5	4	4.5	4.5	4	6.5	3.5	4	4.2	6.2	**65**	**72.5**
Insert Peg	98	37	97.5	35.5	**98.5**	35.5	98	36.5	98.2	36.3	98	**80**
Place 3 Cups	23.5	4.5	21.5	5	16	5	23	6	21.2	5.5	**42**	**30**
Stack Cups	90.5	38.5	**91.5**	38	**91.5**	38	90.5	39	91	38.3	91	**65**
Sweep to Dustpan	**100**	58	**100**	58	**100**	58	**100**	58	**100**	59.5	**100**	**77**
Bimanual												
Handover Item	97.5	66	**99**	68	98.5	63	95.5	64.5	95.3	62.7	92.5	**87**
Lift Tray	56.5	39	56.5	38.5	**58**	39	55.5	36	57.5	37.3	57.5	**41.5**
Bim. Sweep	97	57.5	97.5	59.5	96.5	59.5	97.5	62.5	**97.7**	65.5	97.5	**80.5**
Bottle in Fridge	91.5	42.5	90	40.5	**92**	43	89.5	41	91.8	43.2	90	**63**
Task macro avg.	74.4	40.2	74.5	40.1	74.2	40.3	73.9	40.4	74.2	40.9	**81.5**	**66.3**

TSF leads on every perturbed task (Table 1), with the largest gains over the frozen policy on Open Drawer and Insert Peg and only a modest gain on Lift Tray. Restart and monitor-plus-retry remain close to the frozen-policy baseline, indicating limited benefit from detection followed by repetition. Nominal non-inferiority holds in aggregate, although TSF has lower point estimates on Close Jar and Handover Item.

5.2 Failure Monitoring and Recovery

To evaluate monitoring separately from recovery, all monitors score identical frozen-policy trajectories without intervening. Event-level F1 counts each audited physical violation once. We then fix TSF inference and compare generic retry with relation repair, either continuing from the same state or selecting a legal re-entry state.



(a) Detection trade-off **(b)** Fault-family F1 **(c)** Recovery outcomes

Figure 3. Monitoring and recovery. (a) event F1 versus delay. Labels give false interventions per 1000 nominal cycles. (b) event F1 by fault family. Sup. GRU is a three-seed mean trained with 200 labeled failure trajectories per task. (c) paired final task success on the same 800 scheduled episodes per method. Stage-conditional rates report restoration among detected episodes and completion among resumed or re-entered episodes.

TSF-Monitor exceeds both unsupervised monitors in event F1 but remains below the supervised GRU overall (Figures 3a and 3b). TSF detects missed interactions and relation loss more reliably

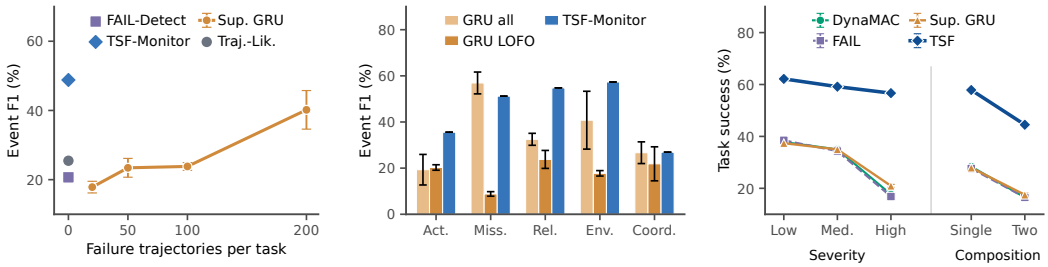
than coordination delays, so sensitivity varies with the type of mismatch. Higher recall also comes with more false interventions and longer detection delay than the GRU.

At a shared calibration target of one false intervention per 1000 nominal cycles, TSF again detects more events than the GRU but incurs more false interventions on test trajectories. Calibration matches the target rate, not the realized test cost (Appendix C.2).

With TSF inference fixed, relation repair improves task success over generic retry, and legal re-entry adds a further gain (Figure 3c). Choosing where to resume therefore matters beyond restoring the physical relation. Stage-conditional rates use different denominators and cannot isolate component-level causal effects. Appendix C.2 reports the common-trigger comparison and audited recovery paths.

5.3 Generalization to Unseen Failure Conditions

On the four-task stress subset, we vary the GRU's failure-data budget and withhold individual failure families. All monitors score the same trajectories without affecting execution. We then evaluate closed-loop task success under stronger disturbances and two-event compositions.



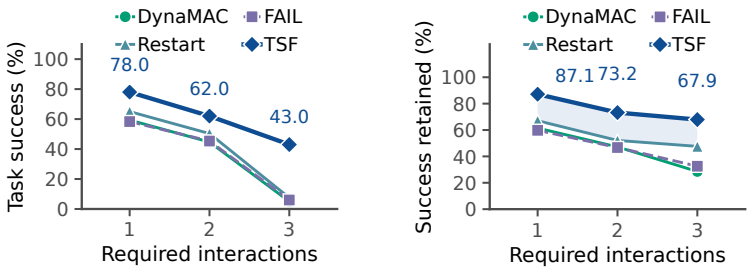
(a) Failure-supervision budget **(b)** Held-out failure families **(c)** Harder test conditions

Figure 4. Generalization beyond enumerated failures. (a) failure-data budget. (b) leave-one-family-out monitoring. (c) task success under increased severity and two-event compositions. GRU error bars are mean ± standard deviation over three seeds. TSF uses no failure trajectories.

On this four-task subset, additional failure labels improve GRU event F1, but TSF retains an advantage without failure trajectories (Figure 4a). The all-family GRU exceeds TSF on missed interactions, but withholding that family reverses the ranking (Figure 4b). The reversal shows how the GRU depends on training coverage of each failure family. TSF instead tests consistency with successful demonstrations. At a shared calibration target, the higher recall still comes with more false interventions on test trajectories.

Greater severity and two-event compositions reduce TSF success, but TSF remains above the detector-plus-retry baselines in every condition (Figure 4c). The larger decline under composed events shows that recovery across multiple disturbances remains difficult.

5.4 Robustness over Longer Task Horizons



(a) Success under per-stage events **(b)** Retention of nominal success

Figure 5. Long-horizon task-macro success and perturbed-to-nominal retention.

We test repeat-count variants of Place Cups, Push Buttons, and Remove Cups with up to three interactions and per-stage perturbations. TSF has the highest perturbed success at every tested horizon, with the largest advantage at three interactions (Figure 5a). Nominal success also differs across methods, so absolute success alone cannot separate task difficulty from perturbation robustness.

Figure 5b therefore averages the perturbed-to-nominal success ratio within each task. TSF retains more nominal success at every horizon, although retention declines for all methods. The advantage thus persists after accounting for nominal performance. Appendix C.4 gives the protocol and full results.

Native-system comparisons. We also compare complete TSF, RVT, and RACER systems (Goyal et al., 2023; Dai et al., 2025), each with the native policy, observations, and training supervision. TSF has the highest absolute success, while RACER has the highest per-task retention. Appendix C.5 gives the matching protocol and results.

5.5 Ablation Study

We test four variants on four tasks. Motion-only task state removes relation and scene compatibility from progress correction and re-entry scoring. Open-loop progress removes posterior correction and guarded advancement, while generic retry replaces relation repair and legal re-entry. Nearest-state inference replaces the coupled belief with the nearest demonstrated state and direct relation thresholding, retaining the same boundary guards and recovery interface.

Table 2. Ablation task-macro success (%) on the four-task subset. Recovery is final task success among physically triggered episodes. All variants share the policy, episodes, and control budget.

Variant	Nominal	Perturbed	Bimanual perturbed	Recovery
TSF (Ours)	**70.0**	**57.9**	**63.8**	**67.9**
Nearest-state inference	22.5	21.5	0.5	31.6
Motion-only task state	46.5	22.4	37.5	25.2
Open-loop task progress	66	52	44	55.3
Generic retry after detection	40.3	23.5	43.3	27.1

Full TSF leads across all four metrics in Table 2. Open-loop progress is the closest variant overall, but the larger bimanual gap is consistent with a greater need for online alignment at shared transitions. Removing relation and scene compatibility weakens performance more broadly. Generic retry also loses much of the recovery benefit, linking reliable execution to the response after detection.

Nearest-state inference performs poorly in both conditions, especially on bimanual tasks, although Open Drawer shows no significant difference between estimators. The comparison supports coupled inference over this particular heuristic in nominal and perturbed execution, without isolating a perturbation-specific benefit. Appendix C.6 reports paired tests, per-task results, infrastructure-error sensitivity, and recovery paths.

6 Conclusion and Limitations

We presented TSF, which learns valid task evolution from successful demonstrations to guide progression and recovery. Across the evaluated single-arm and bimanual tasks, task-state feedback improves robustness to perturbations while meeting nominal non-inferiority. The results show how successful demonstrations can support recovery without failure or recovery training data.

The current formulation uses segmented demonstrations and low-dimensional entity states, with binary kinematic relations and recovery within demonstrated support. Richer interaction models and broader demonstration coverage could extend the range of recoverable states. Beyond the present DynaMAC evaluation in RLBench/CoppeliaSim, future work could test other policy backbones and account for perception uncertainty. Hardware studies would then assess recovery and safety during physical interaction.

AI USE STATEMENT

We used generative AI tools for translation and manuscript editing, including shortening and formatting checks. AI assistance also supported literature search and organization, interpretation of research results, suggestions for experimental parameters, and diagnosis of code errors.

The authors reviewed all AI-assisted work. Literature suggestions were checked against the original publications and bibliographic records, and translations were compared with the source text and revised for technical accuracy. The authors selected experimental parameters and evaluated the suggestions through controlled experiments. Code diagnoses were checked by inspection, issue reproduction, and relevant tests. Interpretations were verified against the original machine-readable results and statistical analyses. Formatting suggestions were checked against the official ICLR 2027 template and author guidelines, followed by manual manuscript inspection. The authors take responsibility for all final content.

ETHICS STATEMENT

The present study uses simulated robot manipulation tasks and does not involve human subjects or private personal data. Potential downstream risks include unsafe deployment of recovery behavior on physical robots without independent safety monitoring. The method should therefore be validated with hardware safety limits, collision monitoring, and human oversight before real-world deployment.

REPRODUCIBILITY STATEMENT

Sections 4 and 5 define the method and controlled comparisons. Appendices A and B describe task-model construction, online inference and execution, and the task and data splits. The appendices also specify perturbations, baselines, metrics, software settings, and hyperparameters. Appendix C provides the complete numerical analyses underlying the main results. The anonymized supplementary archive contains the TSF implementation and pinned configurations. A runnable two-stage demonstration constructs a task model from five packaged successful demonstrations, then executes a seeded physically perturbed rollout. The archive also includes paired execution videos and the final derived numerical inputs. An accompanying script verifies the reported tables and regenerates the supplied figures.

REFERENCES

Christopher Agia, Rohan Sinha, Jingyun Yang, Ziang Cao, Rika Antonova, Marco Pavone, and Jeannette Bohg. Unpacking failure modes of generative policies: Runtime monitoring of consistency and progress. In *Proceedings of the 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 689–723. PMLR, 2025. URL https://proceedings.mlr.press/v270/agia25a.html.

Anastasios N. Angelopoulos and Stephen Bates. Conformal prediction: A gentle introduction. *Foundations and Trends in Machine Learning*, 16(4):494–591, 2023. doi: 10.1561/2200000101. URL https://mlanthology.org/ftml/2023/angelopoulos2023ftml-conformal/.

Suneel Belkhale, Yuchen Cui, and Dorsa Sadigh. HYDRA: Hybrid robot actions for imitation learning. In *Proceedings of the 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2113–2133. PMLR, 2023. URL https://proceedings.mlr.press/v229/belkhale23a.html.

Sylvain Calinon. A tutorial on task-parameterized movement learning and retrieval. *Intelligent Service Robotics*, 9(1):1–29, 2016. doi: 10.1007/s11370-015-0187-9.

Carlos Celemin, Rodrigo Pérez-Dattari, Eugenio Chisari, Giovanni Franzese, Leandro de Souza Rosa, Ravi Prakash, Zlatan Ajanović, Marta Ferraz, Abhinav Valada, and Jens Kober. Interactive imitation learning in robotics: A survey. *Foundations and Trends in Robotics*, 10(1–2): 1–197, 2022. doi: 10.1561/2300000072.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.026. URL https://roboticsproceedings.org/rss19/p026.html.

Tingjun Dai, Mingfei Han, Tingwen Du, Zhiheng Liu, Zihao Zhang, Zhihui Li, Salman Khan, Jun Yu, and Xiaojun Chang. See, plan, rewind: Progress-aware vision-language-action models for robust robotic manipulation. *arXiv preprint arXiv:2603.09292*, 2026. doi: 10.48550/arXiv.2603.09292. URL https://arxiv.org/abs/2603.09292.

Yinpei Dai, Jayjun Lee, Nima Fazeli, and Joyce Chai. RACER: Rich language-guided failure recovery policies for imitation learning. In *IEEE International Conference on Robotics and Automation*, pp. 15657–15664, 2025. doi: 10.1109/ICRA55743.2025.11127799. URL https://arxiv.org/abs/2409.14674.

Norman Di Palo and Edward Johns. Learning multi-stage tasks with one demonstration via self-replay. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 1180–1189. PMLR, 2022. URL https://proceedings.mlr.press/v164/palo22a.html.

Jiafei Duan, Wilbert Pumacay, Nishanth Kumar, Yi Ru Wang, Shulin Tian, Wentao Yuan, Ranjay Krishna, Dieter Fox, Ajay Mandlekar, and Yijie Guo. AHA: A vision-language model for detecting and reasoning over failures in robotic manipulation. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/70a06501001e1820fd1eb9ee821302d2-Abstract-Conference.html.

Bradley Efron. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1): 1–26, 1979. doi: 10.1214/aos/1176344552.

Pete Florence, Corey Lynch, Andy Zeng, Oscar A. Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 158–168. PMLR, 2022. URL https://proceedings.mlr.press/v164/florence22a.html.

Peter Florence, Lucas Manuelli, and Russ Tedrake. Dense object nets: Learning dense visual object descriptors by and for robotic manipulation. In *Proceedings of the 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pp. 373–385. PMLR, 2018. URL https://proceedings.mlr.press/v87/florence18a.html.

Ankit Goyal, Jie Xu, Yijie Guo, Valts Blukis, Yu-Wei Chao, and Dieter Fox. RVT: Robotic view transformer for 3D object manipulation. In *Proceedings of the 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 694–710. PMLR, 2023. URL https://proceedings.mlr.press/v229/goyal23a.html.

Jennifer Grannen, Yilin Wu, Brandon Vu, and Dorsa Sadigh. Stabilize to act: Learning to coordinate for bimanual manipulation. In *Proceedings of the 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 563–576. PMLR, 2023. URL https://proceedings.mlr.press/v229/grannen23a.html.

Markus Grotz, Mohit Shridhar, Yu-Wei Chao, Tamim Asfour, and Dieter Fox. PerAct^2: Benchmarking and learning for robotic bimanual manipulation tasks. In *CoRL 2024 Workshop on Whole-body Control and Bimanual Manipulation: Applications in Humanoids and Beyond*, 2024. URL https://openreview.net/forum?id=nIU0ZFmptX.

Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 1025–1037. PMLR, 2020. URL https://proceedings.mlr.press/v100/gupta20a.html.

Dan Haramati, Tal Daniel, and Aviv Tamar. Entity-centric reinforcement learning for object manipulation from pixels. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/804dbf8d3b8eee1ef875c6857efc64eb-Abstract-Conference.html.

Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002. doi: 10.1162/089976602760128018.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979. URL https://www.jstor.org/stable/4615733.

Ryan Hoque, Ashwin Balakrishna, Ellen Novoseller, Albert Wilcox, Daniel S. Brown, and Ken Goldberg. ThriftyDAgger: Budget-aware novelty and risk gating for interactive imitation learning. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 598–608. PMLR, 2022. URL https://proceedings.mlr.press/v164/hoque22a.html.

Ryan Hoque, Ajay Mandlekar, Caelan Garrett, Ken Goldberg, and Dieter Fox. IntervenGen: Interventional data generation for robust and data-efficient robot imitation learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2840–2846, 2024. URL https://arxiv.org/abs/2405.01472.

Kuo-Han Hung, Pang-Chi Lo, Jia-Fong Yeh, Han-Yuan Hsu, Yi-Ting Chen, and Winston H. Hsu. VICtoR: Learning hierarchical vision-instruction correlation rewards for long-horizon manipulation. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/2c74f005aabbf90a8f1747d99f387321-Abstract-Conference.html.

Shivam Jain, Sachit Sachdeva, and Rohan Paul. Enabling long(er) horizon imitation for manipulation tasks by modeling subgoal transitions. In *Proceedings of the 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 2219–2238. PMLR, 2025. URL https://proceedings.mlr.press/v305/jain25b.html.

Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J. Davison. RLBench: The robot learning benchmark and learning environment. *IEEE Robotics and Automation Letters*, 5(2): 3019–3026, 2020. doi: 10.1109/LRA.2020.2974707.

Liyiming Ke, Yunchu Zhang, Abhay Deshpande, Siddhartha Srinivasa, and Abhishek Gupta. CCIL: Continuity-based data augmentation for corrective imitation learning. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/99831104028c3b7e6079fd8bdcc42c8f-Abstract-Conference.html.

Thomas Kipf, Yujia Li, Hanjun Dai, Vinicius Zambaldi, Alvaro Sanchez-Gonzalez, Edward Grefenstette, Pushmeet Kohli, and Peter Battaglia. CompILE: Compositional imitation learning and execution. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3418–3428. PMLR, 2019. URL https://proceedings.mlr.press/v97/kipf19a.html.

Youngwoon Lee, Joseph J. Lim, Anima Anandkumar, and Yuke Zhu. Adversarial skill chaining for long-horizon robot manipulation via terminal state regularization. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 406–416. PMLR, 2022. URL https://proceedings.mlr.press/v164/lee22a.html.

Junchi Liang and Abdeslam Boularias. Self-supervised learning of long-horizon manipulation tasks with finite-state task machines. In *Proceedings of the 3rd Conference on Learning for Dynamics and Control*, volume 144 of *Proceedings of Machine Learning Research*, pp. 484–497. PMLR, 2021. URL https://proceedings.mlr.press/v144/liang21a.html.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. RDT-1B: A diffusion foundation model for bimanual manipulation. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/49f80e4d2471ad4f2edf4f5f1ab62339-Abstract-Conference.html.

Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 1678–1690. PMLR, 2022. URL https://proceedings.mlr.press/v164/mandlekar22a.html.

Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. MimicGen: A data generation system for scalable robot learning using human demonstrations. In *Proceedings of the 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 1820–1864. PMLR, 2023. URL https://proceedings.mlr.press/v229/mandlekar23a.html.

Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, 1947. doi: 10.1007/BF02295996.

Utkarsh Aashu Mishra, Shangjie Xue, Yongxin Chen, and Danfei Xu. Generative skill chaining: Long-horizon skill planning with diffusion models. In *Proceedings of the 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2905–2925. PMLR, 2023. URL https://proceedings.mlr.press/v229/mishra23a.html.

Soeren Pirk, Karol Hausman, Alexander Toshev, and Mohi Khansari. Modeling long-horizon tasks as sequential interaction landscapes. In *Proceedings of the 4th Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pp. 471–484. PMLR, 2021. URL https://proceedings.mlr.press/v155/pirk21a.html.

Carl Qi, Dan Haramati, Tal Daniel, Aviv Tamar, and Amy Zhang. EC-Diffuser: Multi-object manipulation via entity-centric behavior generation. In *International Conference on Learning Representations, ICLR*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/ba2e53000899e45e6018f639cb7469fa-Abstract-Conference.html.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 627–635, 2011.

Leonel Rozo, Meng Guo, Andras G. Kupcsik, Marco Todescato, Philipp Schillinger, Markus Giftthaler, Matthias Ochs, Markus Spies, Nicolai Waniek, Patrick Kesper, and Mathias Bürger. Learning and sequencing of object-centric manipulation skills for industrial tasks. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 9072–9079, 2020. doi: 10.1109/IROS45743.2020.9341570.

Takaya Saito and Marc Rehmsmeier. The precision–recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PLOS ONE*, 10(3):e0118432, 2015. doi: 10.1371/journal.pone.0118432.

Nur Muhammad Mahi Shafiullah, Zichen Jeff Cui, Ariuntuya Altanzaya, and Lerrel Pinto. Behavior transformers: Cloning k modes with one stone. In *Advances in Neural Information Processing Systems*, volume 35, pp. 22955–22968, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/90d17e882adbdda42349db6f50123817-Abstract-Conference.html.

Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Proceedings of the 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 785–799. PMLR, 2023. URL https://proceedings.mlr.press/v205/shridhar23a.html.

João Silvério, Leonel Rozo, Sylvain Calinon, and Darwin G. Caldwell. Learning bimanual end-effector poses from demonstrations using task-parameterized dynamical systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 464–470, 2015. doi: 10.1109/IROS.2015.7353413.

Amin Soleimani Abyaneh, Mahrokh Boroujeni, Hsiu-Chin Lin, and Giancarlo Ferrari-Trecate. Contractive dynamical imitation policies for efficient out-of-sample recovery. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=lILEtkWOXD.

Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. MIT Press, 2005. URL https://mitpress.mit.edu/9780262201629/probabilistic-robotics/.

Jan Ole von Hartz, Tim Welschehold, Abhinav Valada, and Joschka Boedecker. The art of imitation: Learning long-horizon manipulation tasks from few demonstrations. *IEEE Robotics and Automation Letters*, 9(12):11369–11376, 2024. doi: 10.1109/LRA.2024.3487506. URL https://arxiv.org/abs/2407.13432.

Jan Ole von Hartz, Adrian Röfer, Joschka Boedecker, and Abhinav Valada. The unreasonable effectiveness of discrete-time Gaussian process mixtures for robot policy learning. *IEEE Transactions on Robotics*, 42:2419–2438, 2026a. doi: 10.1109/TRO.2026.3697167. URL https://arxiv.org/abs/2505.03296.

Jan Ole von Hartz, Abhinav Valada, and Joschka Boedecker. One hand watches the other: Dynamic multi-agent cooperation for sample-efficient bimanual manipulation in dynamic environments. *arXiv preprint arXiv:2607.22119*, 2026b. URL https://arxiv.org/abs/2607.22119.

Renhao Wang, Jiayuan Mao, Joy Hsu, Hang Zhao, Jiajun Wu, and Yang Gao. Programmatically grounded, compositionally generalizable robotic manipulation. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=rZ-wylY5VI.

Christoph Willibald and Dongheui Lee. Hierarchical task decomposition for execution monitoring and error recovery: Understanding the rationale behind task demonstrations. *The International Journal of Robotics Research*, 45(3):369–396, 2026. doi: 10.1177/02783649251352112.

Edwin B. Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212, 1927. doi: 10.1080/01621459.1927.10502953.

Chen Xu, Tony Khuong Nguyen, Emma Dixon, Christopher Rodriguez, Patrick Miller, Robert Lee, Paarth Shah, Rares Andrei Ambrus, Haruki Nishimura, and Masha Itkina. Can we detect failures without failure data? uncertainty-aware runtime failure detection for imitation learning policies. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.073. URL https://www.roboticsproceedings.org/rss21/p073.html.

Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3D Diffusion Policy: Generalizable visuomotor policy learning via simple 3D representations. In *Proceedings of Robotics: Science and Systems*, 2024. doi: 10.15607/RSS.2024.XX.067. URL https://roboticsproceedings.org/rss20/p067.html.

Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, and Johnny Lee. Transporter networks: Rearranging the visual world for robotic manipulation. In *Proceedings of the 4th Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pp. 726–747. PMLR, 2021. URL https://proceedings.mlr.press/v155/zeng21a.html.

Xiaoyu Zhang, Matthew Chang, Pranav Kumar, and Saurabh Gupta. Diffusion meets DAgger: Supercharging eye-in-hand imitation learning. In *Proceedings of Robotics: Science and Systems*, 2024. doi: 10.15607/RSS.2024.XX.048. URL https://roboticsproceedings.org/rss20/p048.html.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.016. URL https://roboticsproceedings.org/rss19/p016.html.

Gehan Zheng, Sanjay Seenivasan, Matthew Johnson-Roberson, and Weiming Zhi. Rewind-IL: Online failure detection and state respawning for imitation learning. *arXiv preprint arXiv:2604.16683*, 2026. doi: 10.48550/arXiv.2604.16683. URL https://arxiv.org/abs/2604.16683.

Junzhe Zhu, Yuanchen Ju, Junyi Zhang, Muhan Wang, Zhecheng Yuan, Kaizhe Hu, and Huazhe Xu. DenseMatcher: Learning 3D semantic correspondence for category-level manipulation from a single demo. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/7ba6b5b03ae07151a9a353b51f943290-Abstract-Conference.html.

Yifeng Zhu, Zhenyu Jiang, Peter Stone, and Yuke Zhu. Learning generalizable manipulation policies with object-centric 3D representations. In *Proceedings of the 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 3418–3433. PMLR, 2023. URL https://proceedings.mlr.press/v229/zhu23b.html.

APPENDIX

TABLE OF CONTENTS

A Additional Method Details **16**
A.1 Task-State Model Construction 16
A.2 Relation and Progress Inference 16
A.3 Closed-Loop Execution Algorithm 17

B Experimental Setup and Implementation **19**
B.1 Tasks and Demonstrations 19
B.2 Observation Interface and Information Boundaries 19
B.3 Perturbation Protocol 21
B.4 Baselines and Controlled Recovery Interface 22
B.5 Native-System Matching Protocol 23
B.6 Metrics and Statistical Analysis 23
B.7 Software, Models, and Hyperparameters 24

C Additional Experimental Results **24**
C.1 Controlled-Comparison Statistics 24
C.2 Detection and Recovery Results 25
C.3 Generalization to Unseen Failure Conditions 27
C.4 Robustness over Longer Task Horizons 28
C.5 Native-System Results 29
C.6 Ablation Results 29
C.7 Recovery-Path Analysis 31

D Task Gallery **32**

A Additional Method Details

The frozen DynaMAC policy represents end-effector motion in task-relative frames and fuses the active frame-conditioned distributions as a product of experts (Calinon, 2016; Hinton, 2002; von Hartz et al., 2024; 2026a;b). The following subsections describe how TSF builds a task-state model on this representation and uses online inference to guide execution and recovery.

A.1 Task-State Model Construction

Motion and frame relevance. Candidate entity frames are evaluated by the predictive contribution of each frame relative to a task-relative baseline, using leave-one-demonstration-out validation. Frames with a consistently detrimental contribution are excluded from action fusion. Geometrically useful frames remain available to the corresponding task-state components, as do frames carrying relation semantics. A peer-end-effector stream additionally requires a stable, nonredundant geometric dependency or participation in a demonstrated interaction path. Synchronization alone is insufficient for retaining such a stream. Within each task state, the motion model marginalizes over alternative motion modes until a mode is selected.

Relation priors and events. For each frame f and task state s, cross-demonstration dispersion of the local end-effector pose provides a smoothed empirical relation prior $\pi^{\mathrm{demo}}_{f,s}$. Action-induced comotion evidence refines this prior when the demonstrations contain sufficient excitation. A persistent change from external to linked creates a LINK event and the converse an UNLINK event. Repeated closing behavior may identify a candidate link without enough post-contact motion to establish comotion. TSF records a LINK-PENDING event together with the candidate state and local trajectory. A transition that depends on the pending link requires active verification before committing.

Sparse scene factors. The candidate pool contains task-entity poses and pairwise relative configurations within a bounded neighborhood. A retained scene factor must be stable within a skill and supported across demonstrations. Retention also requires better recognition of the corresponding state neighborhood under leave-one-demonstration-out validation. For each retained continuous factor j, we compare the current configuration with candidate state s using a peak-normalized score $c_{j,t}(s)$. The scene score $L^{\mathrm{scene}}_t(s)$ combines the retained factors as

$$c_{j,t}(s) = \exp\left[-\tfrac{1}{2} r_{j,t}(s)^\top \Sigma_{j,s}^{-1} r_{j,t}(s)\right], \qquad L^{\mathrm{scene}}_t(s) = \prod_{j \in \mathcal{J}_s} c_{j,t}(s)^{\rho_{j,t}}, \tag{7}$$

where $r_{j,t}(s)$ is the residual from state s in factor j's local coordinates and $\Sigma_{j,s}$ is the corresponding covariance. The set $\mathcal{J}_s$ indexes the retained factors, and $\rho_{j,t} \in [0,1]$ measures observation reliability for factor j. For $L^{\mathrm{robot}}_t(s)$, executable streams are transformed and fused in the world frame before scoring the observed end-effector pose. Linked streams contribute relative-pose monitoring terms. The resulting bounded terms enter TSF as state-conditional compatibility factors.

Boundary conditions. For transition $b : k \to k+1$, successful terminal windows immediately preceding the first target-skill action define the source-skill completion region and required physical conditions. The windows also identify participating arms, required relation directions, and stable scene factors. Thresholds are derived from the successful boundary windows with a fixed global margin. The observed terminal dwell determines how long the boundary conditions must persist before a skill transition.

A.2 Relation and Progress Inference

Action-conditioned relation evidence. To evaluate the response to the preceding action, TSF compares the realized end-effector motion $\Delta T^W_{e,t}$ with the frame motion $\Delta T^W_{f,t}$. The relative-pose change is $\Delta\xi_{f,t}$, with translation and rotation expressed in a common scaled tangent representation. With sufficient end-effector excitation, linkage evidence increases when the frame follows the end effector with a small relative-pose residual. Independent frame motion instead supports an external relation. The likelihood contribution is tempered by command-tracking reliability, frame visibility, and an outlier mixture. Cycles below the excitation or reliability threshold update neither direction strongly. The persistent prior in Equation 3 then carries the latest informative evidence. A new linked/external decision is accepted only when posterior probability and entropy pass the shared

confidence rule. An uninformative cycle may retain the latest reliable decision while the binary posterior continues to evolve.

Candidate compatibility and expansion. The nominal progress kernel proposes candidates in a local backward neighborhood and at the executed reference and legal successors. Relation events may add nearby candidates to explain a newly formed or broken link. Candidates across a skill boundary remain unavailable until the guarded transition commits. For each candidate s, we compute the peak-normalized relation overlap $\kappa_{f,t}(s)$ and apply the binary gate $\delta_{f,t}(s)$ to obtain the direction-aware score $c^{\mathrm{state}}_{f,t}(s)$,

$$\kappa_{f,t}(s) = \frac{\sum_z q^f_t(z)\pi^{\mathrm{demo}}_{f,s}(z)}{\max_z \pi^{\mathrm{demo}}_{f,s}(z)}, \qquad c^{\mathrm{state}}_{f,t}(s) = \kappa_{f,t}(s)\delta_{f,t}(s), \tag{8}$$

where the numerator of $\kappa_{f,t}(s)$ is the raw overlap used in the progress posterior. Event expansion and recovery re-entry use the gated score $c^{\mathrm{state}}_{f,t}(s)$. The gate $\delta_{f,t}(s) \in \{0,1\}$ is zero when a reliable online decision disagrees with the unique dominant relation under $\pi^{\mathrm{demo}}_{f,s}$. Otherwise, $\delta_{f,t}(s) = 1$. Directional agreement is therefore required for event-based expansion and re-entry. An unresolved pending event retains the soft relational overlap.

Normalizing the product in Equation 4 over $\mathcal{C}_t$ gives the progress posterior β_t. Absolute compatibility instead requires scores that remain comparable across different numbers of reliable factors. We therefore combine positive factors u_n with reliability weights $w_n \geq 0$ using the weighted geometric mean

$$\mathrm{GM}(\boldsymbol{u};\boldsymbol{w}) = \begin{cases} \exp\left(\dfrac{\sum_n w_n \log u_n}{\sum_n w_n}\right), & \sum_n w_n > 0, \\ 1, & \sum_n w_n = 0. \end{cases} \tag{9}$$

Applying the weighted geometric mean to scene compatibility and relation overlap gives the peak-normalized terms in Equation 5,

$$\begin{aligned} \widetilde{L}^{\mathrm{scene}}_t(s) &= \mathrm{GM}\big((c_{j,t}(s))_{j\in\mathcal{J}_s};(\rho_{j,t})_{j\in\mathcal{J}_s}\big), \\ \widetilde{L}^{\mathrm{rel}}_t(s) &= \mathrm{GM}\big((\kappa_{f,t}(s))_{f\in\mathcal{F}_s};(\nu_{f,t})_{f\in\mathcal{F}_s}\big). \end{aligned} \tag{10}$$

The definition assigns the neutral value one to an evidence family with no reliable observation. The robot term $\widetilde{L}^{\mathrm{robot}}_t$ analogously divides the fused robot-configuration score by the attainable peak of that score. The posterior ranks legal candidates. The quantity $E^\star_t = \max_{s\in\mathcal{C}_t} \mathcal{E}_t(s)$ tests whether any current candidate has sufficient absolute compatibility. A mismatch is declared only after $E^\star_t$ remains below ϵ_{exp} for the configured persistence period.

Control equivalence. Weakly identifiable neighboring states can split posterior mass despite implying the same world-frame command. For the control decision only, we aggregate mass over states that agree in active streams, expected relation directions, stream roles, gripper action, and world-frame action compatibility. Each equivalence set is confined to one skill segment bounded by action and relation events. The original per-state posterior and diagnostic label are preserved.

A.3 Closed-Loop Execution Algorithm

Boundary guards and transactions. The boundary criterion combines progress completion with relation and scene conditions specific to the transition. Because β^i_t already incorporates robot, relation, and scene evidence, posterior mass in the terminal window measures local completion. The set $\mathcal{W}_{i,b}$ contains the terminal states of arm i for boundary b. Each tuple $(i, f, z^\star) \in \mathcal{R}_b$ specifies the required relation $z^\star$ between arm i and frame f, and $\mathcal{J}^{\mathrm{guard}}_b$ indexes the retained boundary-scene factors. When evidence is sufficient, the relation filter emits a binary decision $\hat{z}^{i,f}_t$. If the filter abstains, the required relation condition remains unsatisfied. We use these definitions to compute the

Algorithm 2 Mode routing for Task-State Feedback

function ROUTEMODE($B_t, r_t, m_{t-1}, e_{t-1}$)
  if $m_{t-1} \in \{\text{REPAIR}, \text{REENTRY}\}$ and $\mathcal{R}_t^{\text{rep}}$ is verified **then**
    return $(\text{REENTRY}, e_{t-1})$
  if $\exists (f, z^\star) \in \mathcal{R}_t^{\text{req}} : \hat{z}_t^f = \text{UNKNOWN}$ **then**
    return $(\text{VERIFY}, \varnothing)$
  if PERSIST(RELATIONMISMATCH(r_t)) **then**
    $e_t \leftarrow \text{EVENTANCHOR}(r_t)$
    if $e_t = \varnothing$ **then**
      return $(\text{HOLD}, \varnothing)$
    else
      return (REPAIR, e_t)
  if PERSIST($E_t^\star < \epsilon_{\text{exp}}$) **then**
    return $(\text{REENTRY}, \varnothing)$
  return $(\text{TASK}, \varnothing)$

completion, relation, and scene conditions

$$\begin{aligned} \gamma_{i,b,t}^{\text{end}} &= \mathbf{1}\{E_{i,t}^\star \geq \epsilon_{\text{exp}}\} \sum_{s \in \mathcal{W}_{i,b}} \beta_t^i(s), \\ \gamma_{b,t}^{\text{rel}} &= \prod_{(i,f,z^\star) \in \mathcal{R}_b} \mathbf{1}\left\{\hat{z}_t^{i,f} = z^\star \ \wedge \ q_t^{i,f}(z^\star) \geq \theta_{b,i,f}\right\}, \\ \gamma_{b,t}^{\text{scene}} &= \prod_{j \in \mathcal{J}_b^{\text{guard}}} \mathbf{1}\{\rho_{j,t} \geq \theta_\rho \ \wedge \ c_{b,j,t} \geq \theta_{b,j}\}. \end{aligned} \tag{11}$$

Here, $\rho_{j,t}$ is factor-observation reliability and $c_{b,j,t}$ measures compatibility with the demonstrated boundary condition. A boundary is ready when all of the following conditions hold for H_b consecutive cycles. Every participating arm must satisfy $\gamma_{i,b,t}^{\text{end}} \geq \theta_{i,b}^{\text{end}}$. Both $\gamma_{b,t}^{\text{rel}}$ and $\gamma_{b,t}^{\text{scene}}$ must equal one. Task-dependent thresholds and H_b are estimated from successful boundary windows. Shared confidence and reliability thresholds are fixed before evaluation. For a coordinated boundary, all participating arms evaluate the same observation before one atomic commit, preserving a consistent multi-arm task state.

Algorithm 1 gives the causal, single-pass control loop. The task model $\mathcal{M}_{\mathcal{D}}$ collects the state topology $\mathcal{S}$, demonstration-conditioned models Π^{demo}, scene factors L^{scene}, event anchors $\mathcal{A}^{\text{evt}}$, and boundary support $\mathcal{G}^{\text{bdry}}$. All model-dependent operations use $\mathcal{M}_{\mathcal{D}}$, even when the algorithm omits the model argument for brevity. Initialization places (B_0, s_0^{ref}) at the task start and sets $m_0 = \text{TASK}$ with $e_0 = \varnothing$. A no-motion seed command a_0 then acquires o_1.

Inference and routing. During normal execution, the update $\bar{\beta}_t \to q_t \to \beta_t$ follows Equations 2 through 4, using o_t, a_{t-1}, and the previous reference. Outside TASK, progress remains fixed at $\beta_t = \beta_{t-1}$. The relation filter forms the demonstration prior from the fixed progress belief and updates q_t by Equation 3. Current physical evidence still determines $\mathcal{E}_t$ and $E_t^\star$ by Equations 5 and 1. The tuple $B_t = (q_t, \beta_t, \mathcal{E}_t, E_t^\star)$ collects these values.

The compact call ROUTEMODE(B_t, r_t) in the main algorithm also reads the stored (m_{t-1}, e_{t-1}) and persistence counters. Algorithm 2 gives the first-match priority order used by ROUTEMODE. Here, $\mathcal{R}_t^{\text{req}}$ contains currently required relations, and $\mathcal{R}_t^{\text{rep}}$ contains the active repair targets. The test RELATIONMISMATCH checks for reliable directional disagreement between expected and inferred relations. EVENTANCHOR returns an executable demonstrated LINK or UNLINK anchor, or $\varnothing$ when none is available.

Guarded execution and intervention. Within a skill, the controller regulates the reference according to the control-equivalence rule in Appendix A.2. Depending on the current belief, the reference remains fixed, advances by one legal successor, or realigns with a supported earlier state. A skill boundary commits only after the conditions in Equation 11 persist for the required duration. Independent boundaries may commit separately. Coordinated boundaries use one shared observation

and commit all participating arms atomically. PROBERETURN executes a bounded probe-and-return motion and feeds the observed response back into the relation filter. EVENTSTEP executes the next bounded step of the selected demonstrated event anchor. Both interventions preserve β_t and s_t^{ref}.

Legal re-entry. SCREEN generates re-entry candidates $\mathcal{C}_t^{\mathrm{re}}$ from the repaired event. If no current candidate supports execution and no relation mismatch requires repair, the search instead covers the task state space. The set $\mathcal{C}_t^{\mathrm{legal}}$ contains candidates that satisfy task ordering and permitted boundaries as well as robot, relation, and scene compatibility. Candidates in $\mathcal{C}_t^{\mathrm{align}}$ satisfy every condition except robot alignment and have observable robot evidence. Equation 6 selects $s_t^{\dagger}$ only when $\mathcal{C}_t^{\mathrm{legal}}$ is nonempty, and selects $\tilde{s}_t$ only when $\mathcal{C}_t^{\mathrm{align}}$ is nonempty. Atomic admission jointly commits $(\beta_t, s_t^{\mathrm{ref}})$ to $s_t^{\dagger}$ and sets $m_t = \textsc{Task}$. Validated relation evidence is preserved. The current action remains HOLD. ALIGN$(\tilde{s}_t)$ uses π and q_t with candidate-conditioned stream roles, without changing β_t or s_t^{ref}. If neither set is nonempty, the default HOLD action remains in effect. After admission, normal stream roles are recomputed in the next cycle. Only streams retained by the offline motion model may become action experts. A linked stream monitors relation integrity without adding an action constraint toward a frame already driven by the end effector.

Table 3. Runtime roles of task-relevant reference streams. Unknown denotes insufficient evidence for a binary relation decision. The EXECUTE role additionally requires an action-eligible stream.

Expected relation	Online decision	Stream role
external	external	EXECUTE
linked	linked	MONITOR
linked	external	RECOVER(LINK)
external	linked	RECOVER(UNLINK)
external or linked	unknown	DEFER
uncertain	any	DEFER

B EXPERIMENTAL SETUP AND IMPLEMENTATION

B.1 TASKS AND DEMONSTRATIONS

Table 4 summarizes the interaction structures represented by the evaluation tasks. The suite varies sustained relations between the robot and task entities, repeated skill transitions, contact constraints, and inter-arm dependencies. Six tasks use a single arm, and four require bimanual coordination. The six single-arm tasks also form the common subset for the native-system comparison. All controlled methods receive the same five successful demonstrations for each pairing of a policy and task. TSF uses no failure or recovery trajectories.

Demonstration collection, segmentation, and task-frame extraction are fixed across controlled comparisons. Evaluation episodes are paired across methods by task variation and random seed. Calibration and evaluation use the same prespecified task variations but share neither episode identifiers nor initialization seeds. Thresholds, perturbation magnitudes, and task subsets are fixed before evaluation. The task model uses successful demonstration episodes 0 to 4 and the same semantically unlabeled skill boundaries as the frozen action policy. Close Jar, Insert Onto Square Peg, and Stack Cups use variations 0 to 19. Open Drawer uses variations 0 to 2, and Handover Item uses variations 0 to 4. The remaining tasks use variation 0. The main evaluation contains 200 nominal and 200 perturbed episodes per task. Monitor calibration uses 50 disjoint successful rollouts per task, and supervised training uses up to 200 additional failure trajectories per task.

B.2 OBSERVATION INTERFACE AND INFORMATION BOUNDARIES

The controlled experiments use a common state-based observation interface to evaluate task-state feedback without visual-perception errors. At each control cycle, the interface provides synchronized robot state and simulator-derived poses or configurations of task-relevant entities. Every controlled method has access to the same observation schema and uses the fields required by the corresponding architecture. Table 5 separates runtime inputs from evaluator-only information.

Table 4. RLBench tasks used in the controlled evaluation and the interaction structure represented by each task.

Task	Arms	Interaction structure	Principal execution challenge
Close Jar	1	grasp, transport, and constrained closure	link formation and contact-rich placement
Open Drawer	1	articulated-object manipulation	progress under maintained contact
Insert Onto Square Peg	1	grasp, transport, and constrained insertion	contact-rich terminal alignment
Place Cups	1	repeated pickup and placement	repeated relation and stage transitions
Stack Cups	1	multi-object sequential construction	accumulated errors across stages
Sweep to Dustpan	1	tool-mediated object transport	sustained contact and indirect effects
Handover Item	2	sequential inter-arm transfer	change of relational ownership
Lift Tray	2	jointly constrained transport	synchronized shared transition
Bimanual Sweep to Dustpan	2	asymmetric coordinated roles	persistent inter-arm and object relations
Put Bottle in Fridge	2	coordinated articulation and placement	coupled progress across stages

Table 5. Observation interface for the controlled RLBench experiments. Evaluator-only quantities are excluded from the causal runtime stream.

Quantity	Simulation source	Runtime role
End-effector pose and gripper state	robot observation	motion, progress, and action-response estimation
Task-entity 6D poses	task low-dimensional state	object-relative action frames, relation evidence, and scene compatibility
Articulation or entity configuration	task low-dimensional state when defined by the task	sparse scene factors and transition readiness
Peer end-effector pose	synchronized bimanual robot state	candidate reference frame and coordinated-transition conditions
Entity identity and frame correspondence	fixed task interface	associates runtime entities with demonstration reference frames
Frame visibility and tracking reliability	fixed to visible and fully reliable for available simulator states	weights relational and progress evidence
Fault metadata, attachment truth, task success, and ground-truth progress	evaluator-only or unavailable	excluded from every policy and runtime monitor

The perturbation infrastructure may inspect simulator contact or attachment state only to trigger a prescribed physical intervention and audit its physical effect. These records are stored outside the causal observation stream and are never provided to a policy or runtime monitor. Likewise, task reward and success conditions are used only to terminate and score an episode. The framework receives neither a success signal nor ground-truth task progress.

The evaluation assumes exact, temporally aligned entity states with known correspondences. The observation interface excludes oracle failure labels and task progress. Robustness to perception error is outside the scope of the controlled study.

B.3 Perturbation Protocol

AHA and RACER generate failure data by perturbing execution (Duan et al., 2025; Dai et al., 2025). Here, physical perturbations test recovery without altering policy observations or directly assigning a failure label. Each episode is scheduled either without disturbance or with a perturbation from a family compatible with the task (Table 6). A scheduled event is physically triggered only if the eligibility condition is met during execution. All scheduled episodes remain in the intention-to-treat (ITT) denominator, including episodes in which the event is never triggered.

Table 6. Physical perturbation families. The policy does not observe the family, trigger, severity, or intended mismatch. Exact magnitudes are frozen before retained evaluation.

Family	Eligibility and trigger	Physical intervention	Evaluation purpose
Actuation delay	an eligible commanded motion	hold one or both arms for a fixed number of control cycles	delayed physical progress
Missed interaction	an eligible grasp or contact transition	suppress the first interaction without moving the target	intended relation never forms
Relation loss	a stable grasp or maintained contact	release and displace an attached entity, or open the selected gripper without moving a contact-only entity	established relation becomes invalid without an additional scene intervention
Environment change	after a task-relevant intermediate state	displace a target, support, or articulated entity	previously valid future reference changes
Coordination delay	before an eligible shared bimanual boundary	delay or block one participating arm only	inconsistent readiness across agents
Composed event	two eligible events at separated stages	apply two prespecified interventions in one episode	repeated recovery over a longer horizon

The primary end-to-end comparison samples one medium-severity event from the task-compatible families. The distribution-shift study varies physical quantities one at a time and moves triggers across eligible task stages. Further tests use perturbation families excluded from supervised training and episodes with two events. Task variations and random seeds remain paired across methods. Zero severity is the undisturbed reference. An Oracle-Trigger diagnostic supplies the audited intervention time to the otherwise unchanged TSF recovery controller. Remaining failures then reveal limits of recovery under accurate event timing, without implying physical infeasibility of the task. Compatible perturbation families are balanced within each task. Medium severity uses a 12-cycle delay, a 4 cm translation, a 10° articulation change, or one suppressed interaction. At low severity, delays last 4 cycles, translations span 2 cm, and articulation changes are 5°. The corresponding high-severity values are 20 cycles, 8 cm, and 20°. Triggers occur at 15% (early), 40% (middle), or 65% (late) of the eligible task interval. The stress evaluation adds 600 low-severity episodes and 600 high-severity episodes. Additional evaluations use 600 medium-severity episodes split across trigger stages and 200 two-event episodes. The Oracle-Trigger diagnostic contains 600 high-severity episodes and changes only the event-onset signal supplied to the TSF controller.

B.4 Baselines and Controlled Recovery Interface

Controlled end-to-end baselines. *DynaMAC* is the frozen reproduced action policy and receives no runtime intervention. *DynaMAC+Restart* restarts the current skill when a fixed no-progress rule is met, providing a simple-retry baseline. The remaining controlled monitors share the same action policy, episode pairs, and Skill-Retry executor defined below. Methods requiring alarm calibration use the same disjoint normal calibration split.

Monitoring baselines. *Trajectory-Likelihood* thresholds standardized likelihood under the active motion streams and is a constructed control rather than a published method. *FAIL-Detect* follows the released logpZO and time-varying conformal-prediction procedure (Xu et al., 2025; Angelopoulos & Bates, 2023). RND is evaluated as a secondary score. The *Supervised GRU Monitor* is a causal temporal classifier with the same online observations and nested labeled-failure budgets. Alarm thresholds for Trajectory-Likelihood, FAIL-Detect, and the GRU monitor are selected on disjoint nominal calibration rollouts, never on evaluation episodes. TSF derives the native task-dependent thresholds from the five successful demonstrations. For comparisons at a shared calibration target, every monitor additionally outputs a causal scalar score while leaving execution unchanged. TSF uses the maximum current mismatch-persistence count divided by the corresponding demonstration-derived persistence limit. The scalar score leaves the native binary alarm unchanged. Calibration targets are 0.5, 1, or 2 false interventions per 1000 nominal cycles. For each method and task, we select the most sensitive threshold multiplier whose calibration rate stays within the target. Nominal test episodes measure the actual false-intervention rate (FI), which can differ from the target. Detection delay is measured only for detected physical events, with missed events reported separately.

Skill-Retry. For the end-to-end controlled comparison, each generic monitor invokes the same recovery executor. Skill-Retry returns the action reference to the current skill entry recorded by the shared task executor. Execution then resumes from the current physical scene. Skill-Retry resets the action reference without resetting the simulator or physical scene. The executor receives no injected fault label and obeys the shared recovery and episode horizons. TSF-Monitor+Retry substitutes the TSF alarm for the baseline monitor's alarm while using the same executor.

Nearest-state inference ablation. The grouped ablations change both task-state estimation and how the controller uses the estimate. To test a simpler estimator with the same recovery interface, the nearest-state variant retains Full TSF's observations, five demonstrations, and frozen DynaMAC policy. Boundary guards, the Link/Unlink executor, legal re-entry search, and the 400-cycle recovery budget remain unchanged. The variant replaces the joint belief with a one-hot distribution at the nearest demonstrated state within the committed skill. Relation decisions use direct thresholds at each cycle. Full TSF and the ablation share the same 400 nominal and 800 perturbed task and seed assignments from the four-task stress subset. Paired ITT success is the primary endpoint.

Native external systems. RVT and RACER retain the released RGB-D observations, model checkpoints, and training pipelines (Goyal et al., 2023; Dai et al., 2025). Evaluation is restricted to task versions and fault interventions that can be matched without altering either controller's semantics. The native-system results are reported separately and cannot isolate the effect of the TSF task-state model.

Table 7. Baseline supervision and recovery interfaces. Calibration rollouts are successful policy executions and are listed separately from demonstrations.

Method	Success demos	Calibration	Failure supervision	Recovery executor
DynaMAC	5/task	0	0	none
DynaMAC+Restart	5/task	0	0	current-skill restart
Trajectory-Likelihood + Retry	5/task	50/task	0	Skill-Retry
FAIL-Detect + Retry	5/task	50/task	0	Skill-Retry
Supervised GRU Monitor + Retry	5/task	50/task	20 to 200/task	Skill-Retry
TSF-Monitor + Retry	5/task	0	0	Skill-Retry
Nearest-state inference	5/task	0	0	relation repair and re-entry
TSF (Ours)	5/task	0	0	relation repair and re-entry
RVT	native	native	0	none
RACER	native	native	recovery traj.	native learned policy

B.5 Native-System Matching Protocol

The native-system comparison covers six common single-arm tasks. Each system is evaluated on 100 nominal and 100 perturbed episodes per task. RVT, RACER, and TSF retain the observation interfaces, policies, checkpoints, and execution stacks of each native system. Only task assignment, success conditions, perturbation semantics, and the result schema are matched. The perturbed manifest balances actuation delay, missed interaction, and relation loss. These three families can be instantiated across systems without exposing controller-internal state. Fault eligibility, injection, and physical effect are recorded separately. Every scheduled episode remains in the intention-to-treat denominator.

The common adapter uses simulator time and physical contact or attachment evidence rather than inference wall time or policy-specific milestones. Compatibility corrections instantiate the same physical event definition across systems without changing native policies or action targets. For each system, we record the released checkpoint, environment version, task assignment, perturbation outcome, and episode result. RVT and RACER run in the corresponding native execution stacks, and the resulting outputs are aggregated without replay through the TSF executor. Nominal episodes are rerun whenever a compatibility change could alter nominal execution.

B.6 Metrics and Statistical Analysis

Episode success is the primary outcome throughout the paper. The undisturbed comparison uses a one-sided non-inferiority margin of 5 percentage points for TSF relative to DynaMAC. Per-task proportions receive 95% Wilson intervals (Wilson, 1927). We test paired binary outcomes with exact McNemar tests (McNemar, 1947). Holm's method corrects the prespecified family of pairwise decisions (Holm, 1979). Macro differences across tasks use 10,000 paired bootstrap samples over task and paired episode identifiers (Efron, 1979).

For faulted episodes, intention-to-treat success includes fault-eligible and fault-ineligible episodes exactly as scheduled. Recovery-chain diagnostics supplement the primary outcome with trigger-conditional success and recovery duration. Stage-specific diagnostics measure relation restoration, legal re-entry, and completion after re-entry. Monitor metrics are computed at the physical-event or episode level. We report event precision, recall, and F1 alongside episode balanced accuracy and area under the precision-recall curve (AUPRC). Detection delay is measured from audited violation onset for detected events. False interventions are reported per 1000 nominal control cycles. The same event metrics are computed separately for each perturbation family using identical shadow trajectories. Within each family, recall measures sensitivity, but F1 also penalizes false alarms. For each family-specific comparison, positives are the audited events physically triggered by that family.

Negatives are a fixed set of nominal episodes from the same eligible tasks. Other perturbation families are excluded rather than treated as negatives. The reporting population depends on the experiment. Figure 3a and Table 10 summarize the full monitoring population, while Figures 4a and 4b report task-macro results on the four-task stress subset. Per-event counts are retained in the released result files. TSF-specific quantities are reported only as internal diagnostics and ablations, never as primary cross-monitor metrics. These include inferred relation direction, progress state, and re-entry target. Violation events are less frequent than normal control cycles. We therefore include AUPRC, which is informative under class imbalance (Saito & Rehmsmeier, 2015). Repeated control cycles are not treated as independent samples.

B.7 Software, Models, and Hyperparameters

All global TSF hyperparameters are shared across tasks and fixed before evaluation. Task-dependent TSF quantities are estimated exclusively from successful demonstrations, without individual-task tuning. TSF uses neither failure trajectories nor failure outcomes for model fitting, hyperparameter selection, task-specific calibration, or recovery-policy design. Task interfaces specify the observable entities, candidate frames, and structural bindings supplied as input assumptions. These interface definitions are not tuned numerical parameters.

All controlled methods share the low-level controller and action rate, with the same kinematic limits and collision handling. The prespecified episode horizon is also fixed across methods. External RVT/RACER results retain their native interfaces. Each result is accompanied by the exact repository revision, checkpoint hash, task-data version, image resolution, camera set, and any compatibility patch. Table 8 separates shared settings from quantities estimated from the five successful demonstrations. Demonstration collection, calibration, supervised training, and evaluation use separate deterministic seed namespaces. The released configurations record software versions, model fingerprints, and manifest hashes for each execution stack.

Table 8. Configuration of TSF and the shared execution stack. Task-dependent quantities are estimated only from successful demonstrations. All other values are fixed globally.

Component	Quantity	Value
Relation inference	decision probability, persistence, entropy, information, tracking	0.70, 0.98, 0.62, 0.10, 0.25
Progress inference	confidence, entropy, backward/forward radius	0.55, 0.80, 1/2 states
Evidence and control	relation/scene weights, action-equivalence compatibility	1.0/1.0, 0.80
Persistence and boundary	relation/no-state mismatch, terminal window, boundary relation support	5/20 cycles, 3 cycles, 0.70
Verification and recovery	probe (samples), attempts, verify/re-entry/waypoint, total	1 s (3), 2, 30/40/40, 400 cycles
Execution	control period, episode horizon	0.05 s, 1000 cycles

C Additional Experimental Results

This section provides paired statistical tests and detailed results for the experiments in Section 5. All analyses use the task splits, information boundaries, and metrics defined in Appendix B.

C.1 Controlled-Comparison Statistics

Table 9 reports the prespecified paired tests for the per-task success rates in Table 1. Confidence intervals are computed over task-macro differences, while exact tests use paired episode outcomes.

Table 9. Prespecified tests for the controlled comparison. Differences are absolute success-rate changes for TSF relative to the named baseline.

Population	Comparison	Absolute difference (%)	95% CI	Exact p	Holm p	Decision
Nominal, all tasks	TSF vs. Dyna-MAC	$+7.2$	$[-1.7, 20.9]$	1.87×10^{-16}	1.87×10^{-16}	Non-inferior
Perturbed, all tasks	TSF vs. Dyna-MAC	$+26.1$	$[16.6, 37.7]$	2.30×10^{-114}	1.15×10^{-113}	Superior
Perturbed, all tasks	TSF vs. Restart	$+26.2$	$[16.5, 38]$	1.25×10^{-114}	7.49×10^{-114}	Superior
Perturbed, all tasks	TSF vs. FAIL-Detect	$+25.9$	$[16.5, 38]$	3.06×10^{-112}	1.22×10^{-111}	Superior
Perturbed, single-arm	TSF vs. Dyna-MAC	$+32.3$	$[18.9, 48.6]$	5.57×10^{-88}	1.67×10^{-87}	Superior
Perturbed, bimanual	TSF vs. Dyna-MAC	$+16.8$	$[7, 24]$	4.95×10^{-28}	9.90×10^{-28}	Superior

C.2 Detection and Recovery Results

The monitoring comparison uses an independent physical-state audit to mark the onset and end of each task-state violation. A positive event requires an audited task-state violation caused by the perturbation. In shadow mode, all monitors score identical frozen-policy trajectories without influencing execution. A separate closed-loop comparison connects the alarms to Skill-Retry.

Table 10. Shadow-mode monitoring results. Classification metrics are percentages. Delay is measured in control cycles from audited violation onset, and FI denotes false interventions per 1000 nominal cycles. Sup. GRU is averaged over three seeds.

Monitor	Prec. ↑	Rec. ↑	F1 ↑	Bal. Acc. ↑	AUPRC ↑	Delay ↓	FI ↓
Trajectory-Likelihood	44.4	16.9	24.5	51	48.9	25	0.92
FAIL-Detect	52.2	19.3	28.2	52.5	50.2	**5.5**	1.31
Sup. GRU	**87.1**	45.3	**59.6**	**69.8**	**77.2**	8	**0.38**
TSF-Monitor	61.1	**52.1**	56.2	62.1	55.4	12	2.07

Table 11. Recall at the independently selected calibration budget of 1 FI per 1000 normal calibration cycles. FI is the actual rate on retained nominal trajectories, not the calibration target. Missed is one minus event recall. Delay is conditional on detection. Sup. GRU reports mean ± standard deviation over three seeds.

Monitor	Actual test FI ↓	Recall ↑	Missed ↓	Delay ↓
Trajectory-Likelihood	1.81	27.8	72.2	20
FAIL-Detect	3.18	37.4	62.6	12
Sup. GRU	0.91 ± 0.13	52.4 ± 2.3	47.6 ± 2.3	6.7 ± 0.6
TSF-Monitor	2.38	**65.8**	**34.2**	8

Table 12. Recovery outcomes for physically triggered episodes, aggregated over three 800-episode runs for the supervised GRU and one 800-episode run for each of the other methods. Detection is normalized by triggered episodes. Restoration, resumption, and post-resumption completion are conditional on the preceding stage. Final success is normalized by triggered episodes.

Method	Triggered	Detect	Restore	Resume	Post-res.	Final SR
Likelihood + Retry	764	19.1	23.3	91.2	0	29.5
FAIL-Detect + Retry	762	20.1	34	53.8	10.7	28.6
Sup. GRU + Retry	2300	29.4	32.5	99.1	36.7	28.9
TSF-Monitor + Retry	687	**68.3**	19.2	**100**	0	27.1
TSF repair + same-state continuation	723	68	**39.2**	76.7	38.5	56.8
TSF (Ours)	679	63.3	36.3	70.5	**65.5**	**67.9**

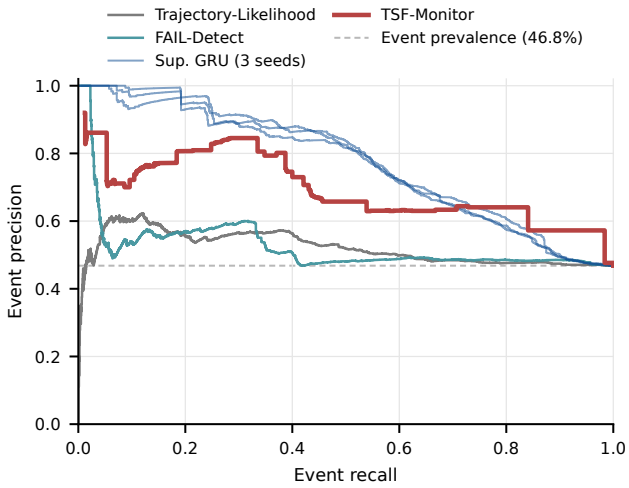


Figure 6. Event-level curves of precision against recall on the common scored shadow population (1991 nominal and 1992 perturbed episodes). An audited physical violation occurs in 1865 of the perturbed episodes. The three supervised-GRU seeds are shown individually. The dashed line is event prevalence. These retained-test curves are descriptive and are never used to select the fixed-FI operating points in Table 11.

Table 13. Event-level F1 (%) by physical perturbation family at each monitor's frozen operating point. Each family uses its audited violations and the same nominal episodes from eligible tasks.

Monitor	Actuation	Missed interaction	Relation loss	Environment change	Coordination delay
Trajectory-Likelihood	35.4	7.1	18.5	32.5	**57.9**
FAIL-Detect	28.6	13.9	39.5	32.2	52.6
Sup. GRU	45.1	79.8	63	**64.6**	46.5
TSF-Monitor	**47.1**	**90.4**	**80.9**	46.8	33.2

Table 14. Recovery success under explicit alternative denominators on the same pre-registered Stress-4 task and seed assignments. Method-specific triggered rows retain each policy's own physical-trigger subgroup. The final row restricts all methods to the intersection of episodes physically triggered for all three. Entries are successes/denominator (%).

Population	TSF monitor + retry	Repair + same-state	Full TSF
Scheduled ITT	188/800 (23.5)	413/800 (51.6)	463/800 (57.9)
Method-specific physical trigger	186/687 (27.1)	411/723 (56.8)	461/679 (67.9)
All-three trigger intersection	186/606 (30.7)	385/606 (63.5)	417/606 (68.8)

Table 15. Cross-tabulation of recovery path and final success on the 800-episode ITT population. Entries are final successes/episodes in that path. "Not restored" and "no resume" mean that the corresponding event did not occur after the preceding audited stage. Success can still occur through ordinary task execution or an event ordering outside the recovery funnel.

Audited path	TSF monitor + retry	Repair + same-state	Full TSF
Not physically triggered	2/113	2/77	2/121
Triggered, not detected	184/218	201/231	214/249
No restoration observed after detection	2/379	108/299	129/274
Restoration without subsequent resume/re-entry	0/0	45/45	46/46
Resumed/re-entered after restoration	0/90	57/148	72/110
All scheduled episodes	188/800	413/800	463/800

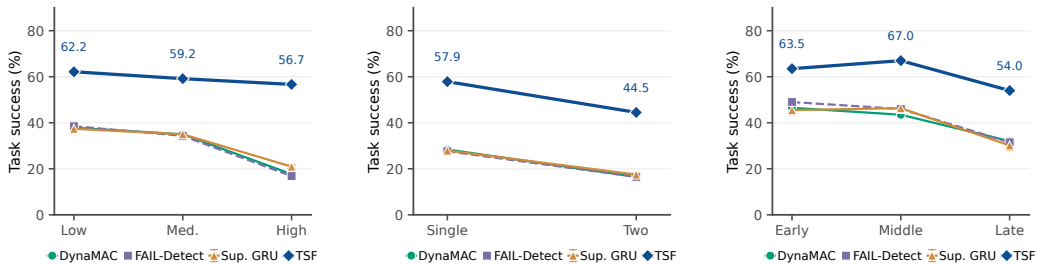
The Oracle-Trigger diagnostic tests whether inaccurate event timing explains the remaining high-severity failures. Audited onset produces little change in aggregate success (Table 16), showing that accurate timing alone does not resolve the remaining failures. Place Cups still fails under environment change with either timing source, identifying a setting where the current recovery procedure needs improvement.

Table 16. High-severity task success (%) with standard TSF detection and audited event timing. Oracle timing is a diagnostic input, not a deployable method.

Timing	Handover	Lift Tray	Open Drawer	Place Cups	Macro
Standard TSF	78.7	**50**	**74**	24	56.7
Audited onset	**79.3**	48.7	**74**	**26**	**57**

C.3 Generalization to Unseen Failure Conditions

We vary the amount of labeled failure data and test a failure family omitted from supervised training. At evaluation, we also vary disturbance severity, event composition, or trigger stage. For the held-out-family evaluation, the test family is removed from both GRU training and failure-based model selection. Every checkpoint uses the same disjoint nominal calibration split.



(a) Disturbance severity **(b)** Event composition **(c)** Trigger stage

Figure 7. Task-macro success on the four-task stress subset under changes in disturbance severity, event composition, and trigger stage. Error bars give the mean ± standard deviation over three supervised-GRU training seeds. The other methods are fixed.

Table 17. Task-macro success (%) under changes in severity, event composition, and trigger stage. Sup. GRU is averaged over three seeds. Medium severity uses a prespecified 150-episode-per-task subset of the matched primary assignments, while single event uses all 200 episodes per task.

	Severity			Events		Trigger stage		
Method	Low	Med.	High	Single	Two	Early	Middle	Late
DynaMAC	37.8	35	17.7	28.4	16.5	46.5	43.5	32
FAIL-Detect + Retry	38.5	34.3	16.8	27.6	16.5	49	46	31.5
Sup. GRU + Retry	37.4	35	20.9	27.9	17.5	45.5	46.3	30.2
TSF (Ours)	**62.2**	**59.2**	**56.7**	**57.9**	**44.5**	**63.5**	**67**	**54**

Table 18. Task-macro shadow-mode monitoring as the number of labeled failure trajectories per task varies. Every row is evaluated on the same trajectories. Zero denotes no failure trajectories, and GRU rows are averaged over three training seeds.

Monitor (failure traj./task)	Prec.	Rec.	Event F1	AUPRC	Delay	Nom. FI
Likelihood (0)	45.2	19.7	25.5	49.4	45.9	2.34
FAIL-Detect (0)	34.3	19.1	20.7	40.6	23.5	3.47
TSF-Monitor (0)	53	**57.7**	**48.8**	51.9	**12**	5.46
GRU Monitor (20)	61.3	11.7	17.8	54.7	82.5	0.66
GRU Monitor (50)	68.7	14.9	23.4	58.8	64.4	0.36
GRU Monitor (100)	71.6	14.6	23.8	60.1	48	**0.31**
GRU Monitor (200)	**78**	28.7	40.2	**65.7**	30.4	0.41

Table 19. Task-macro event F1 (%) on unseen failure families. "GRU all" uses 50 examples from every family, and "GRU LOFO" uses 50 examples per non-test family. GRU rows are averaged over three training seeds. All monitors are evaluated on identical shadow trajectories.

Method	Actuation	Interaction	Relation	Scene	Coordination
Likelihood	26.4	7.2	18.4	35.4	**47.7**
FAIL-Detect	21.2	14.9	18.6	23.5	27.7
GRU all-family	19.3	**56.9**	32.5	40.8	26.7
GRU LOFO	20.3	8.8	23.8	17.7	21.9
TSF-Monitor	**35.6**	51.2	**54.8**	**57.3**	26.9

Table 20. Held-out-family recall/actual nominal FI at the shared calibration target of 1 FI per 1000 cycles. GRU LOFO is the three-seed mean. The calibration and test views contain 200 episodes for the actuation, missed-interaction, and relation-loss families. The corresponding views contain 100 family-eligible episodes for coordination and environment change.

Method	Actuation	Interaction	Relation	Scene	Coordination
GRU LOFO	17.6/0.58	11.8/0.95	27.4/1.14	31/1.31	17.7/0.63
TSF-Monitor	**69.8**/3.56	**74.1**/3.56	**71.5**/3.56	**89.8**/3.71	**86**/3.26

C.4 Robustness over Longer Task Horizons

The stage-count study uses native repeat-count task variations. Scene randomization and control rules remain fixed within each task family, while the number of required interactions changes. The episode limit scales from demonstrated nominal duration by a frozen rule. The per-stage protocol is nested within the single-event protocol. Paired episodes share a seed, initialization, and first scheduled event. Later opportunities open only after the first event is physically triggered. The nested protocol keeps initial conditions and the first disturbance matched when measuring the effect of repeated events. Each task, method, and condition contains 100 episodes. The *All levels* row averages over all available combinations of task family and interaction count.

Table 21. Task-macro success (%) over longer horizons. Each cell reports nominal/single-event/per-stage success. Results at each interaction count are macro-averaged over the available task families.

Method	1 interaction	2 interactions	3 interactions	All levels
DynaMAC	96/59.3/59.3	92.3/45.7/44.7	16/6/5	74.6/40.9/40.3
Restart	96/65/65	94.3/55/50.3	14.5/9/7.5	75/47.3/45.1
FAIL-Detect + Retry	97/58.3/58.3	95/45.7/45.3	19.5/7/6	76.9/40.8/40.4
TSF (Ours)	90/78/78	83.3/64/62	**62/43/43**	**80.5/64/63.3**

Figure 5b divides perturbed success by nominal success within each task before macro-averaging. TSF has higher retention than the strongest baseline at every interaction count, although retention declines as the required sequence grows. The advantage therefore appears both in absolute success (Table 21) and after normalization by nominal performance.

C.5 Native-System Results

We compare TSF with the released RVT and RACER systems (Goyal et al., 2023; Dai et al., 2025) on six common single-arm RLBench tasks. Each method retains its native action policy, observation inputs, and training supervision. All methods follow the same nominal and perturbed evaluation protocol.

Table 22. Native-system performance (%) on six common RLBench tasks, with 100 episodes per task and condition. Each method uses its native policy, observation inputs, and training supervision. Retention is $100 \times S_{\mathrm{pert}}/S_{\mathrm{nom}}$, computed for each task and then macro-averaged.

Method	Failure/recovery supervision	Nominal	Perturbed	Retention
RVT	none	61.7	43	69.7
RACER	recovery trajectories	52	32	**93.6**
TSF (Ours)	none	**79.8**	**66.3**	86.5

TSF has the highest mean nominal and perturbed success in the six-task comparison (Table 22), leading on five tasks under perturbations. RACER has the highest mean per-task retention despite lower absolute success. Retention weights tasks equally by proportional change and can remain high even when absolute success is low. Absolute success and the per-task breakdown in Table 23 provide the complementary view.

RACER uses recovery trajectories for training, while TSF uses only successful demonstrations. Because the complete systems also differ in observation interfaces, policies, and execution stacks, the comparison measures system-level performance rather than the effect of an isolated recovery component.

Table 23. Per-task success (%) for the native-system comparison. Each cell reports nominal/perturbed success over 100 episodes per condition.

Method	Close Jar	Open Drawer	Insert Peg	Place Cups	Stack Cups	Sweep
RVT	99/35	89/96	30/30	38/22	75/61	39/14
RACER	86/30	97/90	12/16	2/4	42/29	73/23
TSF (Ours)	81/82	65/70	97/80	41/40	95/78	100/48

C.6 Ablation Results

The main ablation groups the method into task-state representation, online progress alignment, and state-aware recovery. Finer ablations test individual components by removing relation or scene evidence, or replacing belief-driven stream roles. The evidence ablations omit the corresponding compatibility term from progress correction and re-entry scoring while retaining relation filtering, stream routing, and repair. Other variants disable control-equivalence aggregation or boundary guards, or change verification or re-entry behavior. Each variant changes only the named component and reuses the paired episode assignments and controller budget.

Table 24. Aggregate results (%) for fine-grained ablations on the four-task ablation subset. Recovery is final task success among physically triggered episodes.

Variant	Nominal	Perturbed	Bimanual perturbed	Recovery
TSF (Ours)	70.0	**57.9**	**63.8**	**67.9**
Nearest-state inference	22.5	21.5	0.5	31.6
No relation evidence	**70.2**	57.6	61.7	66.1
No scene evidence	68.5	56.6	63.5	66.9
Static stream roles	52.8	36.9	59.8	41.1
No boundary guards	66	45.5	39.5	52.9
No active verification	52.5	46.4	44.5	54.1
Same-state resumption	62	51.6	61	56.8
No control equivalence	64.8	55.5	61.7	62

Nearest-state inference details. For the nearest-state row in Table 2, the paired Full-minus-ablation difference is $+47.5$ points nominally (95% CI $[42.0, 53.0]$, exact McNemar $p = 2.09\times10^{-44}$). Under perturbations, the difference is $+36.4$ points (95% CI $[32.7, 40.0]$, $p = 2.86\times10^{-70}$). Table 25 gives the per-task breakdown. Open Drawer remains similar across estimators in both conditions. However, the nearest-state ablation fails to complete either bimanual task nominally.

Table 25. Per-task results for the nearest-state inference ablation. Nominal cells contain 100 paired episodes, and perturbed cells contain 200. Differences are Full minus nearest state.

Condition	Task	Full (%)	Nearest state (%)	Difference	95% CI	Exact p
Nominal	Open Drawer	70.0	65.0	$+5.0$	$[-3.1, 13.1]$	0.332
Nominal	Place Cups	51.0	25.0	$+26.0$	$[14.3, 37.7]$	6.88×10^{-5}
Nominal	Handover	95.0	0.0	$+95.0$	$[90.7, 99.3]$	5.05×10^{-29}
Nominal	Lift Tray	64.0	0.0	$+64.0$	$[54.5, 73.5]$	1.08×10^{-19}
Perturbed	Open Drawer	73.5	69.5	$+4.0$	$[-1.0, 9.0]$	0.169
Perturbed	Place Cups	30.5	15.5	$+15.0$	$[8.7, 21.3]$	9.25×10^{-6}
Perturbed	Handover	85.5	0.0	$+85.5$	$[80.6, 90.4]$	6.68×10^{-52}
Perturbed	Lift Tray	42.0	1.0	$+41.0$	$[34.2, 47.8]$	4.14×10^{-25}

The primary ITT analysis includes every scheduled execution. Infrastructure errors are more frequent for full TSF than for nearest-state inference, occurring in 39 versus 5 nominal episodes and 69 versus 7 perturbed episodes. Removing any pair affected by an infrastructure error leaves 358 nominal and 724 perturbed pairs. The TSF advantage then increases from $+47.5$ to $+52.8$ points nominally and from $+36.4$ to $+40.1$ points under perturbations. The advantage therefore persists in the sensitivity analysis. ITT remains primary because infrastructure status is observed after assignment.

Table 26. Perturbed recovery paths for the nearest-state inference ablation. Each stage after Trigger uses the preceding stage as denominator. ITT uses all 800 scheduled episodes. Common-trigger success uses the 494 episodes physically triggered for both methods.

Method	Trigger	Detect	Restore	Re-enter	Finish after re-entry	ITT success	Common-trigger success
Full TSF	679/800 (84.9)	430/679 (63.3)	156/430 (36.3)	110/156 (70.5)	72/110 (65.5)	463/800 (57.9)	325/494 (65.8)
Nearest state	544/800 (68.0)	258/544 (47.4)	89/258 (34.5)	89/89 (100)	46/89 (51.7)	172/800 (21.5)	171/494 (34.6)

Stage-conditional rates describe selected recovery paths and do not identify component-level causal effects. Every restored episode in the nearest-state ablation reaches re-entry, but overall detection and task completion remain lower.

Table 27. Effect of control-equivalence aggregation on repeated commands. Rates are paired per-episode means over the ablation subset. Differences are full TSF minus the ablation, with paired-bootstrap 95% confidence intervals.

Condition	Metric	TSF	No aggregation	Difference
Nominal	repeated reference (%)	21.6	34.5	$-12.9\ [-16.2, -9.7]$
Nominal	repeated target (%)	12.7	25.6	$-12.9\ [-16.4, -9.5]$
Perturbed	repeated reference (%)	30	39.6	$-9.6\ [-11.6, -7.7]$
Perturbed	repeated target (%)	15.6	27.1	$-11.5\ [-13.6, -9.4]$

Removing relation or scene evidence alone produces small decreases in perturbed success. Static stream roles produce a larger aggregate decrease, while removing boundary guards or active verification substantially reduces bimanual success. Control-equivalence aggregation has a modest effect on perturbed success but consistently reduces repeated reference and target cycles (Table 27). Grouping states with equivalent commands therefore reduces repetition even when the gain in final success is small.

We also vary the demonstration-derived boundary threshold from $0.85\times$ to $1.15\times$ the frozen value on calibration demonstrations. All five settings reject all 300 preterminal instances, accept all 300 terminal instances, and pass all 60 evaluated boundaries. The evaluated boundary decisions are unchanged across this local calibration range.

C.7 Recovery-Path Analysis

We classify each physically triggered TSF episode by the first recovery stage not reached after the preceding audited stage, or by the final outcome after resumption. These path categories describe event ordering rather than final failure. Episodes can still finish without traversing the complete recovery chain, as shown in Table 15.

Table 28. Recovery-chain outcomes for the 679 physically triggered TSF episodes. Rates use all triggered episodes as the denominator.

Outcome	Count	Rate (%)	Operational definition
Missed or late detection	249	36.7	no actionable detection was emitted for the audited violation
Condition not restored	274	40.4	a mismatch was detected but the required physical condition was not restored
No execution resumption	46	6.8	the condition was restored but no supported state was committed
Post-resumption failure	38	5.6	legal re-entry occurred but the remaining task did not finish
Successful after resumption	72	10.6	legal re-entry occurred and the remaining task completed

The largest path category contains detected episodes without a subsequent audited restoration. This count alone does not establish restoration as the dominant cause of final failure, because some episodes in the category still succeed. A majority of episodes that reach legal re-entry complete the task, but this conditional result excludes earlier exits. The outcome cross-tabulation in Table 15 is therefore needed to distinguish incomplete recovery paths from unsuccessful executions.

D TASK GALLERY

We present representative successful executions across the full ten-task evaluation suite in Figure 8. The ordered frames expose the distinct manipulation stages.

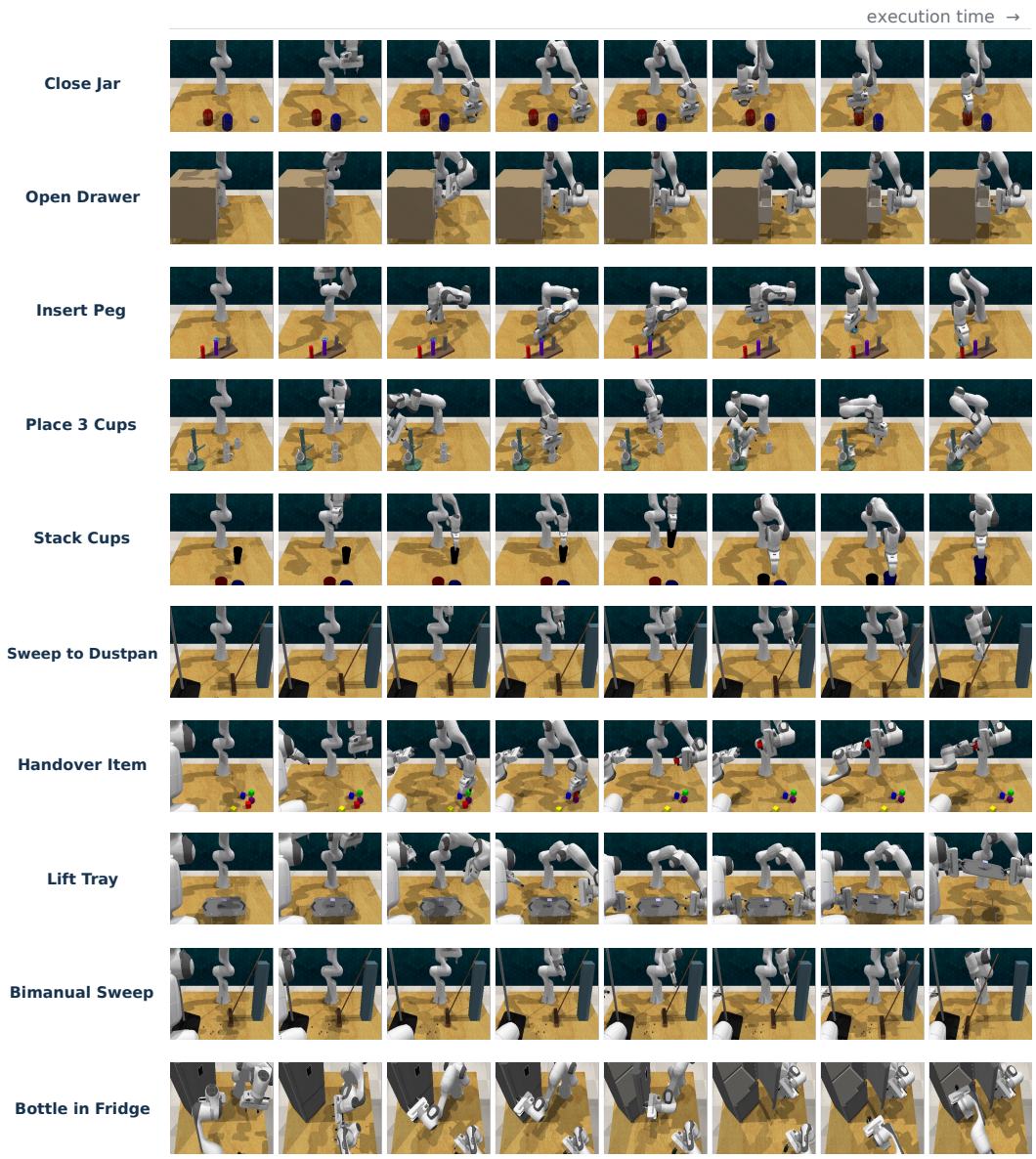


Figure 8. Visualization of successful executions across the ten-task evaluation suite. The displayed sequences progress from the initial scene to task completion, covering six single-arm and four bimanual tasks.