Latent-Space Prediction Error and Fuzzy Risk-Scaled Recovery for Robotic Manipulation

Thanh-Hai Tran [1,2] and Xuan-Bach Le [1,2⋆]

[1] Faculty of Computer Science and Engineering, Ho Chi Minh City University of Technology (HCMUT), Ho Chi Minh City, Vietnam
[2] Vietnam National University Ho Chi Minh City, Ho Chi Minh City, Vietnam
{tthai.sdh241,lexuanbach}@hcmut.edu.vn

Abstract. Industrial pick-and-place lines demand near-zero failure tolerance, yet many safety layers still wait for a fault to manifest before they trigger an indiscriminate Emergency Stop. We ask whether latent prediction error can instead serve as a graded recovery signal. Our closed loop combines a frozen DINOv2 encoder, an RSSM whose KL-divergence prediction error gives us an online risk reading, an ANFIS risk mapper, and a controller that applies bounded trajectory corrections scaled by that risk coefficient. In Webots with a UR5e arm and condition-based TRANSIT dynamics, we evaluate four recoverable motion disturbances ($N = 40$ episodes per mode per scenario), and we treat two semantic or actuation faults as outside joint-space recovery. On the harder disturbances, the deployed ON policy improves RSR over a binary fail-safe halt (Jerky +25.0 pp and Speed +42.5 pp, with non-overlapping Wilson 95% CIs). On the lighter disturbances, ON mainly improves trajectory quality, and we reduce mean joint deviation by 15–45% on three motion disturbances (Pause peaks at −44.8%). Speed stays dominated by the joint-velocity cap. We report OFF/ON as a deployed-policy comparison and use smoothing as the controller-level complement. CPU-only latency is P95 = 34.24 ms, which sits 65.8% below the 100 ms budget.

Keywords: Anomaly detection · World models · Fuzzy risk · Risk-aware control · Robotics

1 Introduction

Industrial robotic arms in pick-and-place lines operate under near-zero failure tolerance, yet most safety layers remain reactive: they wait for a fault to become visible and then halt the task outright. This response protects the cell, but it turns recoverable disturbances into unplanned downtime. Predictive visual models offer a more graceful alternative, but video-diffusion approaches can hallucinate future frames and typically need seconds per clip on GPU hardware, which is far too slow for 20–50 Hz control. Expressive latent predictors such as V-JEPA 2 [1] and DreamerV3 [2] avoid hallucination, but they still lack the

⋆ Corresponding author

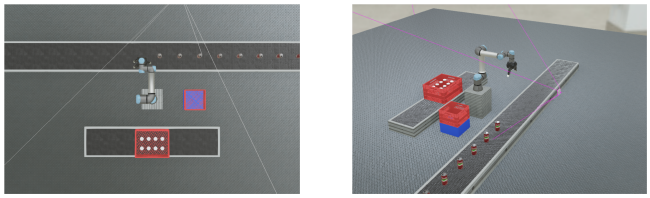


Fig. 1: Workspace layout (left) and mid-transit camera frame (right). Webots R2025a, UR5e with Robotiq gripper.

safety logic and runtime recovery policy needed to translate prediction error into action.

We organize our predictive closed-loop anomaly detection and recovery system around three goals. **(G1) Online Risk Detection**: we read latent dynamics during TRANSIT rather than pixel reconstruction, which removes any dependence on generated visual predictions. Lead-time metrics remain supplementary offline diagnostics. **(G2) Semantic Risk Quantification**: we convert raw prediction error into graduated, human-readable risk levels (Safe / Medium Risk / High Risk / Critical) through fuzzy logic rather than a single binary threshold, following the long-standing fuzzy-set and Takagi–Sugeno modeling tradition [14,15]. **(G3) Risk-Aware Adaptive Control**: instead of halting on the first alarm, we continue through moderate risk and issue bounded corrective actions scaled by the risk coefficient, and we evaluate this against a binary fail-safe halt as a deployed-policy comparison (L7, Sec. 6).

We let three questions guide our evaluation. **RQ1:** can RSSM-based latent dynamics catch faults that calibrated reactive baselines either miss in part or cannot turn into recovery? **RQ2:** can fuzzy risk quantification give us a less disruptive alternative to a fail-safe halt on recoverable disturbances? **RQ3:** can the whole loop run within a CPU-only real-time budget (<100 ms)?

We make three contributions. First, we integrate latent perception, RSSM prediction error, fuzzy risk quantification, and bounded residual control into one closed recovery loop. Second, we provide Webots-online validation under condition-based TRANSIT dynamics ($N = 40$ per mode per scenario) on four recoverable motion disturbances plus two out-of-scope semantic/actuation faults, where the ON policy improves RSR over fail-safe halt on the harder disturbances (Jerky +25.0 pp and Speed +42.5 pp, with non-overlapping Wilson 95% CIs) and improves trajectory smoothness on three motion disturbances (joint deviation reduced 15–45%, while Speed stays dominated by the velocity cap). Third, we report a CPU-only P95 latency of 34.24 ms ($N = 9{,}969$ step samples, 19 runs). Sections 2 and 3 formalize the problem and the architecture. Section 4–6 cover the evaluation and findings, and Section 7 positions the work against ten recent systems along four functional modules.

2 Problem Formulation

We model the task as a Markov decision process (MDP) [18], with state $s_t \in \mathcal{S}$, action $a_t \in \mathcal{A}$, transition $T(s_{t+1} \mid s_t, a_t)$, reward $R(s_t, a_t)$, and discount γ,

which the Webots cell instantiates. The hidden state contains robot joints and velocities, gripper state, object pose, task phase, and injected fault variables. The deployed agent sees only a 140×140 RGB crop $o_t \in \mathcal{O}$ through $\Omega(o_t \mid s_t)$, so the operational model is the POMDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, T, R, \Omega, \gamma \rangle$. Webots R2025a [11] advances at 125 Hz, and we require the recovery stack to keep CPU inference below 100 ms.

The nominal policy π_{nom} executes the scripted task, while the recovery layer acts only in TRANSIT. For motion-level recoverable faults we dispatch a joint residual, $a_t = \pi_{\text{nom}}(\hat{s}_t) + \alpha_t \delta a_t$ with $\delta a_t \in [-0.30, 0.30]^6$, where $\hat{s}_t$ is the runtime controller state (joints, gripper, phase, instrumented can pose, KL proxy, and clipped risk). The gripper command stays nominal, so we can detect premature release and wrong-product faults but cannot recover them with this residual layer.

Our objective is graded intervention rather than a binary halt. During TRANSIT, DINOv2 maps o_t to a latent token, the RSSM produces $e_t = D_{\text{KL}}(q \,\|\, p)$, and ANFIS combines the current KL with $x_{3,\text{std}}$ to assign $\alpha_t \in [0, 1]$ and {Safe, Medium, High, Critical}. The layer must identify anomalous risk, scale the correction by α_t, preserve low-risk behavior, and meet FPR $< 5\%$ and RSR $> 70\%$ on recoverable scenarios. We assume that DINOv2 features retain task structure [12], that RSSM error rises when observations become hard to predict [2], and that risk is monotone in the two ANFIS inputs [13].

3 Methodology

We describe the pipeline in five subsections: a data-flow overview (Section 3.1), followed by the four modules in execution order, namely perception (Section 3.2), the world model (Section 3.3), semantic risk (Section 3.4), and risk-aware control (Section 3.5).

3.1 System Architecture

We organise the pipeline as four modules that move information from raw pixels to a graded corrective action within a single control timestep. At each step t, a frozen DINOv2 backbone encodes an RGB crop into a compact latent $z_t \in \mathbb{R}^{384}$, which then feeds a Recurrent State Space Model (RSSM) that emits a KL-divergence anomaly signal e_t. An ANFIS layer next fuzzifies this signal into a continuous risk coefficient $\alpha_t \in [0, 1]$, which finally modulates the magnitude of a bounded residual correction that we add to a nominal IK policy. We summarise the data flow in Figure 2, and we show the simulation workspace and the camera viewpoint that feeds the perception stack in Figure 1. We describe each module in turn below.

3.2 Module 1: Visual Perception (DINOv2-ViT-Small)

We start perception from a frozen DINOv2-ViT-Small encoder [12], which we export to ONNX for CPU inference. We rely on DINOv2 because its self-supervised features supply task-relevant semantic structure without any anomaly labels, and

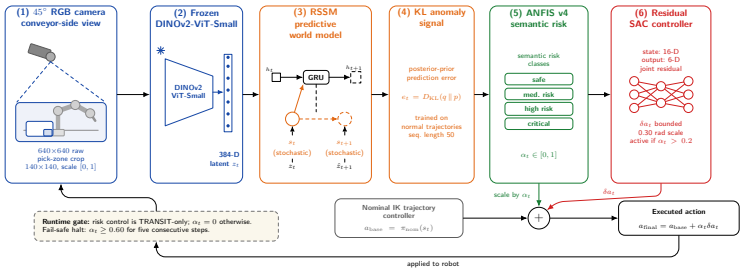


Fig. 2: Four-module closed-loop pipeline.

this matters given our limited simulation dataset. The encoder ingests an RGB crop from the 45-degree conveyor-side camera, resized to 140×140 and scaled to $[0, 1]$, and returns the final-layer CLS token $z_t \in \mathbb{R}^{384}$. Two choices keep this module real-time. We freeze the encoder to avoid overfitting while we preserve its transferable semantics, and we pick ViT-Small (21M parameters) as the largest backbone that fits the 100 ms CPU budget. The 140×140 crop halves the input cost relative to the standard 224×224, which already exceeds the budget on CPU (224×224 gives P95 = 150 ms on onnxruntime, whereas 140×140 with OpenVINO gives P95 = 34.24 ms, see Sec. 5.4). The trade-off is reduced sensitivity to fine gripper-state details, which we revisit in the Limitations.

3.3 Module 2: Predictive World Model (RSSM)

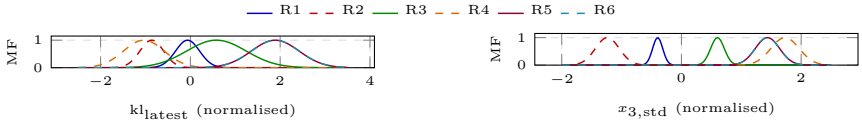
A latent encoder alone cannot tell us which observations are anomalous, so we also need a model of what the latent should look like. We therefore train a Recurrent State Space Model (RSSM), following the latent-dynamics world model DreamerV3 [2], on the encoder's output during nominal trajectories. At deployment we read off the divergence between its posterior and prior as the anomaly signal: when the world becomes hard to predict from the current hidden state, e_t rises.

The RSSM combines a deterministic recurrent state h_t (GRU, hidden size 128) with a stochastic state s_t of dimension 32 governed by a prior $p(s_t \mid h_t)$. An encoder MLP $[384 \to 256]$ (LayerNorm + ELU) preprocesses z_t, and the posterior head concatenates h_t with this representation to produce $q(s_t \mid h_t, z_t)$ (336,384 parameters total). The per-step anomaly signal is the closed-form Gaussian KL,

$$e_t = D_{\mathrm{KL}}(q(s_t \mid h_t, z_t) \,\|\, p(s_t \mid h_t)) \,, \tag{1}$$

which is small when posterior and prior agree and grows whenever the latent observation deviates from prediction. We train for 50 epochs on 200 nominal trajectories with Adam (lr = $3{\times}10^{-4}$), sequence length 50, and free_bits = 0.5. Empirically, e_t alone is informative but not decisive: per-episode maximum KL distributions for nominal and anomalous trajectories overlap (medians 0.31 vs. 0.39, with the B3 calibration threshold 0.4132 = 1.1× the max calibration KL), so a single-threshold detector inherits a hard precision-recall trade-off. We design the next module to resolve this ambiguity.

Fig. 3: Risk-tier action map over the risk coefficient α_t.



(a) $\mathrm{kl}_{\mathrm{latest}}$ input. (b) $x_{3,\mathrm{std}}$ input.

Fig. 4: Learned Gaussian membership functions for the two ANFIS inputs (6 rules R1–R6, normalised feature space).

3.4 Module 3: Semantic Risk Quantification (ANFIS)

To turn the ambiguous e_t into a controller-usable signal, we route it through an Adaptive Neuro-Fuzzy Inference System (ANFIS) [14,15,13], which yields a continuous risk coefficient and an interpretable risk tier from learned membership functions. The deployed model takes two inputs, $\mathrm{kl}_{\mathrm{latest}}$ (the current KL) and $x_{3,\mathrm{std}}$ (the 10-frame sliding std of the absolute pixel difference over the 140×140 crop after a 7×7 box blur, with no ROI mask). From these, a Sugeno-style network ($n_{\mathrm{rules}} = 6$, sigmoid output) produces $\alpha_t \in [0, 1]$, which we map to the semantic risk tiers in Figure 3. The Critical tier is a design provision for future extension, and we exercise it only as the ON-mode emergency backstop ($\alpha \geq 0.90$ for 8 steps), while the lower three tiers carry the active risk-aware control. The illustrative template *IF* KL is "Medium" *AND* $x_{3,\mathrm{std}}$ is "Rising" *THEN* Risk is "CAUTION" shows the intended human-readable form. Because the target $\alpha^* = \mathrm{clip}(\mathrm{KL}/0.8 + x_{3,\mathrm{std}}/0.05, 0, 1)$ is itself a clipped-linear combination, we use ANFIS strictly as *semantic-tier labelling infrastructure* and make no claim of a regression-fidelity improvement over a calibrated linear regressor (L5). We fit this two-input network to 10,000 procedurally-sampled points of the α^* surface (80/20 split) by minimising the logit-space MSE, which the six-rule network reproduces closely. Figure 4 shows the learned Gaussian membership functions.

3.5 Module 4: Risk-Aware Adaptive Control

The final module closes the loop. We want the intervention magnitude to grow smoothly with estimated risk rather than switch abruptly at a threshold, and we implement this with a residual Soft Actor-Critic (SAC) [16,17] whose output we gate and scale by α_t. The policy observes a 16-dimensional state (joint positions, gripper state, can position, phase one-hot, KL proxy, and clipped α_t) and returns a 6-dimensional joint residual. Recovery is intentionally joint-space only, so S5/S6 fall out of scope, and we draw the `can position` from Webots ground-truth (L4). Whenever $\alpha_t > 0.2$, the policy emits δa_t and we dispatch

$$a_{\mathrm{final}} = \pi_{\mathrm{nom}}(s_t) + \alpha_t \cdot \delta a_t, \tag{2}$$

with δa_t bounded by a 0.30 rad action scale, making the correction proportional to risk and preventing arbitrarily aggressive deviations. The training reward $R = R_{\text{task}} - 0.01 \cdot \|\alpha_t \cdot \delta a_t\|^2$ shares the same shape: the squared-residual penalty (scaled by α_t^2) encourages near-zero intervention at low risk and tightens action only when risk warrants it. At evaluation we compare two deployed policies on the same signal: ON (SAC enabled, with an emergency guard at $\alpha \geq 0.90$ for 8 steps) versus a binary fail-safe halt (OFF, with SAC disabled and an abort at $\alpha \geq 0.60$ for 5 steps). This is a deployed-policy comparison, not an isolated residual-SAC ablation (L7).

Training runs in a lightweight offline kinematic simulator with procedurally sampled (kl_t, α_t) pairs calibrated to Webots fault statistics; the deployed pipeline receives live RSSM output. We train for 1500 episodes from a single seed 42 (L8), with a replay buffer of 100,000 and 1024 warmup steps, Adam (lr $= 3\times10^{-4}$) for the actor, critic, and entropy coefficient, $\gamma = 0.99$, $\tau = 0.005$, batch size 256, and automatic entropy tuning ($H^* = -6$ nats). We inject anomaly perturbations with probability 0.85 per episode under domain randomisation, and we keep evaluation seeds disjoint from training, so the reported Wilson 95% CIs capture scenario noise rather than training-seed noise.

Condition-based TRANSIT. TRANSIT is condition-based rather than fixed-budget: it terminates when the joint-position error from the reference pose stays below 0.08 rad for three consecutive steps, or at a 300-step cap (2.4 s at 125 Hz). At the end of an episode, the controller signals the conveyor supervisor over shared IPC to recycle the placed can and deliver the next supply, and we observed no invalid-supply cascade across the 320 recoverable-scenario episodes. We report the end-to-end CPU-only latency in Sec. 5.4.

4 Experimental Setup

We describe the setup in five subsections: the Webots environment (Section 4.1), failure scenarios (Section 4.2), episode workflow and phase gating (Section 4.3), baselines and ablations (Section 4.4), and evaluation metrics (Section 4.5).

4.1 Simulation Environment

We run all experiments in Webots R2025a [11] with a Universal Robots UR5e (6-DOF) and Robotiq 3-finger gripper, under ODE physics at 8 ms timestep (125 Hz). A 45-degree conveyor-side RGB camera (640×640 raw) provides observations, and we resize the pick-zone crop to 140×140 for the pipeline. Our multi-controller architecture communicates over shared IPC files, which lets us inject faults and log state without touching the physics loop.

4.2 Failure Scenarios

Table 1 defines the six evaluated scenarios. We fix the scenario selector at the start of an episode and apply each perturbation through command-level hooks (joint timing/velocity, gripper state, or supplied object), and we leave the Webots

Table 1: Failure scenarios and intervention scope.

ID	Scenario	Injection	Breaks?	Recover?
S1	Vibration	sinusoidal joint perturb.	no	yes
S2	Jerky Motion	step-hold motion	no	yes
S3	Pause	mid-transit freeze	no	yes
S4	Speed Anomaly	reduced joint speed	no	yes
S5	Premature Release	gripper opens mid-carry	yes	no
S6	Wrong Product	wrong object supplied	yes	no

physics engine unchanged. The **Breaks?** column marks faults whose nominal task cannot complete without intervention beyond continuing the trajectory. Because residual correction is joint-space only, S5 and S6 fall outside its scope. We train RSSM and DINOv2 *entirely unsupervised* on normal trajectories, so no labelled anomaly data enters training.

4.3 Episode Workflow and Phase Gating

Each episode runs an eleven-phase pick-and-place cycle in three stages: pick (home, approach, lower, close, lift), TRANSIT carry, and place (approach, lower, open, retract, home). We confine fault injection (S1–S6) and Module 4 intervention to TRANSIT, where ANFIS computes α_t at every step and the residual-SAC layer dispatches bounded corrections whenever $\alpha_t > 0.2$. All other phases execute the nominal IK trajectory with $\alpha_t = 0$. TRANSIT terminates either by convergence (three consecutive steps with joint error < 0.08 rad) or at a 300-step cap. Within TRANSIT we damp α_t with a 100-step confidence ramp (min(1, transit_step/100), while KL < 0.65) so that we suppress transient high-KL artefacts during RSSM warm-up, and the fail-safe halt adds a 35-step grace period before it arms. The runtime ANFIS reads $\mathrm{kl}_{\mathrm{latest}}$ from the RSSM and $x_{3,\mathrm{std}}$ (the 10-frame sliding std of the absolute pixel difference over the 140×140 crop after a 7×7 box blur).

4.4 Baselines and Ablations

We define four diagnostic detector baselines. **B1** is a Reactive Threshold (pixel-motion $x_{3,\mathrm{std}}$, no world model). **B2** is EfficientAD-style [3] (DINOv2 embedding L2 distance to a nominal reference). **B3** is RSSM KL-only (Module 2 only). **B4** is RSSM+ANFIS detect-only (Modules 1–3, no recovery). We calibrate B1/B2 to 0.0% FPR on 10 nominal episodes (conservative, not sweep optima), and we treat them as diagnostic matched-calibration baselines in Section 5.3 because they lack a separately held-out normal set. B3 uses the threshold $0.4132 = 1.1\times$ the maximum calibration KL. B4 and Ours share a fixed $\alpha \geq 0.92$ threshold that we set before anomaly evaluation, and we evaluate B3/B4 FPR on the remaining 70 nominal episodes. The ablation compares deployed intervention policies rather than a matched-threshold module-removal RSR.

Table 2: Webots-online system summary (Wilson 95% CI in brackets).

Method	Role	FPR [95%CI]	Clean RSR [95%CI]
B3: RSSM KL-only	Detector	0.0% [0, 5.2]	N/A
B4: RSSM+ANFIS	Detector+risk	0.0% [0, 5.2]	N/A
Fail-safe halt (OFF)	Binary halt	N/A	65.0% [57.3, 71.9]
Ours (ON)	**Adaptive ctrl**	**0.0% [0, 4.6]**	**86.3% [80.0, 90.8]**

4.5 Evaluation Metrics

FPR is the online false positive rate. **RSR** (Recovery Success Rate) excludes only invalid-supply events (none occur on recoverable scenarios). **Smoothness** reports the mean joint deviation from the TRANSIT reference pose (rad) and the mean residual magnitude $\|\alpha_t \delta a_t\|$ (rad) accumulated over TRANSIT. **TRANSIT outcomes** distinguish TRANSIT-reached (convergence) from TRANSIT-timed-out (300-step cap), and we do not count a timeout as a task failure when placement succeeds. We exclude episode 0 as warm-up, and OFF/ON share matched scenario configurations ($N = 40$ per mode per scenario, with conveyor supply and episode-synchronised recycle). Artifact paths are in the supplement.[3]

5 Results and Analysis

All results below are Webots-online measurements ($N = 40$ episodes per mode per scenario, with episode 0 excluded as warm-up) under episode-synchronised conveyor supply. We retain offline AUROC/EDR values only as supplementary diagnostics and do not report them as operational outcomes.

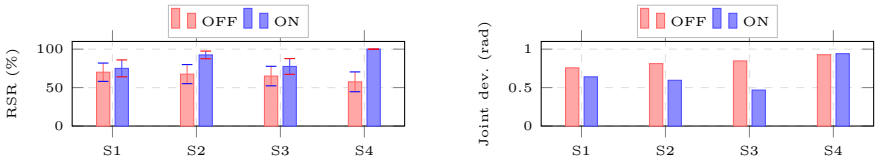
5.1 Overall Performance

Table 2 pools the four recoverable scenarios (S1–S4, $N = 40$ each, denominator 160) into a single Clean RSR metric. We compare two deployed policies on the same RSSM+ANFIS signal: OFF aborts at $\alpha \geq 0.60$ for 5 steps, whereas Ours (ON) continues with a bounded residual correction and an emergency guard at $\alpha \geq 0.90$ for 8 steps. The OFF/ON gap therefore reflects both the threshold relaxation and the corrective layer (L7, Sec. 6). On the pooled denominator, ON improves over OFF by +21.25 pp (138/160 vs. 104/160). B3 and B4 are detector-only diagnostics (recovery N/A), with FPR evaluated over the held-out 70 nominal episodes (10 of the 80 used for B3 calibration). For Ours, the observed FPR is 0/80 on the full nominal set (Wilson CI [0.0, 4.6]). End-to-end CPU latency is P95 = 34.24 ms (mean 25.63 ms, P99 = 38.17 ms, see Sec. 5.4).

5.2 Per-Scenario Performance and Trajectory Smoothing

Figure 5 reports per-scenario OFF/ON RSR (a) and mean joint deviation (b), and we give the per-scenario detail in Table 4. Gains concentrate on the

[3] `https://github.com/haitran-13/acivs2026-supplementary`



(a) RSR (Wilson 95% CI). (b) Mean joint deviation.

Fig. 5: Per-scenario recovery (a) and trajectory quality (b); $N = 40$ per mode, OFF (red) vs. ON (blue). S5/S6 omitted (no recoverable trajectory; see Table 1).

Table 3: Detector TPR / intervention rate on S1–S4 (Wilson 95% CI).

System	S1 Vib.	S2 Jerky	S3 Pause	S4 Speed
B1: Reactive Thresh.	30.0 [18.1,45.4]	20.0 [10.5,34.8]	20.0 [10.5,34.8]	22.5 [12.3,37.5]
B2: EfficientAD-style	40.0 [26.3,55.4]	32.5 [20.1,48.0]	32.5 [20.1,48.0]	35.0 [22.1,50.5]
B3: RSSM KL-only	15.0 [7.1,29.1]	32.5 [20.1,48.0]	27.5 [16.1,42.8]	17.5 [8.7,32.0]
B4: RSSM+ANFIS (detect)	0.0 [0,8.8]	0.0 [0,8.8]	2.5 [0.4,12.9]	0.0 [0,8.8]
$x_{3,\mathrm{std}}$ only	0.0 [0,8.8]	0.0 [0,8.8]	0.0 [0,8.8]	0.0 [0,8.8]
Ours int. ($\alpha \geq 0.20$)	**67.5 [52.0,79.9]**	**100 [91.2,100]**	**100 [91.2,100]**	**100 [91.2,100]**

harder disturbances: **S2 Jerky** (+25.0 pp) and **S4 Speed** (+42.5 pp) show non-overlapping Wilson 95% CIs, while S1 and S3 remain within CI overlap. Smoothing is complementary, with the joint-deviation reduction ranging from 15.4% (S1) to 44.8% (S3). S4 is essentially unchanged because the velocity cap dominates the speed fault. The per-step residual magnitude $\|\alpha_t \delta a_t\|$ stays below 0.20 rad across all scenarios (action scale 0.30 rad), and the ON-mode timeout fractions concentrate on S2 and S4 (we revisit this below).

5.3 Baselines and Feature Ablation

Table 3 reports detector TPR on the condition-based TRANSIT for S1–S4. We calibrate B1/B2 to 0% FPR on 10 normal episodes (diagnostic), we set B3 at 0.4132 (1.1× max calibration KL), and we use B4 at $\alpha \geq 0.92$ for 3 steps. **Ours** engages the deployed controller at $\alpha \geq 0.20$ on the same signal, which is an intervention rate rather than an alarm TPR. Three readings stand out. First, pixel-only $x_{3,\mathrm{std}}$ collapses to 0% TPR, which confirms that the RSSM latent dynamics are necessary. Second, B1/B2/B3 reach moderate TPR (15–40%) but only as binary alarms, with no path to graduated correction. Third, tightening to B4's high-confidence criterion drops detector TPR to $\leq 2.5\%$, which makes plain that conventional thresholding cannot supply both low FPR and useful coverage on this signal. The Ours-intervention row corresponds to the Clean RSR of 86.3% in Table 2.

5.4 Per-Scenario Measurements

Table 4 consolidates the operational metrics for S1–S4: RSR (with Wilson 95% CI), the OFF/ON joint-deviation pair and its smoothing ratio, the ON-mode mean residual magnitude EWR $= \mathbb{E}[\|\alpha_t \delta a_t\|]$, and the ON-mode TRANSIT-timeout fraction. We omit S5/S6 because both modes give 0% RSR (Table 1).

Table 4: Per-scenario measurements ($N = 40$ per mode). j = mean joint deviation (rad); Smooth = $(j_{\mathrm{OFF}} - j_{\mathrm{ON}})/j_{\mathrm{OFF}}$; EWR = mean per-step $\|\alpha_t \delta a_t\|$ (rad, ON); T-out = TRANSIT timeout count (ON). Clean pool aggregates S1–S4 ($N = 160$).

Scenario	**RSR [CI]**		j **(rad)**		**Smooth**	**EWR**	**T-out**
	OFF	ON	OFF	ON		ON	ON
S1 Vibration	70.0 [54.6,81.9]	75.0 [59.8,85.8]	0.76	0.64	−15%	0.086	6/40
S2 Jerky	67.5 [52.0,79.9]	**92.5 [80.1,97.4]**	0.81	0.60	−27%	0.196	37/40
S3 Pause	65.0 [49.5,77.9]	77.5 [62.5,87.7]	0.85	0.47	−45%	0.176	25/40
S4 Speed	57.5 [42.2,71.5]	**100.0 [91.2,100]**	0.93	0.94	−1%	0.199	40/40
Clean (S1–S4)	65.0 [57.3,71.9]	**86.3 [80.0,90.8]**	0.84	0.66	−22%	0.164	108/160

TRANSIT-timeout reading. Timeout is a phase-exit signal rather than a task-failure flag: placement still enters the RSR count. Mean TRANSIT steps (ON/OFF): S1 118/64, S2 300/158, S3 265/96, S4 300/300. S2 and S4 episodes exhaust the full cap, so their RSR gains reflect task completion under a more permissive abort threshold (L7), not faster recovery (L9).

Latency. We measure P95 = 34.24 ms (mean 25.63 ms, P99 = 38.17 ms) on an i5-11320H CPU with OpenVINO, aggregated over 9,969 step-level samples from 19 deployed-controller runs. We meet the real-time CPU budget (< 100 ms) with substantial headroom. The condition-based TRANSIT overhead (the joint-state reach check and the supply-synchronisation poll on every step) accounts for the latency above a fixed 180-step TRANSIT baseline (P95 $\approx$ 25 ms).

6 Discussion

RQ1: Online Risk Detection. RSSM latent dynamics carry information that pixel statistics do not: pixel-only $x_{3,\mathrm{std}}$ collapses to 0% TPR at the calibrated 0% FPR (Table 3). Conventional thresholds on the RSSM signal (B1/B2/B3) reach moderate TPR (15–40%) but only as binary alarms, and tightening to B4 ($\alpha \geq 0.92$/5-step) drops detector TPR to $\leq 2.5\%$. We take this as evidence that binary thresholding cannot extract usable detection at low FPR. Our system uses the same signal but engages the residual controller at $\alpha \geq 0.20$, turning partial detection into the graduated correction analysed below.

RQ2: Risk-Aware Adaptive Control. The ON policy improves RSR with non-overlapping Wilson 95% CIs on the harder disturbances (S2 Jerky +25.0 pp and S4 Speed +42.5 pp), while S1 (+5.0 pp) and S3 (+12.5 pp) remain within CI overlap. Two qualifications shape the reading. First, the OFF/ON gap couples threshold relaxation with residual correction (L7), so the smoothing reduction on S1–S3 gives us the cleaner controller-level view. Second, the largest gains (S2, S4) coincide with high ON-mode timeout fractions, so placement succeeds by absorbing the 300-step cap rather than by faster recovery. Beyond RSR, ON reduces mean joint deviation by 15.4% (S1), 26.7% (S2), and 44.8% (S3) while it keeps the per-step residual bounded ($\|\alpha_t \delta a_t\| \leq 0.20$ rad, action scale 0.30 rad). S4 is essentially unchanged because the velocity cap dominates.

RQ3: Latency. End-to-end P95 latency is 34.24 ms (9,969 step samples, 19 runs), and we meet the 100 ms budget with 65.8% headroom. DINOv2 ONNX dominates the runtime.

Limitations. **(L1)** Recovery is joint-space only: S5/S6 yield 0% RSR in both modes, and physical-obstacle avoidance exceeds what a 0.30 rad joint residual can express. **(L3, L4)** All numbers are from Webots, and the SAC observation uses a Webots-instrumented can pose that we must replace with a perception-derived estimate for physical deployment. **(L5)** Because α^* is a clipped linear combination of KL and $x_{3,\mathrm{std}}$, we use ANFIS for semantic labelling and do not claim that it is superior in MSE to a calibrated linear regressor. **(L7)** The OFF/ON RSR gap couples threshold relaxation ($\alpha \geq 0.60/5 \rightarrow \alpha \geq 0.90/8$) with residual correction, so we do not separately quantify the residual contribution in isolation, and the smoothing analysis is the closest proxy. **(L8)** We use a single training seed (seed 42), so the Wilson 95% CIs capture scenario noise rather than training-seed variance. **(L6, L9)** High ON-mode TRANSIT-timeout counts mean that placement success differs from fast recovery, and we have not yet quantified the placement-time distribution (the RSR-vs-cycle-time trade on S2/S4). **(L10)** We include no image-aligned per-frame KL/α/residual traces in the main paper, which is the most addressable gap for an intelligent-vision-systems audience.

7 Related Work

We organize ten representative works (2023–2026) by four functional modules: (M1) Perception, (M2) World Model, (M3) Risk Assessment, and (M4) Control. Table 5 summarizes coverage, where ✓ / – marks present or absent and "Prop." means proportional non-binary recovery. No prior system couples all four modules to a proportional recovery action.

Table 5: Module coverage.

Method	M1	M2	M3	M4	Prop.
SafeDreamer [4]	✓	✓	✓	✓	–
FARL [5]	✓	✓	✓	✓	–
WM-AD [6]	✓	✓	✓	✓	–
RAPT [7]	✓	✓	✓	✓	–
DreamerV3 [2]	✓	✓	–	✓	–
FIPER [8]	✓	✓	✓	–	–
V-JEPA 2 [1]	✓	✓	–	–	–
Fuz-RL [9]	–	–	✓	✓	–
EfficientAD [3]	✓	–	✓	–	–
AHA [10]	✓	–	✓	–	–
Ours	✓	✓	✓	✓	✓

In robot control, residual RL [17] motivates bounded intervention but does not by itself provide semantic risk tiers from visual prediction error.

Three gaps motivate our design. **Gap 1:** world models (DreamerV3 [2], V-JEPA 2 [1]) predict richly but lack a safety layer over prediction risk. **Gap 2:** no full 4/4-module system scales recovery by a continuous risk coefficient. For instance, WM-AD [6] halts at a threshold, FARL [5] switches policies in a binary fashion, and RAPT [7] delegates correction to a human operator. **Gap 3:** the detection literature (EfficientAD [3], FIPER [8], AHA [10]) flags failures cleanly but likewise hands recovery to operators. We close these gaps by feeding RSSM prediction error through a graduated risk layer into a bounded residual controller, on a CPU-only real-time budget.

8 Conclusion

We presented a four-module architecture for predictive anomaly detection and risk-aware adaptive control in pick-and-place. It combines a frozen DINOv2 encoder, an RSSM KL-divergence anomaly score, ANFIS graduated risk, and a bounded residual-SAC correction to turn one latent prediction signal into proportional intervention, and it addresses three gaps in prior work (world-model prediction without safety logic, recovery without graduated risk scaling, and detection without robot-side response). Under condition-based TRANSIT in Webots ($N = 40$ per mode) on four recoverable disturbances plus two out-of-scope faults, our system reports FPR = 0.0% on 80 nominal episodes and a Clean RSR of 86.3% for ON versus 65.0% for fail-safe halt (a deployed-policy comparison, L7), with non-overlapping Wilson 95% CIs on S2 (+25.0 pp) and S4 (+42.5 pp), smoothing of 15.4–44.8% on three motion disturbances, and P95 latency 34.24 ms. Future work targets L1–L10: a matched-threshold ablation, a multi-seed sweep, image-aligned per-frame traces, time-to-place, an ANFIS-vs-linear comparison, physical UR5e transfer, and S5/S6 recovery.

References

1. Assran, M., et al.: V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. arXiv:2506.09985 (2025)
2. Hafner, D., et al.: Mastering diverse domains through world models. arXiv:2301.04104 (2023)
3. Batzner, K., Heckler, L., König, R.: EfficientAD: Accurate visual anomaly detection at millisecond-level latencies. In: WACV, pp. 127–137 (2024)
4. Huang, W., et al.: SafeDreamer: Safe reinforcement learning with world models. In: ICLR (2024)
5. Li, H., et al.: Failure-Aware RL: reliable offline-to-online reinforcement learning with self-recovery for real-world manipulation. arXiv:2601.07821 (2026)
6. Domberg, F., Schildbach, G.: World models for anomaly detection during model-based reinforcement learning inference. arXiv:2503.02552 (2025)
7. Munn, H., et al.: Model-predictive out-of-distribution detection and failure diagnosis for sim-to-real humanoid robots. arXiv:2602.01515 (2026)
8. Römer, R., et al.: Failure prediction at runtime for generative robot policies. NeurIPS (2025)
9. Wan, X., et al.: Fuz-RL: a fuzzy-guided robust framework for safe reinforcement learning under uncertainty. NeurIPS (2025)
10. Duan, J., et al.: AHA: A vision-language-model for detecting and reasoning over failures in robotic manipulation. arXiv:2410.00371 (2024)
11. Michel, O.: Webots™: professional mobile robot simulation. Int. J. Adv. Robotic Systems **1**(1), 39–42 (2004)
12. Oquab, M., et al.: DINOv2: learning robust visual features without supervision. TMLR (2024)
13. Jang, J.-S.R.: ANFIS: adaptive-network-based fuzzy inference system. IEEE Trans. SMC **23**(3), 665–685 (1993)
14. Zadeh, L.A.: Fuzzy sets. Information and Control **8**(3), 338–353 (1965)
15. Takagi, T., Sugeno, M.: Fuzzy identification of systems and its applications to modeling and control. IEEE Trans. SMC **SMC-15**(1), 116–132 (1985)
16. Haarnoja, T., et al.: Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. arXiv:1801.01290 (2018)
17. Johannink, T., et al.: Residual reinforcement learning for robot control. arXiv:1812.03201 (2018)
18. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction, 2nd edn. MIT Press (2018)